Coalition Formation for Multi-agent Pursuit Based on Neural Network

Zhaoyi Pei1 · Songhao Piao1 · Mohammed El Habib Souidi2 · Muhammad Zuhair Qadir1 · Guo Li1

Abstract

Multi-agent Systems (MAS) often needs to pursue a common goal and in order to achieve that they need to make an effective formation. An approach for coalition formation of multi-agent pursuit based on neural network (NN) and agent group role membership function (AGRMF) model is proposed and thus making a novel algorithm called ARGMF-NN. This new algorithm consists of two parts i.e. feature extraction and group generation. First, the layers of feature extraction can abstract the AGRMF feature for all of the groups. Secondly, those features will be fed to the group generation part based on self-organizing map (SOM) layer which is used to group the pursuers with similar features. Besides, we also come up with the group attractiveness function (GAF) which is used to evaluate the quality of coalitions and the pursuers contribution. It helps in adjusting the main ability indicators of AGRMF and other weights of the whole neural network. The simulation experiment showed that this proposal can improve the effectiveness of coalition formation for multi-agent pursuit and also the ability to adopt pursuit-evasion problem with the scale of growing pursuer team.

Keywords

Multi-agent system (MAS) · Neural network (NN) · Agent group role membership function (AGRMF) · Pursuit-evasion · Group attractiveness function (GAF)

1 Introduction

Nowadays, multi-agent systems (MAS) attracts more attention in research concerning the distributed artificial intelligence. Many research findings on the said subject have been applied to intelligent robots as a result achieving amazing performance. NASA is exploring a multi-agent system known as Swarmie, it is made for the exploration of extraterrestrial resources [1]. Each agent has a complete set of communication devices and sensors. It can independently discover potential value resources and call for the partners to explore that together, such a system has a certain degree of robustness. Even if part of it were get damaged, it would not be all paralyzed. The adaptive system research team of Harvard University designed a multi-robot system in 2014 to achieve complex behaviors such as self-assembly just relying on the mutual communication between adjacent agents [2]. It is an ideal platform for studying complex systems and group behavior. The Autonomous Vehicle Emergency Recovery Tool (AVERT) studies the tasks which need agents closely coordinated to accomplish such as lifting weights, pushing boxes, etc. Four robots based that tool can easily move and transport two tons of vehicles through collaboration [3].

These applications demonstrate state of the art of MAS in which pursuit problem is drawing more attention. It is also due to many applications such as search and rescue in disaster area, patrol on a larger scale and military combat. Cooperation formation of pursuers (i.e. agents hunting the evaders) is one of the key technologies of pursuit evasion because when multi-agent pursuit systems are facing multiple targets, it is bound for them to deal with the optimization problem in which pursuers has the task to coordinate among themselves. For a multi-agent system, the problem of how
to make all the pursuers to convert the material resources to good account relates to the difference of pursuers as well as that of evaders and environmental dynamics.

This paper proposes a coalition formation mechanism of multi-agent pursuit based on neural network and AGRMF model, we apply a novel neural network called AGRMF-NN to abstract each pursuer’s AGRMF feature and learn the main ability indicators. The pursuers use the special main ability indicators from AGRMF-NN in face of given evaders. Then the self-organizing map(SOM) fed by those features will create groups for pursuers. Finally, pursuers in one group will be assigned their target according to AGRMF. In order to adjust the main ability indicators of AGRMF and other weights of all neural network, the learning algorithm and group attractiveness function(GAF) are also discussed.

The paper is organized as follows: in the Section 2, related work about coalition formation for multi-agent system is discussed. In the Section 3, the multi-agent pursuit problem is described. The part of features extraction for pursuers and that of group generation will be discussed respectively in Section 4. The group attractiveness function will be deduced in Section 5 along with the network training process. The complete algorithm of the approach is shown in Section 6.

2 Related Work

Lot of work has been done and applied for coalition formation of multi-agent system such as bionics, sociology and game theory. Broadly, these methods can be divided into two types : behavior-based and plan-based. Behavior-based are inspired by some biological systems such as swarm, ant colony and other social animals. Such multi agent system is composed of large number of robots, each of which interacts with the environment and other agents obeying simple behavior rules. Though each of them only have limited capability and resources just like bee in swarm, well designed behavior rules enables such systems to complete the specific tasks. The presentative algorithms of this method are discussed in ALLIANCE [5], Broadcast of Local Eligibility(BLE) [6], swarm optimization [7], and emotional recruitment [13]. Research on distributed coordination of dynamic agents in network has also achieved results and has been applied to practical MAS. Denis Budaev and Konstantin Amelin proposed a network-centric multi-agent system for real-time task scheduling for UAV’s group [8]. Literature [9] proposed a decentralized multi-copter flock that performs stable autonomous outdoor flight which navigate themselves based on the dynamic information received from other agents in the vicinity.

Some approaches based on game theory can also be regarded as of this type, literature [26] came up with a decentralized team formation based on the Iterated Elimination of Dominated Strategies. Adel. Ghaizikhani proposed an coalition formation algorithm based on Shapley value [10]. An approach to compute the optimal strategy for a pursuer that maximizes the probability of capturing an evader is presented in literature [11]. literature [12] proposed a solution employing strategic multi-agent planning based on game theory which make the best of any shared journey plan found for timetabled transport services.

