Chapter 14
Optimization of AI Tactic in Action-RPG Game

Kristo Radion Purba

Abstract In an Action RPG game, usually there is one or more player character. Also, there are many enemies and bosses. Player should kill as many as possible to get more experience. A smart AI is needed to increase the game challenge. In this research, a method is proposed to optimize the enemy AI strategy, by implementing enemy units grouping, and attacking in group using hit and run strategy against the player. The grouping is done using clustering, while the behavior picking is using Fuzzy Logic. If the player is approaching a group, most likely the group will retreat and the others start attacking. The units' formation is also maintained using clustering and distance calculation to player character. From the testing, this method can slightly increasing the game difficulty because of the enemies are trickier.

Keywords Artificial intelligence · Fuzzy logic · Clustering · Hit and run · Action RPG game

14.1 Introduction

RPGs have their origins in the paper and pen role-playing games pioneered by Dungeons & Dragons. These were defined games with rules [1]. In computer games, RPG is where player and enemy character have certain statistics, and have quests. Hybrid game genre like Action RPG game becomes common nowadays.

In an Action RPG game, usually there is one or more player character. Also, there are many enemies and some bosses. Player should kill as many as possible to get more experience. The effectiveness of AI in video games depends on how well game characters are able to cooperate and react to the opponent player [2].

K.R. Purba (✉)
Petra Christian University, Surabaya, Indonesia
e-mail: kristo@petra.ac.id; kristoradion@live.com

F. Pasila et al. (eds.), *Proceedings of Second International Conference on Electrical Systems, Technology and Information 2015 (ICESTI 2015)*,
Lecture Notes in Electrical Engineering 365, DOI 10.1007/978-981-287-988-2_14

The enemies' AI can be really smart or dull. A dull AI causes a boring game playing, although it can help the player to reach higher experience because player can kill more enemies easily. But a smart AI is needed to increase the challenge.

This research result can create an AI that is more difficult to fight in action RPG game, thus increasing the game's challenge, and contributes in the field of game artificial intelligence.

14.1.1 Related Works

Hit and run is a common strategy used in a RTS (Real Time Strategy) games. A Research has been done to create an AI that controls enemies to do the hit and run tactic against player [3]. This research is implemented in Warcraft 2 game. In this game, unit can dodge enemy attacks by moving away, so the hit and run is mainly used to dodge. The difference between [3] and this research, is the hit and run will be used to avoid enemy from attacking.

Neural network is also common to optimize the AI reactivity against the player, which for example implemented in [4], the aims of the research are to develop controllers capable of defeating opponents of varying difficulty levels.

A research is also done to optimize the AI units in StarCraft game (a real time strategy game) in which the enemies can do micromanagement, terrain analysis, picking up strategies, and controlling attack timing to do maximum damage on player [5]. This research will be different from [4, 5] because of the type of game, and [4] is using neural network to control units globally.

14.2 Literature Review

In this section, we will explain about K-Means Clustering and Fuzzy Logic, which will be used in this research.

14.2.1 K-Means Clustering

K-Means clustering is an algorithm to classify or to group your objects based on attributes/features into K number of group. The grouping is done by minimizing the sum of squares of distances between data and the corresponding cluster centroid [6]. The distance between data are calculated using Pythagoras formula, that takes x and y as the parameter.

14.2.2 Fuzzy Logic in Game

Fuzzy logic is a generalization of the classical set theory [7]. A research is conducted in [8] that optimizes the behavior of enemy in Pacman game. It uses distance (against player character), pellet/bullet time, and average lifetime as the membership function. It has several fuzzy rules grouped into hunting behavior, defense, shy ghost, and random behavior. In general, the enemy will attack the player if it has a good skill and good pellet time. It will defense if the enemy is not so good, shy ghost if worse, and random behavior is worst.

14.3 Methodology

Enemies will attack player mainly using grouped hit and run strategy. The grouping will be done using K-Means clustering, while the strategy is picked using Fuzzy Logic. The whole process is shown in Fig. 14.1.

14.3.1 Units' Grouping

The distance used in the clustering process is done by considering a unit X and Y position, and unit's strength. Unit's strength is determined by remaining HP (hit points/life), attack, speed, defense and range. The formula is shown in Eq. 14.1.

$$strength = \frac{remaining\ HP}{10} + \left(\frac{speed}{100} x\ attack\right) + defense + \frac{range}{10} \tag{14.1}$$

With the strength and unit's position (X and Y coordinate), we can calculate the unit's distance that will be used in the clustering, the formula is shown in Eq. 14.2.

$$distance = strength + \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{14.2}$$

So the distance term used here is not the actual distance. The process of units grouping is shown in flowchart in Fig. 14.2.

