# **Abstract**

Table of Contents

[**Abstract** 1](#_Toc498770109)

[**Chapter 1: Introduction** 3](#_Toc498770110)

[**Chapter 2: Machine Learning** 4](#_Toc498770111)

[**2.1 Introduction** 4](#_Toc498770112)

[**2.2 Types of Machine Learning** 4](#_Toc498770113)

[**2.2.1 Supervised Learning** 4](#_Toc498770114)

[**2.2.2 Unsupervised Learning** 5](#_Toc498770115)

[**2.2.3 Reinforcement Learning** 5](#_Toc498770116)

[**2.3 Machine Learning methods** 5](#_Toc498770117)

[**2.3.1 Decision Tree Learning** 5](#_Toc498770118)

[**2.3.2 Artificial Neural Networks (ANN)** 5](#_Toc498770119)

[**2.3.3 Deep Learning (DL)** 6](#_Toc498770120)

[**2.3.4 Genetic Algorithms (GA)** 6](#_Toc498770121)

[**2.3.5 Temporal Difference (TD)** 6](#_Toc498770122)

[**2.4 Machine learning examples in everyday life** 6](#_Toc498770123)

[**Chapter 3: Artificial Intelligence** 7](#_Toc498770124)

[**3.1 Introduction** 7](#_Toc498770125)

[**3.2 Artificial Intelligence Approaches** 7](#_Toc498770126)

[**3.3 Artificial Intelligence in Games** 8](#_Toc498770127)

[**3.4 Artificial Intelligence Accomplishments** 9](#_Toc498770128)

[**3.4.1 IBM 701 Electronic Data Processing Machine – The First Checker-playing Program** 9](#_Toc498770129)

[**3.4.2 Deep Blue - The First Chess Computer** 9](#_Toc498770130)

[**3.4.3 AlphaGo - The First Computer Go** 9](#_Toc498770131)

[**Chapter 4: Chinese Chess** 11](#_Toc498770132)

[**4.1 Introduction** 11](#_Toc498770133)

[**4.2 Chinese Chess Piece Names** 12](#_Toc498770134)

[**4.3 Chinese Chess Rules** 13](#_Toc498770135)

[**4.2 Techniques used in modern Chess programs** 13](#_Toc498770136)

[**4.2.1 Game Tree** 13](#_Toc498770137)

[**4.2.1 Search Algorithm** 14](#_Toc498770138)

[**4.2.2 Minimax Algorithm** 14](#_Toc498770139)

[**4.2.3 Alpha-Beta Pruning (ABS)** 15](#_Toc498770140)

[**4.2.4 Evaluation Functions** 15](#_Toc498770141)

[**Chapter 5: Methodology and Design** 16](#_Toc498770142)

[**5.1 Review of research undertaken** 16](#_Toc498770143)

[**5.2 Research Question** 16](#_Toc498770144)

[**5.3 Methodology** 16](#_Toc498770145)

[**5.4 Vision Document** 17](#_Toc498770146)

[**5.4.1 Purpose** 17](#_Toc498770147)

[**5.4.2 Scope** 17](#_Toc498770148)

[**5.4.3 Stakeholder and User Description** 17](#_Toc498770149)

[**5.4.4 Requirements** 17](#_Toc498770150)

[**5.4.5 Risk Analysis** 18](#_Toc498770151)

[**5.4.6 Solutions (Tools and Platform)** 19](#_Toc498770152)

[**5.5 Functional Specification** 19](#_Toc498770153)

[**5.5.1 User Stories** 19](#_Toc498770154)

[**5.5.2 Project Plan** 20](#_Toc498770155)

[**5.6 Design** 20](#_Toc498770156)

[**5.6.1 Architecture Diagram** 20](#_Toc498770157)

[**5.6.2 Use Cases** 20](#_Toc498770158)

[**5.6.3 Key Class Diagrams** 20](#_Toc498770159)

[**5.6.4 VOPC** 20](#_Toc498770160)

