2022 International Conference on Artificial Intelligence of Things (ICAIoT) | 979-8-3503-9676-8/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICAIoT57170.2022.10121899

Artificial Intelligence, Game Theory, Programming Used Languages and Platforms, Game Types and Training Methods

Fatih Şahin
Department of Computer Engineering
Topkapi University
Istanbul, Turkey
fatihsahin@topkapi.edu.tr

Abstract—**Artificial Intelligence is one of the most sought-after branches of computer science today, and the branch of artificial intelligence tries to make machines act intelligently, enabling them to cope with more and more diverse problems on their own. Undoubtedly, one of the sectors that has the highest share in the development of artificial intelligence today is the computer games sector. The game development process has a more detailed structure every day. Artificial intelligence applications, which are a comprehensive information technology, have also affected game technologies closely. It is one of the most impressive works that are increasing day by day. Today's computer games can even beat world chess champions and checkers. In this article, which started with this motivation, first of all, the definition of artificial intelligence was examined, and auxiliary calculation and training methods for artificial intelligence were introduced.**

Keywords- Artificial Intelligence, Machine Learning, Game Intelligence, Artificial Intelligence Education

I. INTRODUCTION

A. *Definition of Artificial Intelligence*

In the simplest terms, artificial intelligence (AI) refers to systems or machines that mimic human intelligence to perform tasks and can iteratively improve themselves based on the information they collect.

B. *Turing Test*

Alan Turing was the first to define the concept of a machine being intelligent. According to Turing, for a machine to be considered intelligent; He must pass a three-player quiz, one himself and two people. One of the human players becomes the jury and interacts with both players individually without knowing which is human and which is computer. If the jury is unable to distinguish which of the players, he interacts with is human and which is machine, then the machine has passed the test and can be qualified as an intelligent machine.

This exam is called the "Turing Exam" in the artificial intelligence literature. There are different opinions as to whether the Turing test can give a complete definition of an intelligent machine. Because the Turing machine compares a machine with a human instead of starting from a definition of intelligence, it is obvious that it is a test of human resemblance, not real intelligence. However, Turing gave the first concrete definition, and it is a test known to all artificial intelligence scientific circles.

C. *Game Artificial Intelligence*

Game artificial intelligence, in its most general definition, is the integration of a kind of artificial intelligence into games. While the term game AI may be used to meet a relatively limited function in-game, such as collision detection, it may also refer to the presence of a more complex implementation, such as the implementation of a neural network algorithm in the game.

Shannon's (1950) article, game AI researchers began developing computer games that could play against humans. Deep Fritz, Chinook, Othello, Scrabble game programs can be given as examples. Although each of these programs is powerful, it is not perfect, and the opponent must make a mistake to win. Today, game programs have developed rapidly in terms of graphics, but it is still early to say this for game artificial intelligence.

In fact, the concept of game artificial intelligence is not a clearly defined concept. Game developers and academics understand different things under the concept of game artificial intelligence. When academics say game artificial intelligence, they understand the intelligent behavior of the character in the game. In contrast, game developers understand techniques that can be universally used as game AI, such as pathfinding, animation. There are significant differences between natural intelligence (the human brain) and artificial intelligence. First, a human can play different games intelligently using his brain. For a computer to play different games intelligently, artificial intelligence must also be produced for each game. Games may have common features. Allis (1994) classified the games according to their common features. However, these common features did not save us from writing a separate computer code for each game. Second, none of us can expect the person who plays the game to play perfectly, there is the possibility of making mistakes. In turn, everyone expects the computer to play perfectly. In fact, for the computer to play perfectly, the game must be solvable, free of ambiguities and NP-hard (non-A non- polynomial -hard) algorithm must exist. For example, the game of checkers is a solvable game.

Over the years, the increase in processing power has not only made the games run faster but has also begun to change the format of the games. While there were games based only on moves (chess, checkers or simple intelligence games) and low-quality two-dimensional characters' movements according to a uniform logic in the past, today's computers display the interaction of dozens of characters with each other in real-time and almost lifelike environments. . The increase in games and game companies' day by day has increased the desire of users for more realistic graphics and smarter game characters. At this point, artificial intelligence, which is one of the most difficult and impressive aspects of game programming, comes into play.
try to make the characters interacting with the player or the environment in which the game take place, as much as possible, to resemble the real human, community, and world (or chemical-physical-biologically meaningful environment) living environment. In another sense, the ability of the computer to respond to a talented player with the same skill, to set up plans or traps similar to the plans of the player, to put the player in a difficult situation and defeat him, when necessary, determines the quality of artificial intelligence in the game program.

