A Fuzzy Approach to Generating Adaptive Opponents in the Dead End Game

HAO-MIN HSIEH[1] AND LING-LING WANG[2,*]
[1]*Department of Information and Design, Asia University, Taiwan*
[2]*Department of Information Communication, Asia University, Taiwan*

ABSTRACT

In most of current computer games, computer-controlled opponents are often guided by a limited set of fixed strategies. The behavioral repetition of computer-controlled opponents reduces the human player's enjoyment during gameplay. Furthermore, the predefined fixed difficulty levels of a game are not satisfactory to most human players. Human players prefer challenging games which keep level with them. Hence, in this study we propose a fuzzy approach to adapting opponents' tactics to the behavior of the player such that the player's win/loss rate is kept at their desired rate. The value of the desired rate can be preset by the player according to the player's preference for challenge. Our experiments are conducted on a predator/prey game. The experimental results show the adaptation efficiency and robustness of the proposed method.

Key words: computer game, adaptive opponents, artificial intelligence, fuzzy inference, predator/prey game.

1. INTRODUCTION

Traditionally, game developers have invested most of their resources in creating realistic graphics and sound effects, but in recent years more attention has been paid to developing challenging games. Game artificial intelligence (AI) is the key to providing challenging gameplay experiences for most games (Laird & Lent, 2000). Game AI refers to the techniques needed in computer games to produce the illusion of intelligence in the behavior of computer-controlled opponents, and some AI techniques have been adopted (Rabin, 2004). Although the use of these techniques produces an improvement in game AI, it is currently still unsatisfactory. One reason is that computer-controlled opponents are often guided by a limited set of fixed strategies. Human players can thus predict the behaviors of computer-controlled opponents after playing a game several times, and then feel the game has become boring (Manslow, 2002). Some game developers lend variety to the behaviors of computer-controlled opponents through writing lots of elaborate scripts, even though the process of script writing may be tedious.

To solve the aforementioned problems, research about adaptive game AI (Andrade, Ramalho, Santana & Corruble, 2005; Demasi & Cruz, 2003; Houlette, 2004; Hunicke & Chapman, 2004; Ponsen, Muñoz-Avila, Spronck & Aha, 2006; Spronck, 2005; Spronck, Sprinkhuizen-Kuyper & Postma, 2003, Spronck, Ponsen, Sprinkhuizen-Kuyper & Postma, 2006; Yannakakis & Hallam, 2005; Yannakakis, Levine & Hallam, 2004; Ponsen, Spronck, Muñoz-Avila & Aha, 2007; Wong, 2008) has been proposed. Adaptive game AI is game AI with the capabilities of

[*] Corresponding author. E-mail: ling@asia.edu.tw

self-correction and creativity (Spronck, 2005). It is traditional game AI incorporated with machine learning techniques. Learning can be applied before or after a game's release. Learning is called offline if it is applied before a game's release (or in the period of game development), whereas it is called online if applied after a game's release (or during gameplay). Spronck et al. (2003) proposed a neuro-evolutionary technique to offline train a computer-controlled opponent in a shooting game by playing with a scripted opponent. Ponsen et al. (2006) used an evolutionary algorithm to offline generate tactics for a real-time strategy game. Houlette (2004) proposed player modeling to change opponents' tactics according to the player's profile which is a record of the player's behavior during gameplay. Demasi and Cruz (2003) proposed coevolutionary approaches to online evolving computer-controlled opponents' tactics of a simple action game. Hunicke and Chapman (2004) proposed a probabilistic method based on the inventory theory to modify the environment settings to adapt the game to the player. For instance, when the game is too hard for the player, the quantities of supplies are increased and the opponents' strength is weakened. This approach is only suitable for games whose environment settings can be changed. Yannakakis et al. (2004) and Yannakakis and Hallam (2005) applied a neuro-evolutionary technique to train opponents offline for a multi-agent cooperation game. The trained opponents also evolve during gameplay based on the proposed interest-judging criteria so that the player has a continuous interest in the game. Andrade et al. (2005) applied reinforcement learning to learn tactics of a fighting game. If the game is hard for the player, the opponent chooses a suboptimal tactic to achieve game balance. Spronck et al. (2006) proposed dynamic scripting to online create scripts for the opponents. A script is composed of a set of if-then rules selected from a rulebase. Whether a rule is selected for a script depends on the weight of the rule. The weight of a selected rule is adjusted according to the results of battles after each round ends.

Although the above-mentioned approaches can adapt opponents' tactics to the player, the numbers of rounds needed for adaptation sometimes vary greatly even when playing against the same fixed-strategy opponent (Spronck et al., 2006; Yannakakis et al., 2004; Yannakakis & Hallman, 2005). Hence in this paper, a new fuzzy approach with a rule enabling/disabling mechanism is proposed to adapt the game's tactics to the behavior of the player such that a skill balance between the player and computer-controlled opponents can be achieved effectively. In this study, we model intuitive and subjective human ideas of controlling opponents in a game into fuzzy rules. Fuzzy inference is applied during gameplay to guide the opponents. After each round of the game, we strengthen or weaken the opponents' skill according to the player's performance. Skill balance between the player and opponents is achieved through rule enabling and disabling.

The rest of the paper is organized as follows. In Sec. 2, we present a detailed description of a predator/prey game which is used as the test bed of our research. In Sec. 3, we describe the proposed fuzzy adaptive approach for the predator/prey game. The experiments conducted for evaluating the performance of the proposed approach are described in Sec. 4. Finally in Sec. 5, conclusions are provided.

2. THE DEAD END WORLD

In this study, we chose a two-dimensional multi-agent predator/prey game called Dead End (Yannakakis et al., 2004) as our test bed because it is an interesting multi-agent game and its graphics are simple. Hence, we can place emphasis on the investigation of the game AI but not graphics. Figure 1 shows a snapshot of a Dead End game. In the game field, the *x*-axis directs from left to right and the *y*-axis from top to bottom. The characters in the upper region of the game field are Ghosts, and the one in the lower region is the Player. The top of the game field, where there is no barrier, is the exit. The dimension of the game field is 320 × 480 pixels, and that of Ghosts and Player is 20 × 20 pixels. The player is initially in the lower region of the game field, while Ghosts are in the upper region.

