2021/22

Student Name:	Nan Zhao
Student ID:	201530286
Project Title:	COMP702 Visualization and AI for 3D Tic-Tac-Toe
Primary Supervisor:	Prof. Boris Konev
Second Supervisor:	Prof. Frank Wolter

DEPARTMENT OF COMPUTER SCIENCE

The University of Liverpool, Liverpool L69 3BX

Contents

1 Project Description

The game played on a three-row chessboard dates back to ancient Egypt and was found on roof tiles around 1300BC[6]. Early variations of tic-tac-toe were broadcast in the Roman Empire around the first century BC. Because of the simple structure of the tic-tac-toe game, it became a common example in artificial intelligence in the early years. From the existing gameplay, the winning method of the game can be summarized, and a program can be deduced from the strategy, allowing the computer to play against the user[4].

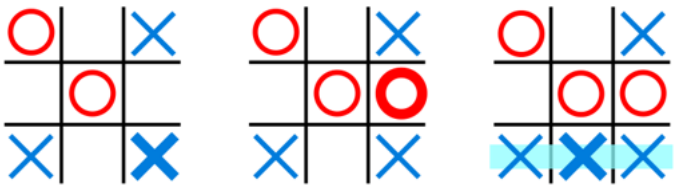

Figure 1: A winning example of the player with a cross (X)

Two players, one with a circle (◯) and one with a cross (X), take turns playing their own symbols on the 3x3 grid, and the first to line up the horizontal, vertical, and diagonal lines is the winner (as shown in figure 1). If both players play correctly, the board would be filled and the game will be tied (as shown in figure 2).

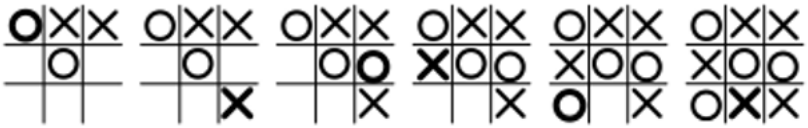

Figure 2: the draw of tic-tac-toe

The game is actually controlled by the first player, the first player is the offense, and the second is the defender. If the first player is in the corner with the first piece, the chances of winning are the largest (as shown in figure 3). If the second player is on the side and the corner with a piece, the first player can contain the second player with two pieces of connecting lines, so the second player has to go down the center. If the first player plays in the center, the second player must play in the corner so that the second player does not lose the game. If the first player plays on the side, the second player can play in the center or corner, or on the side opposite to where the first player played.

Through the project of tic-tac-toe,the common others could know of algorithm and visualization about the interest of AI with playing game. The contribution of the project to the knowledge of others is that AI is easily winning player and tic-tac-toe could use python programming to achieve.

