International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

A Multi-Robots Task Allocation Algorithm Based on Relevance and Ability With Group Collaboration

Yanyan Han, Deshi Li, Jian Chen, Xiangguo Yang, Yuxi Hu, Guangmin Zhang
School of Electronic Information , Wuhan University
Wuhan, China
E-mail: hanyan4981@163.com, dsli@whu.edu.cn, cj@eis.whu.edu.cn,
yangxiangguo123@163.com, hyxwhu@163.com, zgmszxzj@yahoo.com.cn

Abstract: Multi-Robot Task Allocation is a crucial issue before performing a certain task. This paper deals with a distributed task allocation method based on some special relation defined according to the performance of history cooperation between two robots. The algorithm we propose here is named TARARC—a Task Allocation algorithm based on Robot Ability and Relevance with group Collaboration, where robot ability is weighed by reliability, relevance represents a fresh concept of "history relevance" between every two robots to establish reasonable groups for better collaboration, and the group collaboration includes inter and inner group help strategy that are adopted when different nodes failures happen in unknown environment. TARARC emphasizes the role of "agent node" in each group that is responsible for task competition, group leadership, formation maintenance as well as task execution with changing agents. Simulation on Player/Stage shows that our mechanism is feasible and valid.
Keywords: Mobile Robots Team; Task Allocation; Group Collaboration; Mobile Agent; Multi –Robots

1. Introduction

Nowadays, research and application of multi-robot system is drawing more and more attention all over the world due to its alterable network structure, universal application, convenient manual control and monitoring, accessibility of various circumstances..

Multi-robot task allocation is a crucial issue in a multi-robot system, which concerns the strategy of matching tasks and robots to get higher performance efficiency according to certain principles. Whether task assignment is proper has a great influence on the ultimate performance of the whole multi-robot system. What is more, it has a close relation with ways to establish groups and a formation, because in most circumstances tasks are performed by group collaboration rather than individual robots. With increasing complexity and scale of the task to be executed, the significance of validity and efficiency of task allocation method in a robot team becomes more distinct.

Generally speaking, Multi-robot team can be applied in many situations, such as salvaging in emergent and hazardous situations, goods transit, exploration of an unknown environment, environmental monitoring, etc. In this paper, we mainly discuss its application in exploration.

The rest of paper will be organized as following: part 2 will introduce some related works concerning MRTA(Multi-Robot Task Allocation), part 3 will demonstrate the main idea of our algorithm, part 4 is a detailed explanation of this algorithm, part 5 is concerned about the simulation result, and part 6 is the conclusion and guide for future work.

2. Related Works

At present, there are two primary kinds of solutions to MRTA problem: reactive and deliberative[1]. The advantage of the first type is fast decision-making, while the second is more efficient in collaboration. More specifically speaking, algorithms, based on swarm-type coordination, includes ant colony algorithm[2] and contract net, belonging to the first

type, while most collaborative algorithms belong to the second type, which is based on target topology[3], vacant chain[4], emotion[5], market[6] and so on. Ant colony constructs an environmentally embedded, pheromone based solution to the Task Allocation (TA) problem, but it assumes independent task completion time[4]; target topology is based on the geometric toplogy of target, defined impact and suffering factor to form a random allocation model, but it is merely applicable to certain special tasks without much universality. Vacant chain imitates the structure of society and contract net[7] to make full use of available resources and use reinforcement learning to generate vacant chain, but the requirement of vacant resources is rather high. Emotion endows robots with emotions that like humans to assign different tasks in different emotion states for better performance. This is a new concept and needs more research; Market-it utilizes the mechanism of market[8] to design a solution to incompact cooperative task, but it is merely available to tasks that can be completed by single robot without much manifest cooperation. Local eligibility[9]-task allocation is based on the local ability of a robot, and the most efficient robot directly inhibits other robots around it and performs the task.

Most of the above algorithms treat each robot as an individual with own certain task to be completed and once determined, they play almost the same role in the subsequent process. In this paper, we divide robots into two basic kinds: agent node and common node. The agent node plays critical roles while common node is an assistant to agent node. Based on this division, we propose a new concept that the formation could be restricted not to all nodes but to agent nodes only, while member nodes move around them. This may not only reduce communication traffic between nodes, but also decrease the complexity of computation, which results in less energy usage.

