Demonstrating Three Dimensional Tic-Tac-Toe with Gravity

Lok Hei Ng

March 14, 2016

Abstract

Tic-tac-toe is a 2-player paper-and-pencil game which its objective is to have three tokens in a row in the given grid.

In this project, we are going to present the 3x3x3 tic-tac-toe with gravity using VPython. We will also present the traditional 3x3 tic-tac-toe which the computer AI will never lose. Apart from these, the visualisation of the winning strategy of the player whose the first move is the middle pole will also be demonstrated.

Introduction

The classic tic-tac-toe is in 2 dimensions with a 3x3 grid. However, it always ends up with a draw. Therefore, it was modified to 3x3x3. But it is still not very interesting as there is a strategy for the player who has the first move. To increase its complexity and make it interesting again, we add gravity to it. Here are the descriptions of the game :

Setup :

1. 14 tokens of type “X” and 13 tokens of type “O”, a total of 27 tokens
2. 9 poles arranged in 3 in a row and 3 columns for each row. At most 3 tokens can be put onto each pole.

Winning criteria :

- Getting a three in a row in any direction, ie, horizontal, vertical or diagonal.

Flow :

1. Player X uses type “X” tokens and player O uses type “O” tokens.
2. Player X makes the fist move and puts his tokens onto one of the poles. He is not allowed to it onto the middle pole.
3. Then two players take turn putting one of their tokens onto a pole.

Mathematical context

In a 3x3x3 tic-tac-toe with gravity, there are at most $\frac{27!}{(3!)^9}$ possible games (different sequences of placing the tokens onto the poles). Indeed, each possible game can be represent by a word of length 27, composed with 9 letters and each letter must be used three times. However this number can be greatly reduced because the game ends once there exists three in a row.

Player X is not allowed to put his tokens onto the middle pole in his first move. Otherwise, he can always win and we have the following theorem.

Theorem 1:
Suppose player X puts his token onto the middle pole. Then there exists a strategy that he always win.

Proof :
We number the poles as follows :

123
456
789

Player X puts his token onto pole 5. Then player O has 3 distinguishable moves, the corners, the edges, or the middle pole. Player X can use one of the following strategies to win the game :

1. 51823746558(82 or 28)
2. 52197(34 or 43)
3. 55192(38 or 83)

Remark :
It is easier to make 3 in a row using the the middle pole. Also, player X gets the dominance of the game. Therefore, it is quite likely for him to win the game if he puts his token onto the middle pole. The theorem confirms this conjecture.

Programming of the project :

Mathematical visualisation of the proof of theorem 1

The animation of the three strategies of player X, together with the text descriptions will be shown.

The traditional 3x3 tic tac toe :

There are 4 modes, the single player mode with the computer AI taking the first move, that with the player taking the first move, the two players mode and the demonstration mode which two computer AI play as player X and player O.

Flow :
1. Display a window with the 3x3 grid, the players take turn to put their tokens into the grid by clicking it.
2. The player who gets three in a row first will win the game.

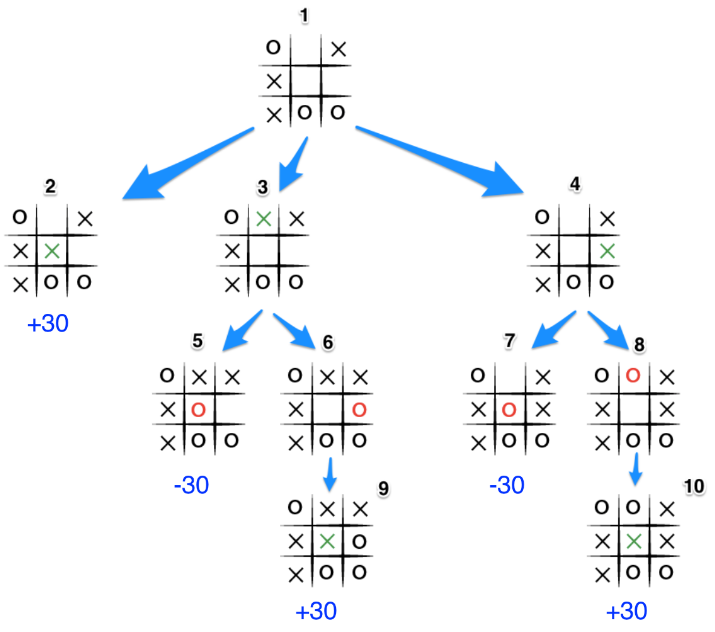
Algorithm of computer AI:
The computer AI adapts the minmax algorithm.