II. GAMES

A. Game Theory

Min-max theorem, which was proved by Hungarian-born mathematician John Von Neumann. Neumann, who made important contributions to many inventions and research from the atomic bomb to the computer, wrote the applications of game theory in economics in 1943 with the economist Oscar Morgenstern. Theory of Games and economist He brought it up with the book "Behaviour". Also, a young mathematician, John Forbes Nash, generalized Neumann's approach and proved the equilibrium theorem in games with many players in 1950, and was awarded the Nobel Prize in 1994 for this discovery.

B. Game Types

TABLE I. CLASSIFICATION OF COMPUTER GAMES BY CHARACTERISTICS

	Decision Based (Deterministic)	Based on Luck (Chance)
Fully Knowledgeable (Perfect information)	Chess Checkers Go Othello Sudoku (Diamond)	Backgammon Monopoly
Incomplete Information (Imperfect information)	Battleships Three Stones Game (Tic-tac-toe) Minesweeper	Bridge Poker Word Puzzle (Scrabble) Nuclear War

In a complete information game, players know their personal strategies and outcome functions like other players. In addition, each player knows that the other players have all the information.

Particularly informed (incomplete in information) games, players know the rules of the game and their personal preferences, but not the sequel functions of other players.
A complete information game is a type of game in which players successively choose strategies and are aware of what the other players' choice is. Chess is an example of this game (see Table I).

Incomplete (incomplete information) game is where players act without knowing another player's move, only guessing what their other player will do.

Sudoku game is a single player game. Fully knowledgeable as the player knows their personal strategies and sequel functions (complete information) is in the game group. In addition, since there is absolutely no luck factor in the Sudoku game, the game is completely focused on deciding according to the position. Therefore, the Sudoku game is also included in the decision-based game group.

C. Classic Games

- "Mastermind": Based on guessing 4 randomly selected colors in order. Genetic algorithms, "simulated annealing" and artificial neural network approaches are solved separately with each of them.

- "Connect Four", "Go" There are examples of AI by college student Victor Allis playing the games Moku" and "Qubic" (4x4x4 "TicTacToe"). They play excellently in tournament conditions.
- "Othello": Named after Shakespeare's Othello, the AI of this game ("Reversi") developed by Michael Buro defeated the world champion in August 1997. An artificial neural network is used as its basis.
- AIs capable of playing the game of Go have been developed, there are no examples that can defeat humans. The samples on the site of the American Go community can easily be eaten after hard work.
- There are many examples of AI playing chess. Many of us have played with them. Those who know chess really well can easily beat them. Most commercial chess programs use iterative search trees and games of good players.
- The biggest shortcoming of the AI example called Poki playing the poker game is the bluffing event. The computer cannot do the bluff that people can easily use when face to face. Probability statistics at the University of Alberta includes estimation based on competitor modeling techniques, learning based on observation and statistical analysis.

III. DECISION MECHANISMS AND PROGRAMMING TECHNIQUES USED IN GAME AI

A. *Finite State Machines (Finite state Machines FSM)*

Finite state machines are a decision-making method in which there are a finite number of states and the transitions between these states are bound by certain rules. Finite state machines basically consist of four components:

- Situation
- Initial Status
- Alphabet
- Transition Function

In finite state machines, the transition between states is made according to the result of the transition function whose domain is alphabet.

It would not be wrong to say that it is the AI technology with the most common application area. FSM can be defined simply as a hierarchy of logical rules and states created for each possible situation. Each state has a specific entry and response, and the layout does not have a choice of which states to visit. To make things clear with an example, let's take a look at the logic-controlled gates used on the USS Enterprise aircraft carrier. In normal situations, when a staff member approaches a door, the door opens, and when he moves away, it closes. In case of red alarm, these doors can only be used by authorized personnel. A few simple states (authority and affinity) control a certain number of transactions (opening and closing). Order is “if the door is not open, it cannot be closed.” with logical facts such as

B. *Fuzzy State Machines (Fuzzy State Machine FuSM)*

FSM in many respects, but there is a clear difference. In FuSM, instead of giving a series of entries for specific responses, these entries are directed to the pool of "probable responses" created, and each of these potential responses has a weight in the pool they are in. Once these response pools have been created, designers’ resort to various methods to select responses to be given. The weights can be used as simple probabilities, or the weights can be summed up and compared with some limit values to see which response will be triggered. The result is a state machine that can generate quite different and appropriate responses to the given series of alerts. Although the response of the FSM-controlled gates in the USS Enterprise example is precise, considering the example of a spaceship controlled by the FuSM, flexibility and selectivity in the ship's responses to changing combat conditions emerge. In the condition of an enemy ship approaching, there is more than one reaction that our ship can give, and in this reaction pool, where the condition is directed, winning profitability is given depending on the characteristics of the situation. If our ship's power and weapon status is bad or damaged, it will immediately choose the way away, otherwise it will go into full attack (see Fig. 1).

