ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/256496551

# Automated Evaluation for AI Controllers in Tower Defense Game Using Genetic Algorithm

**Chapter** *in* Communications in Computer and Information Science · January 2013

DOI: 10.1007/978-3-642-40567-9_12

CITATIONS
4

READS
1,411

**5 authors**, including:

Tse Guan Tan
University of Malaysia, Kelantan
**35** PUBLICATIONS **90** CITATIONS

SEE PROFILE

Jason Teo
Universiti Malaysia Sabah (UMS)
**129** PUBLICATIONS **852** CITATIONS

SEE PROFILE

**Some of the authors of this publication are also working on these related projects:**



Augmented Reality in Heritage View project



Malaysian medicinal plant project View project

All content following this page was uploaded by Rayner Alfred on 01 June 2014.

The user has requested enhancement of the downloaded file.

# Automated Evaluation for AI Controllers in Tower Defense Game Using Genetic Algorithm

Tan Tse Guan, Yong Yung Nan, Chin Kim On, Jason Teo, and Rayner Alfred

School of Engineering and Information Technology
Universiti Malaysia Sabah,
Jalan UMS, 88400, Kota Kinabalu, Sabah, Malaysia
{tseguantan,yongyn89,kimonchin}@gmail.com,
{jtwteo,ralfred}@ums.edu.my

**Abstract.** This paper presents the research result of implementing evolutionary algorithms towards computational intelligence in Tower Defense game (TD game). TD game is a game where player(s) need to build tower to prevent the creeps from reaching their based. Penalty will be given if player losses any creeps during gameplays. It is a suitable test bed for planning, designing, implementing and testing either new or modified AI techniques due to the complexity and dynamicity of the game. In this research, Genetic Algorithm (GA) will be implemented to the game with two different neural networks: (1) Feed- forward (FFNN) and (2) Elman Recurrent (ERNN) used as tuner of the weights. ANN will determine the placement of the towers and the fitness score will be calculated at the end of each game. As a result, it is proven that the implementation of GA towards FFNN is better compared to GA towards ERNN.

**Keywords:** Genetic Algorithm (GA), Artificial Neural-Network (ANN), Feed-forward Neural Network (FFNN), Elman Recurrent Neural Network (ERNN), Tower Defense game (TD game), Strategy Games, Artificial Intelligence (AI).

## 1 Introduction

Tower defense (TD) is a type of games that required player(s) to build tower in order to prevent certain amounts of enemies from approaching their base. It is known as real-time strategy game or turn-based game and several well-known TD games are such as Plants vs Zombies and TD in Warcraft III [1]. This genre of game provide an important testbed for Artificial Intelligence (AI) research by allowing the testing and comparison of new and experimental approaches on a challenging but well-defined problem.

In the gameplays, there are certain amounts of enemy creeps in each waves moving towards the player creeps. Player(s) should either build more defense towers or up-grade the existing tower in order to kill the enemy creeps. As a return, the player will be awarded with certain amount of resources for each kill. Thus, the player can build more as well as stronger towers to defense for the incoming waves. For a fair to the

S.A. Noah et al. (Eds.): M-CAIT 2013, CCIS 378, pp. 135–146, 2013.
© Springer-Verlag Berlin Heidelberg 2013

gameplays, the strength of the enemy creeps is improved proportionately, in order to ensure the difficulty of the game is consistent with the resources awarded. The games ended when certain amounts of waves are being eliminated or certain number of player's lives is eliminated.

The complexity of the TD games can be dissimilar based on the game design. Some of the TD games allow player(s) to build tower along the path whereas some have limited the tower to be build beside the path [1]. In other case, some TD games limited the number of towers to be built while others are having very limited type of tower to be built [1]. Most of the TD games provide different paths design. For example, Plants vs Zombies consists of five fixed lanes [2]. Field Runner and Sentinel provide several open-ended maps [3], [4]. Player(s) must carefully plan, design and allocate the towers in order to slow down as well as kill the enemy creeps that tend to reach the end point. As such, the dynamicity and complexity in TD games lead to the AI cognition for researchers to plan, design, and testing their novel idea and algorithms [5].

