SpringerLink Submit

Journal of Central South University

Science & Technology of Mining and Metallurgy

ISSN: 2095-2899 (Print) 2227-5223 (Online)

This journal was previously published under other titles (view Journal History)

Description

The Journal mainly publishes original academic papers which represent the latest research achievements in such fields as materials science and engineering, metallurgical science and engineering, mineral processing, geology and mining, chemical engineering, and mechanical, electronic and information engineering.

Browse Volumes & Issues

Impact Factor	Available
0.601	1994 - 2017
Volumes	Issues
24	142
Articles	
4,903	

Latest Articles

OriginalPaper

Statistical design and kinetic and thermodynamic studies of Ni(II) adsorption on bentonite

Bahareh Sadeghalvad, Amir Reza Azadmehr… (July 2017)

OriginalPaper

System structural analysis of communication networks based on DEMATEL-ISM and entropy

Kai Fu 付 , Jing-bo Xia 夏靖波, Xiao-yan Zhang 燕… (July 2017)

OriginalPaper

Constitutive model of disturbed soil-structure interface within mining subsidence areas

Hong Chang 虹, Jun-wu Xia 夏 武 (July 2017)

» See all articles

Stay up to Date

Article abstracts by RSS

Register for journal updates

Find a Volume or Issue

Find

Share

Journal Title
Journal of Central South University

Coverage
Volume 1 / 1994 - Volume 24 / 2017

Print ISSN
2095-2899

Topics
» Engineering, general
» Metallic Materials

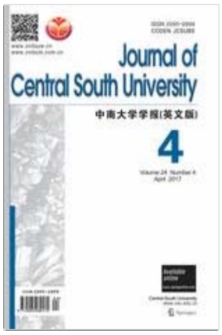
Industry Sectors
» Materials & Steel
» Automotive
» Chemical Manufacturing
» Electronics
» IT & Software

SpringerLink Submit

Journal of Central South University
All Volumes & Issues

Volume 24, Issue 4, April 2017

ISSN: 2095-2899 (Print) 2227-5223 (Online)



In this issue (27 articles)

Page 2 of 2

Page 2 of 2

Over 10 million scientific documents at your fingertips

Browse by Discipline

JOURNAL SEARCH

Search Terms: 2095-2899

Total journals found: 1

THE FOLLOWING TITLE(S) MATCHED YOUR REQUEST:

Journals 1-1 (of 1)

FORMAT FOR PRINT

JOURNAL OF CENTRAL SOUTH UNIVERSITY

Monthly ISSN: 2095-2899

JOURNAL OF CENTRAL SOUTH UNIV TECHNOLOGY, EDITORIAL OFFICE, CHANGSHA, HUNAN, PEOPLES R CHINA, 410083

Coverage

Science Citation Index Expanded

Journals 1-1 (of 1)

FORMAT FOR PRINT

Search Terms:

Search type: Title Word

Database: Master Journal List

SEARCH

J. Cent. South Univ. (2017) 24: 1002−1012
DOI: 10.1007/s11771-017-3502-0

Univector field method-based multi-agent navigation for pursuit problem in obstacle environments

Le Pham Tuyen[1], Hoang Huu Viet[2], Sang Hyeok An[3], Seung Gwan Lee[3], Dong-Han Kim[4], Tae Choong Chung[1]

1. Department of Computer Science and Engineering, Kyung Hee University, Yongin, Gyeonggi, 446-701, Korea;
2. Department of Information Technology, Vinh University, 182-Le Duan, Vinh City, Nghe An, 461056, Vietnam;
3. Humanitas College, Kyung Hee University, Yongin, Gyeonggi, 446-701, Korea;
4. Department of Electronics and Radio Engineering, Kyung Hee University, Yongin, Gyeonggi, 446-701, Korea

Abstract: The pursuit problem is a well-known problem in computer science. In this problem, a group of predator agents attempt to capture a prey agent in an environment with various obstacle types, partial observation, and an infinite grid-world. Predator agents are applied algorithms that use the univector field method to reach the prey agent, strategies for avoiding obstacles and strategies for cooperation between predator agents. Obstacle avoidance strategies are generalized and presented through strategies called hitting and following boundary (HFB); trapped and following shortest path (TFSP); and predicted and following shortest path (PFSP). In terms of cooperation, cooperation strategies are employed to more quickly reach and capture the prey agent. Experimental results are shown to illustrate the efficiency of the method in the pursuit problem.

Key words: pursuit problem; predator agent; prey agent; univector field method; multi-agent systems

1 Introduction

The pursuit problem and its variations (cops and robbers, lion and man, hunters and a rabbit) are one of the most challenging problems in robotics and computer games. The first proposal [1] of this problem modeled a group of four predator agents trying to capture a moving prey agent by surrounding four directions of that prey agent on a finite grid-world. Agent movements are limited to either one horizontal or vertical step in every discrete time interval. This problem has quickly attracted attention from the research community due to its impact on a wide range of applications in robotics [2−7] and computer game [8−11]. Most of the studies employ approaches of path planning with a moving target and cooperation between multiple agents.

In path planning, each predator agent must follow a path, determined from a particular algorithm, to reach the prey agent. Many algorithms are used for this purpose and each algorithm has a certain efficiency in particular contexts. For example, offline search algorithms [12] can generate an optimal path to reach a prey agent. However, a large amount of knowledge about the environment must be known which causes a large problem in data representation, especially if the environment is infinite. In addition, such algorithms are also extremely inefficient because any slight change in position of the prey agent requires re-analysis of the path. Incremental search algorithms (D* [25], D* Lite [26]) are proposed for the purpose of reducing the search space and decreasing search time. However, such algorithms are not sufficient for the dynamic environment of the pursuit problem. UNDEGER and POLAT [13] used an algorithm called MAPS to solve the path planning problem with a moving target. In their work, the pursuit problem is modeled as a group of predator agents trying to capture a moving prey agent in a finite grid-world. The main component of MAPS is an algorithm called real-time moving target evaluation search (MTES) [14], which repeats until reaching the target or determining the target is unreachable. MTES uses a heuristic, real-time target evaluation (RTTE-h), that analyzes obstacles and proposes a moving direction to avoid obstacles and reach the prey agent through a shortest path. A disadvantage of this approach is that it can only be used in a limited environment (size of 150×150). In addition, the capture condition in this model was defined as one of the predator agents having the same position as the prey agent. This capture condition is easier than the capture condition in which all predator agents must collectively surround the prey agent. In Ref. [15], VIET et al used the univector field method [16] to reach the moving target. Each predator agent in a group of eight predator agents

Received date: 2015−12−24; **Accepted date:** 2016−04−11
Corresponding author: Tae Choong Chung, Professor, PhD; Tel: +82−31−201−2569; E-mail: tcchung@khu.ac.kr

will move to the next position guided by a univector field and will reach one of the eight neighbor cells around the prey agent. The prey agent is captured when it cannot move in any directions due to blockage by eight predator agents. The univector field method is fast and efficient in path planning, especially in robotics because predator agents only need to know the position of the prey agent and the direction toward the prey agent. However, this method is only successful in an obstacle-free environment. In an obstacle environment, some modification or other techniques are needed to overcome obstacles. Previous studies [17, 18] have used a modified univector field to guide a mobile robot movement and avoid obstacles. Basically, the modified univector field combines two components: "move-to-goal univector field" and "avoid-obstacle univector field". These two components are combined at a particular ratio decided by an evolutionary algorithm. However, this approach only applies to an environment with simple obstacles such as spheres, cylinders, cubes. It is not easily applied to an environment of non-convex obstacles where agents can be trapped.