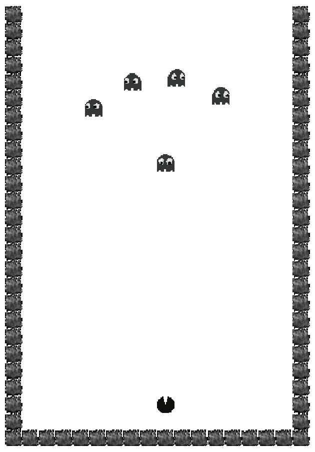


Figure 1. The snapshot of the Dead End game.

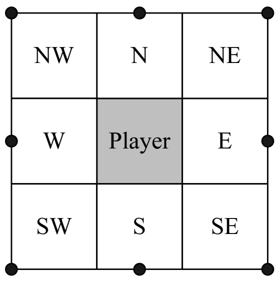


Figure 2. Eight moving directions of CB Player.

The player aims to reach the exit and avoid the Ghosts, whereas the Ghosts aim to defend the exit and to kill the Player. The player owns health points which denote Player's health. A Ghost attacks a Player by touching the Player. Once the Player is touched by a Ghost, the Player's health points decrease by one, and this repeats until the Player leaves the Ghost. The player dies and loses the game if the number of health points equals zero. Additionally, if the Player cannot reach the exit within a predefined period of time, the Player loses the game. After the Player either wins or loses the game, a new round starts. The Player moves at double the Ghosts' speed, and thus it is impossible for a single Ghost to catch the Player. Hence, all the Ghosts must hunt cooperatively.

In this research, our goal is to develop an adaptive approach to controlling these Ghosts effectively. To test the adaptability of computer-controlled Ghosts during the experiments, we designed three types of fixed-strategy for the Player. These fixed strategies are somewhat like those which human players may adopt. They are described below from simple to complicated.

(a) Simple-Avoidance (SA) Player: An SA Player can move in four directions (north, south, east, and west). The Player moves directly toward the closest exit if no Ghost is in its visible area (40 pixels around it). However, if the Ghosts approach and enter its visible area, the Player moves in the direction which is not blocked by Ghosts. It behaves similarly to a novice human player.

(b) Improved-avoidance (IA) Player: The strategy of an IA Player is the same as that of an SA Player except that an IA Player may move in eight directions (plus northeast, northwest, southeast, and southwest). Therefore it can reach the exit more effectively than an SA Player.

(c) Cost-based (CB) Player: This strategy is obtained by modifying that proposed by Yannakakis et al. (2004). Compared with the other two types of Player, the CB Player performs a more effective ghost-avoiding and exit-achieving strategy. Its movement is determined based on the costs of its eight possible moving directions. In Figure 2, the eight grid squares of dimension 20 × 20 pixels denote eight moving directions of the CB Player, and the coordinate of the dot on the perimeter of a grid square denotes the coordinate of the grid square. The cost of each moving direction is calculated through the function $C(x, y)$:

$$C(x,y) = \Delta E(y) + G(x,y)\,, \tag{1}$$

$$\Delta E(y) = y_e - y\,, \tag{2}$$

$$G(x,y) = \sum_{n=1}^{N} \frac{\rho}{\left|x_{g,n} - x\right| + \left|y_{g,n} - y\right|} \tag{3}$$

where (x, y) denotes the coordinates of a grid square, $\Delta E(y)$ is the distance from the grid square to the exit, and y_e is the y-axis coordinate of the exit. We ignore the distance in the x-axis while evaluating $\Delta E(y)$ since the whole top of the game field is the exit. N is the number of Ghosts; $(x_{g,\,n}, y_{g,\,n})$ are the coordinates of the n^{th} Ghost's center. ρ is a parameter that indicates the weight of $G(x, y)$ to $C(x, y)$. A low value of ρ denotes that the CB Player prefers moving toward the exit to avoiding Ghosts, whereas a high value of ρ denotes that avoiding Ghosts has high priority for the CB Player. By changing the value of ρ, CB Players with different ghost-avoiding and exit-achieving strategies can be obtained. Before starting a movement, the CB Player first calculates the costs of the eight moving directions, and then moves 20 pixels in the direction with the minimal cost.

3. GENERATION OF ADAPTIVE OPPONENTS

3.1 The Fuzzy Rulebase

In this study, intuitive and subjective ideas or strategy of controlling Ghosts are modeled beforehand into fuzzy rules. These rules with fuzzy inference are used to guide Ghosts during gameplay. In the following paragraphs, we first describe the proposed strategy for Ghosts, and then explain the implementation of this strategy in fuzzy logic.

To catch the Player efficiently, Ghosts appear in the role of vanguard or fullback. Each round the game starts, one of Ghosts at the lowest place plays the vanguard and the others the fullbacks. The vanguard leads fullbacks to chase Player, and the fullbacks follow the vanguard and form a fan-like formation as shown in Figure 1. We design five types of movement for the vanguard and fullbacks as summarized in Table 1. Type A and type C movements are used to chase the Player to catch it. Type B movement is used to block the Player's way while the Player tries to cut a way through the Ghosts to the exit. The use of type D and type E movements let the Ghosts form a fan-like formation to prevent the Player from approaching the exit.