Eventually, there are few interesting issues involved for researching TD games. First is the design of the map. It is not an easy task in designing a map that can attract player(s) to stay longer with the game. It follows with the issue of type of towers and type of creeps design. Each tower must have its own specialty and attributes. Balancing of designing towers plays an important role. Player(s) can easily complete the game level if the towers are equipped with higher strength and ability. Otherwise, player(s) may uninstall the game if they found the game is not beatable after spending few days to the lost games. Another issue regards to the balancing of awards or resources provide to the player(s) after each kill of the enemy. Player(s) may easily lose a game if the resource given is too limited to build or upgrade towers. The last issue would be related to the testing of the gameplays. Most of the TD games have been designed without proper testing procedures on the designed maps. As the manual testing procedures lead to costly and time consuming issues. Hence, it creates an unbeatable situation for the designed map. At last, player(s) uninstall the application after few games. Basically, the inclusion of AI technology to the game can replace the manual testing procedures as well as reduce the cost and time taken for designing the game.

[6] and [7] show AI controllers could be easily generated to provide a variable level of difficulty as an opponent in Real-Time Strategy (RTS) game. [8] and [9] show the inclusion of simple Finite State Machine could reduce the cost taken to test the level of difficulty in Mario Bros and First Person Shooting games. Hence, this motives us to create AI controllers that use Genetic Algorithm (GA) and two different neural networks in this research in order to generate a possible solution to test the usability of the designed TD game's map.

In this study, a modified GA will be implemented with (1) Feed-forward Neural Network (FFNN) and (2) Elman Recurrent Neural Network (ERNN) towards a TD game in order to generate AI controllers that can be used in testing the designed map. A comparison will be carried out in order to determine which neural network will be best suit with GA in generating better controllers.

The remainder of this writing is organized as follows. In section 2, some related works will be discussed. Then, the ANN representation will be included in section 3. It follows with the modified GA discussion in Section 4. The experimental setup will be discussed in Section 5. Then, the experimental results and discussions are included in section 6. Finally, the paper will be concluded in Section 7.

## 2 Related Works

ANN had been applied at different areas as their ability to learn and easy to adapt to different functions and able to make prediction based on the learning data. [10] explained the usage of ANN in gaming. One of the popular ANN implementation game is Reversi [11]. The implementation of ANN enables the development of AI that discovers complex strategies that able to compute with experience human player. There are also studies on other board games such as checkers [12], tic tac toe [13] and five in a row [14]. Researchers generated superior AI controllers that are unbeatable by human expert.

In other case, [15] implements GA with ANN to increase the accuracy of multiple shooting in a ball and balancing skills between characters in duel games. [16] proposed neural-evolutionary model for case-based planning in RTS games. The study shows that ANN in GA algorithms had helps in reducing the time complexity and the characters are able to react based on current data without using heuristics-based algorithm. Besides, [17] suggest the implementation of ANN towards GA in RTS game. The authors also compared different combination of AI methods such as Back-propagation and Radial Basis Networks models.

## 3 Artificial Neural Networks

Artificial Neural Network (ANN) is a mathematical model that tries to stimulate the structure and functionalities of biological neural networks [18]. The starting of ANN is at 1943 when McCulloch and Pills Whor introduced the first neural model [19]. As in biological model, ANN consists of neurons that are connected among each other. The interconnection neurons are generally divided into three layers, except a multi-layer of neural structure is used. The weights used can be either in positive value or negative value. The weights also can be adjusted in order to specific output for specific input. The process of adjusting the weight can be known as learning or training [20].

There are several concerns when implementing ANN: (1) the type of problem to be solved and (2) the type of topology used in the ANN. Till now, there is no evidence that show a single type of ANN that is superior and can be suit with any type of evo- lutionary algorithms for solving all problems. Hence, most of the researches are still conducted in comparing different neural networks topology in different problem solving cases.