All papers mentioned above deal with path planning in the pursuit problem. In the aspect of cooperation between the agents, the pursuit problem is a well-known class of test problems for the study of cooperative behavior in distributed artificial intelligence (DAI) systems [19−21]. The MAPS algorithm in Ref. [13] employs two strategies: blocking escape directions (BES) and using alternative proposals (UAL), which demonstrate coordination between agents. BES determines the blocking location, which is the estimated point at which the predator agents may possibly capture the prey agent. Therefore, the path planner will determine a path for reaching the blocking location instead of the current prey location. UAL selects the best estimated direction from the alternative movement directions proposed by the path planner. The model in Ref. [15] uses communication in the cooperation among agents to share information about the prey and themselves based on either a local goal or a common goal to choose a suitable surrounding direction.

Based on approaches mentioned above, in this work, we introduce hybrid algorithms that combine the univector field method with strategies to solve the pursuit problem in infinite environments (obstacle-free or obstacle). Modeled obstacles have a variety of shape types (convex or non-convex). In our approaches, a group of eight predator agents try to capture a moving prey agent. Each predator agent follows a univector field to reach the prey agent. These predator agents work together and using one of the cooperation strategies to surround the prey agent. In addition, each predator agent will encounter the obstacles on the path to the prey agent, demonstrating the need for algorithms used to overcome the obstacles and continue toward the prey agent. All strategies in this work are considered in a partially observable environment, so each predator agent is aware of only the obstacles in a limited view. We will also define strategies of the prey agent with increasing intelligence, which will be a factor in the evaluation of the approaches of predator agents. The experimental results show the effects of every tested strategy.

2 Univector field method

The univector field method is a navigation method designed for mobile robots. In the univector field method, the magnitude of each vector is ignored, and only information about the direction is considered. The advantage of univector field method is to help the agent (or robot) reach the goal at a desired posture. A univector field $\boldsymbol{F}(\boldsymbol{s})$ at position $\boldsymbol{s}(x_s, y_s)$ is defined as in Eq. (1), where n is a positive constant, $g(x_g, y_g)$ is the desired goal position, $r(x_r, y_r)$ is a guiding point, and the symbol $\angle$ denotes the angle of a vector mapped onto the range (−π,π].

$$\boldsymbol{F}(\boldsymbol{s}) = \angle \boldsymbol{sg} - n\alpha \tag{1}$$

where

$$\alpha = \angle \boldsymbol{sr} - \angle \boldsymbol{sg}; \tag{2}$$

$$\angle \boldsymbol{sr} = \arctan\frac{y_r - y_s}{x_r - x_s};\ \text{and}\ \angle \boldsymbol{sg} = \arctan\frac{y_g - y_s}{x_g - x_s}.$$

All univector fields constitute a univector field space of the environment. Figure 1 depicts the univector field space with n=5, where each tiny circle represents a position, and the straight line attached to it represents the moving direction of the agent. The value of the univector field $\boldsymbol{F}(\boldsymbol{s})$ at position $\boldsymbol{s}(x_s, y_s)$ is equivalent to the desired

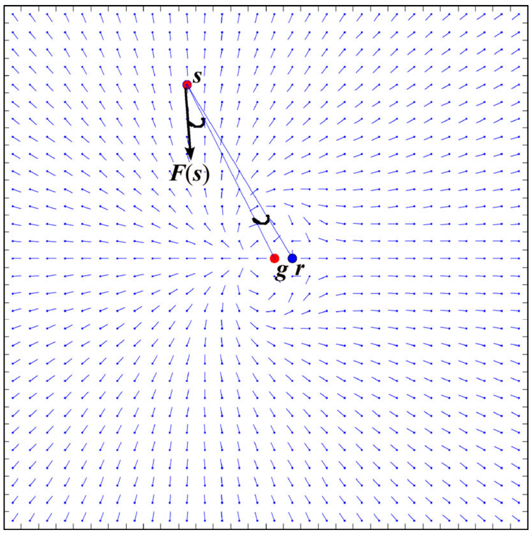
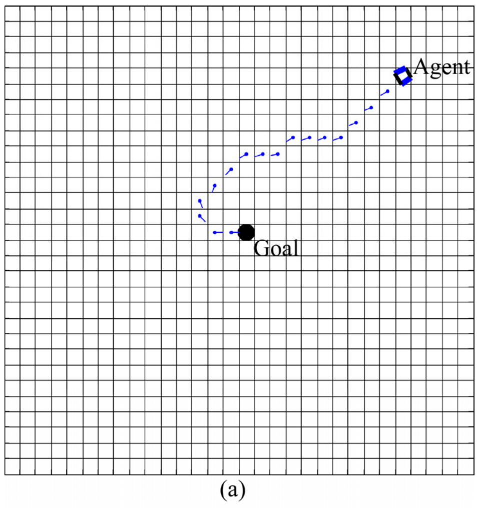


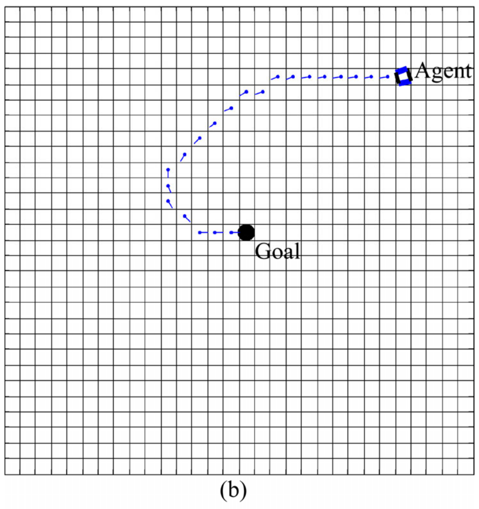
Fig. 1 Univector field space of environment

heading angle of the agent at that position.

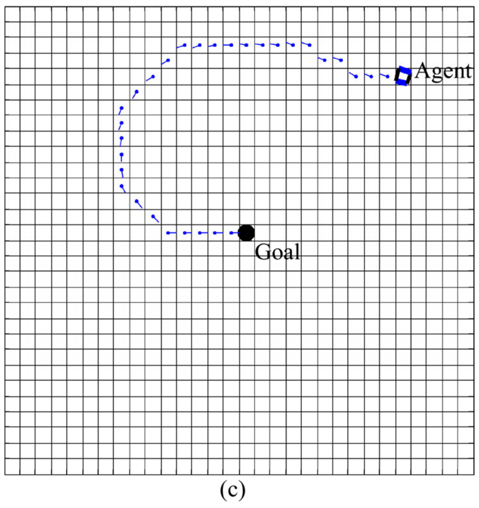
It is clear that the univector field $\boldsymbol{F}(\boldsymbol{s})$ of the agent at position $\boldsymbol{s}(x_s, y_s)$ depends on the parameter n. The larger is n, the smaller is the $\boldsymbol{F}(\boldsymbol{s})$ at the same position. Thus, if n increases, the univector field will spread out over a larger area, increasing the length of the path to be traversed by the agent in reaching its goal. Figure 2 shows an example showing the effect of parameter n on the trajectory of an agent which follows the univector field to reach the goal. According to Ref. [15], predator agents, which get stuck in a local minima region, raise a low unsuccessful capture rate. This phenomenon occurs when an agent has the same position as a guiding point where $\angle \boldsymbol{sr}$ cannot be measured using Eq. (2). To overcome this issue, $\angle \boldsymbol{sg}$ is added as a condition in Eq. (3) to make sure that a valid $\angle \boldsymbol{sr}$ is generated at every position in the space.



