Proceedings of the 2010 14th International Conference on Computer Supported Cooperative Work in Design

Design and Implementation of Multi-robot Cooperative Tracking

Li Shuqin, ZhangLe
Department of Computer Science
Information Science & Technology University
Beijing 10085, P.R. China
Email:lishuqin_de@126.com

Yuan Xiaohua
College of Information
Shanghai Ocean University
Shanghai 201306, P.R. China
Email: yuanxia8631_cn@sina.com

***Abstract*—Multi-robot cooperation is a Cutting-edge topic in multi-robot research, in which objects tracking is an important and typical problems. Based on the reinforcement learning and multi-agent theory, a decision-making algorithm and a cooperative formation control algorithm for multi-robot tracking are proposed, and more than one cooperation forms for chaser robot are presented. Finally the corresponding simulation experiments are conducted to show the validation of the proposed algorithms.**

Keywords- Multi-robot tracking; reinforcement learning; decision-making algorithm; formation control algorithm

I. INTRODUCTION

With the development of robot technology, there is a growing emphasis on the research of multi-robot system. One reach focus of nowadays multi-robot system is how to build a flexible and efficient team in which the robot can be in harmony with each other and act as a whole. And the research of multi-robot cooperative tacking is an ideal test platform for this kind of team.

Currently there are many a literature in this area, such as in literature [1,2,3], robots take formation vector to control a team to encircle and capture a single mobile. In [4,5] the robots was divided into two parts, the pursuit and containment , so as to get the same aim. This paper is mainly around the question that how to make chasers to adapt to a given environment by learning and cooperate to form an effective formation to catch up with the escaper successfully.

II. TRACKING TASK DESCRIPTION

In robot tracking, there are two kinds of roles, the chaser and the escaper. There is one escapee who is fast, and there are three chasers, whose speed is only the half of the escapee. So that only by some effective organization and well cooperation, and to form a besiegement or drive the escapee into one corner, can the three chasers capture the escaping robot.

To make the whole study more challenging, the robot designed in this paper owning the capacity of perceiving the world approaching the people. Each robot is set with a tag, a position, field of view, moving speed and other information, the chaser even has communication capabilities. In our system, the robot is expressed with a circle, the number in circle shows the role of the robot, -1 indicates escapee, and 1, 2, 3 represent the three chaser respectively. Each robot has its own coordinate system, this is a polar coordinates, taking the direction that robot's head point to as zero angle, and the counterclockwise as positive direction, along which the angle degree will increase. The radian on the circle cut by two line educed from the center is the robot's visual range, in which there are two important view fields, the far field and the near field. The far-field is at a distance of units from the robot center, and is in a fan-shaped area between the two directions of left and right 45 degree refer to robot's head. Whichever robot comes into this area, the robot will "see" it, and obtain two kinds of information around it. The near field is the circle with a 50 unit radius and taking robot as the center. In this area, other robot around can obtain the relative information around the angle and distance of the robot's.

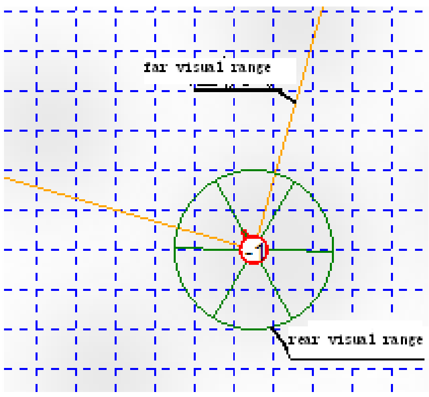


Figure 1. Description of robot and his visual range

Relative angle refers to the degree angle between the line that connecting two robot center and the line extended positively from robot head. The relative distance refers to the distance between the two robots, namely, the length of line connecting the two robots' center. In addition, through communicating, each Chaser can inform other two peers of two messages, one is its own place in global coordinate system, which including its own X, Y coordinate , the absolute orientation of the head, as well as its own absolute speed. The other is around escapee's relative distance and angle it perceived. The above information will be passed through Message Bus built in the robot environment by our method proposed in this paper. After the system having transferred the messages to the robot, the robot will trigger the corresponding event ON_MESSAGE to receive the messages.