Table 1. *Five types of Ghosts' movement*

Role	Movement	
Vanguard	A:	Chase Player
	B:	Block Player's way
Fullback	C:	Chase Player
	D:	Approach or part from the vanguard to keep an appropriate distance
	E:	Part from the closest fullback to keep an appropriate distance

To model the five types of movement into fuzzy rules, we define five fuzzy and one crisp input variables for use in the antecedents of fuzzy rules, and five fuzzy output variables for use in the consequents of fuzzy rules. The six input variables provide the Ghost with information about distances to the Player, to the exit, to the vanguard, to the closest fullback, and whether the Player's y-axis coordinate is smaller than that of Ghost's, as listed in Table 2. The five fuzzy output variables are used to derive the Ghost's moving directions, as listed in Table 3. The defuzzified output of a fuzzy rule is used as a weight of the derived moving direction. For example, the defuzzified output of a rule whose consequence is *chasePlayer* is used as a weight of the moving direction toward Player. When two or more rules are activated at a time, the weights corresponding to the same derived moving direction are summed. Then Ghost's moving direction is the derived one with the highest weight.

The fuzzy sets of each fuzzy input and output variable are designed from our experience. We define each fuzzy variable with only two or one fuzzy set whose membership function is a triangular or trapezoid shape. The smooth transitions in the Ghosts' behavior are achieved by overlapping each fuzzy set with its adjacent set by at least 25% (Bourg & Seeman, 2004). To implement the five types of movement for Ghosts, we design 40 fuzzy rules in total, 12 rules for the vanguard and 28 rules for the fullbacks, as shown in Tables 4 and 5, respectively. In these tables, each row represents a rule; the checked columns denote which fuzzy sets of input variables are taken as the antecedent of the rule, and the last column denotes which fuzzy set of output variables is taken as the consequence of the rule. All rules can work coordinately through a little trial-and-error tuning of the membership functions of the fuzzy variables. These membership functions are shown in Figures 3 to 12. The correlation-minimum inference method (Mamdani, 1977) is applied to fuzzy inference and the weighted average method (Babuska, 1998) to defuzzification. The use of the fuzzy approach simplifies the design of rules and shortens the period of development.

Table 2. *Input variables*

Variable name	Meaning
distToPlayer	The distance from Ghost to Player
distToExitY	The y-axis distance from Ghost to the exit
distToPlayerX	The x-axis distance from Ghost to Player
distToVanguard	The distance from Ghost to the vanguard
distToFullbackX	The x-axis distance from Ghost to the closest fullback
getPast	A Boolean value denoting whether Player's y-axis coordinate is smaller than that of Ghost's

Table 3. *Output variables*

Variable name	Meaning
chasePlayer	Move toward Player
partFromPlayerY	Part from Player in y-axis
approachVanguard	Move toward the vanguard
partFromVanguardY	Part from the vanguard in y-axis
partFromFullbackX	Part from the fullback in x-axis

Table 4. *Fuzzy rules for the vanguard*

Rule	Antecedent							Consequent
	distToPlayer		*distToExitY*		*distToPlayerX*	*getPast*		
	near	*far*	*near*	*far*	*far*	*false*	*true*	
1	V		V			V		*chasePlayer*: *many*
2	V		V				V	*chasePlayer*: *many*
3	V			V		V		*chasePlayer*: *many*
4	V			V			V	*chasePlayer*: *many*
5		V	V			V		*chasePlayer*: *few*
6		V	V				V	*chasePlayer*: *many*
7		V		V		V		*chasePlayer*: *few*
8		V		V			V	*chasePlayer*: *many*
9	V		V		V	V		*partFromPlayerY*: *few*
10	V			V	V	V		*partFromPlayerY*: *many*
11		V	V		V	V		*partFromPlayerY*: *few*
12		V		V	V	V		*partFromPlayerY*: *few*

Table 5. *Fuzzy rules for the fullbacks*

Rule	Antecedent								Consequent
	distToPlayer		*distToVanguard*		*distToFullbackX*		*getPast*		
	near	*far*	*near*	*far*	*near*	*far*	*Fals e*	*true*	
1	V		V		V		V		*chasePlayer*: *many*
2	V		V		V			V	*chasePlayer*: *many*
3	V		V			V	V		*chasePlayer*: *many*
4	V		V			V		V	*chasePlayer*: *many*
5	V			V	V		V		*chasePlayer*: *many*
6	V			V	V			V	*chasePlayer*: *many*
7	V			V		V	V		*chasePlayer*: *many*
8	V			V		V		V	*chasePlayer*: *many*
9		V	V		V		V		*chasePlayer*: *few*
10		V	V		V			V	*chasePlayer*: *few*
11		V	V			V	V		*chasePlayer*: *few*
12		V	V			V		V	*chasePlayer*: *few*
13		V		V	V		V		*chasePlayer*: *few*
14		V		V	V			V	*chasePlayer*: *few*
15		V		V		V	V		*chasePlayer*: *few*
16		V		V		V		V	*chasePlayer*: *few*
17	V			V			V		*approachVanguard*: *few*
18	V			V				V	*approachVanguard*: *few*
19		V		V			V		*approachVanguard*: *many*
20		V		V				V	*approachVanguard*: *few*
21	V		V				V		*partFromVanguardY*: *few*
22	V		V					V	*partFromVanguardY*: *few*
23		V	V				V		*partFromVanguardY*: *many*
24		V	V					V	*partFromVanguardY*: *few*
25	V				V		V		*partFromFullbackX*: *few*
26	V				V			V	*partFromFullbackX*: *few*
27		V			V		V		*partFromFullbackX*: *many*
28		V			V			V	*partFromFullbackX*: *many*

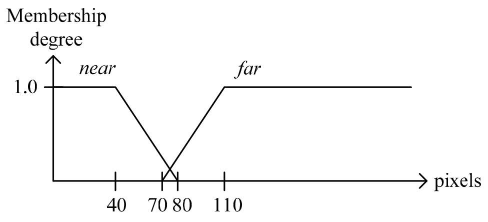


Figure 3. The membership functions of the input variable *distToPlayer*.

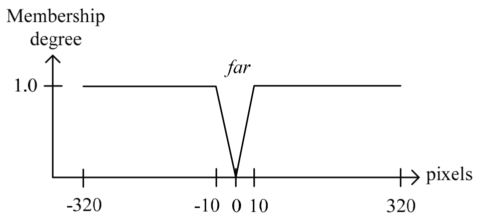


Figure 4. The membership function of the input variable *distToPlayerX*.