3. Main Idea

The algorithm we put forward is named TARARC (Task Allocation algorithm based on Robot Ability and Relevance with group Collaboration), which is used for task competing, group building, inter and inner group cooperation. The main idea of TARARC is following: an initialized relevance array R is used to group the robots in order to make the robots in a group have better collaboration ability when executing a task. Then belief degree of each robot will be calculated to choose an agent node to represent the whole group. After that all the agents will compete for the subtasks by comparing their efficiency functions, and agent with the largest value will win this task. Then certain formation will be established in which agents move to the exact position as expected, with in-group nodes staying around them. In the process of movement, some robots may fail due to energy exhaustion or unexpected collision. At this time, rescue strategy will be employed according to different conditions as Fig.1 shows. At last, when arriving at the destination, robots will encircle the target around them as a group. If tasks are finished, all the relevant arrays used in this algorithm will be updated for the next task.

The flow chart of proposed TARARC is demonstrated as Fig.1.

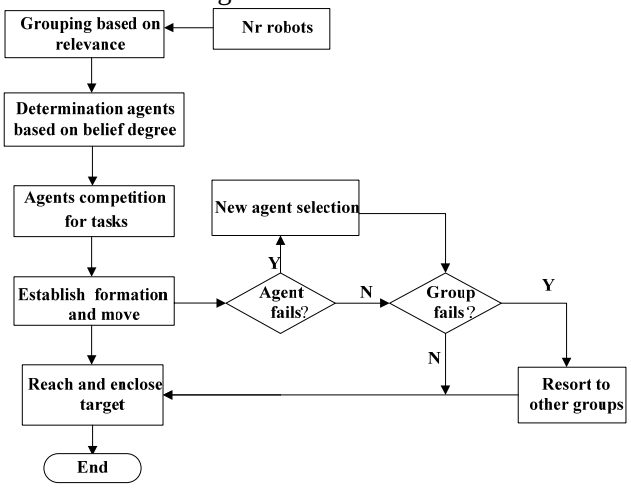

Figure 1. Flow chart of TARARC

4. Model Setup

4.1 Model Assumption

Here，we assume：

a) The number of robots is supposed to be less than 50.

b) All robots are homogeneous, that is all robots are mounted with the same sensors and their initial energy are also the same.

c) Task allocation happens before formation estab−lishment.

d) Tasks are separable and can be sorted with priority.

4.2 Definitions

Def.1 - assume r is the abbreviation of robots, the total number of robots is represented as N_r, constituting a set $\{R_i\}$;

Def.2 - after the decomposition of one task, the number of subtasks is represented as N_g;

Def.3 - assume k and j are two nodes, then the geometrical distance between them is represented as d_{kj};
Def.4 - assume node i has cooperated with j, then the incremental belief degree is represented as F_{ij};
Def.5 - assume node i competes for task j, than the function of benefit of i to j is denoted as B_{ij} , and the cost of node i to task j is denoted as C_{ij};
Def.6 - assume i and j has cooperated before, then the history relevant degree between node i and j is denoted as R_{ij};
Def.7 - assume node i and j has cooperated before, then the probability of successful collaboration between two robots is denoted as S_{ij} ;
Def.8 - the residual energy of robot j is denoted as S_j, the threshold value of belief degree is denoted as F_{thres}, and the order of node j in a group used in belief degree is denoted as j_{ord}.

4.3 Algorithm Description of TARARC

1) Task decomposion

The divided task set can be demonstrated by N subtasks as: $\{T_0, T_1, \ldots, T_{N-1}\}$ satisfying $T_{total}= T_0 \cup T_1 \cup \ldots \cup T_{N-1}$.Here it is assumed that current tasks in right order in terms of priority, and they will match different groups correspondingly. The attributes of a task T_i include: position--(x,y) ,difficulty--d_t, group number--*grnum* and the order of task in the task set demonstrating its priority.

2) Robots Grouping

If $N>N_r$, we should re-combine the tasks. Assuming the regulated number of sub-task is N', the average difficulty of tasks $dt_{aver}=\sum_{i=0}^{N'-1} dt(Ti)/N'$, and the mean variation $\delta_{dt}=\sum_{i=0}^{N'-1}[dt(Ti)-dtaver]^2/N'$. The principle of re-division is that the complexities of each task should tend to be equal—that is δ_{dt} should be minimized. After that, do as follows:

The task should be partitioned by the ratio of N_r and N, then the number of every group is n or $n+1$(If the relevance got by looking up the relevance matrix is small, allocate the residual nodes to groups with n nodes by random).