Sponsored by KM200910772011and PHR20100713

Escapee will run away with some simple strategy according to the obtained environmental information. It will first detect the locations of the chaser robot, and judge which chaser is the nearest, then move in the opposite direction of that chaser. If no chaser having being detected, then it will move randomly.

III. Chaser Strategies

The proposed chasing strategies are not some sort of given tracking, but obtained by a Q-learning algorithm, by which the chasers train their cooperation in more than one tracking rounds for different escapee, and finally fulfill their tasks. In Q-algorithm, the proposed system only prescribe the action set, the status set and the reward, the last is for driving the robot to fulfill the tracking. According to the characteristics of Q-learning, robot abiding by Q-learning will always select the action strategy that will bring much more rewards, thus whether the whole tracking strategy is good or not has huge influence on the rewards and punishment definition.

A. Robot's action set

In order to fulfill some task, robot must complete some actions within a certain time, and combine and perform some actions together so as to get the aim. In this paper, there are many simple actions(from now on we call it Simple Action Set), which includes moving forward, stop moving, turning head to the left, turning head to the right, backward, moving to the left behind, moving to the right behind, and etc. They fulfill during one period. And in order to fulfill some complicated tracking, we also define an Advanced Action Set. The Advanced action is an action serial selected from the Simple Action Set, and they fulfill during the N decision-making periods. In Advanced Action Set there includes two actions, hunting and searching.

1) Hunting

Hunting action aims at making the chaser move directly alone escapee moving direction and capture the escapee in some special circumstances. In completing hunting, there need five decision-making cycles, during which, either moving forward or backward, the chaser will move along with the escapee's direction acquired currently. In the hunting action, the chaser will adjust its posture according to the location of the escapee.

2) Searching

Search action is carried out when the chase robot can not apperceive any information of the escapee. It is also need five decision-making cycles to fulfill. During the five cycles the robot will adjust its head to find the escapee. If finding the escapee or receiving the information reported by other robots in cycles, the chase robot will terminate the search action, that is, the search action will be completed in a variable-length cycle.

B. Robots' state set and its discretization

The robot's operating environment is simulated according to the description of the researched problem, so there are many state and different state will has different effect on the robot.

The data can be apperceived by the chaser are as following, robot's x, y coordinate, robot's orientation angle (a 2-dimensional vector), the expected speed and head towards of the nearest companion, companion's distance, companion's relative angle (2-dimensional vector), the distance of the escapee (2-dimensional vector), the escapee's relative angle, the distance from the wall, its own head towards, whether to find escapee, etc. we combined them into a numerical state set. Most of these data are continuous quantity. Because the reinforcement learning is suited to solve in the discrete space, the system must discrete all the input dates, so as to fit the application of reinforcement learning algorithms. How to select and how to discrete the state is very important for multi-robot reinforcement learning.

C. Chaser's strategy

Chaser's strategy is that the next action is determined by the rewards or punishments obtained by the previous step. After the above-mentioned state and action discrete, the learning problem of the robot moves is transformed into a discretized reinforcement learning problems, to solve which the Q-Learning algorithm can be used, in which the Q value is taken as the evaluation on the pair of state-action, and a Q (s , a) reinforcement learning can be conducted. Before the learning beginning, their also need to define a set of reward rule. In this paper the reward rule is defined through experiment and is shown in table 1.

TABLE 1. Act and the corresponding reward value

action	**reward and punishment Value**
Single hit the wall	-0.3
Repeatedly hit the wall	-0.2
Solved from hit the wall state	+0.1
Single hit teammate	-0.3
Repeatedly hit teammate	-0.2
Solved from the state of hit teammates	+0.1
Lost target	-0.2
Find the target	+0.2
Close to the target	Maximum distance -(target distance last time-target distance this time) * 0.01;
Solely seize the object	+3.0
Team victory	0.05* target distance
Formation determine	discussed in formation control

IV. Collaboration Algorithm

A. Teammate Behavior Forecast

Multi-robot tracking is actually a mutually cooperation. Each chaser gets other teammate's position through communication and apperception, find the nearest teammate according to the relative distance, and then according to the absolute head direction and moving speed of teammates to predict the behavior of the related teammate by applying a Kalman filter. The two behaviors can help robot to select one behavior conjugate the teammate.