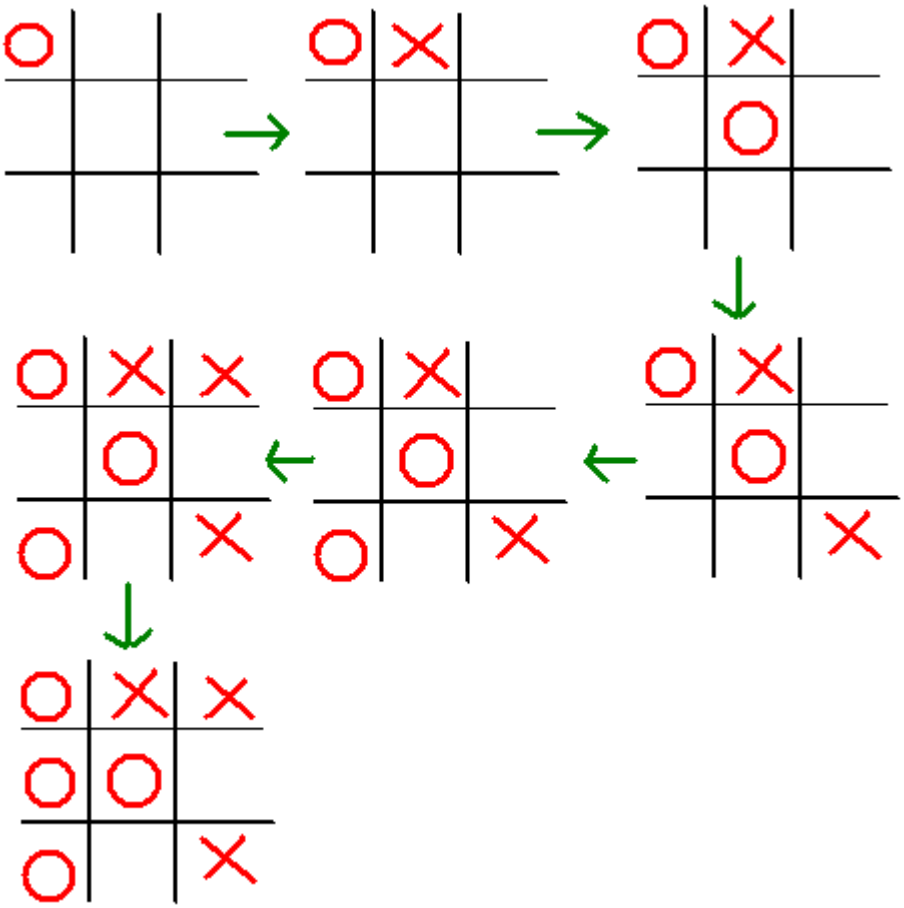

Figure 3: the first player is in the corner with the first stone

2 Aims and Objectives

From the original plane to pass the three levels, change it to a three-dimensional 3x3x3 to pass the three levels. When the three symbols of the player side are connected in a three-dimensional, vertical, horizontal, and diagonal direction, the player would win (there are 49 winning lines in total). However, it is not very interesting, because as long as the first player is in the center of the cube, it is guaranteed to win (the center of the cube can control up to 13 connections) and there is no possibility of a tie in this game, no matter how many squares each side occupies. If the first player is prohibited from playing in the center of the cube, the second player would win. If the grid in the center of the cube could be deleted, it becomes the first player would win.The component of the grid is the current state of each grid, whether it is placed by player A (◯ and

X), placed by player B (W and Y), or empty. The component of the turn is whether it is Player A (◯ and X) or Player B (W and Y).

In this game,AI uses two pieces of Y and W fairly to play with two humans who have X and O, and also AI uses two pieces of Y and W to play with a player who has one piece(X or ◯). Only having the same four pieces become a line will increase a score, finally, summing up the scores and getting higher scores will win the game.

3 Key Literature and Background Reading

The topic area is about minimax algorithm and visualization through playing the game of tic-tac-toe. key literature include 3D tic-tac-toe,minimax algorithm,interface and visualization.

About the background reading, I read the brief introduction of tic-tac-toe from wikipedia.I also read the review of interface that is using 3D Graphics and Interfaces in Training Games,the user interface is a space where human-machine interaction takes place. The aim of this interaction is to ensure that the machine functions effectively and is controlled by the person, while the machine simultaneously feeds back information that helps the person make decisions.From the review I know that interface function and how to make a interface in game. I have a clear goal about creating a clear and simple game interface,using 3D graphics to create a game environment analysis of the interface interaction between the player and the game[2].

I read the article that is Alpha-Beta Pruning in Mini-Max Algorithm –An Optimized Approach for a Connect-4 Game, from the review I know that The moves of the AI will be optimized by using mini-max algorithm and alpha beta pruning. There are two algorithm, one of algorithm is mini-max, the other is mini-max with alpha beta pruning which is an optimized version of mini-max algorithm.For the same level of difficulty the two algorithms behave very differently in terms of number of iterations performed and time taken with alpha beta pruning taking much less time and performing very few iterations than mini-max to generate the game state[5].

In the project,I use the min-max with alpha beta, because alpha beta is helpful to find better location of piece on the board.Therefore,the time of performed iterations is much less than using only mini-max algorithm.

4 Development and Implementation Summary

The design diagram is about analysis of algorithm(as shown figure 4),it is quite evident that mini-max and alpha-beta pruning are not entirely different algorithms rather alpha-beta is just an optimized version of simple min-max algorithm[5].

Comparison both min-max algorithm and alpha-beta pruning shows the figure 5.