(a)



(b)



(c)

Fig. 2 Trajectory of an agent with different values of parameter n: (a) n=5; (b) n=10; (c) n=20

$$\angle \boldsymbol{sr} = \begin{cases} \arctan\dfrac{y_r - y_s}{x_r - x_s} & \text{if } \boldsymbol{s} \neq \boldsymbol{r} \\ \angle \boldsymbol{sr} + \dfrac{\pi}{2n} & \text{if } \boldsymbol{s} = \boldsymbol{r} \end{cases} \tag{3}$$

In our approaches, the univector field method is the main component and is used to guide predator agents to reach the prey agent in a specified direction. Strategies to avoid obstacles are also based on the univector field to determine if an agent has escaped an obstacle or not. For our experiment, the default parameters are n=5 which are also default values in Ref. [15].

3 Modeling pursuit problem

In this section, we show how the pursuit problem is modeled through the representation of the environment, obstacles, agents, and moving rules.

3.1 Environmental representation

The grid-world is used to represent the environment in this work. It is defined as a grid of square cells overlaid in the environment, where each cell is considered either traversable if no part of an obstacle overlaps the cell or non-traversable otherwise. This representation is the most common in the computer game and robotics fields because the environment can be implemented easily using a sequence of cells, and many path planning techniques can work on it quickly. A disadvantage of this representation is that, if obstacles that are not aligned with the axis, the precision of the modeled environment depends on the resolution of grid cells. Figure 3(a) depicts an example of this problem in grid-world representation. Imprecision in the grid-world representation can lead to a sub-optimal path or no path. A solution to this issue is to increase the resolution of the grid by decreasing the square size, as shown in Fig. 3(b).

In this work, we ignore imprecision in representation and assume that the grid-world is generated with a suitable resolution so that all paths determined by a path planning technique are optimal.

3.2 Movement rules

At each discrete time interval, agents choose

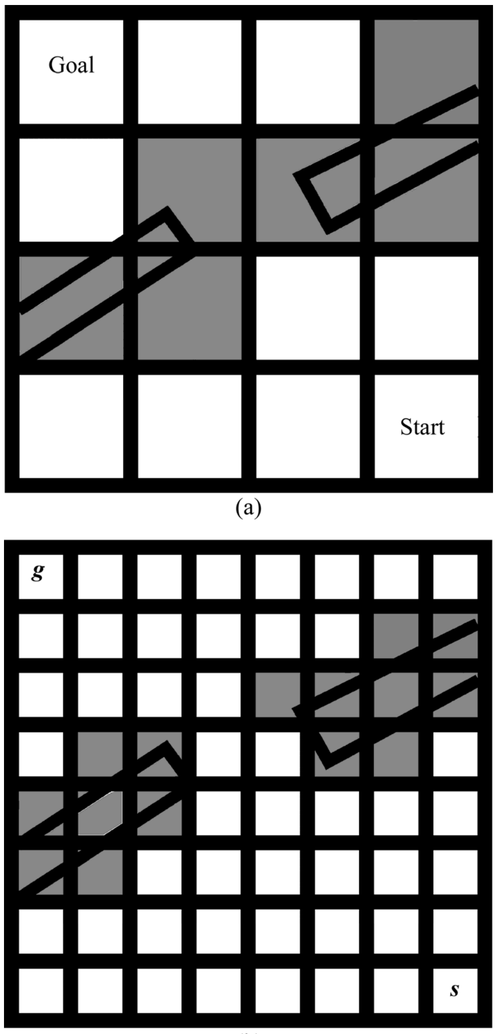
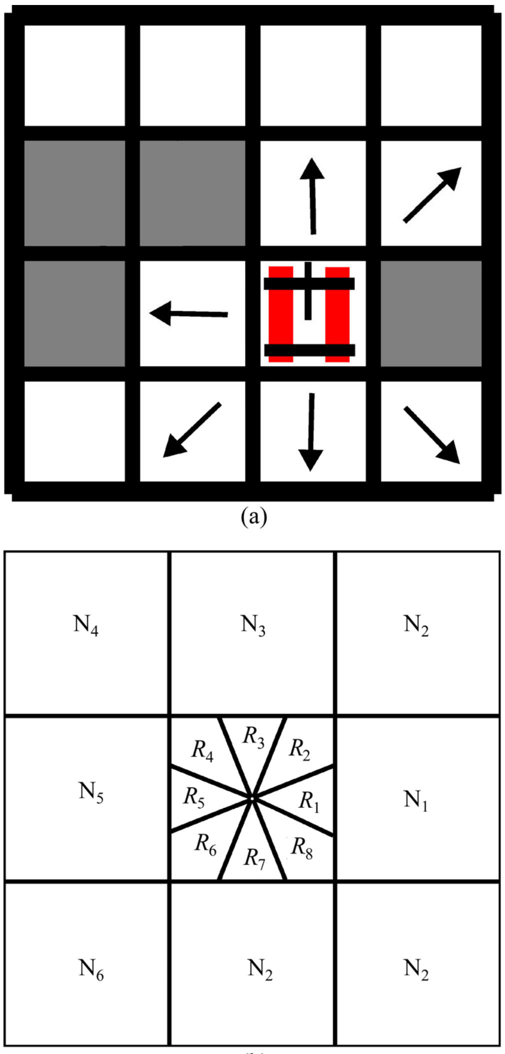


Fig. 3 An illustration to show impact of environmental representation: (a) Optimal path cannot be determined with this representation; (b) A path is found after increasing resolution of grid-world

between remaining in the current cell and moving to one of the neighboring traversable cells. The maximum number of movement positions is eight, equivalent to the number of neighbor cells around a position (Fig. 4(a)). We do not allow an agent to share a cell with another agent.

A univector field is computed for every cell of the grid-world, and the next position of the predator agent is determined using this univector field. However, the value of the univector field varies in the range of (−π, π], although the agent can only move into one of eight possible positions. Therefore, we need to determine which value corresponds to the next position of the predator agent. Figure 4(b) demonstrates our approach to divide the range of the values of a univector field into eight equal parts, defined in Eq. (4), corresponding to the eight neighboring cells. If the univector field at the agent’s position belongs to this range, then the agent will



(b)

Fig. 4 Movement Rules: (a) Directions of possible agent movement; (b) Eight regions and action selection rule

move to the corresponding neighboring cell. For instance, if the predator agent is in a position where the univector field is in the range of R_2, then the predator will move to neighbor cell.

$$R_i=\begin{cases}(-\pi/8,\ \pi/8], & \text{if } i=1\\ (\pi/8,\ 3\pi/8], & \text{if } i=2\\ (3\pi/8,\ 5\pi/8], & \text{if } i=3\\ (5\pi/8,\ 7\pi/8], & \text{if } i=4\\ (-\pi/8,\ -7\pi/8]\cup(7\pi/8,\pi], & \text{if } i=5\\ (-7\pi/8,\ -5\pi/8], & \text{if } i=6\\ (-5\pi/8,\ -3\pi/8], & \text{if } i=7\\ (-3\pi/8,\ -\pi/8], & \text{if } i=8\end{cases}$$