B. Formation control algorithm

Team cooperation is the key to study the multi-robot goal tracking, and the core question of team cooperation is the determination and the formation of formation. In this paper, the method of artificial potential field is applied to decide the formation of the chasers, determine immediate rewards, estimate the selected action, and applying the rewards and punishment coefficient obtained in the formation control to restrict the formation. Suppose that all the attraction from the escapee to the chaser be the same, that is 1 unit, then the joint force on the escaped is

$$F(t)=\sqrt{\left(\sum_{i=1}^{n}\cos\theta_i\right)^2+\left(\sum_{i=1}^{n}\sin\theta_i\right)^2}\geq 0$$

Where s_t is the angle between the line connected the escapee and the ith chaser and the horizontal axis of the coordinate system. Since the pursuit goal is to make all chaser distributed uniformly around the escapee, so on an ideal condition when $\theta_i=\frac{2\pi}{n}$, the joint force on the escaped F(t) is zero. The larger F(t), the chaser distibuted more non-uniform. The joint forces difference between two neighbor time is $\Delta F(t)=F(t)-F(t-1)$, where, $\Delta F(t)$ expresses the distribution tendency of chaser. When $\Delta F(t)< 0$ the chasers distribute evenly, and should be reward by an immediate repayment $r > 0$. $\Delta F(t) > 0$ indicated that the chaser distribute non-uniform, and should be punished by an immediate repayment $r < 0$. Here the evaluate function is $r=2\left[\frac{1}{1+e^{\Delta F(t)}}-\frac{1}{2}\right]$, and the function value serve as the immediate repayment for the robot movement.

C. Formation control in moving target tracking

As shown in figure 2, in each tracking round, the chaser will start to obtain the environment information, according to which the chaser will update it's own state, and select the action related to its state and perform the action. In the new decision-making period, the rewards and punishment obtained in the previous period will be passed to the Q-algorithm in order to update the Q-table. Below are the realization steps.

Step 1: Initialize Q by a random value, robot collect the state information s_t, and convert the information into discrete form;

Step 2: select one action a_t to perform according to probability formula;

Step 3: Obtain the rewards and punishment $R_t(s_t,a_t)$ from the action result;

Step 4: get new state from environment information, and update the value of Q iteratively by using the formula

$$Q_{t+1}(s_t,a_t)=(1-a)Q_t(s_t,a_t)+[R_t(s_t,a_t)+\max Q(s_{t+1},a_i)];$$

Step 5: add the rewards and punishment and the optimized Q value into the new state;

Step 6: go back to Step 2.

After some rounds, the chaser will obtain its final rewards and punishment.

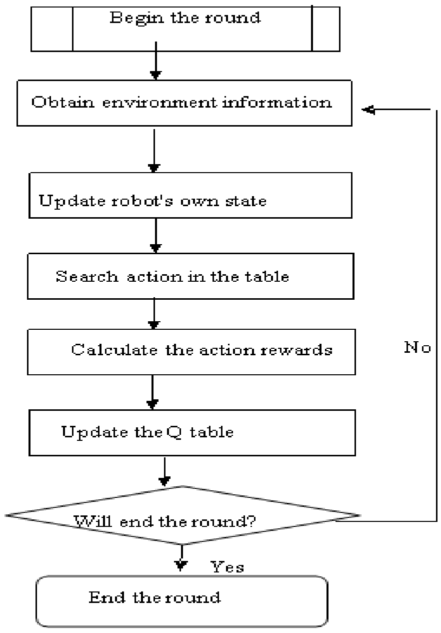


Figure 2. Flow chart for the tracking algorihm

V. SIMULATION AND RESULTS ANALYSIS

The simulation system is implemented on a PC by language VC ++6.0. In each round, the start position of the three pursuit chaser and the escapee was randomly set by system. Each round contains N Decision-making cycles, in each cycle the system will give each robot a chance to update their state, such as communicating or completion some action. Before the end of the N decision-making cycles, if any chaser captures the escapee successfully, then the chasers success, otherwise, the chasers fail, and a new round begin.