It is difficult to analyze the behavior-based methods because the accuracy of macro behavior emerged cannot be guaranteed effectively. In other words, it is hard to conduct the behavior of the agent directly and users can only set the simple rules for the agents to obey. On the contrary, the explicit plan-based method is according to the goal and has an easy implementation comparatively. It is more effective in performing the specific tasks due to designed plan. Baofu Fang and LuChen proposed a two step-auction mechanism to recruit teams based on emotional cooperation factor [14]. A. Rauniyar, PK. Muhuri applied immigrants based on random immigrants genetic algorithm and elitism to optimal coalition formation considering dynamic variants [18]. The privacy model is proposed to ensure the competitiveness and autonomy of agent when plan-based methods are applied for coalition formation [15]. Vaibhav Katewa and Fabio Pasqualetti designed a noise-adding mechanism according to the classic differential privacy framework which is implemented by agents [17].Literature [16] propose a class of iterative algorithms for solving private distributed optimization problem in order to achieves differential privacy and convergence to a common value.

Besides, there is a new branch these years called synergy graph which uses graphs to represent task-based compatibility and teamwork ability of all agents. It applies graph theory to find the optimal team formation [19]. Then they combine learning agent theory and synergy graphs to promote agent’s learning instances; instead of considering their capabilities which are static [20]. It will be hard for plan-based method to determine the ability indicators [4] which represent the extent to which each attribute contributes to the task when the scale of MAS is too large. So the goal here is to make up for this deficiency.

3 Description of Multi-agent Pursuit Problem

3.1 Pursuers

The set of pursuers is denoted by:

\[ P = \{p_1, p_2, ... p_n\} \]
Each agent should be described by two additional Capacity parameters in order to determine membership function of AGR model:

**Self-conference Degree** There are many ways that can represent one pursuer properly but one of the most intuitional representations is the success rate of tasks. This parameter can be considered as self-conference degree which can be calculated by follows:

\[ \forall Conf \in [\lambda, 1] : Conf = \max \left( \lambda, \frac{C_s}{C_t} \right) \]  

(1)

where \(C_s\) is the number of tasks that the agent has finished with success. \(C_t\) is the number of tasks that the agent has participated. The function of lamda is to control the magnitude of the change of Conf range like a rectified linear function.

**Credit** Unlike Conf, Credit measures the ability of agent to work with other agents. If that credit is low that means agent may not able to execute the services requested by the other agents properly. Because the failure of the task will also affect the performance of other agents. The credit of an agent is denoted as follows:

\[ \forall Credit \in [0, 1] : Credit = \min \left( 1, 1 - \frac{C_b}{C_t - C_s} \right) \]  

(2)

while \(C_b\) is the number of tasks that the agent has to abandon.

**Distance** Distance from pursuer to evader in environment is a crucial criterion for the pursuit-evasion. The distance Dist between pursuer \(P\) and evader \(E\) is computed as follows:

\[ Dist_{PE} = \sqrt{(cc_{P_i} - cc_{E_i})^2 + (cc_{P_j} - cc_{E_j})^2} \]  

(3)

Here \((cc_{P_i}, cc_{P_j})\) is the Cartesian coordinates of the pursuer and \((cc_{E_i}, cc_{E_j})\) is the Cartesian coordinates of the evader. Credit and Conf can be considered as the main abilities of one agent, however the dist is one of the limitation of completing tasks which cannot be determined by the agent itself.

### 3.2 Evaders

The set of evaders is denoted by

\[ E = \{e_1, e_2, ... e_n\} \]

each evader has its own pursuit difficulty which means how many pursuers are needed to catch that evader, the set of pursuit difficulty is denoted by

\[ D = \{d_1, d_2, ... d_n\} \]

Pursuer who catch evader \(e\) will get the reward which is equal to \(d_e\).

### 3.3 Organizer

This agent is the mainstay of the chase as it will create the chase groups and extract features of pursuers, therefore it is the place where AGRMF-NN is deployed. The organizer should be trusted by all pursuers and can’t disclose the parameters of a pursuer to others.

### 3.4 Strategy of Pursuit

Multi-agent cooperative pursuit-evasion is that multiple agents, through a certain cooperation to complete the tracking, intercept and finally capture of multiple targets. One evader will be considered as arrested if the neighborhood are invaded by enough agents as Eq. 4 and Fig. 1.

\[ d_n \leq \sum_{p \in P} \text{neighbor}_e(p) \]  

(4)

the neighbor of \(e\) represents a fixed area which is centered on \(e\), \(\text{neighbor}_e(p)\) means whether \(p\) come into the neighborhood or not.