We next discuss the simulation of the speed of the agents in the grid-world. Basically, at each discrete time interval, agents (predators and prey) can move to neighbor cells with the same speed. However, there are some special cases in which all predator agents are initialized on the same side of the prey agent's position. In these cases, predator agents may not capture the prey agent because the prey agent tends to run from the predator agents at the same speed at which they pursue the prey. In order to overcome this situation and allow predator agents to always capture the prey agent, we assume that the speed of the prey agent is low than that of the predator agents. A parameter q_0 ($0<q_0<1$) is proposed to indicate the probability that the prey can move in that step. The smaller the q_0 is, the slower the simulated speed of the prey agent is. This concept is easy to understand because the prey agent has fewer opportunities for movement. We also simulate the speed of the prey agent in each discrete time interval. This speed is not greater than the speed of predator agents and will decrease if some predator agents already surround the prey agent. The prey agent cannot move if the number of surrounding predator agents is greater than its number of traversable cells.

Finally, we define the rule to capture the prey agent. Predator agents are determined to capture the prey agent when they occupy all neighbor cells of the prey agent. In an obstacle environment, not all eight predator agents are required to surround the prey because the prey agent sometimes gets stuck against obstacles. For instance, the prey agent (the circle in back color) in Fig. 5 is captured by only five predator agents (the shapes in different colors); the remaining positions are occupied by obstacles (rectangles in gray color).

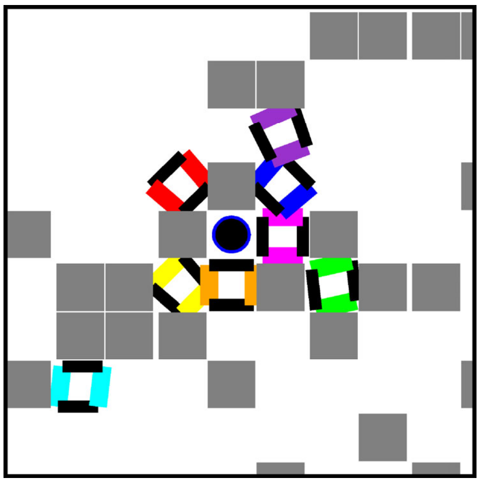


Fig. 5 Only five predator agents are needed to capture prey agent

3.3 Predator strategies

As mentioned above, the proposed approach is a combination of a univector field method for navigation and strategies for avoiding obstacles and cooperation. In this section, these strategies are introduced and classified into two categories: obstacle avoidance and cooperation. Cooperation strategies (described in "Cooperation Strategy" section) are executed to determine the movement direction of each predator agent. After that, each predator agent will navigate toward the prey agent based on direction chosen in the first step. Obstacle avoidance strategies are triggered when a predator agent encounters or predicts an obstacle in the path. All strategies are considered in a partially observable environment. Each predator agent is assumed to know the positions of the prey agent and other predator agents. In addition, each agent has knowledge of only a part of the environment around it and uses this limited knowledge to determine its next move. In the real-world, a centralized system can be used to control and share information between predator agents.

Definition 1: An escape position is a location along the boundary of an obstacle from which if an agent follows the univector field in a partially observable environment, it will not hit the obstacle again.

Definition 2: A trapped position is a location inside the rectangle surrounding an obstacle into which a predator agent falls. If the predator agent keeps following the univector field from this position, it will hit the obstacle (see Fig. 6).

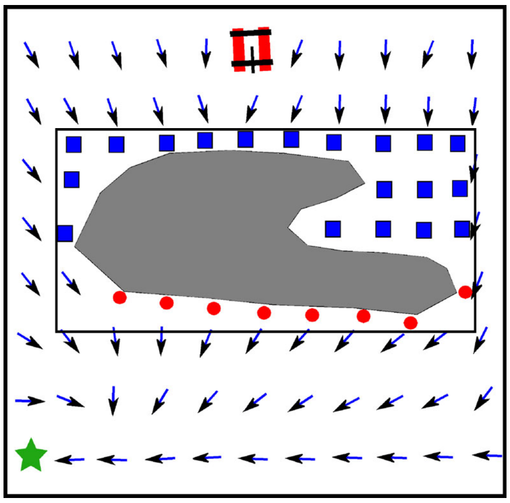


Fig. 6 An agent following univector field to reach goal (star cell) (Rectangle positions are trapped positions and circle positions are escape positions)

3.4 Obstacle avoidance strategies

3.4.1 Hitting and following boundary strategy (HFB)

In this strategy, predator agents move toward the prey agent along the univector field. However, if the univector field leads them to an obstacle in the next step, they will avoid that obstacle by following its boundary until they reach the nearest escape position (See Definition 1). This strategy is a derivation of Bug algorithms which are well known mobile robot navigation method [22]. It is the simplest strategy to avoid obstacles. In robotics, detecting and avoiding an obstacle can be archived with simple sensors such as ultrasonic, light or touch sensors. The disadvantage of this strategy is that the path generated by following the boundary of the obstacle is not usually optimal. Figure 7 demonstrates this issue: the agent needs 10 steps to reach the nearest escape position using HFB strategy, but it only needs six steps if it follows the optimal path.

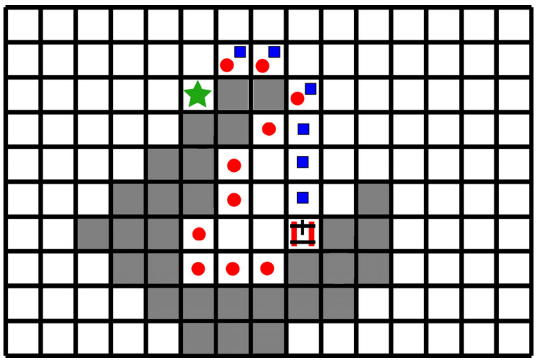


Fig. 7 Boundary path (path of circles) longer than optimal path (path of rectangles) to reaching the nearest escape position (star cell)

3.4.2 Trapped and following shortest path strategy (TFSP)

This strategy can solve the issue of the HFB strategy by following an optimal path generated from a shortest path algorithm such as those in Ref. [12]. A shortest path algorithm is only executed to determine the path from a trapped position (See Definition 2) to an escape position in the partially observable environment around the agent and that obstacle. Therefore, this algorithm has little effect on performance. Figure 8 depicts a scenario in which an agent encounters a non-convex obstacle. An optimal path to the nearest escape position is determined in fewer steps than if the agent has applied the HFB strategy.

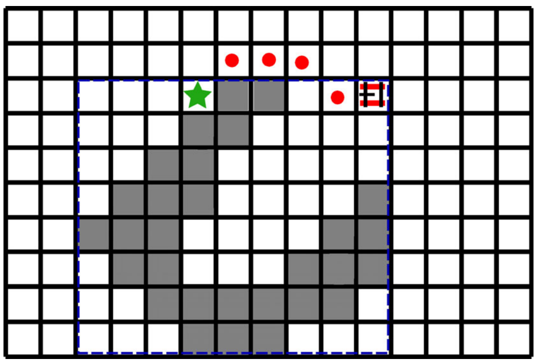


Fig. 8 Trapped and following shortest path strategy

3.4.3 Predicted and following shortest path strategy (PFSP)

Predicting obstacle ahead is not a new idea in robot navigation [23, 24] and PFSP is a kind of strategy which combines obstacle prediction and univector field method. The PFSP strategy predicts the next obstacle based on the univector field in a partially observable environment and determines the optimal path to an escape position of that obstacle. Agents employing this strategy can avoid an obstacle before it falls into that obstacle. Figure 9 demonstrates the path of an agent (the path of circles) after predicting an obstacle ahead.