[**5.6.5 Sequence Diagram** 20](#_Toc498770161)

[**5.6.6 Screen Layout** 20](#_Toc498770162)

[**5.7 Prototype** 21](#_Toc498770163)

[**Chapter 6: Implementation** 22](#_Toc498770164)

[References 22](#_Toc498770165)

# **Chapter 1: Introduction**

# **Chapter 2: Machine Learning**

## **2.1 Introduction**

Machine learning (ML), a subfield of artificial intelligence (AI), is concerned with the construction of computer programs that automatically improve with experience (Mitchell, 1997). It is about making computer modify or adapt their actions so that these actions get more accurate. The aims of machine learning is to establish procedures, known as learning algorithm, that allow a machine to learn from examples presented to it and to ultimately let machines teach themselves (Bengio, 2016). The idea of a computer machine and intelligence began in the 1950 when Alan Turing questioned “Can machine think?” (Turing, 1950). Machine learning has led to the development of machines that can learn intelligent behaviour directly from data rather than being explicitly programmed to display such behaviour (Schölkopf, 2015).

Machine learning methods are sometimes called subsymbolic because no symbols or symbolic manipulation are involved (Marsland, 2015).

## **2.2 Types of Machine Learning**

### **2.2.1 Supervised Learning**

Supervised learning is the most widely used technique in machine learning. In machine learning, systems are trained to infer pattern from observational data. A particularly simple type of pattern, a mapping between input and output can be learned through supervised learning. It involves given training data which consists of example inputs and the corresponding outputs and comes up with a model to explain those data (Schölkopf, 2015). Decision tree learning is a type of supervised learning algorithm.

### **2.2.2 Unsupervised Learning**

The goal of unsupervised learning is to identify and explore regularities and dependencies in data. Like supervised learning, unsupervised learning proceeds from a finite sample of training data, meaning that the learned concepts are stochastic variables depending on the particular training set (Hansen & Larsen, 1996).

### **2.2.3 Reinforcement Learning**

Reinforcement learning is another type of machine learning technique. The training information provided to the learning system by the environment is in the form of scalar reinforcement signals that measure how well the system operates. The learner is not told which actions to take but and must discover which actions to be the best by trying each action in turn (Maglogiannis, et al., 2007).

## **2.3 Machine Learning methods**

### **2.3.1 Decision Tree Learning**

Decision tree learning is a logic-based type of supervised learning algorithm. It is a tree that classify instances by sorting them based on feature values. Each node in a tree represents a feature in an instance to be classified, and each branch represents a value that a node can assume. Instances are classified starting at the root node and sorted based on their feature values (Maglogiannis, et al., 2007).

### **2.3.2 Artificial Neural Networks (ANN)**

Artificial neural networks (ANN) is a perception-based type of supervised learning algorithm from biology, designed to simulate the way in which the human brain processes information. ANNs gather their knowledge by detecting the patterns and relationships in data (Agatonovic-Kustrin & Beresford, 2000). The basic unit of ANN is neuron. An artificial neuron corresponds to a nonlinear threshold apparatus with multiples inputs and a single output (Wang & Li, 2008).



Fig 2.1: Structure of an Artificial neural networks (ANN) (Agatonovic-Kustrin & Beresford, 2000)

### **2.3.3 Deep Learning (DL)**

Deep learning is a form of machine learning that allows computers to learn from experience. It refers to the simulation of networks of neurons that gradually learn to recognised images, understand speech or even make decisions on their own (Bengio, 2016).

### **2.3.4 Genetic Algorithms (GA)**

Genetic algorithm is a global optimization algorithm that introduces the idea of biology genetics, enhancing the adaptability of each individual by the genetic operation mechanism such as selection, crossover, thus simulating the evolution process of natural selection (Wang & Li, 2008). The aim of Genetic algorithm is to obtain the approximate optimal solution from all feasible solutions under genetic operators (Li , et al., 2016).