Gravitational field algorithm [14]: There is a virtual force field in the environment according to the distance between pursuers and evaders, the force from target to the pursuers is the attraction and denoted as:

\[ F_a = \gamma \frac{1}{Dist_e} \]

where \(\gamma\) is scale factor; \(Dist_e\) means the distance between \(p\) and his target \(e_i\) on the contrary, the force from pursuers to each pursuer is the repulsion.

\[ F_r = \gamma \left( \Sigma_{i=0}^{n} \frac{1}{Dist_{pi}} \right) \]

Fig. 1 Instances of successful arrest
The strategy of evaders and pursuers that we have chosen is the combination of gravitational field algorithm and bug2 algorithm. The change of velocity of agents is determined by the resultant force. Bug2 algorithm is a simple and an effective obstacle avoidance algorithm. If agent meets obstacles in the direction of velocity, it will take this algorithm to avoid obstacles as shown in Fig. 2.

Taking into account the impact of obstacles on pursuit operations, \( Dist \) should be calculated as total distance of the path which is determined by obstacle avoidance algorithm. Take Fig. 3 as an example, \( Dist_{e1} = Dist_{p1} = |p1A| + |AO| + |OB| + |Be1| \), and the direction of the force is pointing to the nearest end point, so the direction of \( F_{e1} \) is \( p1A \) instead of \( e1p1 \). Agent’s velocity changes under the effect of force. The MAS for pursuit-evasion simulation will calculate the position of each agent at set intervals, and check whether the pursuit is successful or not. We assume that the change in the agent speed is instantaneous. So agent can be seen as a uniform motion at one interval.

4 Structure of AGRMF-NN

4.1 Agent Group Role Membership Function

A new variable is needed to represent the fitness of an agent to perform a task in a group. Agent group role membership Function(AGRMF) is an extension of agent-group-role model. Unlike other institutions, this approach allows agents to consider group as fuzzy set. Consequently, each group can be managed via the utilization of a membership function. Instead of figuring out one explicit result that whether one special agent is supposed to one of the groups or not, AGRMF admits to the degrees of membership to a given group. This change will provide more space to use different algorithms to optimize the result for the organization and more flexibility for the reorganization compared with the ordinary AGR model. Roles of one group can not be played by any agent without considering its property. In AGRMF model, one group means one task. When the task is over ,the group should be dismissed so \( \mu_{t}(p) = \mu_{g}(p) \) here the \( t \) is the task that group is executing. In our algorithm, AGRMF of \( P \) to group \( g \) \( \mu_{g}(p) \) determined by \( \text{credit}_p, \text{conf}_p, \text{dist}_p \) is used to represent fitness of pursuer \( p \) to join group \( g \) and perform task \( t \) as Eq. 5.

\[
\mu_{g}(p) = \frac{\text{coef}_t^1 \times \text{Pos}_p + \text{coef}_t^2 \times \text{Conf}_p + \text{coef}_t^3 \times \text{Credit}_p}{\sum_{i=1}^{3} \text{coef}_t^i (5)}
\]

The main idea of this algorithm is to select the pursuer with the maximal membership degree of target to join group. AGRMF is the most important part of the whole algorithm. Here \( (\text{coef}_t^1 - \text{coef}_t^3) \) are called as main ability indicators which are used to determine the Contribution by the degree of Conf, Credit and Dist to the result e.g. \( \text{coef}_t^1 \) is relatively greater while the others \( (\text{coef}_t^2, \text{coef}_t^3) \) of that are relatively small which means that the task \( t \) require the agents which have a high degree of Conf to accomplish. The more suitable agent will also obtain higher membership. For instance, if one vector \( \{\text{coef}_t^1, \text{coef}_t^2, \text{coef}_t^3\} \) of task \( t \) is \( [0.2, 0.7, 0.1] \), the ability vector of pursuer \( p1 \) is \( [0.2, 0.7, 0.1] \) and that of \( p2 \) is\( [0.7, 0.2, 0.1] \), here we can see that obviously the \( p1 \) is more suitable and get the higher \( u_t \). Main ability indicators will be hard to define when there are large number of agents in large environment and it is not appropriate to set them as constants because of the variability of the environment.
Algorithm 1 AGRMF

**input:** $U$ the set of vector of AGRMF of evader, $n$ size of $U$

**output:** group

1: function GETGROUPOFPURSUERS($U$, $n$)
2: pursuersgorup:empty dictionary
3: for each $e \in E$ do
4: $x \leftarrow 0$
5: repeat
6: for each $p \in P$ do
7: if $(p \notin \text{group}[e]) \& \& u_e(p) = \text{Max}(u_e)$ then
8: Add (gorup[e],p);
9: Delete (P,p);
10: $x \leftarrow x + 1$
11: end if
12: end for
13: until $x = e.d$
14: end for
15: end function

