Communications on Applied Nonlinear Analysis
ISSN: 1074-133X
Vol 31 No. 2s (2024)

Optimal Strategy Formulation for Tic-Tac-Toe Using Minimax Algorithm for Interactive Gaming

Megha Chakole[1], Rohini Nawathe[1], Sanjay Dorle[2] ,Gargi Bhakre[1], Nita Nimbarte[1]

[1].Department of Electronics & Telecommunication Engineering , Yeshwantrao Chavan College of Engineering Nagpur,India

[2]. Department of Electronics Engineering, G.H.Raisoni College of Engineering Nagpur,India

mdorle@gmail.com, rdnahate@gmail.com, sanjay.dorle@raisoni.net, bhakaregargi@gmail.com, nitangp@gmail.com

Article History:

Received: *22-03-2024*

Revised: *11-05-2024*

Accepted: *24-05-2024*

Abstract:

Introduction: This research presents a Python implementation of the classic game Tic Tac Toe utilizing the Tkinter library for the graphical user interface (GUI). The code offers a comprehensive framework for creating an interactive gaming experience, featuring both single-player and multiplayer modes. Key functionalities include mode selection, game board initialization, player moves, win and draw condition checks, and game restart options.

The graphical interface provides a visually appealing environment for players to engage with the game, facilitating intuitive interaction through buttons and labels. Notably, the single-player mode incorporates a basic AI component powered by the minimax algorithm, offering players the opportunity to challenge themselves against a computer opponent.

Objectives: The main objectives include developing a comprehensive framework for Tic-Tac-Toe, encompassing essential functionalities such as mode selection, initializing the game board, managing player moves, conducting win/draw condition checks, and offering game restart options.

Methods: The implementation employs Tkinter to create a user-friendly GUI featuring interactive buttons and visual elements. Game logic is meticulously implemented to regulate player turns, validate moves, and assess win/draw conditions. Additionally, the single-player mode integrates a basic AI component driven by the minimax algorithm for a challenging gaming experience.

Results: The resulting implementation boasts a visually appealing interface, complete with dedicated mode selection buttons, interactive game board buttons, thorough win/draw condition checks across rows, columns, and diagonals, and a convenient restart button. The single-player mode successfully emulates human-like gameplay through the AI component.

Conclusions: This implementation's simplicity and clarity render it an invaluable resource for beginners delving into GUI development and game programming with Python. Furthermore, the integration of AI functionality caters to advanced users keen on exploring algorithmic game strategies. Ultimately, this paper offers a valuable foundation for learning and comprehending the fundamentals of game development using Python and Tkinter.

Keywords: Python, Tic Tac Toe, Tkinter, graphical user interface (GUI), single-player, multiplayer, mode selection, game board initialization, player moves, win condition checks, draw condition checks, game restart options.

1. Introduction

Python is a general-purpose, high-level programming language. With a strong emphasis on indentation, its design philosophy prioritizes code readability. Python uses garbage collection and dynamic typing. It is compatible with several programming paradigms, such as functional, object-oriented, and structured programming.

Many developers and learners choose Tkinter initially because it is user-friendly and fast. With the help of the Python package Tkinter, simple graphical user interface (GUI) programs can be created. It is the most used Python package for graphical user interfaces.

Known as a classic two-player game, Tic-Tac-Toe is played by people all over the world. Usually, one person performs for "X" and another for "O". The distraction is played on a 4x4 TicTac-Toe square leading body or one that is noticeably larger. The goal of this game is for the player to load any line, sector, or corner of the main diversion board with "X" or "O" in order to win [1].