In this research, there are two types of neural networks involved (1) FFNN and (2) ERNN. FFNN is differs to ERNN in its learning process although both neural networks involved only three layers in their structure. FFNN involves straight forward propagation in its topology, direct from input layer to hidden and from hidden layer to the output layer.

In ERNN, the propagation process starts from input to hidden layer. Then, a learning rule is applied. After that, there are connections from the hidden layer to the context units fixed in the input layer for the first propagation process. Then, the learning will be continued straight forward from hidden layer to the output layer. The basic topology of FFNN and ERNN is shown in Fig. 1.

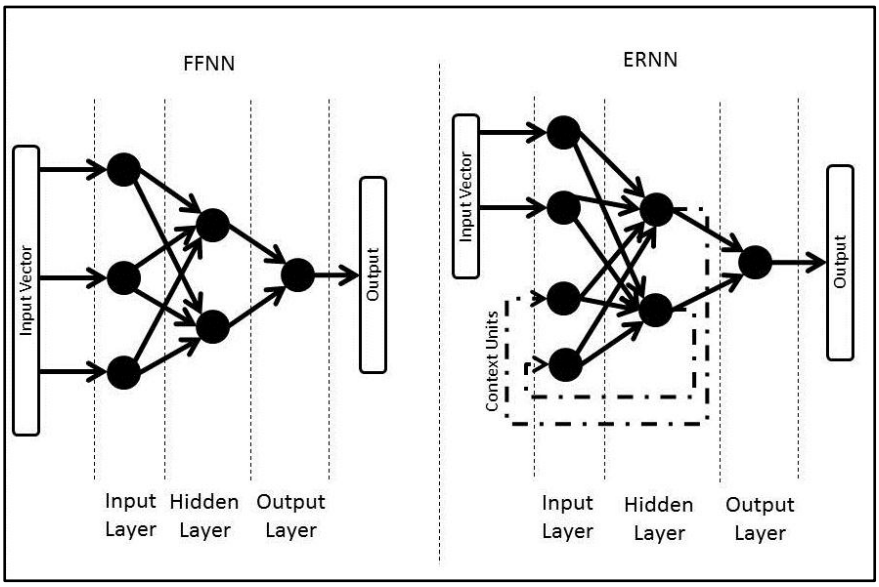


**Fig. 1.** FFNN and ERNN topologies

## 4 Genetic Algorithm

Genetic algorithm (GA) is one of the popular techniques used among all of the evolutionary algorithms. Its self-adaptation and self-organization had make GA a good techniques for solving optimization problems [21]. It had been well implemented at varies of field such as document clustering, tuning ANN weight, gaming and etc [21] and [22]. The data is presented as a fit-length chromosome. Before any of the operator is been triggered, a set of chromosome is randomly generation and it is known as individual in a population. Generally, there are three basic operators in GA that is selection, crossover and mutation.

Selection is a process of selecting fittest individuals in the populations to pass their genes on the next generations. A fitness function is been used to determine the fitness score of each individual before it can be selected. There are lots of selection methods that had been introduced and it can be classified into six categories which are proportionate selection methods, ranking selection, tournament selection, range selection, gender-specific selection and GR based selection [23]. Ranking selection is used in this research.