### 4.2 Feature Extraction of Pursuers

In this section, we will discuss the feature extraction part for pursuers of AGRMF-NN. Pursuer $i$ can be represented by one feature vector $x_i = \{Cedit_i, Conf_i, Dist_{i1}, Dist_{i2}, ..., Dist_{im}\}$ where the $Dist_{im}$ means the distance between pursuer $i$ and evader $m$. By Eq. 5, we can see that numerator of $\mu_t(p)$ can be expressed as the excitation of the input vector $[Conf_t Cedit_t Cist_t]$ to a neuron whose weight vector is $[Coef f_1 Coef f_2 Coef f_3]$. Each neuron $t$ can be seen as a filter that selects the agent which is appropriate for the task $t$. Therefore, the weight vector of neuron $t$ $w_t = [\text{coef } f_1, \text{coef } f_2, \text{coef } f_3]$ has the same purpose of main ability indicator vector of task $t$. All of filters make up of one filter layer, in another words, one layer of neurons. The layer is a convolutional layer instead of full connection layer because one neuron of that layer is not connected with all neurons. Denominator of right part of formula (5) $\sum_{i=1}^{3} Coef f_i$ is designed to normalize the vector $w_t = [\text{coef } f_1, \text{coef } f_2, \text{coef } f_3]$. The regularization process, limiting the initial weights, and adjusting the activation function can make the convolution layer achieve this goal. To sum up, the output of the convolution layer can represent $u_t(p)$ for each pursuer $p$ and task $t$ so that the update of weights of this layer also represents the change of the system’s cognition to the task attributes according to the pursuit process, the method of weight update will be explained detailedly in section 5. Input layer is followed by this convolutional layer. Then the hidden layer can improve the approximation ability of the neural network [21]. The structure of this part is shown in Fig. 4. The feature extraction layer finally output the feature of each pursuers which can be called as AGRMF-feature.

### 4.3 SOM Layer for Group Generation

Only feature extraction is not enough for team generation. AGRMF-NN should learn how to assign pursuers to proper groups to make full use of each agent. Obviously, pursuers with similar AGRMF-features are supposed to be assigned to a group with great probability. Therefore, a Self Organized Map(SOM) layer is added after the feature extraction part. The SOM is one of the most popular neural networks for unsupervised learning and clustering.

**Fig. 4** Feature extraction part for pursuers
network models which is a type of unsupervised learning approach [23]. SOM is widely applied for analyzing the intrinsic characteristics of the data [24, 25]. SOM layer is a single and fully connected layer. Provided training set \( X = \{X_1, X_2, X_3 \ldots X_p \} \), for each \( x \) in training set and the network outputs one winner node whose response is the maximum among all the node like Eq. 6 represents.

$$ y = \arg\max_{k=1,2,3 \ldots m} (W_k \cdot X) $$

The learning algorithm of SOM is competitive learning which will promote the weight vector connected to the winner node of \( X \) to represent the \( X \)'s characteristic best in the whole layer. The weights updates as Eq. 7 represents.

$$ \begin{align*}
    w_{ji}(t+1) &= w_{ji}(t) + \alpha(x_i(t) - w_{ji}(t)) & y_j = \text{winner}(x) \\
    w_{ji}(t+1) &= w_{ji}(t) & y_j \neq \text{winner}(x)
\end{align*} $$

This process can be shown as Figs. 5 and 6 where the input space is a 2-dimensional space, \( w_1, w_2 \) and \( w_3 \) are weight vectors. All the weights reaches the center of clusters after competitive learning. For our application, the \( W \) will represent the characteristic of Groups.

The complete construction of neuron network for coalition formation is shown as Fig. 7. The intermediate layer is output layer of AGRMF features as well as input layer of Generation Groups. Error is passed from the output layer to the hidden layer which will be discussed in section 5. When the neural network converges, the weight vector of neurons \( i \) helps represents the center of AGRMF features of pursuers whose winner neuron is \( i \). The first part algorithm i.e. the training of SOM and the other part i.e. getting the final group result are both represented in algorithm 2.

The algorithm’s steps of training the SOM inspired by AGRMF feature vectors and getting final group result are presented as algorithm 2. The algorithm’s steps are explained in the following manner: the step 2 to 7 are used to initialize the parameters and SOM, the step 8 to 18 are the learning process of SOM where step 13 are inspired by Eq. 7 and step 14 is inspired by Eq. 7. Finally the list consisting

**Algorithm 2 Group-generation**

**input:** \( FU \): The set of AGRMF feature vectors of pursuers, \( n \): Size of \( FU \)  
**output:** List : The list consisting of index of group each pursuer’s belongs to

1: function GETGROUPOFPUSHERS(\( FU, n \))  
2: List:empty list  
3: \( \lambda \) : time of iteration  
4: \( m \) : the number of output neuron  
5: for \( j \) from 0 to \( m \) do  
6: random initialize \( W_j \)  
7: end for  
8: repeat  
9: \( k \) \leftarrow 0  
10: repeat  
11: \( i \) \leftarrow 0  
12: \( f_{\mu_i} \in FU \)  
13: \( o := \text{win}(W, f_{\mu_i}) \)  
14: update weight(\( o \))  
15: \( i \) \leftarrow \( i+1 \)  
16: until \( i \) \( = n \)  
17: \( k \) \leftarrow \( k+1 \)  
18: until \( k \) \( = \lambda \)  
19: repeat  
20: \( i \) \leftarrow 0  
21: \( f_{\mu_i} \in U \)  
22: \( o := \text{win}(W, f_{\mu_i}) \)  
23: add(\( o, \text{List} \))  
24: \( i \) \leftarrow \( i+1 \)  
25: until \( i \) \( = n \)  
26: return List  
27: end function
of index of group each pursuer belongs to will be generated. The pursuers who get the same output from SOM will be assigned to the same group, the pursuers of group \( g \) decide their target evader \( e \) according to their AGMF as Eq. 8. Each group will only own one target at the same time.