The principle of grouping is: firstly, calculate the history relevance of all robots to constitute a relevance matrix $R=[R_{ij}]$. Then robots with higher relevance may be grouped into the same group and those with lower relevance will be divided into different groups. It can be formulated like as:

if $R_{ij}>R_{ik}$,

P(nod[i].grnum=nod[j].grnum)>P(nod[i].grnum= nod[k].grnum).

Hereinto, history relevance is determined by two factors: if nodes i and j ever collaborated, C_{ij}=1, otherwise C_{ij}=0 ; the probability of successful collaboration in history S_{ij}.

The definition of the relevance function is:

$$R_{ij} = C_{ij} \cdot S_{ij}^{\beta} \qquad (1)$$

$R_{ij}=R_{ji}$, R_{ii}=0, $\beta=1/n_s$ and n_s is the number of successful collaborations.

$$S_{ij} = \frac{n_s}{n_s + n_f} \qquad (2)$$

n_f is the number of failed collaboration.

The process of grouping by relevance is as following:

Grouping begins after the relevance matrix R is obtained. Obviously, R is a symmetric matrix, whose elements in line i denote the relevance of node i and other nodes. Firstly, calculate the sum of elements in every line $\sum_{j=1}^{Nr} R_{ij}$, sort the sums in decreasing order. The largest sum $\sum_{j=1}^{Nr} R_{ij}$ denotes that node i has the largest relevance with other nodes. In line i, select the first $n-1$ nodes that make R_{ij} largest, then node i and the $n-1$ nodes make up a group. Secondly, set the relevance between nodes already in a group as 0, and continue until the residual nodes are too few to form a group. At this time, the result obtained is not complete, for instance, a group with expected $n+1$ nodes perhaps only has n nodes. Thus we should mark up groups with expected number of nodes. Then, after agents[9] have been selected, determine the ultimate groups by comparing the distance between those ungrouped nodes and every agent as illustrated in part (4) .

3) Agent Determination

Compare the belief degree (defined as (3)) of all nodes in a group and choose the one with the largest value as the agent, which represents the whole group to compete for task[10], establishes and maintains the formation in the process of moving and serves as the main executor of a task. Other nodes assist the agent to complete a task, especially when the agent fails, then they may become a new agent. Consequently, each agent $1 \ldots k$ can be obtained.

4) Match the ungrouped nodes to suitable groups

Assuming that the number of ungrouped nodes is x, the last x groups in N_g will receive one respectively. Calculating the distance between these nodes and every agent, choose the node with the smallest value in the corresponding group and make a mark. In this way, a complete group is fixed--$\{k , i , j \ldots\}$. At last, there

will be x groups with $n+1$ nodes and N_g-x groups with n nodes. It should be pointed out that all the information is obtained by mutual communication, most of which occurs between agents and member nodes. Robots form an ad-hoc network with self-organized ability and dynamic topology to unknown environment.

For example, there are 50 nodes, 20 subtasks, thus robots should be divided into 20 groups, in which there are 2 nodes, and 10 groups, with 3 nodes.

Definition of belief degree:

Belief degree denotes self-evaluation of a node. The larger it is, the more believable the node is, and the more it is likely to succeed in a task. It is an innate attribute of each node, marked as $F=\{F_{ij}\}$, which is decided by P_j, d_t, S_t , j_{ord} and formulated as :

$$F_{ij} = \frac{k_1 + k_2 \cdot p_j^{\beta} + k_3 \cdot d_t + k_4 \cdot S_t}{k_5 \cdot j_{ord} + 1} \quad (3)$$

Where $k_1 \ldots k_5$ are constant coefficients. The different influence owns by every factor can be realized by controlling the coefficients. Before the implementation of the first task, an initial value is given to every belief degree. Every time a task is completed, if the implementation is successful, F_i will increase and vice versa.