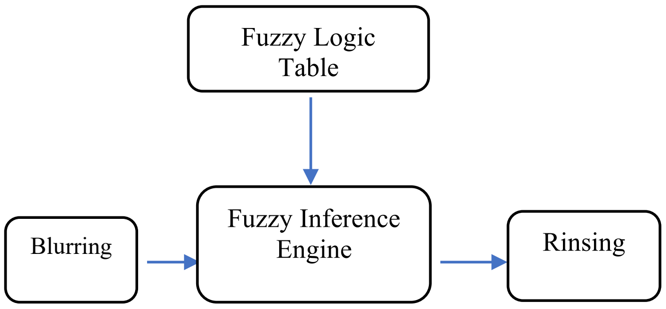

Figure 1. Tsukamoto Model Fuzzy System Chart

C. *Game Trees*

In Game Theory, the game tree is the list of all possible moves from the very beginning of the game or from a current state of the game, with the current state of the game to the top, and all the valid moves in the game ordered from top to bottom in order of occurrence. The important thing here is that the move below is shaped according to the state of the move above. Here, all moves are shaped depending on the next move. The game tree of many zero-sum games can be extracted. Theoretically, game trees of games such as chess, checkers, go can be extracted (see Fig. 2).

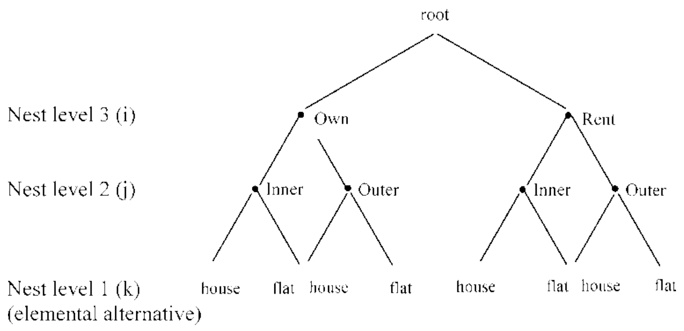

Figure 2. shows the structure of the game tree.

Game trees have a particularly important place in writing computer programs for zero-sum games. Computers make decisions using game trees. All possible moves are taken out and a decision is made according to the situation.

D. *MinMax Method*

One of the algorithms used in Artificial Intelligence applications in computer games is the MiniMax method, which comes from decision theory. The function is primarily intended for use in turn-based, full information, and zero-sum games, but can be modified for use in other genres. The method can be defined as minimizing the highest possible harm.

There are two players in the game, Min and Max, and the computer tries to get the best game situation for itself by subtracting all the possibilities for these two players to play. The algorithm is based on a search tree (Fig. 3). Each node of the search tree holds a state of the game (or a game for short). The child nodes of a node hold the possible states after that state. Levels marked with Min and Max indicate the player who will play at that level.

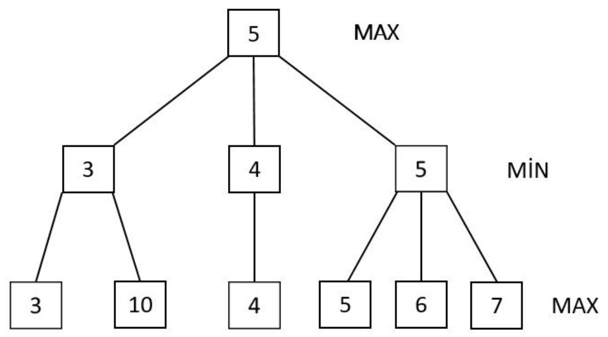

Figure 3. MiniMax Search Tree

The tree is calculated in depth (depth-first) from the current game state to the final game state, and when a game-ending move is found, the game result is interpreted according to Max. Then the nodes in the branches of the tree are filled from the bottom up with the calculated values. Nodes showing situations where Max plays get the highest value of their own children, while nodes showing situations where Min plays get the lowest value of their children. As a result, the values in the search tree indicate how well a game result that situation is for Max. Max will want to choose the highest value of these nodes when making a move. On the other hand, Min will make a move that will try to worsen Max's situation (i.e. improve his own), which will make it difficult for Max to choose a move.