\[
e = \max_{\arg} \sum_{p \in \mathcal{g}} (\mu_{e_1}(p), \mu_{e_2}(p)...\mu_{e_n}(p))
\]

(8)

5 Training Algorithm of AGRMF-NN

5.1 Group Attractiveness Function

Group attractiveness function(GAF) is proposed to evaluate the correctness of the coalition formation obtained by AGRMF-NN. Assigning a pursuer to the group that is attracted to it and it will be considered as a correct decision. \( GA_F_g(p') \) can be expressed as the ratio between the willingness of \( p \) to accomplish the task which is executed by group \( g \) and the existing difficulty of that. The willingness is the success rate \( U_t(p') \) which is calculated according to main ability vector of \( p' \). The existing difficulty can be represented by the average distance between the evader and pursuers which are responsible for catching it. The \( GA_F_g(p') \) is shown in Eq. 9.

\[
GA_F_g(p') = \frac{\mu_g(p') * d}{\sum_{p \in \mathcal{g}} dis(p, E)}
\]

(9)

It is beneficial for the MAS that each pursuer leaves the original group to pursue more interests [26], this is also the means of reorganization. Each agent has to make a decision between the interests in the original group \( GA_F_g(p) \) and the interests in the other groups \( \sum_{(g \in \mathcal{G} - g)} GA_F_g(p) \). The results are also determined by inertial factor \( \mu_p \) which represents the nature of pursuers to keep in the original group. This factor between 0 and 1 can be used to control the speed of the reorganization. The coalition evaluation function(CEF) is shown in Eq. 10.

\[
\begin{align*}
CEF(p) &= \begin{cases} 
1 & \mu_p * GA_F_g(p) > (1 - \mu_p) \sum_{(g \in \mathcal{G} - g)} GA_F_g(p) \\
0 & \mu_p * GA_F_g(p) < (1 - \mu_p) \sum_{(g \in \mathcal{G} - g)} GA_F_g(p)
\end{cases}
\end{align*}
\]

(10)

5.2 Back Propagation Based on CEF

The learning algorithm of AGRMF feature extraction updates the weights of NN whose purpose is to promote NN and to further understand the properties of all tasks which helps to assign tasks more optimally. The learning algorithm is a new back-propagation [22] algorithm based on CEF. The label for each training sample of pursuer \( p \) can be defined according to CEF\( ^{(p)} \) which is calculated in last iteration. \( CEF(p) = 0 \) indicates that \( p \) does not satisfy the task assigned by AGRMF-NN. So the label of the training vector \( x^p \) should be the feature vector of the group \( g' \) that is most attracted to it. In another words, the weight vector \( w_{g'} \) of the neuron represents \( g' \). The result of updating weights is that the AGRMF feature vector of \( p \) is closer to \( w_{g'} \), on
the contrary, when the CEF(p) = 1, the label of \( x^p \) is weight vector of winner neuron \( w^g \). The cost function is shown in Eq. 10.

\[
E(\overrightarrow{w}) = \frac{1}{2}(CEF(p) \sum_{d \in D} \sum_{k \in output} (win_k(d) - o_k(d))^2
+ (1 - CEF(p)) \left( \sum_{d \in D} \sum_{k \in output} (win'_k(d) - o_k(d)) \right)^2
\]

\( d \) is the feature vector of pursuer \( p \), \( o_k(d) \) is the output vector of AGRMF feature extraction part as well as the input vector of SOM layer. \( w_o(k) \) is the weights of the winner neuron in SOM layer, \( w'_o(k) \) is the weights of the neuron of group \( g' \) which is \( \max_{ARGF} GAF(p) \), \( CEF'(p) \) can be treated as a constant because it is a part of label and can not be changed at current iteration. \( E(\overrightarrow{w}) \) is a continuous derivable function so that the back propagation algorithm can be used to update the weights of feature extraction part of AGRMF-NN as Eq. 12.

\[
\delta(w) = -\eta \nabla_w \frac{1}{2}(CEF(p) \sum_{d \in D} \sum_{k \in output} (win_k(d) - o_k(d))^2
+ (1 - CEF(p)) \left( \sum_{d \in D} \sum_{k \in output} (win'_k(d) - o_k(d)) \right)^2
\]

6 Sequences of our Coalition Algorithm

The coalitions are short-lived and goal-oriented [4]. If a coalition lasts longer than the default requirement \( life \), it will be considered as a failure. The algorithm shown in Algorithm 3 is explained in the following manner: The organizer initializes the neuron network(01). Then the pursuers start to catch evaders under the coordination of the organizer until all evaders are captured(02–33). The organizer broadcasts position and value of each evader found in environment and wait the response from pursuers(03-06). The pursuers respond with their own feature vector \( \overrightarrow{x}_p \)(07–09). Then the organizer trains the SOM layer with the set of feature vectors \( X \) and sends target as well as \( \overrightarrow{GAF} \) to each pursuer after creating group \( p \)(10-14). The pursuer \( p \) respond their \( CEF(p) \) and start to pursuit its target(15-30). Finally, the organizer trains its feature extraction part with \( \overrightarrow{CEF} \)(31–32). The sequence diagram describes the organizer and the pursuers’ communication as shown in Fig. 8.