### **2.3.5 Temporal Difference (TD)**

Temporal difference method introduces the idea that learning is based on difference between temporally successive predictions. The goal of learning in temporal difference method is to make the learner’s current prediction for the current input pattern more closely match the next prediction at the next time step (Tesauro, 1995).

## **2.4 Machine learning examples in everyday life**

**Siri:** A voice recognition system uses machine learning to help answer questions and make recommendations.

**Facebook:** A social media service that uses image recognition algorithm to recognised people in photo.

**AVG:** An antivirus software that uses machine learning to detect malicious software on computer device.

**Google:** A search engine that uses machine learning algorithm to improve search results and search suggestions.

**PayPal:** An online payment platform that uses machine learning algorithm to detect fraud.

# **Chapter 3: Artificial Intelligence**

## **3.1 Introduction**

Artificial Intelligence (AI), also known as Machine Intelligence (MI), is the area of computer science dedicated to produce software capable of sophisticated, intelligent, computations similar to those that the human brain routinely performs (Agatonovic-Kustrin & Beresford, 2000). There are four approaches to define artificial intelligence system.

|  |  |
| --- | --- |
| Systems that think like humans | Systems that think rationally |
| Systems that act like humans | Systems that act rationally |

Fig 3.1 Approaches that define Artificial Intelligent systems (Russell & Norvig, 2010)

## **3.2 Artificial Intelligence Approaches**

**3.2.1 The Turing Test Approach**

Alan Turing proposed The Turing Test (Turing, 1950) which provided a satisfactory operational definition of intelligence. He stated to test a machine’s ability to exhibit intelligent behaviour, it would need to possess the following capabilities:

* **Natural Language** Processing to enable it to communicate in English.
* **Knowledge Representation** to store what it knows or hears.
* **Automated Reasoning** to use the stored information to answer questions and to draw new conclusions.
* **Machine Learning** to adapt to new circumstances and to detect and extrapolate pattern.
* **Computer Vision** to perceive objects.
* **Robotics** to manipulate and move objects.

(Russell & Norvig, 2010)

**3.2.2 The Cognitive Model Approach**

This is the interdisciplinary field of study that combines computer model artificial intelligence and psychology techniques to construct precise and testable theories of the human mind. This study is based on experimental of humans or animals. (Russell & Norvig, 2010)

**3.2.3 The Law of Thought Approach**

This approach uses logic to solve any solvable problem described in logical notation. It emphasis on correct inferences. (Russell & Norvig, 2010)

**3.2.4 The Rational Agent Approach**

The idea of this approach is to create a rational agent able to operate autonomously, adapt to changes, create and pursue goals and acts so that it can achieve the best outcome. (Russell & Norvig, 2010)

## **3.3 Artificial Intelligence in Games**

Artificial Intelligence can play multiple different roles in gaming. The goals of Artificial Intelligence in games is to simulate intelligence behaviour, providing the player with a challenge, a challenge that the player can overcome.

**3.3.1 Decision Making**

This is the core concept of AI. To be able to execute these choice, the system needs to be able to affect the entities using the AI system. When the AI makes a decision, that decision is then broadcast t the entities involved. This approach works well in real-time strategy games, where AI is concerned with the big picture. (Kehoe, 2015)

**3.3.2 Basic Perceptions**

To allow AI to make meaningful decision, it needs to perceive its environment. In a simple system, this perception can be checking the position of the player entity. As system become more demanding, entities need to identify key features of the game world and it is up to the designer and developer to come up with a way to identify these key features important to the system. (Kehoe, 2015)

**3.3.3 Prediction**

This is the ability to effectively anticipate an opponent’s next move. A basic method of this is to allow the system to keep track of past decisions and evaluate their success. This can then be evaluated to determine the success of previous actions and whether a change in tactics is required. (Kehoe, 2015)

## **3.4 Artificial Intelligence Accomplishments**

### **3.4.1 IBM 701 Electronic Data Processing Machine – The First Checker-playing Program**

Built by Arthur Samuel, IBM 701 was also known as the first self-learning program. It implements an optimization to search trees called alpha-beta pruning. The program recorded reward for specific move, allowing the application to learn with each game played. Samuel also programmed it to play itself, increasing the rate the program learn and play (Jones, 2007).