E. α – β (Alpha / Beta Cuts) Pruning Method

The MiniMax method uses the Alpha/Beta interrupts technique because all possible moves in the game are kept in a tree structure in memory, and the search is made over the whole tree, increasing the processing load, especially in games with a large move space such as chess. Alpha-Beta interrupts are a method that can be used on trees similar to Fig. 4.

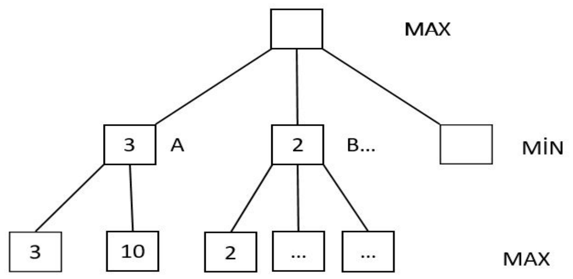

Figure 4. A Search Tree Suitable for Alpha-Beta Interruption

Since nodes A and B are in MIN order, if a number less than 3, which is the value of A, is chosen as the value of B, then it is certain that B will not pass A in the next step, and there is no need to waste any more time with node B. Now when it is known that the value of B's first child is 2, the smallest value of 2 will be B's value if the other children's values are greater than 2, and if the other children's values are less than 2, the value of B will be less than 2. There is no need to worry about B as it will take a small value (and therefore cannot exceed A's value of 3 during MAX).
Or in short;
Alpha = best known MAX value and Beta = best known MIN value,
1. On MAX nodes, compare the value of the previous path with the beta value before starting any path. Skip this node if the value is greater than Beta.
compare the value of the previous path with the alpha value before starting to trace any path. Skip this node if the value is less than Alpha.

Alpha-Beta interrupts can result in significant speed gains in the MiniMax algorithm without degrading the AI quality, but the scale of this acceleration depends on the structure of the search tree. Alpha-beta interrupts do not contribute to performance if the values of the MAX nodes are in ascending order, or if the values of the MIN nodes are in ascending order. The basis of this approach is the use of two variables, α and β, suggested by J.McCarty.

F. Neural Networks

Artificial Neural Networks are an interactive, learning-based prediction method inspired by the nervous systems of vertebrate animals. Nerve cells (nodes) that have good communication with each other transmit the data from the previous node to the next node by performing simple operations (using the weight multiplier and the node weight additive). The result obtained at the last node is compared with the training data. If the result needs to be improved, the ANN parameters are updated. The output of a node depends on the inputs it receives, and the weights assigned to each input.

G. Fuzzy Neural Networks

Adaptive Network Fuzzy Inference Systems (ANFIS – Adaptive Neuro-Fuzzy Inference System), or in short Fuzzy Neural Networks, is a form of standard Fuzzy Inference Systems based on the Artificial Neural Network Model. The system is defined with a network structure that will generate the outputs of an initially defined fuzzy Sugeno or Tsukamoto model, and the membership function parameters of the fuzzy model are updated with the training methods used in artificial neural networks.

H. Genetic Algorithms

Genetic algorithms are adaptive stochastic optimization algorithms with search and optimization. Genetic algorithms were first used by Holland in 1975.
The basic idea is to choose the best among the given solutions by operating a simple example of natural selection. The first stage consists of mutation or random modification of sample solutions. The second step is a selection step and is

usually accompanied by an evaluation of a fitness function that emulates natural selection. These two steps are iterated until the optimal solution is found.

İ. *Artificial Life*

Y-Life technology is the general name given to scientific studies for recreating the biological system in computer environment. Y-Life tries to revive biological life by building systems that act like living organisms. Depending on the design, complex rules can be coded, genetic algorithms and other methods can be used in this technology, or they can all be integrated into the same system.
This issue, which came to the fore with games such as Sims, attracted a lot of attention. Generally, real life has been tried to be modeled on a computer using genetic algorithms. In artificial life, the characters acquire distinctive characteristics according to the environment with the interventions of the player, and the learning process takes place by gaining experience.

In order to make this process easier, the environment in which the game takes place has been designed accordingly. Objects in the environment have the ability to broadcast. For example, the kitchen emits a signal of saturation in a certain area. The character who enters this field receives the food signal emitted by the refrigerator and the scenario proceeds in this way. After the other living spaces are designed in this way, the life model as a whole can be mentioned.