7 Simulation Experiment

The experiments are performed to verify the effectiveness of our overall algorithm, which includes both feature extraction part and the group generation part separately. The pursuit environment in these simulations is a “closed environment” with 100m * 100m, with 16 pursuers and 2 evader whose \( d \) is 4, 1 evader whose \( d \) is 3, 1 evader whose \( d \) is 2. “closed environment” means the map consists
of a couple random shaped obstacles that both pursuers and evaders cannot pass through within a defined bounded region. All the agents’ size is 20 cm * 20 cm. The neibor($p$) is a circle whose radius is 50 cm. $\lambda = 0.1$, $\gamma = 1.5$, $C_p = 0.7$. The evaders disappeared after being captured and the pursuers who captured them disappeared as well. One iteration represents one second in simulation environment.

Algorithm 3 Complete algorithm

```java
1: initialize the neuron network
2: repeat
3: for each $e \in E$ do
4: Broadcast (e.Pos, e.Re);
5: end for
6: for each $p \in P$ do
7: SendResponse($x_p$)
8: end for
9: GetResponse(X)
10: train the SOM layer
11: create-group()
12: SendMessage($e$,$\text{GAF}$)
13: Waiting-Response()
14: for each $p \in P$ do
15: if $p \neq e$ then
16: $p$.life = 0
17: end if
18: Launch of the chase
19: if $e$.captured = true then
20: $p$.Ct ← $p$.Ct + 1
21: $p$.Cs ← $p$.Cs + 1
22: end if
23: if $p$.life = $p$.life then
24: $p$.Ct ← $p$.Ct + 1
25: $p$.Cs ← $p$.Cs + 1
26: end if
27: GetResponse($CEF$)
28: Train the feature extraction part
29: until $\forall e \in E$ $e$.captured = true
```

Experiment 1 This experiment Verifies the feasibility of coalition formation of our algorithm based on AGRMF. The Fig. 9 shows the result of three different ways which are explained below after 100 experiments for each case:

- Case AGR: Formation of the groups without mechanism integration (without membership function) based on AGR model referred to in literature [4].
- Case AGRMF: Formation and reformation of the groups with the application of the coalition algorithm which was proposed in literature [4] before any new chase iteration.
- Case AGRMF-NN: Coalition formation with the application of our coalition algorithm.
In case AGR, the capture of all evaders has consumed an average of 229.78 chase iterations. For case AGRMF, capture of evaders is performed after an average of 175.10 chase iterations. For case AGRMF-NN, capture of the evaders is achieved after an average of 148.57 chase iterations. On the other hand, the error $E$ of feature extraction of AGRMF-NN’s when it converges or stops its learning process for each iteration is shown as Fig. 10, when the evaders are caught or when pursuers changes their targets, the error drops quickly. This part proves that our algorithm for reorganization can promote pursuit which can help in minimization of error from a new state of AGRMF-NN.

Experiment 2 This experiment mainly tests the function of feature extraction part of AGRMF-NN according to comparison between two cases:

- Case 1: AGRMF-NN without feature extraction part which means main ability indicators should never change and group generation part creates coalitions according to AGRMF directly.
- Case 2: AGRMF-NN.

The result is shown as Fig. 11, the average chase iterations for case 1 is 194.780.

Experiment 3 We select KMEANS and DBSCAN which represents the cluster algorithms based on partition and density respectively. They are contrasting objects to explain the advantage of group generation part inspired by our
algorithm. These approaches all have their own advantage for data set with different characteristic [27]. However, for this kind of problem, the algorithm inspired by SOM perform best as shown in Fig. 12. The result shows the density and the number of centers are changing dynamically during the process of chase, the competitive neuron network has a better ability to adapt to this change.

**Experiment 4** In order to validate our algorithm’s ability to adapt to larger environments, we designed experiment 4. The number of pursuers and evaders scales up as well as the size of environment for this experiment. Additional pursuer and evader are added in the previous experimental scene and the scope of the scene are expanded as well. The capture time of the three cases is shown as Fig. 13. Rate of decline for capture time of case AGRMF-NN compared with case AGRMF can be represented by a linear line as shown in Fig. 14. Because AGRMF-NN can learn the attributes of some agents in the previous environment, it has advantages in the face of larger environments.

**Experiment 5** This experiment shows the effectiveness of AGRMF-NN compared with the emotional robots model and auction model. The result is shown in Fig. 15.

- Case 1: Emotional robot model.
- Case 2: Auction model.
- Case 3: AGRMF-NN.