### **3.4.2 Deep Blue - The First Chess Computer**

Deep Blue, developed by IBM, was the first chess-playing supercomputer that beat against world chess champion Garry Kasparov after a six game match in 1997 (Hsu, 1999). Deep Blue had an impact on computing in many different industries. It was programmed to solve the complex strategic game of chess, thus allowing researchers to explore and understand the limits of massively parallel processing. This research gave developers deeper understanding into ways they could design a computer to tackle complex problems in other fields, using deep knowledge to analyse a higher number of possible solutions (IBM , n.d.).

### **3.4.3 AlphaGo - The First Computer Go**

AlphaGo, developed by Google DeepMind, is the first Computer Go program to defeat a professional human Go player in 2016. This was considered to be a significant milestone in the quest of artificial intelligence (AI). AlphaGo uses combination of advanced search tree with deep neural networks. These neural networks record the description of the Go board as an input and process it through a number of different network layers containing millions of neuron-like connections. One neural network, the policy network, selects the next move to play while another neural network, the value network, predicts the winner of the game (DeepMind Technologies Limited, n.d.). These deep neural networks are trained by combination of supervised learning from human expert games, and reinforcement learning from games of self-play (Silver, et al., 2016).

# **Chapter 4: Chinese Chess**

## **4.1 Introduction**

Chinese Chess also known as Xiang Qi (Xiang means elephant and Qi means chess) or The Elephant Game, is a popular two player strategy board game in China. Similar to Chess, the aim of the game is to capture the opponent’s General piece or putting the opponent’s General piece in checkmate to win. The Chinese chess is played on a 9x10 board (Fig 4.1), the board contains a river in the middle that divides the board between two sides. Each side also contains a 3x3 intersection at its centre with diagonal lines which represent the imperial palace. Chinese Chess is played with 32 pieces, each player has 16 pieces (one King, two Advisors, two Elephants, two Rooks, two Horses, two Cannons and five Pawns).

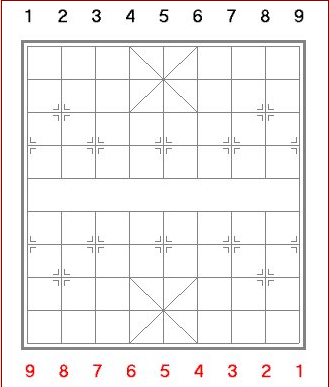


Fig 4.1 Chinese Chess Board (Donnelly, 2014)

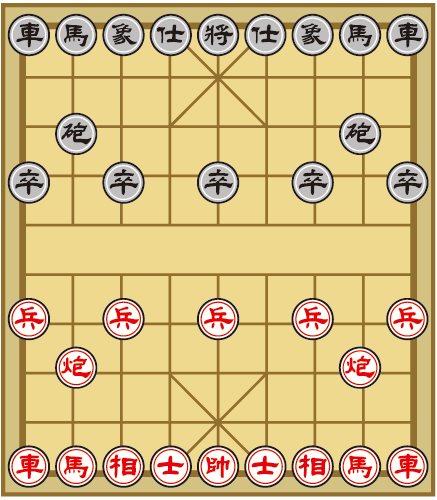


Fig 4.2 Chinese Chess Board with Pieces (Yellow Mountain Imports, n.d.)

## **4.2 Chinese Chess Piece Names**

|  |  |
| --- | --- |
| Piece Name in Chinese | Piece Name in English |
| 帥 | King |
| 士 | Guard |
| 相 | Bishop |
| 車 | Knight |
| 馬 | Rook |
| 炮 | Cannon |
| 兵 | Pawn |

## **4.3 Chinese Chess Rules**

//working on a clearer version of this, will transfer this to here when done

## **4.2 Techniques used in modern Chess programs**

The structure of a chess program is based on the following.