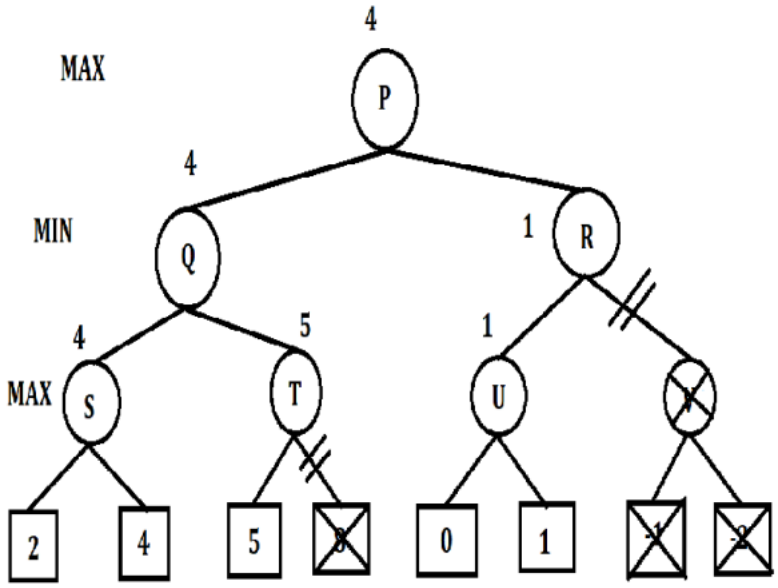

Figure 4: Game Tree using Min-Max Algorithm in case of Alpha-Beta Pruning

In the study carried out it is observed that when the user plays the game in single player mode and chooses minimax as the operating algorithm then AI takes 7 iterations to generate the game state and computation time is 0.00 milliseconds for a difficulty of depth 1 whereas for the same difficulty level alpha-beta pruning also takes 7 iterations and 0.00 milliseconds to compute.The difference between the two algorithms becomes starker when difficulty level is increased as it is clear from the observation that for a difficulty of depth 4 AI takes 2799 iterations to generate the game state and computation time is 33.00 milliseconds in case of mini-max whereas for the same difficulty level alphabeta pruning takes only 477 iterations and 6.00 milliseconds to compute. Further it is observed that when difficulty level is increased to 8 which is the toughest level AI takes 5847005 iterations to generate the game state and computation time is 55441.00 milliseconds in case of minimax algorithm whereas for the same difficulty level alphabeta pruning takes only 71773 iterations and 1009.00 milliseconds to compute.The result has been summarized in figure 5[5].

The application of heuristic function is to coordinate robot agents in adversarial environment[1]. The AI will exhaustively evaluate the tree to a certain depth (the number of moves to 'think ahead') and then evaluation of the board is done using heuristic function. Eventually, as the end of the game draws near, the AI finds winning or losing states confined to the specified depth and plays perfectly. The whole idea here is that AI can be guided to a winnable position by heuristic function. A function is implemented that calculates the value of the board depending on the placement of pieces on the board. This function

Algorithm Used	Mini Max		Alpha-Beta Pruning	
Difficulty Level	**No. of Iterations performed**	**Computation Time (ms)**	**No. of Iterations performed**	**Computation Time (ms)**
Depth 1 (Easiest)	7	0.00	7	0.00
Depth 4 (Normal)	2799	33.00	477	6.00
Depth 8 (Hardest)	5847005	55441.00	71773	1009.00

Figure 5: Comparison between Two Algorithms

is often referred to as Evaluation Function and sometimes also called Heuristic Function[3]. The basic idea behind the heuristic function is to allot a high value for a board if maximiser's turn is being played or a low value for the board if minimizer's turn is being played[5].

In the project, I am planning to use unity to achieve visualization by using python programming.

Unity has many advantages, The first advantage is a visual development environment, the second advantage is cross-platform support. The first factor includes not only visual modeling tools, but also an integrated environment, the build chain, which is aimed at improving the productivity of developers, in particular, the stages of prototyping and testing. The third advantage is the modular system of Unity components, which is used to construct game objects, when the latter are combined packages of functional elements. Unlike inheritance mechanisms, objects in Unity are created by combining functional blocks, rather than being placed in nodes of the inheritance tree. This approach makes it easier to create prototypes, which is important when developing games[2].

Interface development is one of the most important stages of creating a game. The intuitive interface allows the player to understand the game mechanics better. The quality of the interface depends on the amount of fun that the user will get during the gameplay[2].

5 User Interface Mockup

Firstly, I visualise the cube(as shown figure 6).Secondly, I use unity to interface of tic-tac-toe game(as shown figure 7)

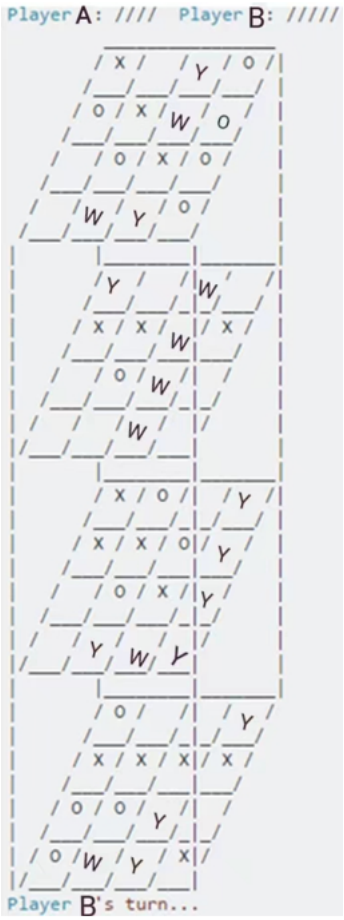

Figure 6: 4x4x4 cube of tic-tac-toe

6 Data Sources

In the project, I do not use data, my supervisor know of it.