With these rules, the game components written below are required:

1. Game Board: A 3x3 grid where players place their "X" or "O" marks. It's where the game happens and where players interact.
2. Player Marks ("X" and "O"): Each player has their symbol, "X" or "O". Players take turns placing their marks aiming to get three in a row horizontally, vertically, or diagonally.
3. Mode Selection Buttons: These buttons let players choose between single-player or multiplayer modes, giving them control over their game experience.
4. Restart Game Button: This button resets the game, allowing players to start fresh or replay after it ends.
5. Winning and Draw Checks: These components analyze the game after each move to see if a player has won by getting three marks in a row or if the game ends in a draw.
6. AI Component (Single-Player Mode): In single-player mode, there's an AI opponent using the minimax algorithm. It offers a challenging solo experience by simulating human-like gameplay.
7. Labels and Text Elements: These elements provide information to players, like the game title ("Tic Tac Toe"), announcing the winner, declaring a draw, or guiding players in selecting game modes [2].

Throughout this paper, we will delve into the design principles, implementation details, and user feedback that shaped the evolution of TicTac GUI. Through empirical analysis and qualitative assessment, we aim to elucidate the impact of intuitive design on user engagement and satisfaction. Moreover, we will explore potential avenues for future research and development, highlighting the untapped potential inherent in the convergence of classic games and modern technology. TicTacGUI represents more than just a reimagining of tic-tac-toe; it embodies a testament to the enduring relevance of user-centric design in software development.

2. Objectives

The development of the tic-tac-toe game in a graphical user interface (GUI) environment will utilize Tkinter for creating the game window. The window consists of two main components: the top display, which presents information about the game's status and previous moves, and the grid cells, representing the game board.

For the grid cells, Tkinter Button widgets arranged in a grid layout will be employed. When a player clicks on one of these buttons, the game logic will be invoked to process the move and determine if there's a winner. The game logic serves as the model, managing the game's data, logic, and rules. This document aims to outline the requirements necessary for users to engage in playing tic-tac-toe through a graphical user interface. Additionally, it provides insight into the underlying logic governing the game's operations.

3. Related work

In ancient Rome, tic tac toe was referred to as Terni Lapilli and dates back to the first century BC. To continue playing, players had to move their three pieces to available spots. Although some claim it originated in ancient Egypt, archaeological data demonstrates that the Roman Empire was the home of its early rendition. Tic-Tac-Toe may seem easy, but there's a strategy to winning, according to a research paper written by authors. Starting in a corner provides you the best chance of winning later if you go first (X). Proceeding to the second (O)? X's opening move should be strategically countered. Play the center if X travels to the corner. Take a corner if X takes the center. You can block X with the opposing corner or the center if they start on an edge. Depending on who gets the first turn, you can outmaneuver your opponent or force a tie by using these maneuvers [1-2].

Ancient Egypt was the time when games having three-in-a-row boards first appeared. The Roman Empire played a version of tic tac toe that predated modern versions. Students at MIT also used tic tac toe in 1975 to show off the computing capability of Tinkertoy components. Perfectly able to play tic tac toe, the Tinkertoy computer is constructed using (almost) just Tinkertoys. Currently, the Computer History Museum is displaying it [3-4].

In this work, authors have implemented a genetic algorithm for the purpose of developing no-loss tactics in the Tictactoe game. First, authors covered the initialization scheme and GA operators for the minimization of the proportion of losses in all potential game scenarios. Then defined a technique for encoding a strategy in a GA. Through meticulous design of the GA technique, we have discovered up to 72,657 distinct strategies that guarantee a win in every game. This is an amazing outcome for the TicTacToe game. After further analysis of these solutions, we have identified four distinct no-loss strategies that also result in a high win-to-draw ratio [5].

Players can select who goes first in this Tic-Tac-Toe game by using the Python environment to implement it for a gaming application (X or O). Moves are tracked in the game on a graphic board as well as on the console. Depending on the condition of the game, the computer can act as an opponent and make its own strategic movements. Every move in the game counts wins, and if a player gets three in a row, the winning line is shown. To manage user input and show messages, the code makes use of conditional statements and fundamental Python functions. Anyone can play the straightforward game of tic tac toe, and this Python version is simple to use and comprehend [6].