* **Representation:** This refers to the way in which the board, the pieces and the movements of the pieces are described numerically so they can be used appropriately by the search and evaluation algorithms.
* **Search for moves:** This is done by using algorithm that searches for an adequate move that retains a list of all the legal movements.
* **Position Evaluation:** The evaluation module is called on continuously by the search algorithm to pass on the scores of each position.

(Rasskin-Gutman, 2009)

### **4.2.1 Game Tree**

Computer chess games are usually represented using game trees. A game tree is an instance of a tree in which the root node represent possible states of the game or positions, and arcs in the tree represent moves. Leaf nodes in the tree represent final states, where the game has been won, lost or tied (Coppin, 2004).

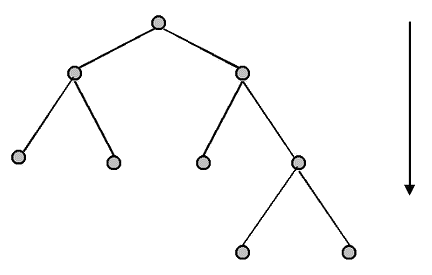


Fig 4.1 Example of a Game Tree (Ross, 2014)

### **4.2.1 Search Algorithm**

Search algorithm works by considering various possible action sequences. The possible action sequences starts at the initial state from a search tree with the initial state at the root. The branches are actions and the nodes correspond to states in the state space of the problem (Russell & Norvig, 2010).

### **4.2.2 Minimax Algorithm**

The minimax algorithm is the most widely used game tree search algorithm. The aim of the algorithm is to maximise the lowest possible score that can be achieved. When evaluating game trees, it is usual to assume that the computer is attempting to maximise some score that the opponent is trying to minimise. This score is usually the result of the evaluation function for a given position, so high positive score means a good position for the computer and a high negative score means a good position for the opponent (Coppin, 2004).

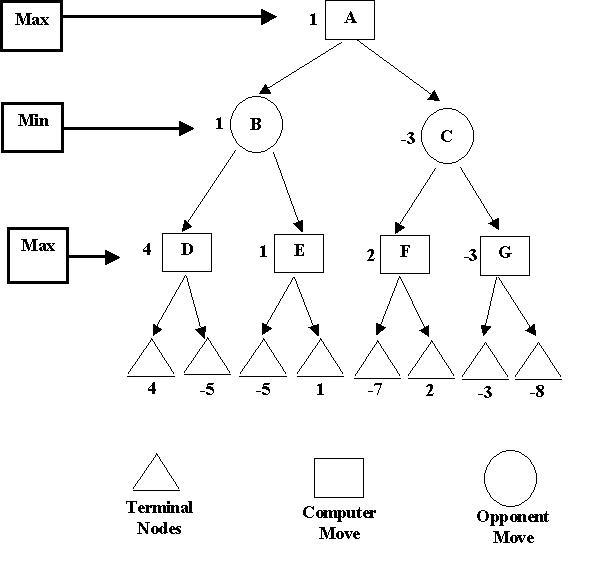


Fig 4.2 Example of Game Tree using Minimax Algorithm (Kendall, 2001)

### **4.2.3 Alpha-Beta Pruning (ABS)**

Alpha-Beta Pruning is an adversarial search algorithm which helps to decrease the complexity of nodes in the game tree thus allowing a deeper and efficient search to be performed. This is done by computing the correct minimax decision without looking at every node in the game tree (Russell & Norvig, 2010).