5) Sub-Tasks Allocation

The definition of the efficiency function is:

$$B_{it}=F_{it}/C_{it} \quad (4)$$

In the process of competing[11] for the tasks, for each subtask, the agent with the highest B will get the task. The belief degree is the benefit for node i completing task j. Cost function is defined as $C_{it} = l_1\sqrt{d_{it}} + l_2 d_t$, where d_t is one attribute of task T_t in range of (0,1), t denotes the order of task or priority, d_{it} is the distance of node i to task t.

Then match the partitioned groups with the sequenced task--task allocation. And the priority of tasks decided in step 1) will determine the order of matching tasks with groups.

6) Formation Establishment

It must be satisfied that the agents are at the expected position and the common nodes are distributed around the agent. Then, the whole troop begins to move towards the target. Owing to the dynamic change of the formation, maintaining formation means that the agents should be at desired position and common nodes should move within effective range around agents. It can be formulated as:

$Agent.coordinate \in [desiredcoordinate- \triangle , desiredcoordinate+\triangle]$, where $\triangle$ is the deviation of coordinate and its value is smaller than the minimum distance between two nodes.

7) Help strategy on node failure

a). The remaining energy of the agent node is not enough to continue the task. When $S_j<S_{Thres}$, agent M should declare its state to subordinate nodes and choose another node M' as the new agent to maintain the task. The node with max belief degree will become the new agent. The remaining energy can be reflected by battery voltage through inner sensors fixed on robots. At this time, M must be deleted from the robot list and replaced by M', and the node matrix must be upgraded at the same time. What's more, the new agent M' should move to the position where former agent M was to maintain the basic formation. After one execution of a task, new agent will compete for new task, and evaluate the belief degree of member nodes in a group.

The energy of a node includes three parts: *tim*--the times a node reads data denoting the communication traffic occurs on this node; *dist*--the distance between current position and initial position; *sped*--the average speed from the beginning to the end. The mathematical formula is defined:

$$S_j = k_1 \times (tim)^m + k_2 \times (dist)^n + k_3 \times (sped)^t \quad (5)$$

In order to ensure S_j to belong to [0, 1], restrict $k_1+k_2+k_3=1$, and m, n, t are all negatives, thus S_j is the decreasing function of *tim*, *dist* and *sped*.

b). If more than one node fails in a group, that is, the available nodes $n<n_{thres}$, it is time to resort to another group. The distance between two nodes D_{Aij} can be obtained by mutual communication. Choose group with smallest D_{Aij} and group j will send one of its subordinate nodes to the failed group i, and then choose the second least D_{Aij}. Repeat this process until in group i, $n \geqslant n_{thres}$.

5. Simulation

5.1 The result of task allocation

TARARC is implemented by C++ client in Player/Stage[12]. Here, 10 robots are taken as an example to justify TARARC. The initialization section includes node array, task array, relevance array, agent array, and group array. The coefficients $k_1 \ldots k_5$ in F_{ij} are set 0.2, l_1,l_2 in C_{ij} are set to be 0.05. After running, each array is updated. In Player, the initial scene is as Fig.2, and after grouping section, Fig.3 is obtained.

Figure 2. Initial scene

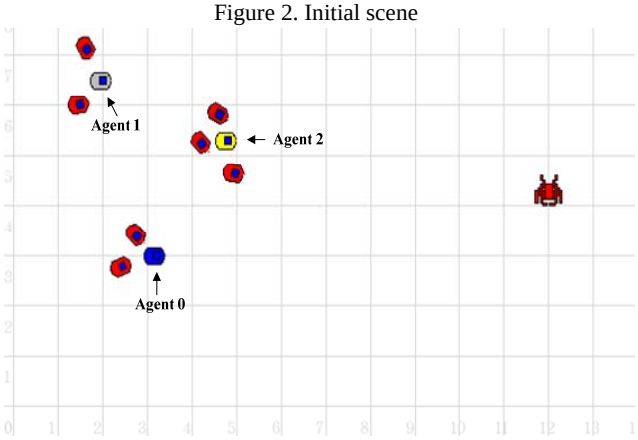

Figure 3. Establish groups

In initial state, 10 robots are distributed around in the simulation environment stage. After the algorithm of grouping, 3 groups are formed with several robots lying around the agents. The blue, grey and yellow nodes denote the agent of each group respectively, while the red ones are common nodes. By changing the values of coefficients in F_{ij} and C_{ij}, different agents can be obtained. When arriving at the destination, robots move to enclose the object "invader" as Fig.4.