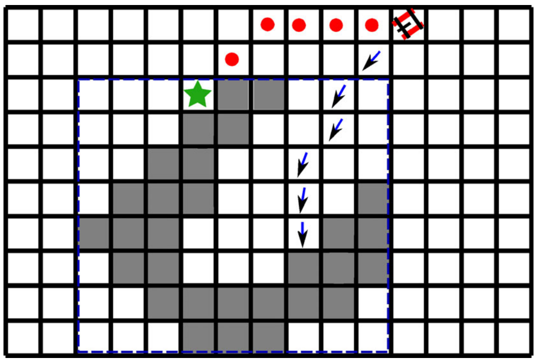


Fig. 9 Predicting obstacle ahead and avoiding it

3.5 Strategies for cooperation among predator agents

With the model of the pursuit problem using the univector field proposed in this work, cooperation among predator agents focuses on communication between them to determine the most suitable direction for each of them and help them to more quickly surround prey agent. In addition, cooperation among predator agents is also reflected in the moving order of predator agents. Basically, predator agents move in an orderly manner. However, sometimes, this movement order can cause difficulty for one or many predator agents. Figure 10 provides an example of this issue, where agent 6 wants to move to the current position of agent 7, agent 7 wants to move to the current position of agent 8, and agent 8 wants to move to an empty position (The top-left prey's neighbor cell). If agents move in an orderly manner, at least three steps are needed for all agents to archive the desired positions. However, we only need one step if we

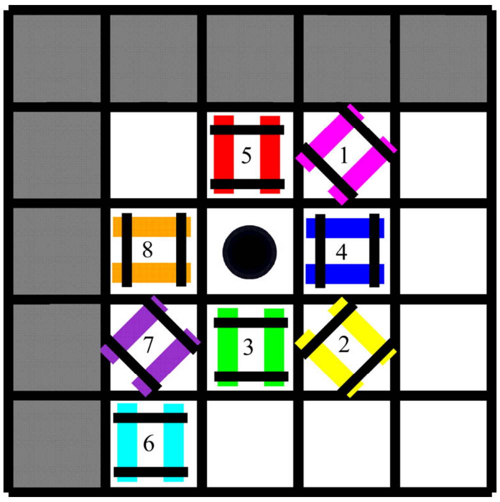


Fig. 10 Scenario needing re-order moving order of predator agents

make a slight change in moving order. In this example, a good moving order is agent 8 first, then agent 7, and finally is agent 6.

3.5.1 Greedy strategy

Each predator agent will choose a neighbor cell in the direction of the prey agent in an orderly manner (from the first to the eighth). The agent will measure the distances from its position to all neighbor cells of the prey agent and move to a cell that minimizes that distance from its position. If the chosen cell has been assigned to another predator agent, it will be ignored and another of the remaining neighbor cells that also minimizes the distance from its position will be chosen. The communication among predator agents in this strategy only involves information about prey agent neighbor cells. A predator will only choose an available neighbor cell and ignore the neighbor cells chosen by other predator agents, although these cells may be more suitable than the cell that it is choosing. Some of the predator agents achieve the best suitable path (local goal) when following this strategy.

3.5.2 Cooperation strategy

In contrast to the greedy strategy, it is the cooperation strategy. The greedy strategy requires less communication among agents. However, the paths generated by this strategy are not always the most suitable paths when considering them in the context of all eight predator agents (a common goal). The cooperation strategy will consider this problem in achieving the common goal of all predator agents and aims to find an optimal solution from all solutions generated by the eight predator agents and eight neighbor cells of the prey agent. The optimal solution is a combination of the predator agents and the specified directions, respectively, so that the total of the distances from each predator agent to the chosen cell is the minimum value.

3.6 Prey strategies

In response to predator strategies, the prey agent chooses among three strategies defined by increasing intelligence. In consequence, the computational resources required for each strategy also increase based on intelligence of each strategy. These strategies consist of the behaviors shown in “Movement rules” section such as prey agent is limited in the speed and its speed will continuously decrease as the number of predator agents surrounding the prey agent increases. Other features of each strategy are shown below.

The first strategy is a random strategy. At each discrete time interval, the prey agent randomly moves to an empty neighbor cell among its eight neighboring cells. This strategy is the simplest because the prey agent does not need to know the positions and moving directions of predator agents. Predator agents will quickly reach the prey agent using this strategy.

The next strategy is a greedy strategy. With this strategy, the prey agent measures the distances to the eight predator agents and moves to the empty cell that maximizes the distance to the nearest predator agent. This strategy requires knowledge of the positions of all predator agents. The disadvantage of this strategy is that the prey agent sometimes gets stuck against an obstacle and may not be able to escape before being captured.

In order to solve the above issue, the prey agent applies a complex strategy called A*. This strategy, based on Ref. [13], is modified to suitable for an infinite environment. A* needs not only information about predator agents, but also information about the partial observation environment around the prey agent. Although it requires more computation, this strategy is powerful enough to challenge predator strategies. Initially, the strategy must identify the nearest predator agent based on the positions of all predator agents. After that, the strategy measures $\text{cost}_{\text{predator}}$ and $\text{cost}_{\text{prey}}$ at each cell within a limited search window centered at the prey agent position. $\text{cost}_{\text{predator}}$ and $\text{cost}_{\text{prey}}$ represent the distance of the optimal path from that cell to the nearest predator agent and prey agent, correspondingly. The objective of A* strategy is to find a cell from which the distance from the nearest predator agent to it is maximized and will not bring the prey agent into contact with any predator agents during travel to this cell through optimal path. This is determined by ensuring that each cell on the optimal path satisfies $\text{cost}_{\text{predator}}\,[\text{cell}] - \alpha * \text{cost}_{\text{prey}}\,[\text{cell}]>0$, where α is computed by the formula $\text{speed}_{\text{predator}}/\text{speed}_{\text{prey}}$.

4 Experiments

In this section, we present the experimental results of the pursuit problem using MATLAB software. Cooperation strategies are combined with obstacle avoidance strategies to counter prey strategies. With every combination of strategies, we perform 100 simulations to measure the average number of steps needed to capture the prey agent (capture time) and the successful capture rate. In these simulations, predator agents and the prey agent are randomly initialized at distinct positions within a radius of 100 cells. The coefficients of univector field in Eq. (1) are n=5 (as mentioned in “Univector field method” section). The environment of each simulation is also randomly generated in the size of 100 ×100 and will be applied to the next part of the environment if the pursuing space exceeds 100×100 unit. The pursuit problem is also simulated in an obstacle-free environment to make sure that the proposed strategies can normally work on that environment. In addition, there are four types of environments containing obstacles. The first three types are environments with different discrete obstacle cells ratios of 10%, 20%, and 30%, respectively. The obstacle ratio is chosen to make sure that the environment has enough space for predator pursuit and prey escape. The fourth type is an environment with non-convex U-type obstacles (See Fig. 11). In the simulations, if a pursuit exceeds 1000 discrete time intervals, it is considered unsuccessful. The computational complexity of each combination of strategies depends on the nature of each strategy. Particularly, they both depend on the size of the obstacle and the range of the partial environment and then a predator agent can observe. For PFSP strategy, in the number of steps a predator agent can predict also effects to computational resources. The experimental results are shown in Table 1

A cooperation strategy always shows efficiency regarding capture time in all environments. Figure 12 compares capture times between greedy strategy and cooperation strategy. A cooperation strategy produces a 6.3% shorter capture time than greedy strategy. In particular, capture time is reduced by 17.84% (from 149.13 time steps to 117.6 time steps) in the environment of 10% obstacles and using PFSP strategy. Figures 13(a) and (b) show the greater efficiency of cooperation strategy compared to greedy strategy. In these figures, the capture times of the greedy strategy and cooperation strategy are 100 and 80 discrete time units, respectively. The trajectories of predator 2 and 4 in greedy strategy require more steps to surround the prey agent than their trajectories in cooperation strategy.