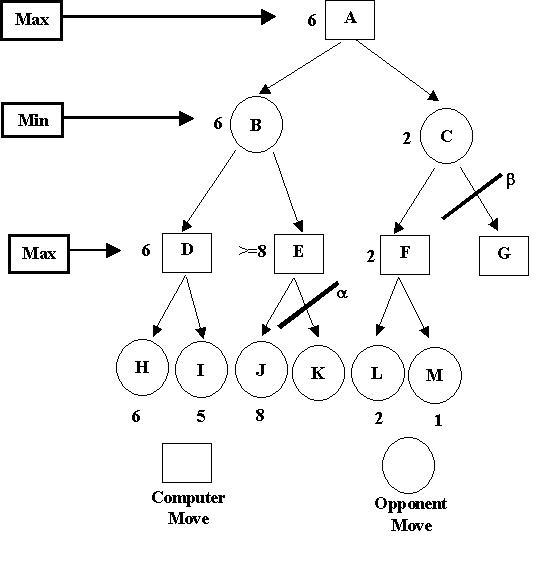


Fig 4.3 Example of Game Tree using Alpha-Beta Pruning (Kendall, 2001)

### **4.2.4 Evaluation Functions**

Evaluation Functions is used to examine a particular position of the board and estimate how well the computer is doing, or how likely it is to win from this position. Evaluation Functions is also known as Static Evaluators because they are used to evaluate a game from just one static position (Coppin, 2004).

# **Chapter 5: Methodology and Design**

## **5.1 Review of research undertaken**

The above research shows the structure, techniques and algorithms used to create a computer Chinese chess game. Like standard computer chess games, computer Chinese chess games uses game tree to represent the state of the game or a particular position. When determining possible moves or position, searching of the game tree using minimax algorithm is often used. However, due to the complexity of Chinese chess game (the complexity of Chinese chess is between Standard Chess and Go), so it is quite often impossible to search the whole game tree. This is where other techniques such as the alpha-beta pruning may be implemented to help reduce the searching complexity and to be evaluate the likelihood to win or lose.

## **5.2 Research Question**

Computer Chinese Chess and Artificial Intelligence. This study involves developing a computer Chinese Chess game using techniques and algorithms learned in research chapters.

## **5.3 Methodology**

The context of this study is to develop a computer Chinese Chess game. This will involves the following:

* Creating the Game board (with code)
* Possible movements for each chess pieces
* Capturing moves on each chess pieces
* How each chess pieces deals with exceptions, i.e. what to do when a piece reaches outside the board etc.
* Creating an Artificial Intelligence that can perform the above

## **5.4 Vision Document**

### **5.4.1 Purpose**

The purpose of this project is to create and develop a computer Chinese Chess game using Unity game engine.

### **5.4.2 Scope**

To create an Artificial Intelligence that can play the game and to be able to determine the next possible and capturing moves.

### **5.4.3 Stakeholder and User Description**

|  |  |  |
| --- | --- | --- |
| **Name** | **Description** | **Responsibilities** |
| Game Board | Game board for the game | To allow chess pieces to be placed on and move. |
| King | Game Piece for the game | To be use in the game |
| Guard | Game Piece for the game | To be use in the game |
| Bishop | Game Piece for the game | To be use in the game |
| Knight | Game Piece for the game | To be use in the game |
| Rook | Game Piece for the game | To be use in the game |
| Cannon | Game Piece for the game | To be use in the game |
| Pawn | Game Piece for the game | To be use in the game |

### **5.4.4 Requirements**

Using the **MoSCoW** method.

|  |  |  |
| --- | --- | --- |
| **Item ID** | **Description** | **M = Must Have, S = Should Have, C = Could Have, W = Won’t Have** |
| 000 | Game board creation | M |
| 001 | Possible movements and capturing moves for King piece | M |
| 002 | Possible movements and capturing moves for Guard piece | M |
| 003 | Possible movements and capturing moves for Bishop piece | M |
| 004 | Possible movements and capturing moves for Knight piece | M |
| 005 | Possible movements and capturing moves for Rook piece | M |
| 006 | Possible movements and capturing moves for Cannon piece | M |
| 007 | Possible movements and capturing moves for Pawn piece | M |
| 008 | Pieces set up at start of game | M |
| 009 | Exceptions handling | S |
| 010 | Game menu | C |
| 011 | GUI | C |
| 012 | Multiplayer | W |
| 013 | Game Difficulties | W |