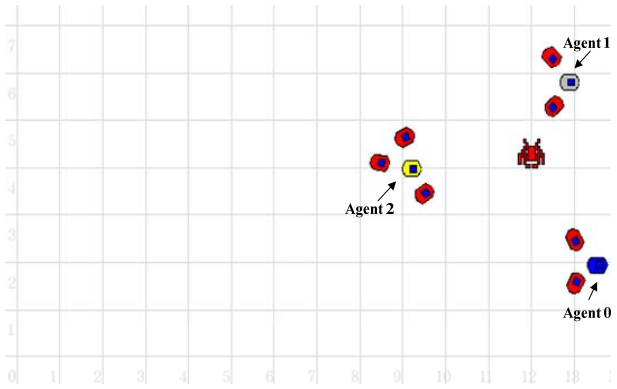

Figure 4. Enclose the target

Fig.5 shows the relation between the time from group state (as Fig.3) to enclosing the target and the density of robots. The horizontal coordinates are ratios of the area robots occupied to the area robots occupied in Fig.2, which reflects the relative density of robots. From 0.6 to 1.4 on the horizontal axis, the variations of convergence time are not obvious, but when density is larger than 1.4, time consumed increases sharply. This is because when robots get too close to each other, there is not enough space left to avoid each other, so it becomes much easier to collide with others. But when the density is smaller, robots can be free to move at optimal path with no collision.

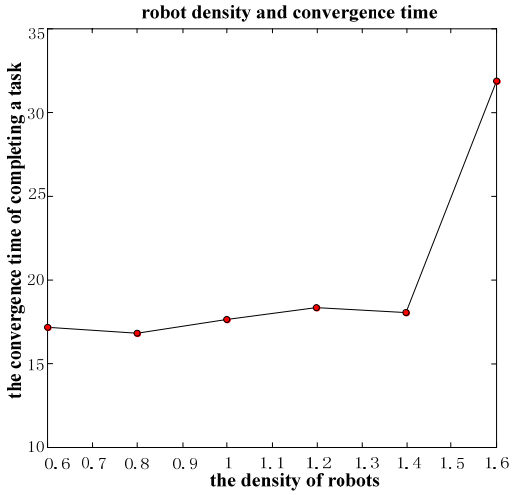

Figure 5. Influence of density of robots

Fig.6 shows the relation between the time from the initial state to forming groups and the number of robots. With the increase of robots, although the time single robot consumes arriving at the destination is almost similar, total time will increase as a linear line, movement becomes discontinuous and the occupancy ratio of CPU increases.

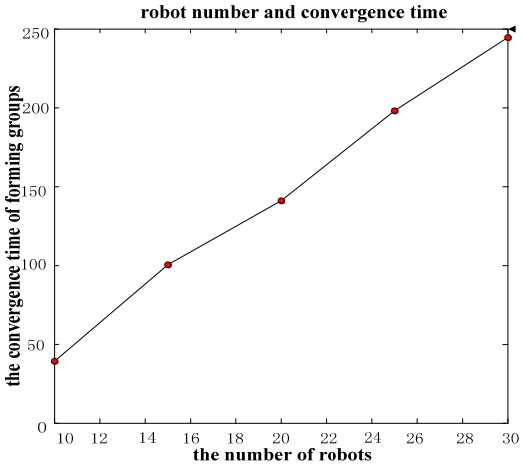

Figure 6. Influence of number of robots

Fig.7 shows the time from grouped state to enclosing the target and the speed of robots. When the speed is smaller than 1.0, running time becomes shorter with the increase of speed, and when the speed is larger than 1, running time becomes long with the increase of speed. This is because if the speed is more than a certain value as 1.6 m/s in the figure, the ability of avoiding obstacle of robots decreases because the larger of the speed is, the smaller is the sensitivity of robots' rotation, so there is not enough time for robots to change direction in order to avoid collision. As a result, it is more difficult to arrive at the destination due to mutual collision.