Compared with emotional robot model, AGRMF-NN reduced the capture time by 12.234% On the other hand. Compared with emotional auction model, AGRMF-NN reduced the capture time by 19.146% The main ability vector of AGRMF-NN can represent the confidence and
selfishness of the agent in the emotional model to a certain extent. The role of the feature extraction layer can also be seen as the pursuers’ bid to tasks in auction model. However, the two original models neither can learn characteristic of evaders and feature of groups.

Experiment 6 This experiment verifies the effectiveness of our algorithm when evaders took other three different strategies with stronger confusion and randomness.

Strategy 1: The evaders take the strategy introduced in section 3.4 to determine the main path of the motion. If space is enough, they will move forward with periodical turning like Fig. 16. Different evaders have different angular speed and radius of turning which can reflect their different characteristics more deeply. This will also disturb the judgment of pursuers and result in winning more escape space for themselves. This result is shown as Fig. 17.

Strategy 2: The evaders still move forward in periodical turning manner but the main path of motion is random. This method can avoid its entry to a dead angle to a certain extent. This result is shown in Fig. 18.

Strategy 3: Combined with the above two strategies, we put forward the strategy 3: When all pursuers are far away from a evader, its main path is random. When there are some pursuers closer to it, the virtual force field is used to determine the main direction for moving forward. When there are some pursuers very close to it, it will escape in the direction of the gap between the pursuers who is very close to it. If the conditions permit, they will still move forward periodically. This result is shown in Fig. 19.

The results of the total experiment are recorded in Table 1. The data format in Table 1 is (Mean value, Standard deviation). Strategy 1-3 represents the case 1-3 mentioned in Experiment 5. AGRMF-NN’s reduce capture time by 18.44% and 19.35% compared with emotional model and auction model in the best case. From the experimental results, we can see that when the evading strategy is more complex and deceptive, our algorithm is more effective. The learning ability of AGRMF-NN makes pursuers not be confused by the temporary action of the evaders, but determine the group and target based on the characteristics of the evaders.

8 Conclusion

In this paper. Coalition formation for multi-agent pursuit based on a novel neuron network called AGRMF-NN is discussed. It consists of feature extraction part and group generation part to learn main ability indicators. Besides, learning algorithm of AGRMF based on group attractiveness function(GAF) is also proposed. Compared with the traditional model, AGRMF-NN has the ability to learn evader characteristics, and pursuers can determine the coalition formation based on what have been learned by AGRMF-NN instead of the temporary behavior of the evaders. The simulation results shows AGRMF-NN’s reduce capture time by 25.549%, 12.234% and 19.146% compared with AGRMF, emotional model and auction model which is drastic improvement on previously researched techniques. Our algorithm also passed the test when evaders adopt more deceptive strategy and get the result that it reduce capture time by 18.44% and 19.35% compared with emotional model and auction model in the best case.

Applying machine learning to evaders can be of great research value. Next we want to try to apply Generative Adversarial Networks (GAN) to the game between pursuers and evaders.

Funding Information This paper is supported by National Natural Science Foundation of China [grant number 61375081]; a special fund project of Harbin science and technology innovation talents research [grant number RC2013XK010002].

Compliance with Ethical Standards

Conflict of interests The authors declare that there is no conflict of interest regarding the publication of this article.