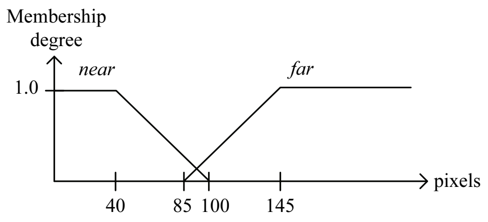


Figure 5. The membership functions of the input variable *distToExitY*.

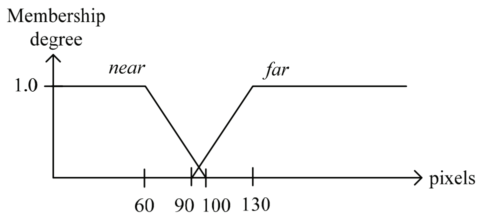


Figure 6. The membership functions of the input variable *distToVanguard*.

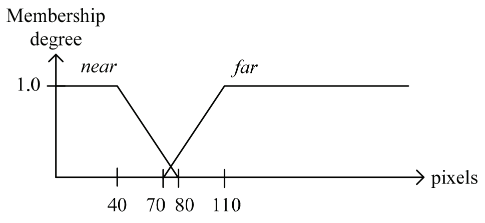


Figure 7. The membership functions of the input variable *distToFullbackX*.

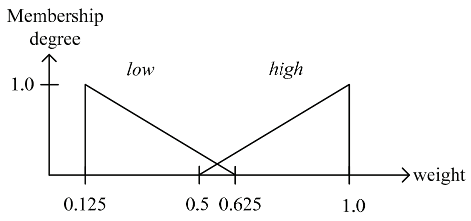


Figure 8. The membership functions of the output variable *chasePlayer*.

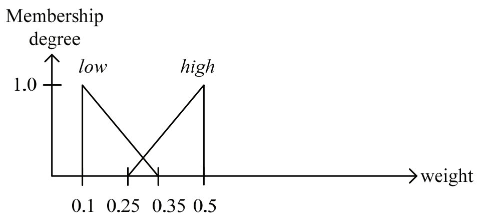


Figure 9. The membership functions of the output variable *partFromPlayerY*.

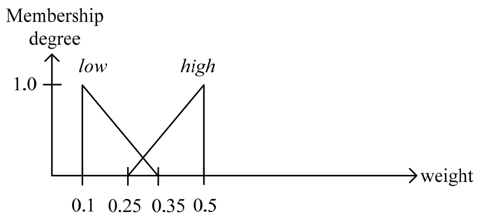


Figure 10. The membership functions of the output variable *approachVanguard*.

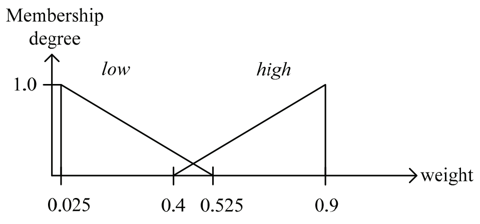


Figure 11. The membership functions of the output variable *partFromVanguardY.*

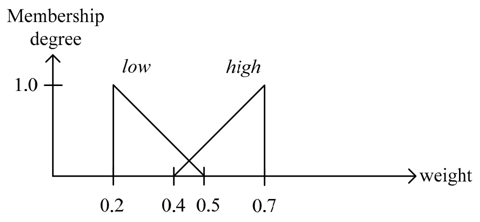


Figure 12. The membership functions of the output variable *partFromFullbackX.*

3.2 Online Rule Enabling/Disabling

In this study an online rule-selection method is used to adapt the Ghosts' tactics to the behavior of the Player. This method is based on a rule enabling/disabling mechanism. As mentioned, the Ghosts are controlled by a fuzzy controller with a rulebase of 40 fuzzy rules. They behave well when all 40 rules are enabled. However, if some rules in the rulebase are disabled, the Ghosts may behave unskillfully and become weaker due to the incompleteness of the strategy. Thus, if the game is hard for the Player, we weaken the Ghosts' skill level by disabling rules in the rulebase. In contrast, if the game is easy for the Player, we increase the Ghosts' level by enabling rules which were disabled previously. Through the rule enabling/disabling mechanism, the Ghosts may keep level with the Player after a few rounds of adaptation. Figure 13 shows the flowchart of the online rule-selection. The details are described in the following.

Whenever a win or loss occurs during gameplay (that is, a round ends), we evaluate whether the difficulty level of the game suits the Player according to the Player's current win/loss rate. The win/loss rate is defined as the ratio of the number of wins for the Player to the number of losses from the beginning of the game. Player's and Ghosts' skill levels are said to be balanced if the Player's win/loss rate achieves a desired level. The value of the desired rate can be preset by the player according to the player's preference for a challenge. If the player prefers a game with more (or less) challenge, they can set a desired rate with the value smaller (or larger) than 50%. The default value of the desired rate is 50% if the player does not set its value. The game is considered to be hard for the Player if the Player loses the game and the current win/loss rate is lower than the desired rate. On the other hand, if the Player wins the game and the current win/loss rate is higher than the desired rate, we consider the game to be easy for the Player.