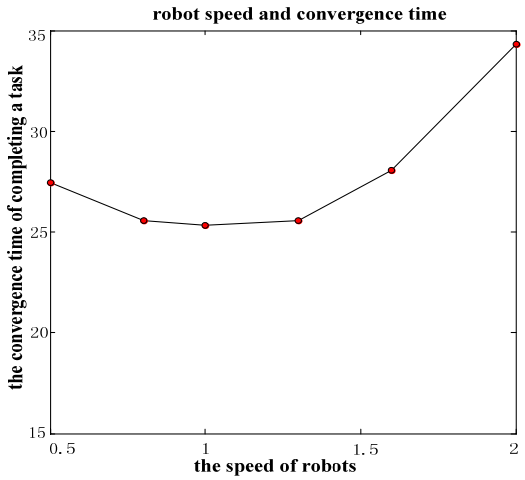

Figure 7. Influence of robot speed

Table 1 shows the influence of different priorities of 3 tasks on the convergence time from the state as Fig.3. Three tasks with different priorities have six combinations shown in Table 1. When it is the 6th combination, due to collision, robots can't complete tasks, and when it is the 3th combination, robots can enclose the target in the shortest time. This is because the priorities of tasks determining the order of allocating tasks, and influents ultimate allocation of tasks. Different tasks decide different paths of movement, so there may be possibility of collision occurred resulting in more time consumed.

Table 1. Influence of combination of priorities

Combination of 3 tasks	Convergence time
(1, 2, 3)	33. 05
(1, 3, 2)	29. 25
(2, 1, 3)	20. 15
(2, 3, 1)	33. 00
(3, 1, 2)	21. 525
(3, 2, 1)	∞

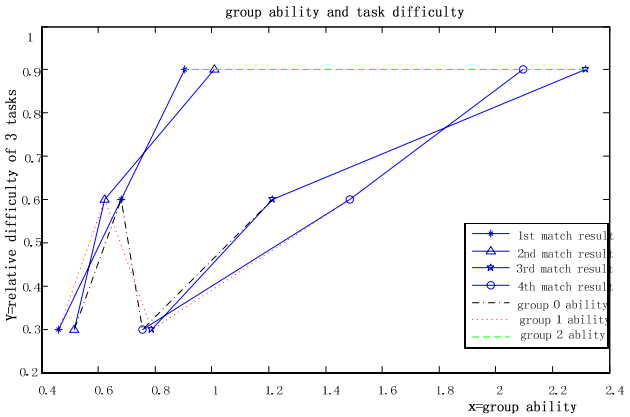

Figure 8. Relation between group ability and task difficulty

Fig.8 shows the relation between task difficulty (or priority) and the ability of a robot group which is demonstrated by the efficient function Bit. The horizontal coordinate denotes the values of efficient function of three groups, and the vertical coordinate denotes the relative difficulty of 3 tasks. The result is derived from continuous 4 task allocations which are differentiated by 4 marks in Fig.8. We can see that the ability of the same group is changing after a task has been completed, and it could increase or decrease according to the performance of one task. Clearly, group 2 is always of most ability, so it always obtains the most difficult task with difficulty coefficient-0.9. And the ability of group 1 is also keeping increasing, so group 1 sometimes obtains the second difficult task. From the curve of group 0, it is obvious that its ability decreases sharply after it completes task 3, because it implements the task very poorly, even fully fails. So at the forth completion of the task, it only obtains the easiest task.

Therefore, it can be inferred that the method we use to allocate tasks to different groups is valid and reasonable. Although relative scenes are assumed, it is still convincing that the performance of the whole team can be improved greatly with the above allocation mechanism.

From above analysis, it is obvious that reasonable groups can be easily obtained from relevance array and distance array. Using the belief degree array, agents of each group can be obtained to lead their group to build the formation, move to the destination and execute task (here refers to enclose target). However, in simulation, to avoid collision among robots, the speed is quite small, so the time is sometimes quite large.

5.2 The result of formation establishing and maintaining

Fig.9-1 shows the initial state of 3 groups after making groups in the third stage of Part 4 where they are differentiated by blue, yellow and red. At this time, the general shape is a triangle in regard to the three agents and the agents have not started moving. When the agents finish competing for subtasks, the team begins to move. Here we want to see the formation transition from a triangle to a vertical line. Therefore, the agent marked as blue on the left side first heads on with its member nodes keeping along with it. When it comes to the horizontal coordinate of the agent marked as red, the two agents begin to move side by side.