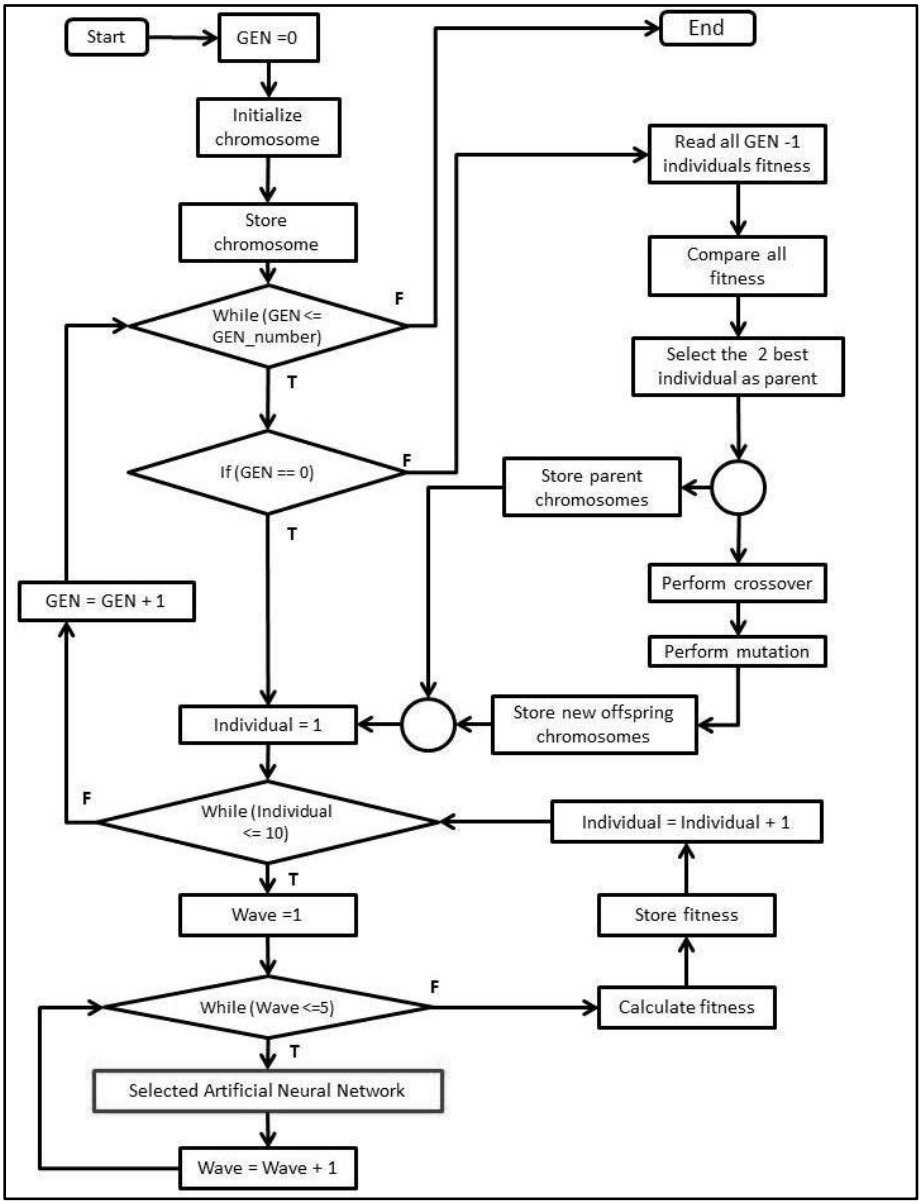


**Fig. 2.** Flow chart of GA toward ANNs

Reproduction started with the implementation of crossover towards two or more parent chromosomes. During this process, two new individuals are formed by swapping a sub-sequence between the parent chromosomes [24]. The length of swapping is determined by the crossover probability rate. Mutation is a process

that changes a gene of a chromosome in order to produce a new offspring. It also depends on the mutation rate to determine the position of gene to be mutated.

In this research, the uniform crossover and uniform mutation methods are used. Uniform crossover will combine part of parent chromosomes to produce new offspring. The part of the parents to be taken for producing new offspring are depends on the crossover rate. In order to generate outperform controller, elitism selection and age-based selection are used separately during the optimization process. Two of the best fittest parent chromosomes are maintained for next generation using elitism method. The rest of the eight chromosomes are selected using age-based selection where the child individuals will replace the parent individual for each generation based to its fitness value.