### **5.4.5 Risk Analysis**

|  |  |  |
| --- | --- | --- |
| **Risk** | **Risk Description** | **Actions** |
| Game Rules | Knowledge on the game rules | Research on game rules |

### **5.4.6 Solutions (Tools and Platform)**

|  |  |
| --- | --- |
| Game Engine | Unity3D |
| IDE | Visual Studio |
| Coding Language | C# |
| Source Control | GitHub |

## **5.5 Functional Specification**

### **5.5.1 User Stories**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Title** | **Description** | **Priority**  **H – High**  **M – Medium**  **L - Low** | **Estimate (Hours)** | **Acceptance Criteria** |
| 000 | Game Board | As a game board, I want to create a game board for the game | H | 6 | Have a game board for the game pieces |
| 001 | King | As a game piece, I want to be able to move around the board to capture other game pieces | H | 8 | List of possible moves and capturing moves defined |
| 002 | Guard | As a game piece, I want to be able to move around the board to capture other game pieces | H | 8 | List of possible moves and capturing moves defined |
| 003 | Bishop | As a game piece, I want to be able to move around the board to capture other game pieces | H | 8 | List of possible moves and capturing moves defined |
| 004 | Knight | As a game piece, I want to be able to move around the board to capture other game pieces | H | 8 | List of possible moves and capturing moves defined |
| 005 | Rook | As a game piece, I want to be able to move around the board to capture other game pieces | H | 8 | List of possible moves and capturing moves defined |
| 006 | Cannon | As a game piece, I want to be able to move around the board to capture other game pieces | H | 8 | List of possible moves and capturing moves defined |
| 007 | Pawn | As a game piece, I want to be able to move around the board to capture other game pieces | H | 8 | List of possible moves and capturing moves defined |

### **5.5.2 Project Plan**

|  |  |  |
| --- | --- | --- |
| **Week** | **Description** | **Estimate (Hours)** |
| 13/11/17-19/11/17 | * Game pieces Script (Pawn) * Continue with updating FYP document (Methodology and Design part) | 8 |
| 20/11/17-26/11/17 | * Game pieces Script * Continue with updating FYP document (Methodology and Design part) | 8 |

## **5.6 Design**

### **5.6.1 Architecture Diagram**

### **5.6.2 Use Cases**

### **5.6.3 Key Class Diagrams**

### **5.6.4 VOPC**

### **5.6.5 Sequence Diagram**

### **5.6.6 Screen Layout**

## **5.7 Prototype**

|  |  |  |
| --- | --- | --- |
| **Prototype** | **Start Date** | **Finish Date** |
| 1 | 01/10/2017 | 13/12/2017 |

|  |  |  |
| --- | --- | --- |
| **Task Number** | **Details** | **Status** |
| 1 | Install Unity Game Engine and Visual Studio 2015 | Complete |
| 2 | Created Chess Board using 2D-Array | Complete |
| 3 | Created abstract class PieceControl | Complete |
| 4 | Created Pawn script to handle movement of the Pawn piece | More required as project progresses, Continue in Sprint 2 |
| 5 | Created BoardAI Class to handle Chess Pieces Set up | More required as project progresses, Continue in Sprint 2 |

Write up of key development involved (1/2 paragraphs usually for the prototype, maybe 1 or 2 pages for Sprints)

# **Chapter 6: Implementation**

# References

Agatonovic-Kustrin, S. & Beresford, R., 2000. Journal of Pharmaceutical and Biomedical Analysis. *American Association of Pharmaceutical Scientists,* 22(5), pp. 717-727.

Bengio, Y., 2016. Machines Who Learn. 01 June, 314(6), pp. 46-51.

Coppin, B., 2004. *Artificial Intelligence Illuminated.* s.l.:Jones & Bartlett Publishers.

DeepMind Technologies Limited, n.d. *AlphaGo.* [Online]   
Available at: https://deepmind.com/research/alphago/  
[Accessed 07 10 2017].