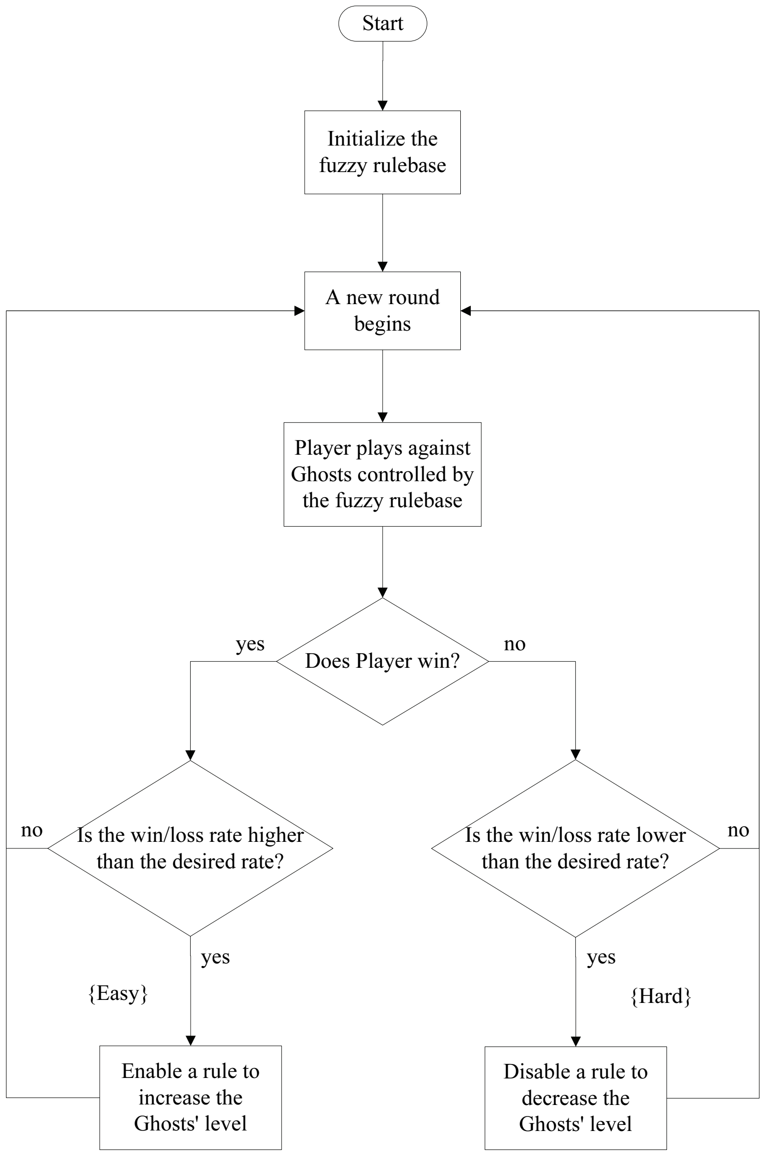

Start
Initialize the fuzzy rulebase
A new round begins
Player plays against Ghosts controlled by the fuzzy rulebase
Does Player win?
yes
no
Is the win/loss rate higher than the desired rate?
Is the win/loss rate lower than the desired rate?
no
no
yes
yes
{Easy}
{Hard}
Enable a rule to increase the Ghosts' level
Disable a rule to decrease the Ghosts' level

Figure 13. Flowchart of online rule enabling/disabling.

When the game is hard for the Player, we disable a rule in the rulebase to decrease the Ghosts' skill level. Which rule is disabled depends on the contribution of the rule. The contribution of a rule is defined as the number of times the rule is adopted in the current round. A rule is said to be adopted if the moving direction derived by the rule is selected as the Ghost's moving direction. The least-contribution rule (a rule with the least contribution but with a contribution larger than zero) is disabled if the game is hard for the Player. Zero-contribution rules are not considered for disabling since the Ghosts' tactics may not be changed if they are disabled or not. The reason why we choose the least-contribution rule but not the most-contribution one is that the Ghosts' skill level may change dramatically if the most-contribution rule is disabled. Since there are four fullbacks but only one vanguard in the Ghosts' team, the probability that the rules for fullbacks are adopted is four times that for the vanguard. Therefore the contributions of the rules for the fullbacks are divided by four while calculating.

If the game is easy for the Player, we enable a rule which is disabled previously in the rulebase to increase the Ghosts' skill level. The procedure of enabling rules is different from that of disabling rules. First, we divide all rules in the rulebase into groups according to the consequences of rules and then calculate the contribution of each group. The contribution of a group is the sum of contributions of all enabled rules in this group. Second, we calculate the sum of input membership degree values for each rule disabled previously. Then, the rule with the highest sum value is enabled. If there are two or more rules with the same sum of input membership degree values, the rule whose group has a lower contribution is prioritized. The reason why we prioritize the least-contribution group but not the most-contribution one is that if the game is easy for the Player, the Ghosts' current strategy might be ineffective, and thus the rules in the most-contribution group might also be ineffective.

4. EXPERIMENTS

In the study, we implemented the proposed approach in Java language and utilized the NRC FuzzyJ Toolkit (2007) (a comprehensive API of fuzzy logic) for fuzzy inference.

In the first set of experiments, there are five Ghosts and one Player in the Dead End game. Initially, the Player has 40 "health points," and the value of the parameter ρ of the CB Player is 2950. The time limit of a round is 20 seconds. That is, if the Player does not reach the exit after playing the game for 20 seconds, the Player loses the game and a new round starts. During gameplay if all rules are disabled or if the outputs of all enabled rules are zero, a specific rule which guides the Ghost directly toward the Player is enabled. The use of the specific rule prevents the Ghosts from being at a standstill.

In the experiment, we ran the game program with different numbers of initially enabled rules: these being 10, 20, and 40. The Ghosts' skill levels corresponding to the three cases can be considered inexperienced, moderate, and experienced respectively. The 10 and 20 initial rules are selected by hand but not randomly because the Ghosts' skill level being controlled by randomly selected rules is unpredictable. The experiments were conducted by playing against three types of fixed-strategy Player, that is, SA, IA, and CB Players. We set three desired win/loss rates: 25%, 50%, and 75%. For each Player, we ran 100 rounds and recorded the win/loss rates every round. The experimental results are shown in Figures 14 to 16.

We ran two other sets of experiments with four and six Ghosts where the number of initial rules was 20 and the desired win/loss rate was set at 50%. The parameter ρ of the CB Player was set to 4000 for the experiments with four Ghosts and 2800 for six Ghosts. The experimental results are shown in Figures 17 and 18.

These experimental results show that the online rule enabling/disabling mechanism can effectively adapt the Ghosts' level to the Player's desired level after several dozen rounds. Furthermore, the desired win/loss rates can be achieved and maintained while the types of Player, the number of initially enabled rules, and the number of Ghosts are varied. These illustrate the efficiency and robustness of the proposed approach.