In this research, GA is implemented with FFNN and ERNN in order to generate the required controllers to the designed TD map. The flow of GA with ANN is as shown in Fig. 2.

## 5 Experimental Setup

Warcraft III World Editor was used to design the TD map. It allows user to customize and build its own custom map. It also enable user to modify object attributes such as appearance and ability of the units, buildings and items. Warcraft III World Editor has its own scripting language which is known as Just Another Scripting Syntax (JASS programming). Due to its simplicity and ease of use, the GA, FFNN and ERNN had been successfully coded in few hours.

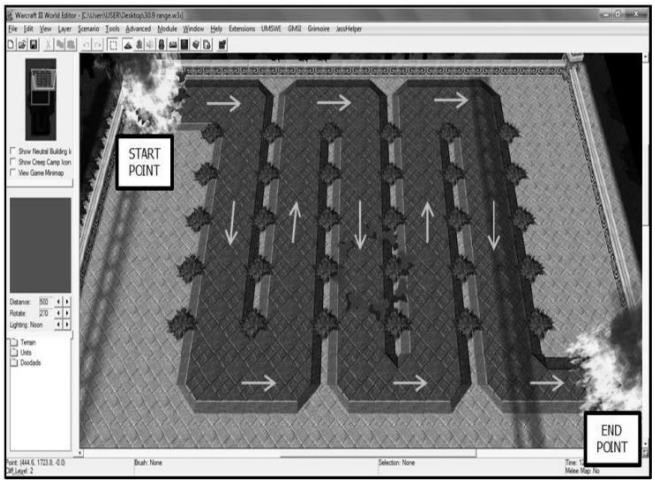


**Fig. 3.** Overview of Designed Tower Defense Map

The designed TD map is shown in Fig. 3. The flow of the creeps' movement is shown in arrow. The enemy path is straight forward and towers are able to be built at specific region which is limited in certain region as shown in Fig. 4. The game is designed for five waves with 20 creeps per wave. The strength and the movement speed of the creeps increase proportionate with the number of waves. Player is given with 10 lives. The game will be ended either all the creeps had been eliminated within five waves or no life of the player is remained.

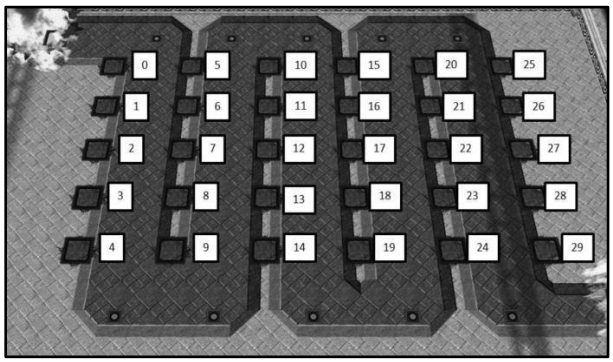


**Fig. 4.** Tower Building Region

Fig. 4 above shows that there are 30 regions that allow towers to be built. The list of tower regions represents the input of ANN. If a tower is built in a region then value 1 will be assigned otherwise, value 0 will be assigned.

The FFNN consists of 31 inputs (30 actual inputs and 1 bias), 16 hidden neurons (15 neurons and 1 bias neuron) and 30 output neurons. ERNN used 31 true input neurons, 31 context input neurons, 16 hidden neurons (15 neurons and 1 bias neuron) and 30 output neurons. The output of ANN shows the next tower to be built by returning 30 neurons which represents the region in the TD game. The activation function used is Binary Sigmoid function.

Eventually, there are ten chromosomes have been generated during early of the experiment. Individual chromosome will be evaluated in each generation. The evaluation function used is referred to the total number of kills minus by 90. The crossover rate used in this research is 0.7 and the mutation rate used is 0.02. There are 150 generations involved in any of the optimization processes.

## 6 Results and Discussions

There are a total of 20 games have been conducted, 10 games for GA and FFNN, another 10 games for GA and ERNN. The results for fitness over generation have been collected and represented using single-line graphs as shown in Fig. 5 and Fig. 6 below.