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

References

1. Ducan, B., Ulam, P.: Behavior as a model for multi-robot systems[C]. In: 2009 IEEE international conference on robot and biomimetics (ROBIO), pp. 25–32 (2009)
2. Rubenstein, M., Cornejo, A., Nagpal, R.: Programmable self-assembly in a thousand-robot swarm[J]. Science 345(6178), 795–799 (2014)
3. May, R., Bickcht, I.B.: The avert project[EB/OL].(2015-3-1)[2016-1-5] http://new.avertproject.eu/
4. Soudi, Habib El, M., Songhao, P.: Multi-agent cooperation pursuit based on an extension of AALAADIN organisational model. J. Exp. Theor. Artif. Intell. 28(6), 1075–1088 (2016)
5. Li, M., Cai, Z.: ALLIANCE-ROS: a software architecture on ROS for fault-tolerant cooperative multi-robot systems. In: Pacific Rim international conference on trends in artificial intelligence. pp. 233–242 (2016)
6. WergerMaja, B., Matarić, J.: Broadcast of local eligibility for multi-target observation. Distributed Autonomous Robotic Systems 4, 347–356 (2000)
7. Yang, J., Zhang, H.: Task allocation for wireless sensor network using modified binary particle swarm optimization. IEEE Sensors J. 14(3), 882–892 (2014)
8. Budaev, D., Amelin, K.: Real-time task scheduling for multi-agent control system of UAV’s group based on network-centric technology. In: International conference on control, pp. 378–381 (2016)
9. Vasárhelyi, G., Virág, Cs., Tarcai, N., Szőrényi, T., Somorjai, G., Nepusz, T., Vics, T.: Outdoor flocking and formation flight with autonomous aerial robots. In: Intelligent robots and systems (IROS 2014), pp. 3866–3871 (2014)
10. Ghazikhani, A., Mashadi, H.R.: A novel algorithm for coalition formation in Multi-Agent Systems using cooperative game theory. In: 2010 18th Iranian conference on electrical engineering May 11–13 (2010)
11. Amigoni, F., Basilico, N.: A game theoretical approach to finding optimal strategies for pursuit evasion in grid environments. In: 2012 IEEE international conference on robotics and automation, May 14–18 (2012)
12. Hrnč Jr, J., Rovatsos, M., Michal, J.: Ridesharing On timetabled transport services: a multiagent planning approach. J. Intel1. Transp. Syst. 19 (2015), 89–105  (2015)
13. Delgado-Mata, C., Ibáñez-Martínez, J.: An emotion affected action selection mechanism for multiple virtual agents. In: International conference on artificial reality & telexistence-workshops. pp. 57–63 (2006)
14. Fang, B., Chen, L.: Research on multirobot pursuit task allocation algorithm based on emotional cooperation factor. Sci. World J., 2014, (2014-7-22) 2014(4), 864180 (2014)
15. Torreño, A., Oaindida, E.: Cooperative multi-agent planning: a survey. ACM Comput. Surv. 50(6), 1–32 (2017)
16. Huang, Z., Mitra, S.: Differentially private distributed optimization. In: Proceedings of the 2015 international conference on distributed computing and networking, pp. 4:1–4:10. ACM (2015)
17. Katewa, V., Pasqualetti, F.: On privacy vs. cooperation in multi-agent systems. Int. J. Control. 91(7), 1693–1707 (2018)
18. Rauniar, A., Muhuri, P.K.: Multi-robot coalition formation problem: Task allocation with adaptive immigrants based genetic algorithms. In: IEEE international conference on systems, pp. 000137-000142S (2017)
19. Liemhetcharat, S., Veloso, M.: Weighted synergy graphs for effective team formation with heterogeneous ad hoc agents. Artif. Intell. 208(1), 41–65 (2014)
20. Liemhetcharat, S., Veloso, M.: Allocating training instances to learning agents for team formation. Auton. Agent. Multi-Agent Syst. 31, 905–940 (2017). https://doi.org/10.1007/s10458-016-9355-3
21. Cybenko, G.: Continuous valued neural networks with two hidden layers are sufficient(Technical Report). Department of Computer Science. Tufts University, Medford (1988)
22. Backpropagation applied to handwritten zip code recognition. Neural Comput. 1(4), 541–551 (1989)
23. Stephanakis, I.M., Chochliouros, I.P.: Anomaly detection. In: Secure cloud environments using a self-organizing feature map (SOFM) model for clustering sets of r-ordered vector-structured features international conference, pp. 1–9 (2015)
24. Vesanto, J., Alhoniemi, E.: Clustering of the Self-Organizing map. IEEE Trans. Neural Netw. 11(3), 586–600 (2000)
25. Nakao, R., Abe, T., Funayama, S.: Horizontally transferred genetic elements in the tsetse fly genome: an alignment-free clustering approach using batch learning self-organising map (BLSOM). Biomed Research International (2016)
26. El Habib Soudi, M., Songhao, P.: A new decentralized approach of multiagent cooperative pursuit based on the iterated elimination of dominated strategies model. Math. Probl. Eng. 2016(3), 1–11 (2016)
27. Karishmatyagi, V.D.: A comparison of sofm with k-mean clustering and am pattern reorganization:a review. International Journal of Engineering Research and Development 7(3), 44–55 (2013)

Zhaoyi Pei received the B.S. degrees in computer science from Harbin Institute of technology,Harbin, China in 2015. He is currently a PhD student in the Computer science and Technology School at Harbin Institute of Technology, Harbin, China. His primary research interests include multi-agent systems, machine learning and robotics.

Piaosong Hao is currently a professor and doctor supervisor in school of computer science and technology in Harbin Institute of Technology. He received his PhD from Harbin Institute of Technology (HIT) in 2004. During 2006 to 2009, He worked as a postdoctoral in national key technology in robot technology and system in the Harbin Institute of Technology. His research interests mainly are robot intelligence control, pattern recognition, motion planning, and robot vision.

Mohammed El Habib Soudi received the B.S. and M.S. in computer science from Abbess Laghrour University, Khanchela, Algeria, in 2011 and 2013, respectively. He is received a PhD degrees in the Computer science and Technology School at Harbin Institute of Technology, Harbin, China. His primary research interests include coordination mechanisms, game theory, and learning in multi-agent systems.

Muhammad Zuhair Qadir received his Bachelors in Computer Engineering from University of Engineering and Technology, Lahore in year 2010. He received his Masters in Computer Sciences in the year 2013 from Lahore University of Managements Sciences (LUMS). Currently, he is working as a PhD scholar at Harbin Institute of Technology, Harbin, P.R. China. Apart from that he is a faculty member of COMSATs Institute of Technology, Lahore on study leave and has 2 years of industry experience in Systems Limited Pakistan. His research interests includes Computer Vision and Robotics.

Guo Li obtained his bachelor’s degree in computer science from Northeast Forestry University NEFU 2012. And in 2012, Li Guo received his master’s degree in computer science and technology at Harbin Institute of Technology. Now Li Guo is a PhD student in computer science at Harbin Institute of Technology concentrating on SLAM and Multi robot system.