5. CONCLUSIONS

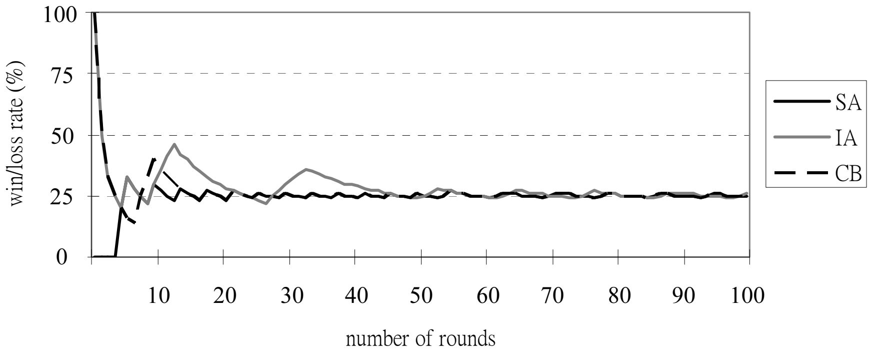
In this study we proposed a fuzzy approach to adapting opponent's tactics to the player based on a rule enabling/disabling mechanism. This approach was tested on a multi-agent predator/prey game called Dead End. The experimental results show that the proposed method can effectively adapt opponents' tactics to different types of fixed-strategy player and achieve the desired win/loss rates within several dozen rounds. The experiments with different parameters of the game prove the robustness of the proposed method. In the future, we will design an offline rule-generation method to create fuzzy rules automatically and then integrate the offline and online learning techniques to more complicated games. The process and results of the research will be of help to game developers in creating adaptive game AI in their games.

ACKNOWLEDGEMENTS

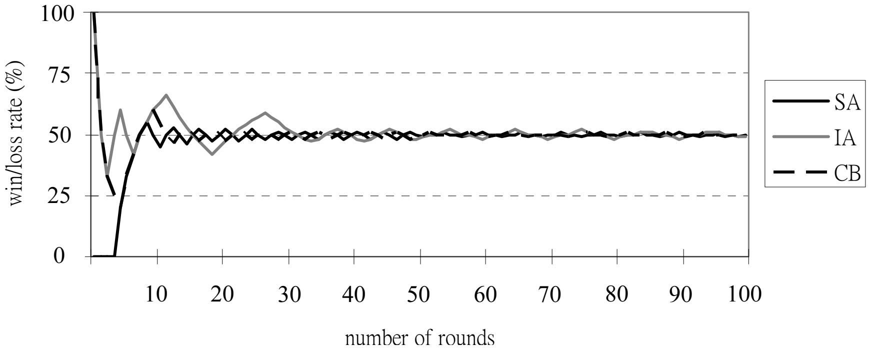
This work was supported by the National Science Council of Taiwan under the grant NSC 96-2221-E-468-010-MY2.

REFERENCES

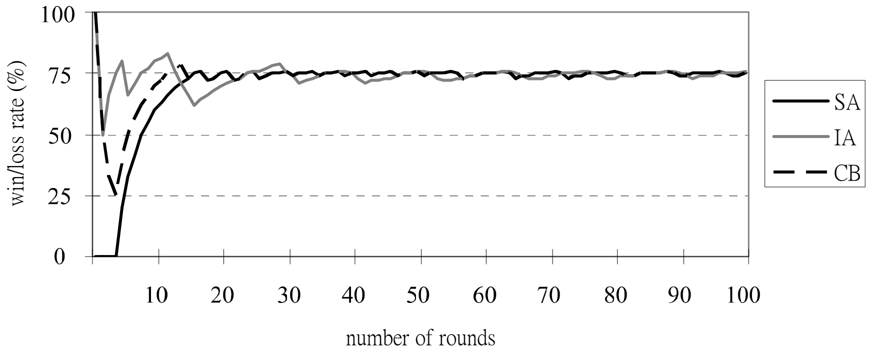
Andrade, G., Ramalho, G., Santana, H., & Corruble, V. (2005). Challenge-sensitive action selection: an application to game balancing. *Proceedings of IEEE/WIC/ACM International Conference on Intelligent Agent Technology*, 194-200.



(a) Desired rate = 25%.

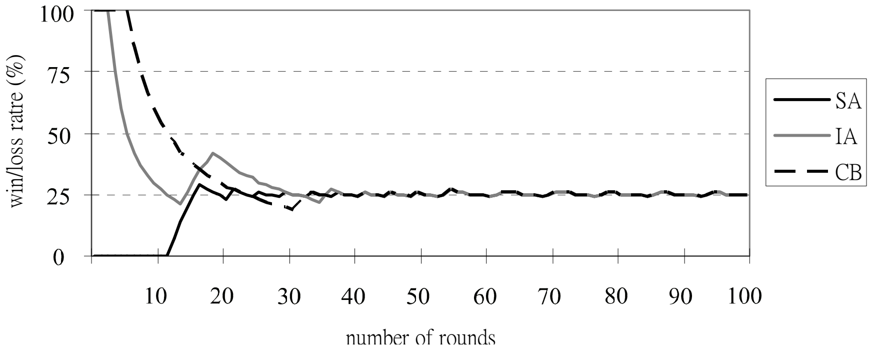


(b) Desired rate = 50%.