Fig. 5 and Fig. 6 show the results for one of the TD game using FFNN and ERNN, separately. The generated controllers using FFNN lost the games during early stage of

the experiments. Then, the fitness score increased from -7 to 6 after generation 7th. The fitness scores maintained until generation 12th. Furthermore, the fitness score increased again from generation 13th – 16th, to the maximum fitness of the game. It shows GA has successfully maintained the optimal score until last of the generation with the integration of elitism concept in the GA.

In other case, the generated controllers using ERNN performed slightly worse compared to the FFNN case. The weakest controller reached -62 fitness score during early stage of the experiment. The GA improved the controller capability in generating -25 fitness scores during generations 9th – 13th. Then, the fitness score reached -6 during generation 16th. The fitness score reached 6 after generation 51st. Then, slowly the fitness score reached the 9 after generation 118th.

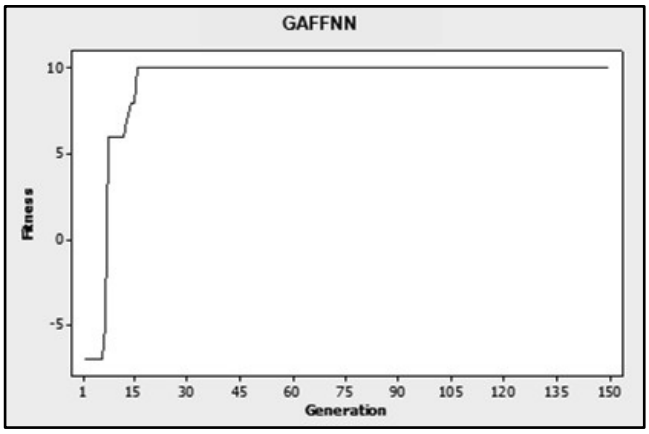


**Fig. 5.** One of the results with GA and FFNN used

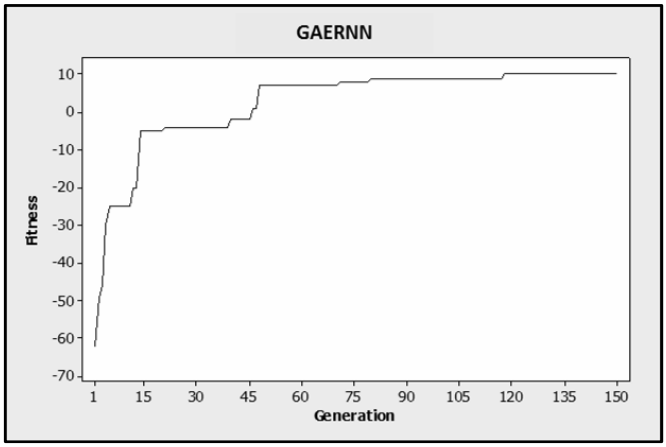


**Fig. 6.** One of the results with GA and ERNN used

Table 1 summarizes the average fitness scores for all the games. The average fitness score is collected by dividing the highest fitness of each generation by 150. Positive value indicates that the controllers managed to win in most of the games and negative value meant the controllers often lost the games.

**Table 1.** Average Fitness Over 150 Generations

| RUN | GAFFNN | GAERNN |
|---|---|---|
| 1 | 8.23 | -1.68 |
| 2 | 8.57 | 8.55 |
| 3 | 7.00 | 7.53 |
| 4 | 8.26 | 6.65 |
| 5 | 8.56 | 4.60 |
| 6 | 8.40 | 9.00 |
| 7 | 7.51 | -1.68 |
| 8 | 6.02 | 9.09 |
| 9 | 7.91 | 8.89 |
| 10 | 7.75 | 7.53 |
| MINIMUM | 6.02 | -1.68 |
| MAXIMUM | 8.57 | 9.09 |
| MEAN | 7.821 | 5.848 |
| STANDARD DEVIATION | 0.805 | 4.193 |