Consider the tree which exhausts all the possible moves. Each node of the tree represents a state of the game. Note that the root is the current state of the game. The leaves represent the end game conditions. The nodes on the same level are the possible states after the AI's move (resp. player's move) if they have an odd (resp. even) distance from the root.

The AI chooses to move to the state with the highest score. To define the score of the state, we need the following 3 definitions.

1. Scores of end game condition :
+ 30 : if AI wins the game
- 30 : if AL loses the game

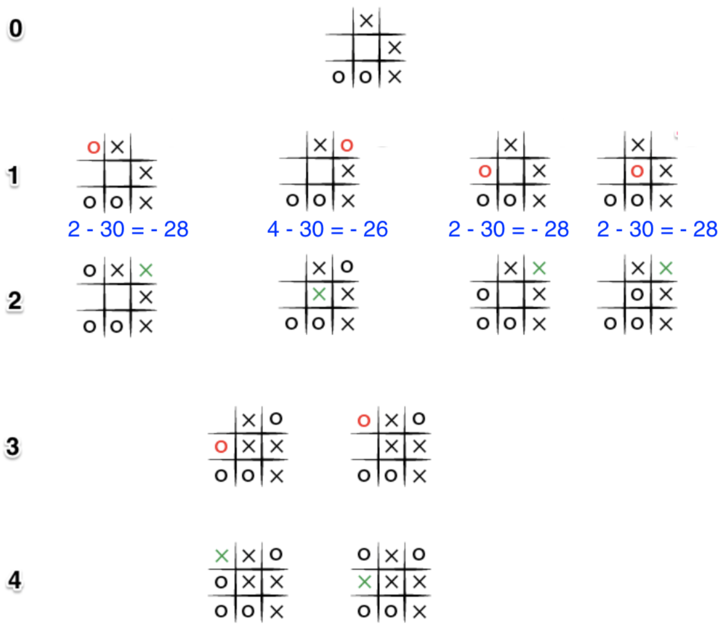
2. Winning scores of each node (defined recursively) :
If it represents a end game condition, its score would be the score of the corresponding end game condition.
If it represents the state right after the AI's (resp. the player) move, then it is the minimum (resp. maximum) of the winning scores of its children.



This figure shows the winning score of the nodes in 3x3 tic tac toe. The AI is player X and the current state is node 1.
The source is http://neverstopbuilding.com/minimax with modifications.

3. Depth of the node :
For the nodes with an even distance from the root, we only consider one with the minimal winning score and its corresponding subtree. The depth of move is defined to be it maximum distance to the farthest leaf.

The score of the node is the sum of its depth and its winning score. The figure below show the scores of each moves.



This figure shows the score of the nodes in 3x3 tic tac toe. The AI is player O and the current state is the node at level 0.
The source is http://neverstopbuilding.com/minimax with modifications.

The 3x3x3 tic tac toe with gravity :

There are 4 modes, the single player mode with the computer AI taking the first move, that with the player taking the first move, the two players mode and the demonstration mode which two computer AI play as player X and player O.

Flow :

1. Display a window for the users to choose one of the 4 modes that they would like to run.
2. If one of the two single players modes or the two players mode is chosen, the users will be brought to the game. An window will pop up, showing the instructions of the game. The users will use their mouse to control the screen and put their tokens onto the poles.
3. If the demonstration mode is chosen, the poles will be displayed and an animation of the tokens being put onto the poles will be shown.
4. After the games or the demonstration, the users will be brought to the window described in (1) and repeat the game.

Algorithm of computer AI:

The computer AI checks if it has 2 in a row and gets the win. Then it checks if the player has 2 in a row and prevents him from winning. Otherwise, it will randomly choose a move among all of the possible moves.

This algorithm is adapted instead of the minmax algorithm since the time for running the latter will be long. A program which counts the number of possible games was written and it takes more than an hour to run. Therefore, I believe that the minmax algorithm takes much more time to run.

Reference

1. Tic Tac Toe: Understanding the Minmax Algorithm http://neverstopbuilding.com/minimax by jasonrobertfox
2. Wikipedia article about Tic-Tac-Toe https://en.wikipedia.org/wiki/Tic-tac-toe