J. *Unit Behaviors*

Unit AI means programming real-life features to add realism to the game or provide combat. An unmoving and unresponsive sentry in a game can remain extremely contrived. If this guard is made to look in different directions at different times or if his posture is changed, an air of realism can suddenly be captured. Now, let's add a second guard walking on a predetermined line to our imaginary scene, and this guard will come and stand in front of the stationary one and try to talk to him. In this case, we can say that we have succeeded in increasing the realism many times over.

Unit AI is divided into two categories, reactive and natural movements. The actions of units in response to changes in their environment are reactive movements. If an enemy soldier starts running by shooting at you as soon as he sees you, this is due to his reaction upon seeing you. Reactive behaviors develop depending on the senses of the character. The character's field of view and distance, his sense of hearing and even his ability to smell are all factors used to determine changes in his environment. In order to capture more realistic approaches, different reaction levels are created for different sensations in today's games. If an enemy soldier enters the direct line of sight of a unit, it will immediately go into alert mode and react to counter the enemy. If the same soldier does not see the enemy and only hears the sound of gunfire, he can go on a search in the direction of the sound. Again, in a situation where he does not see the enemy, this time he will try to find the necessary position to ambush him if he hears footsteps.

Natural behaviors, on the other hand, are not dependent on environmental effects and changes, but represent the normal actions of the character and are one of the most important elements that add vitality to the game world. If every character encountered in the game is waiting for you to come and kill him or talk to him, or worse, he is walking around pointlessly, this is not a convincing situation for the player. In order to prevent such situations, the designers try to provide the richness of the game by applying different methods such as the character commuting on a determined line or wandering randomly in a determined area, and two characters having a conversation with each other

IV. EDUCATIONAL METHODS

The term learning is defined in psychology as the change of an entity's behavior due to repeated experiences with or in a given situation. In Artificial Intelligence, machine learning (or training) can be defined as increasing the performance of an AI system over time.

A. *Error-Guided Learning*

Error-driven learning is to ensure that the machine first makes mistakes when solving a problem and does not repeat those mistakes in subsequent steps. This method of learning shows similarities with the way people learns. A machine can be programmed in a similar way, just as a person learns from a mistake he made once and does not make the same mistake or similar mistakes while solving the same or similar problems. At first glance, " Supervised " in artificial neural networks Although it may seem similar to "Training”, it is actually similar to "Supervised Education" because there is no outside training data as in supervised education. The system has to extract its own error functions.

B. *Tutorial by Trainer*

Instructor-based learning is a type of learning in which the machine's behavior is transferred to the machine by an expert. For example, when solving an equation with one unknown, the instructor can help the machine to solve the problem by saying "move the unknowns to the left of the equation and the knowns to the right". The main problem here is the way the trainer deals with the machine.

C. *Discovery Learning*

Discovery learning is a little different from other learning methods. The purpose of learning is not to achieve a goal, but only to gain more knowledge and increase the richness of concepts in the database. All the machine does is look for sources of interesting information from which it can learn something new. The end of the training is not the point where there is nothing left to be discovered, but the point where you have enough information about the given tasks.
The machine puts the given tasks in an "interesting" order and does not look at some of the information in some tasks

as they are below the required interest. Interestingness is also stated as a mathematical function and is one of the biggest problems of discovery learning because a poorly chosen interest function can cause the machine to ignore very necessary information.

V. PROGRAMMING LANGUAGES AND PLATFORMS

A. *Game programming*

C++ language is generally used to write games. Today's evolving technology environment and increasing graphical advances require very complex and unified systems to write a commercial game. Simply, a very good knowledge of physics and mathematics is required to write a commercial game. At the stage of game programming, subjects such as movements and positions, 3D environments, motion and position information at every moment, vectors in physics, motion and momentum need to be known very well. In terms of mathematics, modeling of figures, modeling of characters, modeling of the environment and operations to be performed in this environment and new positions require advanced knowledge of many subjects such as trigonometry, complex numbers, derivatives, integrals and differential equations in terms of mathematics. These are only required at the initial stage.

One of the first things to be done to program a game is to have an advanced knowledge of a design program such as 3dmax for environment modeling. All characters and places in the game must be drawn in detail. It takes a very long time to learn such programs and is often not successful. Many of those who set out to learn do not even come halfway. The fact that the most known 4-5 game companies in the world today confirms this.