Fig.9-2 shows the state after 5 seconds, we can see that the three agents have been on the same vertical line, and this implies that the formation of the team has transmitted from a triangle to a vertical line.

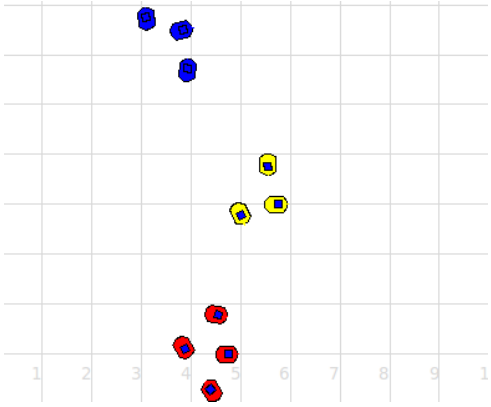

Figure 9-1.Initial scene of 3 groups

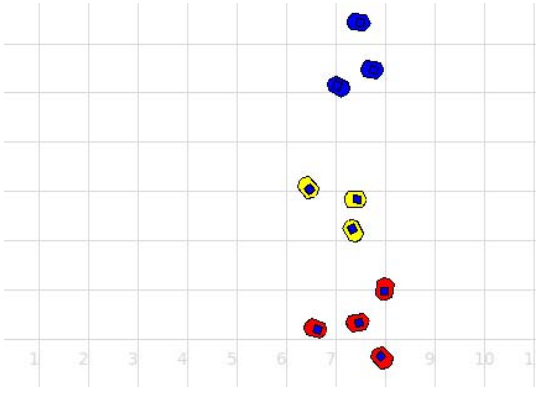

Figure 9-2. After 5 seconds

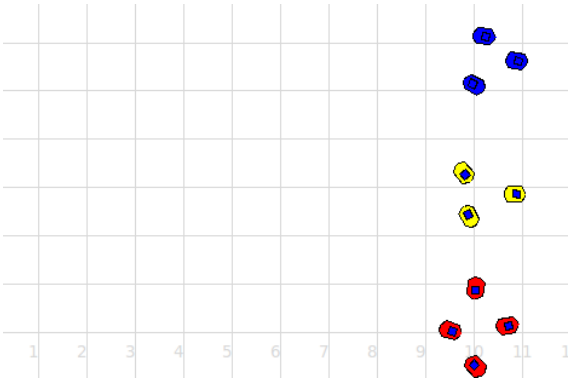

Figure 9-3.After 23 seconds

Figure 9-4. The time arriving at the destination

Fig.9-2 shows the state after 5 seconds, we can see that the three agents have been on the same vertical line, and this implies that the formation of the team has transmitted from a triangle to a vertical line.

Fig.9-3 to Fig.9-4 reflects the process of maintaining a vertical line formation. By means of mutual communication periodically, agent nodes can adjust their speed and positions to the requirement of a vertical line. From Fig.9-4, it can be deduced that the vertical line has always been maintained until robots arrive at the position of the target.

5.3 Simulation Platform

Player/Stage is a special simulation environment for multi-robots. The interface to access to those sensors is the same regardless of the underlying hardware platform [13].

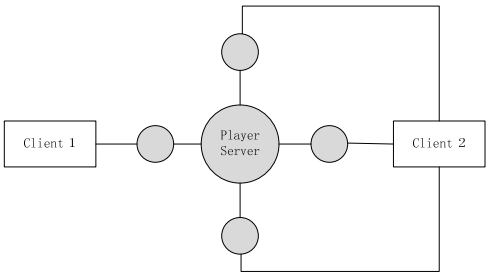

Figure 10. The working mode of Player

When Player is running on the mobile robot, it connects to the controlling software of the client by standard TCP Socket[14]. Any robot client can obtain sensor data from other robots' Player and even send control instructions, which is one of Player's advantages[15].

There are still some disadvantages of Player, one of which is that it is not suitable for large number of robots concurrently running in a PC. When the number is increased to 50, the running speed of the team will decrease sharply, and possibly lead to a computer crash.

5.4 Communication Mechanism between Robots

Traditionally, surrounding circumstances of the robots are always acquired by the sensors fixed on the robots, for example, laser sensor used for obstacle detection and avoidance, speed sensor used for speed controlling, angle sensor used for direction controlling, and so on.