7 Testing

I choose to use unity to show the visualization, which is playing tic-tac-toe game by human and computer, both sides have two pieces respectively, and use python programming to run the project to show the 4x4x4 cube.

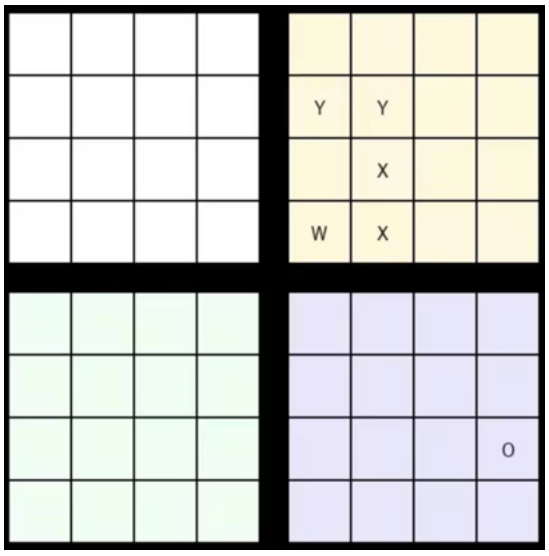

Figure 7: interface of tic-tac-toe

8 Evaluation

In the 4X4 chess board,firstly, initialization of the game. Secondly, the player puts the pieces and finds the corresponding array element changes, and pays attention to judging the end of the game and changing the round. Thirdly, AI chess placement strategy, AI should do local optimal operations, for each square, if AI can win, AI will win first, then if the opponent can win, AI will block first, otherwise, playing randomly. Especially, AI should first traverse the squares to judge that AI won, and then traverse the squares to judge that the opponent won, otherwise AI will not get the optimal solution. Finally, judging the game over.

9 Ethical Considerations

I have read our ethical guidelines and will follow them.

10 Project Plan

My project plan shows the figure 8, I think that the most time-consuming parts is about the visualization.

Time	plan
Jul-18 - Jul-22	Refer to tic-tac-toe from git hub website, and design own program
Jul-25 -Aug-12	Design and achieve the visualization by reading different 3D game engine
Aug-15 – Aug 26	Make the ppt and have a discussion with my supervisor
Aug-29 - Sep-02	Make a presentation
Sep-05-Sep-23	Write my dissertation

Figure 8: project plan

11 Risks and Contingency Plans

I make a list about the possible risk and contingency(as shown the figure 9)

Risks	Contingencies	Likelihood	Impact
Unity could not achieve visualization by using python	Using pygame to achieve the visualization by python programming	middle	A little impact

Figure 9: Risks and Contingency Plans

References

[1] A. Kovarsky I. Levner. Heuristic search for coordinating robot agents in adversarial domains. *International Conference on Robotics and Automation,IEEE*, pages 720,doi:10.1109/ROBOT.2006.1641770, 2006.

[2] R A Filippov L B Filippova. Using 3d graphics and interfaces in training games. *IOP Conf. Ser.:Earth Environ*, 720:012091,doi:10.1088/1755–1315/720/1/012091, 2021.

[3] Simon M. Lucas and Thomas Philip Runarsson. on imitating connect-4 game trajectories using an approximate n-tuple evaluation function. *Conference on Computational Intelligence and Games (CIG)*, pages 2325–4270,doi:10.1109/CIG.2015.7317961, 2015.

[4] Marla Parker. She does math!: Real-life problems from women on the job. *Mathem -atical Association of America,ISBN 978-0-88385-702-1*, 1995.

[5] Rishabh Didwania Rijul Nasa1. Alpha-beta pruning in mini-max algorithm–an optimized approach for a connect-4 game. *International Research Journal of Engineering and Technology (IRJET)*, 05:6, 2018.

[6] Claudia Zaslavsky. Tic tac toe: And other three-in-a row games from ancient egypt to the modern computer. *Crowell,ISBN:0-690-04316-3*, 1982.