Taking account of algorithm extendibility, the robot is implemented by the dynamic-link library (DLL) procedures for reuse. Firstly, the chaser class inherits directly from the robot class, and implements the standard methods in robot class and some corresponding expansion. In this mode, State class describes robot's various states, and in this class a N-dimensional table of states is implemented; The Action class describes the robots' actions, the Hunter Agent class is the core class of the Chaser Robot, it contains a specified algorithm of the evaluating function, action implementation, messages communication, event response function; The RunnerAIBase class is the base class of the escape algorithm, in which there encapsulated some standard interfaces. BackAI, RandomAI and Human Control are three derived class which has realized all the functions of RunnerAIBase. The BackAI class has realized the strategy to escape in the opposite direction, RandomAI class has realized a randomly moving algorithm, and HumanControl class has realized manual control function for the escapee. RunnerAgent class is the core class for the escapee, which includes an algorithm container and the realization for the event function. This article also designed some communication interfaces, using which the robot can issue the appropriate information to the system, then the system will pass the information to the corresponding robot and trigger the event handle function of robot.

In the system the Q table is a 2-dimensional array. Each robot has its own Q table document, and the document record the Q value under different conditions. In each running, the system will automatically load these documents and will update the Q table. This system updates the Q value of by a discount rate $\gamma = 0.9$, $\gamma (0 \le \gamma < 1)$ and a learning rate ∂ =0.1.

Designation of the system interface is shown in Figure 3. The main interface is divided into three regions.

Region1 is at the top of the main interface. It is the system control area, in which many system functions can be called, these functions are Run (Run), stop (Stop), pause (Pause), Reply to run (Pause again, press), save the learning outcomes (Save) and select those who escape different decision-making algorithm (drop-down box).

Region 2 is at the lower right part of the main interface. It is the system data area, and is further divided into the environment (Environment) area, pursuers (Hunter) area, trying to escape (Runner) and network (Network) area. Environmental area displays the current rounds (Round), the capture number (Cached), the system speed adjustment (Sleep Time), the capture rate (Rate), the rounds used in the previous capture (Used Steps). Hunt area and escapee areas display the current coordinate different robot.

Regional 3 is at the left the bottom of the main interface. Two-dimensional images can be displayed here. It is used to show the 2-dimensional simulation of the current system, and here display all robot's physical location, current view scope, current terrain, etc.

Below are the two comparative experiments.

A. *Comparison between using and not using Advanced Action Set*

When not using Advanced Action Set, the chaser will be more slowly, and has no purposes entirely. After a train of 100 rounds, the state has no change, and the success rate is only 20%. Even after 1000 rounds, the success rate is still only 47.5%, but at this time, the chaser's action includes more aims, and chaser will not move randomly again.

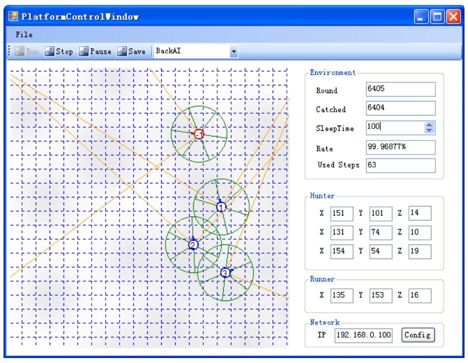
But if adding the Advanced Action Set, robot's action will improve distinctly, and it's action will have much more purpose also. Even after the first one hundred rounds, the success rate arrive at 90.3%, and after a 10 thousand of rounds, the chasing purpose become more clearly, and the chaser can successfully force the escapee into some dead ends. The comparison between the result of using and not using Advanced Action Set are list in the below table 2.