Figure 12(b) shows the efficiency of three obstacle avoidance strategies. Results from the TFSP and PFSP strategies are better than those of the HFB strategy in a complex environment, especially in the environment with non-convex U-type obstacles. Figures 14(a) and (b) show the trajectories of predator agents using HFB strategy and TFSP strategy, respectively. The trajectories of predators 5 and 8 require more steps to avoid obstacles, resulting in inefficiency.

With the PFSP strategy, the results are better than other strategies if the prey agent uses a random or greedy

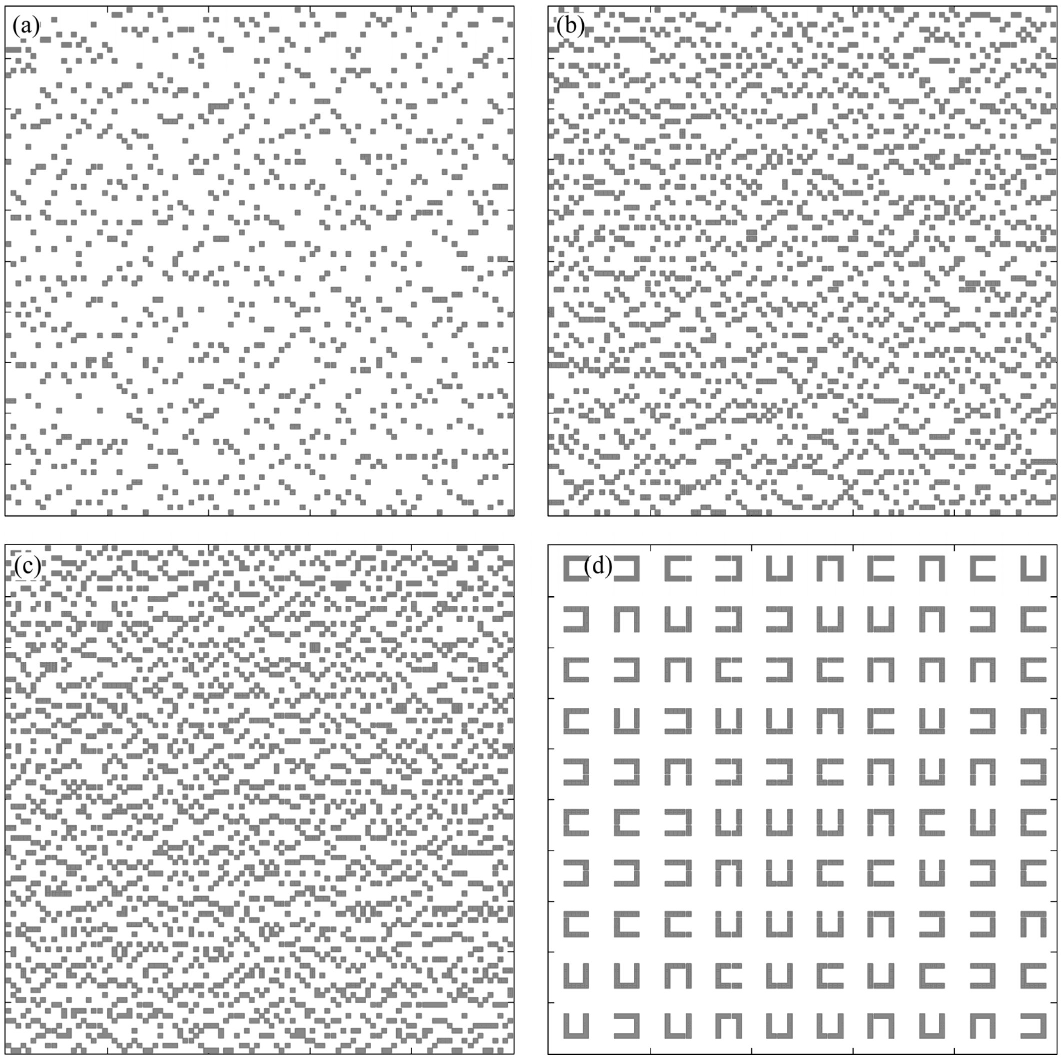


Fig. 11 Obstacles with ratios 10% (a), 20% (b), 30% (c) in environment and U-type obstacles (d)

Table 1 Average number of moving to capture prey agent using different strategies in different obstacle environments

Strategy	Algorithm	Greedy strategy						Cooperation strategy					
		HFB		TFSP		PFSP		HFB		TFSP		PFSP	
		Rate	Step	Rate	Step	Rate	Step	Rate	Step	Rate	Step	Rate	Step
Obstacle-free	Random	100	71.0	100	70.9	100	69.9	100	68.3	100	68.1	100	68.1
	Greedy	100	115.9	100	114.2	100	113.7	100	105.9	100	105.7	100	103.3
	A*	100	120.4	100	126.7	100	122.5	100	105.7	100	108.4	100	109.9
Maze grids with 10% obstacles	Random	100	71.9	100	71.3	100	72.4	99	69.2	100	69.7	100	69.7
	Greedy	100	106.0	100	104.3	100	106.6	99	100.6	100	101	100	99.1
	A*	100	112.1	100	118	100	143.1	98	100.7	100	104.2	100	117.6
Maze grids with 20% obstacles	Random	100	69.4	100	72.4	99	71.7	97	70.3	99	67.6	100	67.5
	Greedy	99	107.6	100	105.2	99	105.9	98	96.5	98	97.1	100	99.2
	A*	100	102.9	100	103.2	98	128.9	97	98.5	99	94.8	99	112.5
Maze grid with 30% obstacle	Random	99	68.7	97	70.5	98	69.7	100	70.8	100	67.5	100	68.3
	Greedy	98	99.8	99	96.4	98	100.6	98	95.3	100	91.3	99	92.9
	A*	95	115.1	98	108.4	98	123.1	98	104.1	100	101.5	100	115.9
Maze grid with *U* type obstacle	Random	94	76.5	96	70.7	96	70.9	100	78.7	100	70.2	100	68.9
	Greedy	99	120.1	92	95.9	93	88.6	99	128.1	100	89	99	86.2
	A*	90	185.6	89	146.9	92	158.7	95	185.6	100	142.8	100	149.1
Total	Random	98.5	72.20	99	71.3	98.8	71.2	99	71.6	99.8	68.9	100	68.6
	Greedy	99.5	112.4	98	104.9	98	103.7	99	107.8	99.5	98.2	99.75	97
	A*	97.5	130.2	97.3	123.7	97.5	138.3	97.5	122.6	99.8	112.5	99.75	122.3