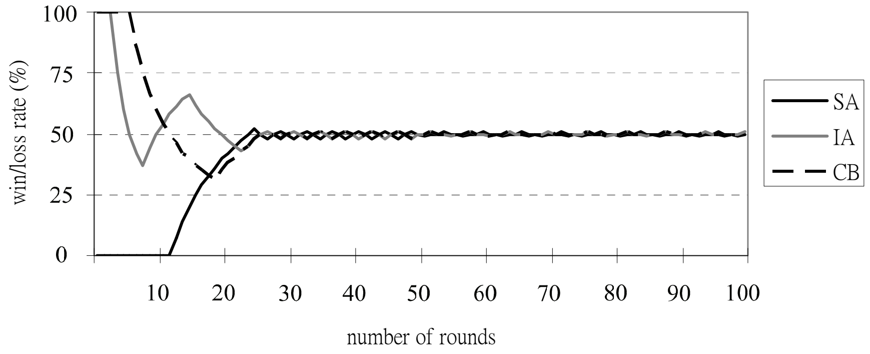


(c) Desired rate = 75%.

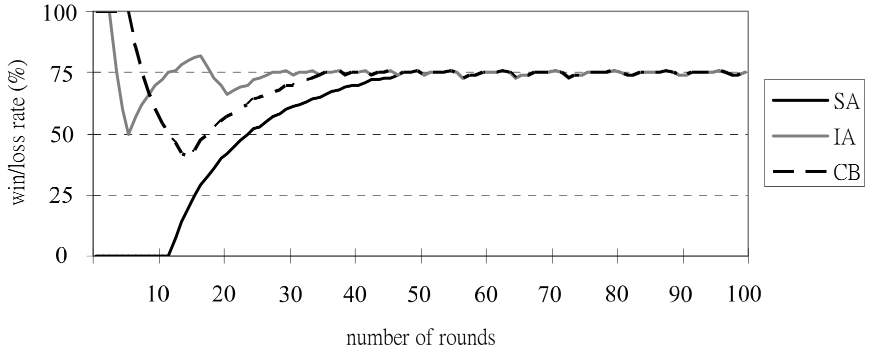
Figure 14. The variations of win/loss rates (number of initially enabled rules: 10).



(a) Desired rate = 25%.

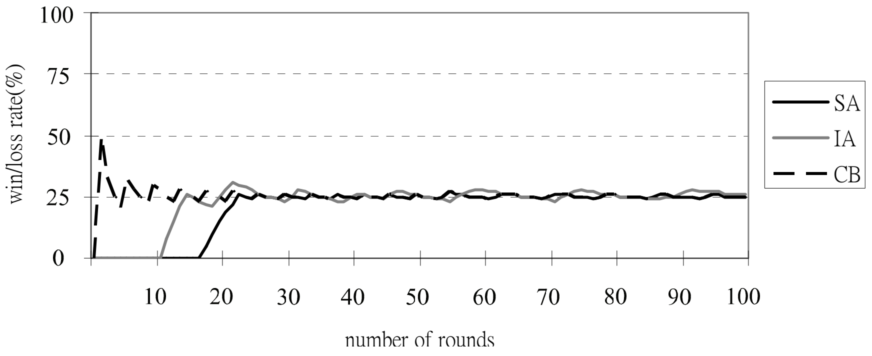


(b) Desired rate = 50%.

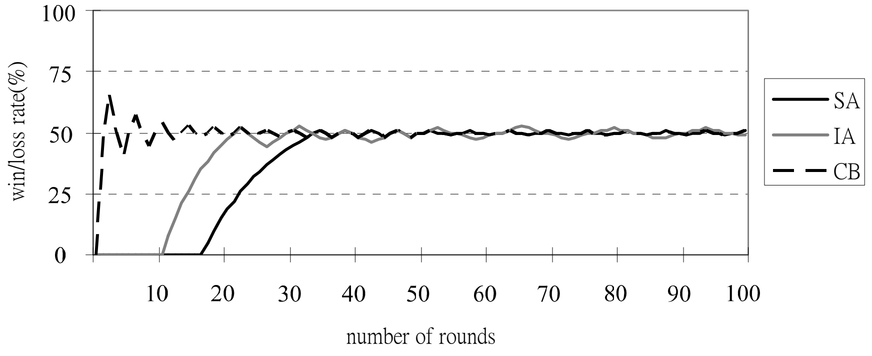


(c) Desired rate = 75%.

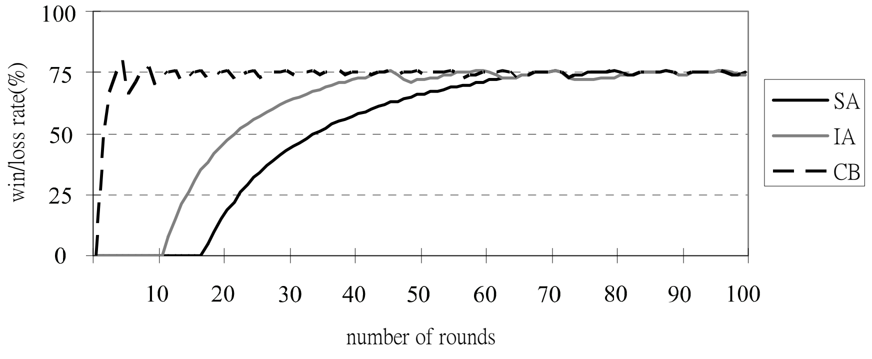
Figure 15. The variations of win/loss rates (number of initially enabled rules: 20).



(a) Desired rate = 25%.



(b) Desired rate = 50%.



(c) Desired rate = 75%.

Figure 16. The variations of win/loss rates (number of initially enabled rules: 40).

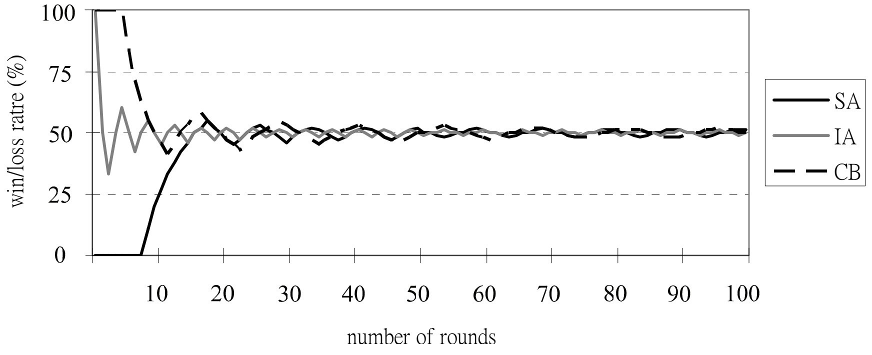


Figure 17. The variations of win/loss rates (number of Ghosts: 4).