The Author used C programming language to create a Tic-Tac-Toe. People of various ages play the game tic tac toe. C programming has been utilized in its development vocabulary. Any decision-maker with a purpose can possess intelligence. After being tested and presented in the system, a Tic Tac Toe algorithm has shown to be successful. The system is error-free, first and foremost [7].

As per James Shah, creating a little game or any other enjoyable project is the finest and most enjoyable approach to learn a programming language. He recommends reading "Automate the Boring Stuff with Python," a book released by No Starch Press, if you want to learn Python in an engaging way. This is how he first got into the language. It covers nearly all of the fundamental subjects in Python and is very beginner-friendly. So, James Shah discovered this Python TicTacToe game implementation while working through the exercises in this book [8].

4. Methodology

The Tic-Tac-Toe game is played on a 3x3 grid. The board can be represented as a

3× 3 matrix B where each cell can have one of three values:

0 for an empty cell

1 for a cell occupied by player 1 (X)

−1 for a cell occupied by player 2 (O)

- **Columns**:

$$\sum_{i=1}^{3} \mathrm{b_{ij}} = 3 \, or \sum_{i=1}^{3} \mathrm{b_{ij}} = -3 \;\; for\, any\, \mathrm{j} \in \{1,2,3\}$$

- **Diagonals**:

$$\sum_{i=1}^{3} \mathrm{b_{ii}} = 3 \, or \sum_{i=1}^{3} \mathrm{b_{ii}} = -3$$

$$\sum_{i=1}^{3} \mathrm{b_{i}(4-i)} = 3 \, or \sum_{i=1}^{3} \mathrm{b_{i(4-i)}} = -3$$

The Minimax Algorithm

The minimax algorithm is a recursive algorithm used for decision-making in game theory. It computes the optimal move for a player assuming that the opponent also plays optimally [4].

The minimax value of a player is the lowest score they can guarantee without knowing what the other players will do. It's also the highest score they can expect when they know the other players' moves. It is given by equation 1.

$$\bar{x}_i = \min_{p_{-i}} \max_{p_i} x_i \left(p_i, p_{-i} \right) \text{ --------------------------------- } (1)$$

Where,

i is the index of the player of interest.

-i denotes all other players except player i.

pi is the action taken by player i.

p-i denotes the actions taken by all other players,

x_i is the value function of player i.

In the minimax algorithm, we look at the maximizing player and the minimizing player separately. This equation summarizes it:

$$\max(p,q) = -\min(-p,-q) \text{ -------------------------------- } (2)$$

Where,

p = Player 1

q = Player 2

The Minimax algorithm was illustrated in Figure 1 to develop the game Tic Tac Toe.

1. The Minimax algorithm is a powerful strategy used in two-player zero-sum games like tic-tac-toe. It navigates through the game tree, assessing potential moves and their outcomes to determine the best possible move for a player.
2. In this algorithm, one player aims to maximize their score, while the other player seeks to minimize it. This dynamic creates a strategic balance where each player anticipates the other's moves to optimize their own chances of winning.
3. By exploring various move possibilities and their consequences, the Minimax algorithm helps players make informed decisions, considering both immediate gains and potential future outcomes.
4. Through iterative evaluation, Minimax enables players to select moves that lead to the most favorable outcome, whether it's a win, loss, or draw, ultimately enhancing their strategic gameplay in games like tic-tac-toe.