Table 1 shows that the controllers generated from GA with FFNN won for all the 10 games and GA with ERNN only won 8 games. The controllers lost the games on the first run and the seventh run with the ERNN used. The controllers generated with GA and FFNN performed well in attaining of an average 7.821 fitness score which is higher compared to the controllers generated with GA and ERNN that scored an average of 5.848 fitness score. However, the maximum average fitness score obtained with GA and ERNN is higher compared to the GA and FFNN, 9.09 and 8.57 respectively. The tabulated data shows the controllers generated using GA with FFNN scored lower standard deviation compared to the GA with ERNN, 0.805 and 4.193 respectively. This shows the controllers generated using GA with FFNN were consistently performed well during the optimization stage.

Furthermore, tests have been conducted for all the optimized controllers with winning rate involved. Table 2 shows the winning rates of GA with FFNN compared to GA with ERNN. Based on the comparison, GA with FFNN controllers produced higher average winning rate compared to GA with ERNN controllers. The data also shows six out of ten of the controllers generated using GA with FFNN outperformed

the controllers generated using GA with ERNN. Hence, it can be concluded that the controllers generated using GA with FFNN is slightly superior compared to the controllers generated using GA with ERNN.

**Table 2.** Average Winning Rate for GAFFNN and GAERNN

| RUN | GAFFNN | GAERNN |
|---|---|---|
| 1 | **63.07%** | 25.80% |
| 2 | 59.80% | **67.27%** |
| 3 | **59.47%** | 49.53% |
| 4 | **52.47%** | 46.13% |
| 5 | **77.13%** | 34.80% |
| 6 | 60.93% | **69.40%** |
| 7 | **67.33%** | 25.80% |
| 8 | 49.27% | **82.00%** |
| 9 | 51.87% | **77.27%** |
| 10 | **54.67%** | 53.40% |
| Average Winning Rate | **59.60%** | 53.14% |

## 7 Conclusion and Future Works

The experimental results showed the implementation of GA had successfully tuned the weights of the FFNN and ERNN in the TD game. The FFNN and ERNN acted as machine learning to build towers without human intervention. The ANNs used also represented two different individuals in playing the TD games. The individual with FFNN used achieved better average fitness scores compared to the individual with ERNN used.

With referring to the hypothesis of this research, is that possible in reducing the cost and time used to design the TD game's map with AI involvement? Yes. The implementation of GA with FFNN and ERNN showed the designed map provided medium level of difficult to the player(s). It is impossible for a good player(s) to win the game easily. In other sense, it is also impossible for a weak player(s) to often lose the game.

In future works, game score will be considered in the evaluation instead of fitness score. The game score will be increased consistently with the increment of level of the game.

Besides, more types of tower and more types of creeps will be implemented into the map. The controller should not be limited to predict the placement of tower rather to learn how to spend the resources either to build different type of tower or upgrade the existing tower.