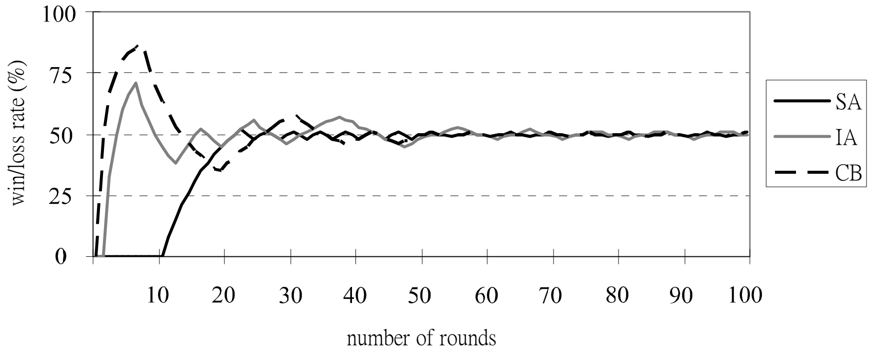


Figure 18. The variations of win/loss rates (number of Ghosts: 6).

Babuska, R. (1998). *Fuzzy Modeling for Control*, Boston, Massachusetts, USA: Kluwer Academic Publishers.

Bourg, D. M., & Seeman, G. (2004). *AI for Game Developers*. Sebastopol, California, USA: O'Reilly Media.

Demasi, P., & Cruz, A. J. de O. (2003). Online coevolution for action games. *International Journal of Intelligent Games and Simulation*, *2*(2), 80-88.

Houlette, R. (2004). Player modeling for adaptive games. In S. Rabin (Ed.), *AI Game Programming Wisdom 2* (pp. 557-566). Hingham, Massachusetts, USA: Charles River Media.

Hunicke, R., & Chapman, V. (2004). AI for dynamic difficulty adjustment in games. *Proceedings of Challenges in Game Artificial Intelligence AAAI Workshop*, 91-96, San Jose, CA, USA.

Laird, J. E., & Lent, M. (2000). Human-level AI's killer application: computer game AI. *Proceedings of AAAI 2000 Fall Symposium on Simulating Human Agents*, 80-87.

Mamdani, E. H. (1977). Application of fuzzy logic to approximate reasoning using linguistic synthesis. *IEEE Transactions on Computers*, *26*(12), 1182-1191.

Manslow, J. (2002). Learning and Adaptation. In S. Rabin (Ed.), *AI Game Programming Wisdom* (pp. 557-566). Hingham, Massachusetts, USA: Charles River Media.

NRC FuzzyJ Toolkit (2007). Retrieved from http://www.iit.nrc.ca/IR_public /fuzzy/fuzzyJToolkit2.html.

Ponsen, M., Muñoz-Avila, H., Spronck P., & Aha, D. W. (2006). Automatically generating game tactics with evolutionary learning. *AI Magazine*, *27*(3), 75-84.

Ponsen, M., Spronck P., Muñoz-Avila, H., & Aha, D. W. (2007). Knowledge acquisition for adaptive game AI. *Science of Computer Programming*, *67*(1), 59-75.

Rabin, S. (2004). Common game AI techniques. In S. Rabin (Ed.), *AI Game Programming Wisdom 2* (pp. 3-14). Hingham, Massachusetts, USA: Charles River Media.

Spronck, P. H. M. (2005). *Adaptive game AI*. Unpublished Ph. D. dissertation, Maastricht University, Maastricht, The Netherlands.

Spronck, P., Sprinkhuizen-Kuyper, I., & Postma, E. (2003). Improving opponent intelligence through offline evolutionary learning. *International Journal of Intelligent Games and Simulation*, *2*(1), 20-27.

Spronck, P., Ponsen, M., Sprinkhuizen-Kuyper, I., & Postma, E. (2006). Adaptive game AI with dynamic scripting. *Machine Learning*, *63*(3), 217-248.

Yannakakis, G. N., Levine, J., & Hallam, J. (2004). An evolutionary approach for interactive computer games. *Proceedings of the 2004 Congress on Evolutionary Computation*, 986-993, Portland, Oregon, USA.

Yannakakis, G. N., & Hallam, J. (2005). A generic approach for generating interesting interactive Pac-Man opponents. *Proceedings of the IEEE Symposium on Computational Intelligence and Games*, 94-101, Essex University, UK.

Wong, K. W. (2008). Adaptive computer game system using artificial neural networks. In M. Ishikawa, K. Doya, H. Miyamoto, & T. Yamakawa (Eds.), *Neural Information Processing*, *Lecture Notes in Computer Science* (pp. 675-682). Berlin, Germany: Springer-Verlag.

Hao-Min Hsieh received his B. S. degree in Electronic Engineering from National Taipei University of Technology, Taipei, ROC in 2004. In 2008, he received the M.S. degree in Information and Design from Asia University, Taichung, ROC. Currently, he is an engineer in KanBo System Information Co., Ltd., Taipei, ROC. His current research is in image processing, pattern recognition, and artificial intelligence.

Ling-Ling Wang received her B. S., M. S., and Ph. D. degrees in Computer Science and Information Engineering from National Chiao Tung University, Hsinchu, Taiwan, ROC in 1984, 1986, and 1990, respectively.

From 1986 to 1987, she was an associate engineer in the System Software Department of ERSO, ITRI, Hsinchu, Taiwan. From 1991 to 1997, she was an associate professor of Computer Science at National Tsing Hua University, Taiwan. Currently, she is a professor of Information Communication at Asia University, Taichung, Taiwan. Her current research is in image processing, pattern recognition, computer vision, and artificial intelligence.