Donnelly, P., 2014. *Basics of XiangQi Play.* [Online]   
Available at: http://www.xqinenglish.com/peterdonnellybasicsofplay.html  
[Accessed 07 10 2017].

Gui, W. & Jun, T., 2016. Chinese Chess Algorithm Design and Implementation in the Computer Game. *Proceedings of the 35th Chinese Control Conference,* pp. 10380-10384.

Hansen, L. & Larsen, J., 1996. Unsupervised learning and generalization. *IEEE International Conference on Neural Networks,* Issue 96, pp. 25-30.

Hsu, F.-H., 1999. IBM's Deep Blue Chess grandmaster chips. *IEEE Micro,* 19(2), pp. 70-81.

IBM , n.d. *Deep Blue.* [Online]   
Available at: http://www-03.ibm.com/ibm/history/ibm100/us/en/icons/deepblue/  
[Accessed 27 09 2017].

Jones, M. T., 2007. *A beginner's guide to artificial intelligence, machine learning, and cognitive computing.* [Online]   
Available at: https://www.ibm.com/developerworks/library/cc-beginner-guide-machine-learning-ai-cognitive/index.html  
[Accessed 11 10 2017].

Kendall, G., 2001. *Introduction to Artificial Intelligence.* [Online]   
Available at: http://www.cs.nott.ac.uk/~pszgxk/courses/g5aiai/005gameplaying/game-playing.htm  
[Accessed 14 10 2017].

Li , Y., Liu , X., Zhang , H. & Xi , Y., 2016. Study on system modeling and optimization of static evaluation based on the computer game of draughts. *Chinese Control and Decision Conference (CCDC),* pp. 4283-4287.

Maglogiannis, I., Karpouzis, K., Wallacce, B. & Soldatos, J., 2007. *Emerging Artificial Intelligence Applications in Computer Engineering: Real Word AI Systems with Applications in EHealth, HCI, Information Retrieval and Pervasive Technologies.* s.l.:IOS Press.

Marsland, S., 2015. *Machine Learning: An Algorithmic Perspective.* 2nd ed. s.l.:CRC Press.

Mitchell, T. M., 1997. *Machine Learning.* s.l.:McGraw-Hill Science/Engineering/Math.

Ross, D., 2014. *Game Theory.* [Online]   
Available at: https://plato.stanford.edu/entries/game-theory/  
[Accessed 14 10 2017].

Russell, S. J. & Norvig, P., 2010. *Artificial intelligence : a modern approach.* 3rd ed. s.l.:Prentice Hall.

Schölkopf, B., 2015. Artificial intelligence: Learning to see and act. 518(7540), pp. 486-487.

Silver, D. et al., 2016. Mastering the game of Go with deep neural networks and tree search. *Nature,* Volume 529, pp. 484-489.

Tesauro, G., 1995. Temporal Difference Learning and TD-Gammon. *Comunications of ACM,* 38(3), pp. 58-68.

Turing, A. M., 1950. Computing Machinery and Intelligence. *Mind,* 49(236), pp. 433-460.

Wang, J. & Li, S., 2008. *Representing evaluation of computer Chinese chess by artificial neural network using genetic algorithm.* Yantai, Shandong, China, Chinese Control and Decision Conference (CCDC), pp. 1226-1232.

Yang , Z. et al., 2017. Research and Implementation of Static Evaluation Algorithm for Checkers. *Chinese Control And Decision Conference (CCDC),* pp. 7665-7670.

Yellow Mountain Imports, n.d. *How to Play Xiangqi / Chinese Chess / 象棋.* [Online]   
Available at: https://www.ymimports.com/pages/how-to-play-xiangqi-chinese-chess  
[Accessed 11 10 2017].

Yen, S.-J., Chen, J.-C., Yang, T.-N. & Hsu, S.-C., 2004. Computer Chinese Chess. *ICGA Journal,* p. 18.

Yu, W.-G., 2012. Designing a high-performance Chinese chess computer player for middle game position. *Proceedings of the 2012 International Conference on Machine Learning and Cybernetics,* pp. 571-57.