However, the focus in this paper is not on the ability and application of various sensors, but on the communication efficiency between two robots or two groups. Current position and residual energy are both acquired through mutual communication periodically. Since the period of communication can be controlled manually, it reflects the accuracy and flexibility of the movement of a robot in certain formation. Besides, as communication consumes extra energy, it also influences the efficiency of energy usage. We hope to give more enthusiasm on the whole network robots constitution and communication efficiency.

6. Conclusion and future work

In this paper, we propose a new algorithm-TARARC. From the result of the simulation, it can make robots complete simple tasks within a reasonable time. We use group collaboration for inner and inter help strategy, history relevance for forming groups, robot ability defined by belief function for tasks competition and task priorities for efficient allocation.

At present, a simulation on larger scale network is in progress; in future we will improve the algorithm at the basis of increasing the number of robots to adapt to larger scale of networks in real application such as environment monitoring. Besides, more attention will be devoted to applying distributed scheduling architecture for higher calculation efficiency and network designing as well as the maintenance which is the foundation of a mobile robot team.

7. Acknowledgements

The authors would like to thank the reviewers for their detailed comments that have helped improve the quality of the paper. This research was funded in parts by China National High Technology Plan grant 2007AA01Z225 and NSF China 60972044.

References

[1] Chen Zonghai,"Mechanism and Strategy of Multi-robot Coordination for Exploring Unknown Environments", *The Research and Development of Worldwide Technology*, Vol.27, No.6, 2005.

[2] Yu Zhang, Shuhua Liu, "large-scale multi-robot task allocation based on Ant Colony Algorithm", IEEE, 2008

[3] Dandan Zhang, Long Wang, "Target Topology Based Task Assignment for Multiple Mobile Robots in Adversarial Environments", *Proc. the 46th IEEE Conference on Decision and Control,* 2007, pp.5323-5328

[4] Torbjørn S. Dahl, Maja J. Matari´c and Gaurav S. Sukhatme , "Multi-Robot Task-Allocation through Vacancy Chains", *Proc. IEEE International Conference on Robotics and Automation*, Vol.2,pp.2293- 2298, 2003.

[5] Sajal Chandra Banik, Keigo Watanabe and Kiyotaka Izumi, "Task allocation with a cooperative plan for an emotionally intelligent system of multi-robots", *Proc. of SICE Annual Conference*, pp. 1004-1010, 2007.

[6] Ashley Stroupe, Terry Huntsberger, Avi Okon, Hrand Aghazarian and Matthew Robinson, " Behavior-Based Multi-Robot Collaboration for Autonomous Construction Tasks", *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1495-1500, 2005.

[7] ZhangHaijun,ShiZhongzhi, "Dynamic Contract Net Protocal", *Computer Engineer* , Vol.30, No.21, 2004.

[8] Aaron Gage, "Multi-Robot Task allocation Using Affect", PhD thesis, University of South Florida, August, 2004.

[9] Brian P. Gerkey, Maja J. Mataric, "Murdoch: Publish/Subscribe Task Allocation for Heterogeneous Agents", *Proc. International Conference on Autonomous Agents,* 2000

[10] VIG Lovekesh, ADAMS Julie A, "Market-Based Multi-robot Coalition Formation", *Proc. IEEE transactions on robotics*, Vol. 22, pp. 637-649 , 2006.

[11] Maria Gini, Richard Voyles, "A Distributed Multi-robot Cooperation Framework for Real Time Task achievement", *Distributed Autonomous Robotic Systems*, Vol.7, pp.187-196, 2006.

[12] Brian P. Gerkey, Richard T. Vaughan, Andrew Howard, "The Player/Stage Project: Tools for Multi-Robot and Distributed Sensor Systems", *Proceedings of the International Conference on Advanced Robotics*, Coimbra, Portugal, pp. 317-323, 2003.

[13] Nils Tippenhauer, "Introduction to Player/Stage/ Gazebo", *http://pami.uwaterloo.ca/groups/asrtdd/psg.pdf*, 2005.

[14] Li Wenfeng, "Wireless Sensor Network and Mobile Robot Control", *the Science Press*, 2009.

[15] http://playerstage.sourceforge.net