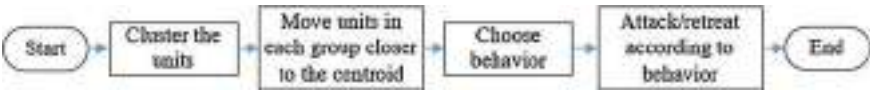

Fig. 14.1 Enemy AI strategy

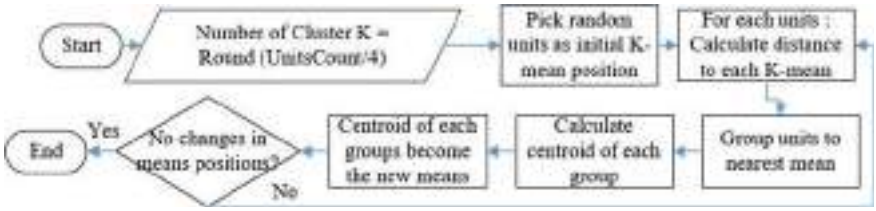

Fig. 14.2 Units' grouping

14.3.2 Picking the Behavior

The behavior is determined by fuzzy membership functions, calculated using average of group units' statistics, i.e.:

a. Group strength
   Each unit in the group has strength (Eq. 14.1). Each group will calculate its average units' strength, to be used in the membership function, shown in Fig. 14.3.
b. Player direction
   Player direction determines whether the player is currently going toward a group or not. This is calculated by the angular difference (degree), illustrated in Fig. 14.4.
   The membership function is shown in Fig. 14.5.
c. Distance to player
   Each group will calculate its centroid distance to the player. The membership function is shown in Fig. 14.6. oR defines whether the group is in the optimum range, based on the mean of attack range of every units. For example, if the mean of range is 350, and distance of group's centroid to player is 350, we say that the oR is 1, calculated using Eq. 14.4.

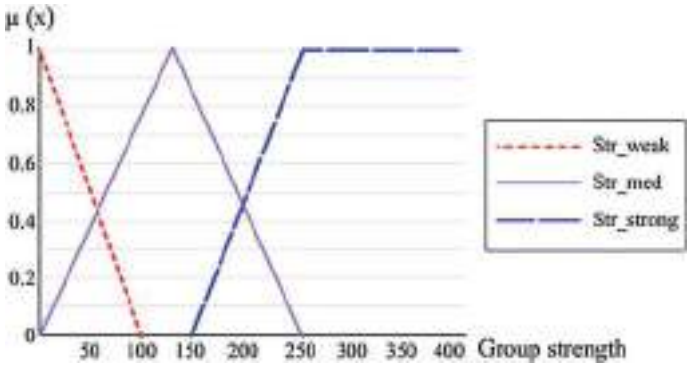

Fig. 14.3 Group strength membership function

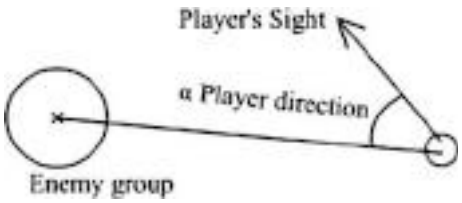

Fig. 14.4 Player direction

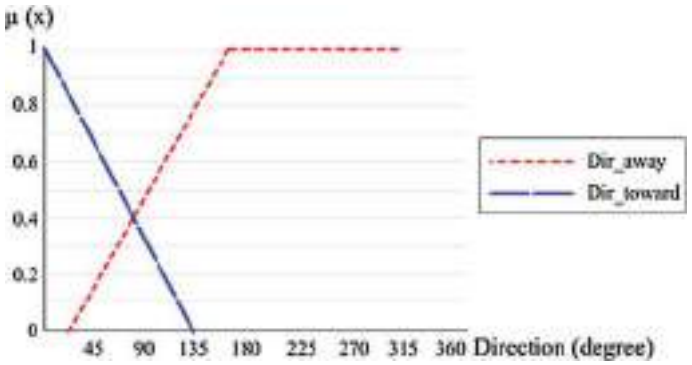

Fig. 14.5 Player direction membership function

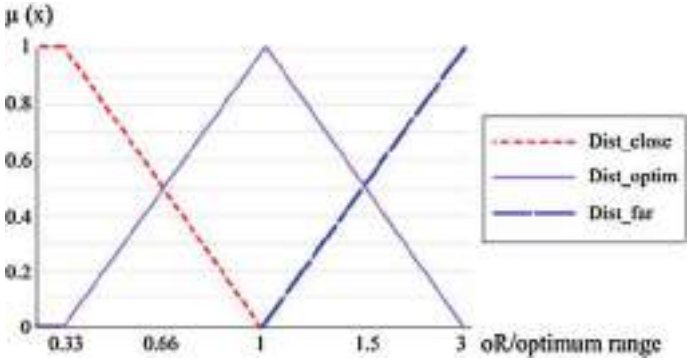

Fig. 14.6 Distance to player membership function

$$oR = distance\,To\,Player \;:\; \frac{\sum_{i=1}^{N} range(i)}{N} \tag{14.4}$$

The behavior that will be used in the defuzzification is divided into 4, i.e. Attack (z = 100), Hold (z = 70), Retreat (z = 40), Retreat far (z = 10). The fuzzy output will