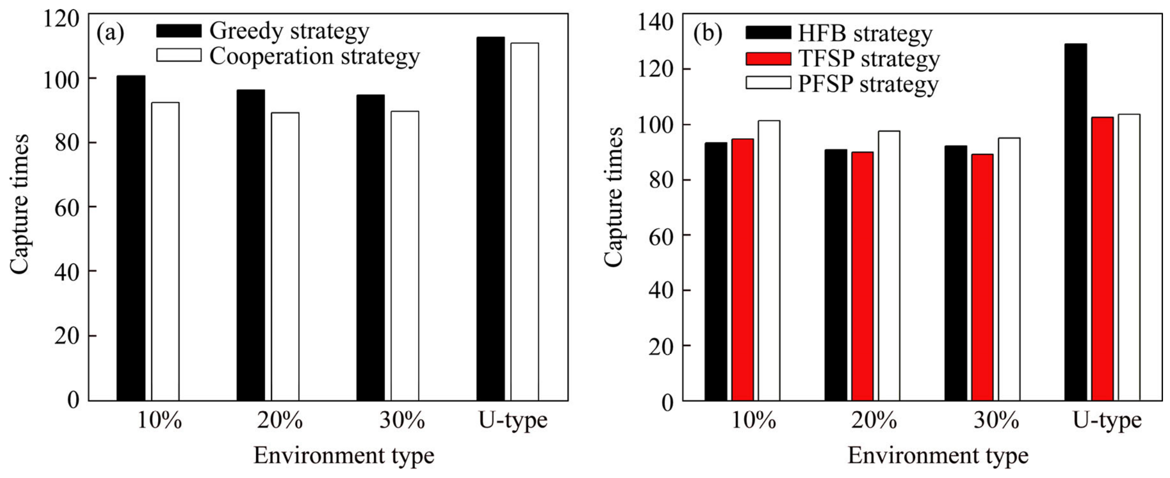


Fig. 12 Efficiency of different strategies: (a) Cooperation strategy; (b) Obstacle avoidance strategy

strategy. However, if the prey agent uses A* strategy, the results show that it is less efficient than the other strategies. The PFSP strategy lets a predator agent generate the predicted shortest paths to an escape point in a partially observable environment. However, it does not ensure that the path in a partially observable environment is an optimal one. This problem intensifies when the prey agent employs a smart strategy that requires the predator agent to follow many predicted paths.

Finally, we compare the capture rate of the different strategies. The capture rate in an obstacle-free environment is always 100%. In an obstacle environment, there is a small unsuccessful capture rate occurring randomly and it tends to happen in U-type obstacle environment. No unsuccessful cases are found in cases involving local minima, in Ref. [15]. Figure 15 depicts an unsuccessful case in which the predator agents cannot reach the prey agent because the univector field and involved strategies let it move repeatedly between two obstacles.

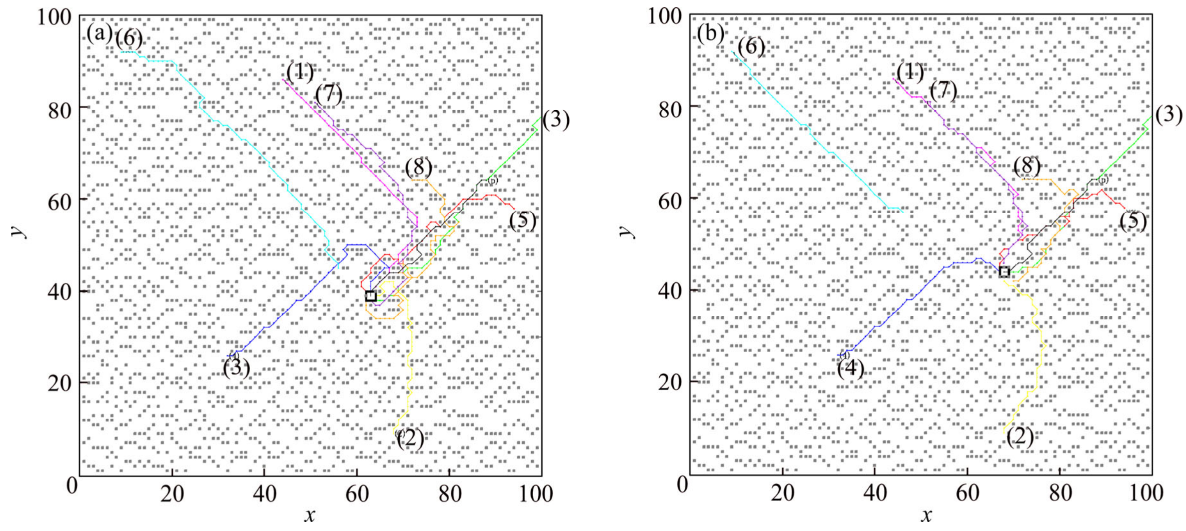


Fig. 13 Trajectories of eight predator agents and the prey agent using greedy strategy (a) and cooperation strategy(b)

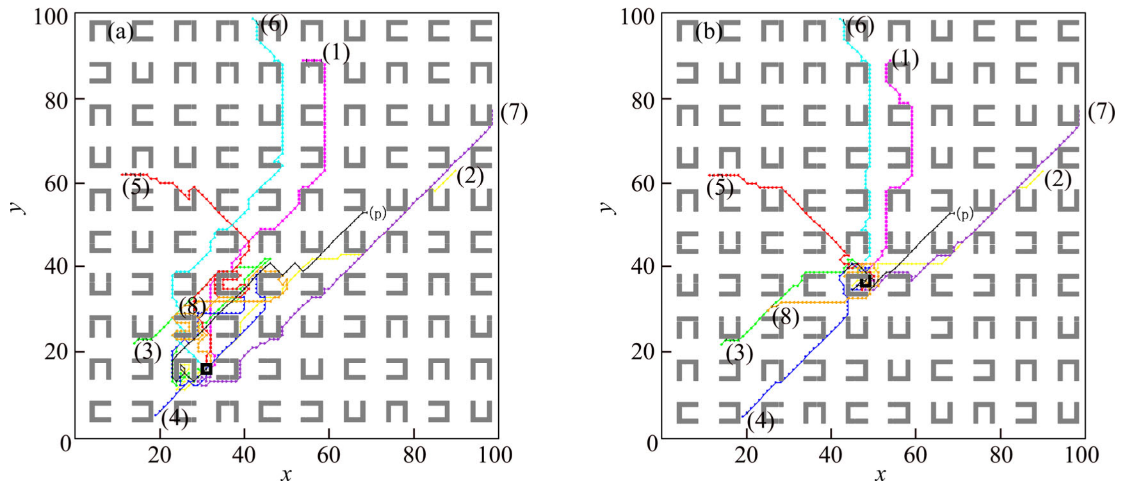


Fig. 14 Trajectories of eight predator agents and prey agent using HFB strategy (a) and TFSP strategy (b)