Another stage for game programming is to have a more than advanced command of an object-oriented language. As of today, the language that has been accepted in this sense in the world is C ++. Because there are thousands of libraries written for this language. Among the programming languages, it has the status of using Latin in the medical literature. Along with C++, it is necessary to know the use of the directX library, which is the graphic library of the Microsoft company, at an advanced level.

Finally, after all this, you need to know artificial intelligence in order to write a game. You should also be familiar with artificial neural networks. About 3000 games are released commercially in the world every year, but even 100 of them hardly find a place in the market.

B. *Development Packages*

AI.Implant: Developed by Biographic Technologies. It advertises itself as "the world's first real-time 3D AI SDK". It is middleware (library) that allows developers to add advanced AI features quickly and efficiently. It works very well as a plugin in Maya.

GALib: Contains standard genetic algorithm implementations. It is the basic C ++ library. It was developed at MIT.

Memetic: Contains behavioral tools for game development. It is a powerful application development interface. never winter It was written using NWScript, which he used for his game Nights Renderware AI: This product offers developers a layered architecture:

• Base layer: Offers Architecture, Core C, C++ classes. Basic scheduling, queuing, agent creation and deletion etc.

• Management layer: Services, basic “do I own?” and “how do I do it?” includes functions. Pathfinding, field of view, dynamic object, etc.

• Actions layer: Agents provide the underlying agent framework.

• Thinking layer: Brains are the layer that allows each agent to think according to the given inputs. Inputs are taken from the service layer. Most user-defined information is found here.

VI. CONCLUSIONS AND COMMENT

Artificial intelligence is a collection of optimizations that enable today's computers to deal with more and more diverse problems. Artificial intelligence, in which complex world problems are handled mathematically and made programmable on a computer, is one of the most open branches of computer science today and is increasing its prevalence.

Artificial intelligence in computer games can basically be defined as developing programs that can amuse people at a level that can fight them because playing games that they have never won can get boring after a while. For this reason, game manufacturers often choose to change the artificial intelligence programs they write for their games to certain variables, at the request of the user, in a way that makes the artificial intelligence blunt or intelligent. Of course, a computer game is not just about artificial intelligence programming, but the part of game programming that is wanted to be revealed in this study is artificial intelligence. Therefore, in this article, some of the methods that can be used in artificial intelligence are listed.

REFERENCES

[1] Sercan Türkmen, Hilmi Yalın Mungan and Selma Tekir, “Artificial Intelligence Adaptation to a Platform Game Based on User Performance”, CEUR Workshop Proceedings, 2015.

[2] Uğur Erkan, “Artificial Intelligence Aided Simulation of Trading Game”, Gaziosmanpaşa University Graduate School of Science Thesis, 2012.

[3] uğba Karaoğlu, “A New Approach Model and Application of the Sudoku Problem to Algorithm Design Supported by Artificial Intelligence”, Istanbul Aydın University Graduate School of Science Thesis, 2012.

[4] Gökalp Gürbüzer, “Game Development with Artificial Intelligence Methods”, Kocaeli University Graduate School of Natural and Applied Sciences Master's Thesis, 2008.

[5] Wikipedia, “Applications of Artificial Intelligence”, (Online) https://en.wikipedia.org/wiki/Applications_of_artificial_intelligence, 2008.

[6] Tahir Emre Kalaycı, “Artificial Intelligence in Games”, (Online) https://docplayer.biz.tr/12727537-Oyunlarda-yapay-zeka-tahir-emre-kalayci.html

[7] Şahin Emrah, “Development of a Distributed Turn-Based Game Artificial Intelligence”, Ankara University Scientific Research Projects, 2010.

[8] Yunus Sarıca, “Game Leveling with Artificial Intelligence”, Pamukkale University Institute of Science Master Thesis, 2019.

[9] [9] Michele Pirovano, “The use of fuzzy logic for Artificial Intelligence in Games”, Department of Computer Science, University of Milan, 2012.

[10] ThinkTech, “Deep Differences: Artificial Intelligence, Machine Learning and Deep Learning”, (online) https://thinktech.stm.com.tr/uploads/docs/1608899913_stm-blog-deep-differences-artificial-intelligence- machine-learning-and-deep-learning.pdf?

[11] Bourg, David M. and Seemann, Glenn, " AI for Game Developers”, O'Reilly Media, 2004.

[12] Judith Yates and Daniel F. Mackay “Discrete Choice Modelling of Urban Housing Markets: A Critical Review and an Application” Urban Studies, Vol. 43, No. 3, 559–581, March 2006