## References


1. Avery, P., Togelius, J., Alistar, E.: Computational Intelligence and Tower Defence Games. In: IEEE Congress on Evolutionary Comp. (CEC), pp. 1084–1091 (2011)
2. Plants vs. Zombies - Walkthrough/guide, http://www.ign.com/faqs/2009/plants-vs-zombies-walkthroughguide-992681
3. Fieldrunners 2 – An Absolute Masterpiece in Every Regard, http://applenapps.com/review/fieldrunners-2-an-absolute-masterpiece-in-every-regard
4. 'Sentinel Earth 2 – Earth Defense' – Does It Live Up to the Original?, http://toucharcade.com/2009/07/08/sentinel-2-earth-defense-does-it-live-up-to-the-original/
5. Rummell, P.A.: Adaptive AI to Play Tower Defense Game. In: The 16th International Conference on Computer Games, CGAMES, pp. 38–40. IEEE (2011)
6. Chang, K.T., Chin, K.O., Teo, J., James, M.: Game AI Generation using Evolutionary Multi-Objective Optimization. In: Evolutionary Computation (CEC), pp. 1–8. IEEE (2012)
7. Chang, K.T., Chin, K.O., Teo, J., Chua, B.L.: Automatic Generation of Real Time Strategy Tournament Units using Differential Evolution. In: Proceedings of the IEEE Conference on Sustainable Utilization and Development in Engineering and Technology, pp. 101–106. IEEE (2011)
8. Ng, C.H., Niew, S.H., Chin, K.O., Teo, J.: Infinite Mario Bross AI using genetic algorithm. In: IEEE Conference on Sustainable Utilization and Development in Engineering and Technology (2011 IEEE STUDENT). The University of Nottingham, Malaysia Campus (2011)
9. Chang, K.T., Ong, J.H., Teo, J., Chin, K.O.: The Evolution of Gamebots for 3D First Person Shooter (FPS). In: IEEE the Sixth International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA 2011). Universiti Sains Malaysia, Penang (2011)
10. Bourg, D.M., Seeman, G.: AI for Game Developers. O'Reilly (2004)
11. Moriarty, D., Miikkulainen, R.: Discovering Complex Othello Strategies Through Evolutionary Neural Networks. Connection Science 7, 195–209 (1995)
12. Chellapilla, K., Fogel, D.B.: Evolving Neural Networks to Play Checkers without Relying on Expert Knowledge. IEEE Transactions on Neural Networks 10(6), 1382–1391 (1999)
13. Fogel, D.B.: Using Evolutionary Programming to Create Neural Networks that are Capable of Playing Tic-Tac-Toe. In: IEEE International Conference on Neural Networks (ICNN), vol. 2, pp. 875–880 (1993)
14. Freisleben, B.: A Neural Network that Learns to Play Five-in-a-row. In: Second New Zealand International Two-Stream Conference on Artificial Neural Networks and Expert Systems, pp. 20–23 (1995)
15. Wong, S.K., Fang, S.W.: A Study on Genetic Algorithm and Neural Network for Mini-Games. Journal of Information Science and Engineering 28, 145–159 (2012)
16. Niu, B., Wang, H., Ng, P.H.F., Shiu, S.C.K.: A Neural-Evolutionary Model for Case-Based Planning in Real Time Strategy Games. In: Chien, B.-C., Hong, T.-P., Chen, S.-M., Ali, M. (eds.) IEA/AIE 2009. LNCS(LNAI), vol. 5579, pp. 291–300. Springer, Heidelberg (2009)
17. Wang, H., Ng, P.H.F., Niue, B., Shiu, S.C.K.: Case Learning and Indexing in Real Time Strategy Games. In: Fifth International Conference on Natural Computation, pp. 100–104. IEEE (2009)

18. Krenker, A., Bester, J., Kos, A.: Introduction to the Artificial Neural Networks. In: Artificial Neural Networks – Methodology Advances and Biomedical Applications. InTech, pp. 3–18 (2011)
19. McCulloch, W., Pitts, W.: A Logical Calculus of the Ideas Immanent in Nervous Activity. Bulletin of Mathematical Biophysics 5, 115–133 (1943)
20. Carlos, G.: Artificial Neural Networks for Beginners. In: Artificial Neural Networks – Methodology Advances and Biomedical Applications. InTech (2011)
21. Xiang, W.J., Liu, H., Sun, Y.H., Su, X.N.: Application of Genetic Algorithm in Document Clustering. In: International Conference on Information Technology and Computer Science, pp. 145–148. IEEE (2009)
22. Mitchell, M.: An Introduction to Genetic Algorithms. MIT Press (1999)
23. Sivaraj, R.: A Review of Selection Methods in Genetic Algorithm. International Journal of Engineering Science and Technology (IJEST) 3(5), 3792–3797 (2011)
24. Talib, S.H.: An Introduction to Evolutionary Computation (1998)

View publication stats