be calculated using the following rules, and will result a floating number between 0 and 100:

- IF str_strong AND dir_away AND dist_far THEN attack
- IF str_strong AND dir_away AND dist_med THEN attack
- IF str_strong AND dir_away AND dist_close THEN hold
- IF str_strong AND dir_toward AND dist_far THEN attack
- IF str_strong AND dir_toward AND dist_med THEN hold
- IF str_strong AND dir_toward AND dist_close THEN hold
- IF str_med AND dir_away AND dist_far THEN attack
- IF str_med AND dir_away AND dist_med THEN attack
- IF str_med AND dir_away AND dist_close THEN hold
- IF str_med AND dir_toward AND dist_far THEN hold
- IF str_med AND dir_toward AND dist_med THEN hold
- IF str_med AND dir_toward AND dist_close THEN hold
- IF str_weak AND dir_away AND dist_far THEN hold
- IF str_weak AND dir_away AND dist_med THEN hold
- IF str_weak AND dir_away AND dist_close THEN retreat
- IF str_weak AND dir_toward AND dist_far THEN retreat
- IF str_weak AND dir_toward AND dist_med THEN retreat far
- IF str_weak AND dir_toward AND dist_close THEN retreat far

The executed behavior will be a crisp behavior, i.e.: Attack (if the fuzzy output is 80–100), hold (60–79), retreat (40–59), retreat far (20–39), random behavior (0–19).

14.4 Testing and Result

This research is tested using a simple battle simulator program by applying generic unit statistics available in most RPG games. In this section we will compare the methods implemented (MI) to random behavior of enemies (RB) in terms of game

Table 14.1 Method testing

Enemies test case	Game Dur. (RB) (s)	Game Dur. (MI) (s)	Life Rem. (RB)	Life Rem. (MI)
5 Archer, 5 Marksman, 5 Catapult	12.5	17.3	720	601
15 Catapult	18.8	20.1	522	355
15 Marksman	14.2	15.7	641	556
15 Archer	16.1	22.4	748	590
10 Marksman, 5 Catapult	13.1	15.4	691	614
10 Archer, 5 Catapult	17.9	21.8	709	652
7 Archer, 8 Marksman	15.7	16.2	662	611
5 Marksman, 10 Catapult	19.2	23.3	531	342
Average	15.9375	19.025	653	540.125

duration (seconds) and how many player's life remains after killing all enemies, displayed in Table 14.1. From Table 14.1, It can be seen that the game duration increases, from average of 15.93 to 19.025 s (19 %). It is also seen that the life remaining is increasing, from average of 653 to 540.12 (−21 %). It means, the enemies are trickier.

14.5 Conclusion

It can be concluded that the enemies grouping, and hit and run strategy works well to create a trickier enemy. It is also seen that the methods implemented (MI) gives the better result than random behavior of enemies in terms of game duration and remaining life after battle. The methods also can be applied to other RPG games using the generic unit's statistics and game rules that are used in this research.

References

1. Janssen, C.: Role-Playing Game (RPG) (2015). Accessed 1 Sept 2015
2. van der Marcel, H., Sander, B., Pieter, S.: Dynamic formations in real-time strategy games. In: IEEE Symposium on Computational Intelligence and Games (2008)
3. Weber, B.: Reactive planning for micromanagement in RTS games. Department of Computer Science, University of California, Santa Cruz (2014)
4. Tong, C.K. et. al.: Game AI generation using evolutionary multi-objective optimization. In: IEEE Congress on Evolutionary Computation (CEC) (2012)
5. Weber, B. et. al.: Building human-level ai for real-time strategy games. In: Proceedings of the AAAI Fall Symposium on Advances in Cognitive Systems (2011)
6. Teknomo, K.: K-Means clustering tutorial (2007)
7. Dernoncourt, F.: Introduction to fuzzy logic. MIT, Cambridge (2013)
8. Adnan, S., Brady, K., Luke, R.: Real-Time game design of Pac-Man using fuzzy logic. Int. Arab J. Inf. Technol. **3**(4), 315–325 (2006)