TABLE 2. CAPTURE RATE OF USING AND NOT USING ADVANCED ACTION SET

round	**Capture rate of using Advanced Action Set**	**Capture rate not using Advanced Action Set**
100	90.3%	20.1%
500	99.1%	29.4%
1000	99.4%	35.2%
10000	100%	47.5%

B. *Comparison between using and not using formation strategy*

When not using formation strategy, at beginning, because of lacking of any knowledge around the pursuit, the chaser will not form any stable action status. According to our observation, after the first one hundred rounds, the chaser does not appear any clear action tendency. After 3 thousand rounds, the tendency appears clearly but goes against the whole pursuit. This situation is showed in figure 3, in which after the escapee the three chasers get in a line, and when the escapee run along a circle, then the chasers will run after along a circle too, thus lead to a fail.



Round 6405
Catched 6404
SleepTime 100
Rate 99.96877%
Used Steps 63

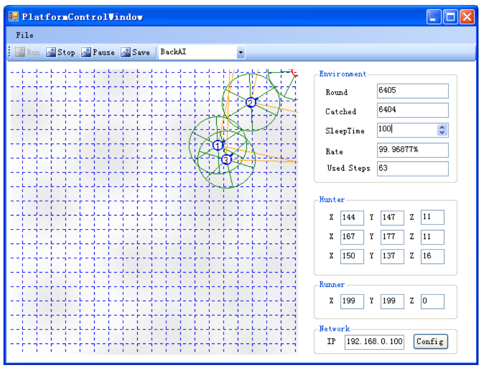
X 151 Y 101 Z 14
X 131 Y 74 Z 10
X 154 Y 54 Z 19

X 135 Y 153 Z 16

IP 192.168.0.100

Figure 3. Pursuit result without formation strategy

When with formation strategy, even after the first 50 to 2 hundred rounds, there is not any clear action tendency, but after 3 thousand rounds, the tendency becomes more clearly, and which can avail the pursuit. Just as shown in figure 4, after having driven the escapee into a corner, the chasers do not use a straightaway pursuit again, but using a fan-shaped pursuit. There is no longer any circular pursuit, and even if the escapee run along a circle, one of the three chaser will first change it's moving direction. So pursuit with formation strategy can increase the success rate, and if combine formation strategy and high-level action together, the success rate can arrive at 99.7%.



Environment

Round 6405

Catched 6404

SleepTime 100

Rate 99.96877%

Used Steps 63

Hunter

X 144 Y 147 Z 11

X 167 Y 177 Z 11

X 150 Y 137 Z 16

Runner

X 199 Y 199 Z 0

Network

IP 192.168.0.100

Figure 4. Pursuit result with formation strategy

From the two comparisons we can obtain the conclusion that it is imperative to add high-level action set into and use formation strategy in a pursuit. From the result we know that adding hiAdvanced Actions the same as to add robot's knowledge directly, this make the robot has some prior Advanced Action go on to learn some better pursuit strategy. The above experiments have proved that in the multi-agent cooperation of pursuit, the formation have a great influence. The proposed formation checking algorithm can help to avoid any worthless queuing pursuit. The experiments have validated the validation of the proposed algorithm.

ACKNOWLEDGMENT

This research is sponsored by science and technology development item of Beijing Municipal Education Committee (KM200910772011), and funding project for academic human resources development in institutions of higher learning under the jurisdiction of Beijing Municipality (PHR201007131).

REFERENCES

[1] Yamaguchi H. A distributed motion coordination strategy for multiple nonholonomic mobile robots in cooperative hunting operations. Robotics and Autonomous Systems 43(2003) 257-282.

[2] Yamaguchi H. A cooperative hunting behavior by mobile-robot troops . The Internattonal Junaral of Robotics and Research 1999, 20(9) : 93l-940.

[3] Han X., Hong B., Meng W. Distributed control for generating arbitrary formation of multiple robots. Robot, 2003,25(l):66-72.

[4] Song M., Gu G., Zhang R. Distributed control system for the cooperative task of multi-mobile robots. Robot, 2003, 25(5): 456-460.

[5] CAO Zhi- Qiang ZHANG Bin WANG Shuo TAN Min, Cooperative Hunting of Multiple Mobile Robots in an Unknown Environment, ACTA AUTOMATICA SINICA,July 2003,29(4):536-543.