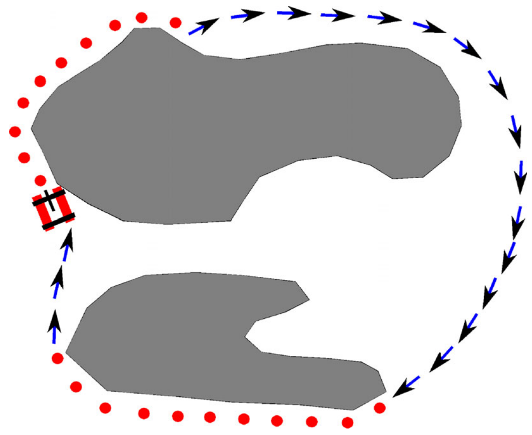


Fig. 15 Predator agent escaping an obstacle but being trapped against another obstacle

5 Conclusions

We have focused on the pursuit problem derived from the original proposed in Ref. [1] but consider it in an infinite obstacle environment. We have introduced algorithms allowing predator agents to capture prey agent in the finite time interval. Algorithms mainly based on the univector field method were successful in allowing predator agents to capture a prey agent. In addition, predator and prey strategies were also defined to test the proposed approaches. Experimental results of the TFSP strategy were better than those of the HFB strategy in obstacle avoidance, and a cooperation strategy of predator agents helps them to move more quickly surround the prey agent compared to a greedy strategy. The PFSP also shows the efficiency of some prey strategies. Collectively, all these results indicate that the algorithms can solve the pursuit problem in some environments with some types of obstacles. In the future, the pursuit problem will be extended to situations involving multiple prey agents, which is definitely more difficult than the pursuit of a single prey agent. Considering the pursuit problem in the environment of dynamic obstacles is also an interesting topic for future research.

Acknowledgements

The authors are grateful to the Basic Science Research Program through the National Research Foundation of Korea (NRF-2014R1A1A2057735) and the Kyung Hee University in 2016 [KHU-20160601].

References

[1] BENDA M, JAGANNATHAN V, DODHIAWALA R. On optimal cooperation of knowledge sources-an empirical investigation [R]. Seattle, WA, USA: Boeing Advanced Technology Center, Boeing Computing Services, 1986.

[2] CHUNG T H, HOLLINGER G A, ISLER V. Search and pursuit-evasion in mobile robotics [J]. Autonomous Robots, 2011, 31(4): 299−316.

[3] SINCÁK D H V. Multi-robot control system for pursuit-evasion problem [J]. Journal of Electrical Engineering, 2009, 60(3): 143−148.

[4] STIFFLER N, O'KANE J. A sampling-based algorithm for multi-robot visibility-based pursuit-evasion [C]// IEEE/RSJ International Conference on Intelligent Robots and Systems. Chicago, USA: IEEE, 2014: 1782−1789.

[5] BILGIN A, KADIOGLU-URTIS E. An approach to multi-agent pursuit evasion games using reinforcement learning [C]// International Conference on Advanced Robotics. Istanbul, Turkey: IEEE, 2015: 164−169.

[6] WANG H, YUE Q, LIU J. Research on Pursuit-evasion games with multiple heterogeneous pursuers and a high speed evader [C]// 27th Chinese Control and Decision Conference. Qingdao, China: IEEE, 2015: 4366−4370.

[7] STIFFLER N A O J. A complete algorithm for visibility-based pursuit-evasion with multiple pursuers [C]// IEEE International Conference on Robotics and Automation. Hong Kong, China: IEEE, 2014: 1660−1667 .

[8] ALONSO L, REINGOLD E M. Bounds for cops and robber pursuit [J]. Computational Geometry, 2010, 43(9): 749−766.

[9] LIN W, QU Z, SIMAAN M. Nash strategies for pursuit-evasion differential games involving limited observations [J]. IEEE Transactions on Aerospace and Electronic Systems, 2015, 51(2): 1347−1356.

[10] KLEIN K, SURI S. Capture bounds for visibility-based pursuit evasion [J]. Computational Geometry, 2015, 48(3): 205−220.

[11] KEHAGIAS A, MITSCHE D, PRAŁAT P. Cops and invisible robbers: The cost of drunkenness [J]. Theoretical Computer Science, 2013, 481: 100−120.

[12] NEAPOLITAN R, NAIMIPOUR K. Foundations of algorithms [M]. Jones & Bartlett Learning, 2010.

[13] UNDEGER C, POLAT F. Multi-agent real-time pursuit [J]. Autonomous Agents and Multi-Agent Systems, 2010, 21(1): 69−107.

[14] UNDEGER C, POLAT F. Real-time moving target search [M]// Agent Computing and Multi-Agent Systems. Berlin, Heidelberg: Springer, 2009: 110−121.

[15] VIET H, AN S, CHUNG T. Univector field method based multi-agent navigation for pursuit problem [J]. International Journal of Fuzzy Logic and Intelligent Systems, 2012, 12(1): 86−93.

[16] KIM Y J, KIM J H, KWON D S. Evolutionary programming-based univector field navigation method for past mobile robots [J]. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, 2001, 31(3): 450−458.

[17] HONG Y, KIM J. Footstep planning based on univector field method for humanoid robot [M]// Advances in Robotics. Berlin, Heidelberg: Springer, 2009: 125−134.

[18] LIM Y, CHOI S H, KIM J H, KIM D H. Evolutionary univector field-based navigation with collision avoidance for mobile robot [C]// 17th World Congress, The International Federation of Automatic Control. Seoul, Korea, 2008: 12787−12792.

[19] GASSER L, ROUQUETTE N, HILL R W, LIEB J. Representing and using organizational knowledge in DAI systems [C]// Distributed Artificial Intelligence. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1990, 2: 55−78.

[20] KORF R. A simple solution to pursuit games [C]// Distributed Artificial Intelligence, Michigan, USA: Springer Verlag, 1992, 2: 183−194.

[21] REVERTE J, GALLEGO F, LLORENS F. Extending Korf's ideas on the pursuit problem [C]// International Symposium on Distributed Computing and Artificial Intelligence 2008. Spain: Springer, 2009: 245−249.

[22] NG J, BRÄUNL T. Performance comparison of bug navigation algorithms [J]. Journal of Intelligent and Robotic Systems, 2007, 50(1): 73−84.

[23] FERGUSON D, DARMS M, URMSON C, KOLSKI S. Detection, prediction, and avoidance of dynamic obstacles in urban environments [C]// IEEE Intelligent Vehicles Symposium. Eindhoven, Netherlands: IEEE, 2008: 1149−1154.

[24] YUNG N, YE C. Avoidance of moving obstacles through behavior fusion and motion prediction [C]// 1998 IEEE International Conference on Systems, Man, and Cybernetics. San Diego, CA, USA: IEEE, 1998: 3424−3429.

[25] STENTZ A. The focussed D^* algorithm for real-time replanning [C]// Proceedings of the International Joint Conference on Artifical Intelligence. Montreal, Quebec, Canada: AAAI, 1995: 1652−1659.

[26] KOENIG S, LIKHACHEV M. Fast replanning for navigation in unknown terrain [J]. Transactions on Robotics, 2005, 21(3): 354−363.

(Edited by YANG Hua)

Cite this article as: Le Pham Tuyen, Hoang Huu Viet, Sang Hyeok An, Seung Gwan Lee, Dong-Han Kim, Tae Choong Chung. Univector field method-based multi-agent navigation for pursuit problem in obstacle environments [J]. Journal of Central South University, 2017, 24(4): 1002−1012. DOI: 10.1007/s11771-017-3502-0.