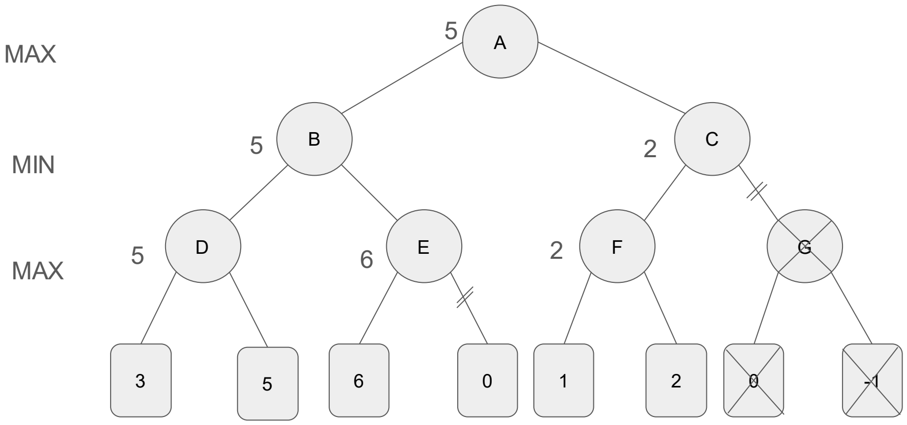

Figure 1 Illustration of Minimax algorithm to develop Tic Tac Toe game

Implementation steps for designing Tic Tac Toe GUI game are as follows and flow diagram is depicted by figure 2

Step 1: Launch the game and configure the participants, GUI, and game board. : First, the players, board, and graphical user interface (GUI) components are set up for the game. This includes initializing player data i.e. Single player mode or Multiplayer mode and building a grid for the Tic Tac Toe board.

Step 2: Make the game board visible.: The game board is then shown on the GUI, with players able to perform moves on an empty grid.

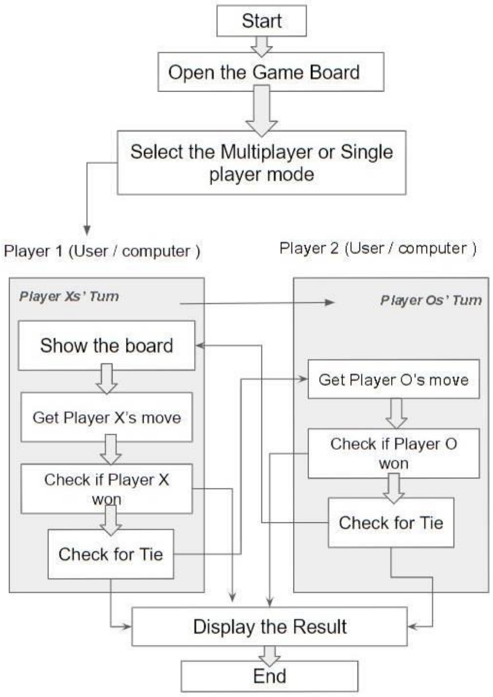

Figure 2. Implementation flow of Tic Tac Toe GUI

Step 3: Await player response: After that, the player must wait for the game to respond, usually by clicking on a grid cell.

Step 4 : Verify the Valid Move : The min-max algorithm determines whether the move is valid after receiving input. This involves making sure the chosen cell is both inside the grid's bounds and empty.

Step 5: Confirm that the relocation is legal.: The player's sign (X or O) is updated in the selected cell on the game board if the move is judged valid.

Step 6: Switch up the game board : The program then determines if the current board configuration leads to a win or a tie after the move. To find out if a player has attained three of their symbols in a row or if the board is full without a winner, this entails looking at rows, columns, and diagonals.

Step 7: Update the player's symbol on the game board if the move is valid.: In the event that a victory or tie condition is satisfied, the game continues as normal, showing the outcome on the GUI and perhaps asking the players to start over.

Step 8: Determine if the recent move wins or ties the game. Ultimately, the game ends when a win/tie condition is met or when the players choose to give up.

5. Results

In this section we will see how this game works. And how it displays the win and tie conditions.

***Case 1*: Following is the condition where 'O' wins:**

Figure 3 images showcase the conclusive state of the game board where the player utilizing the 'O' mark has attained victory. Specifically, the 'O' marks are arranged diagonally from the top-left to bottom-right cells, forming a winning diagonal line. This alignment of three consecutive 'O' marks signifies the successful completion of the winning condition by the player representing 'O'.

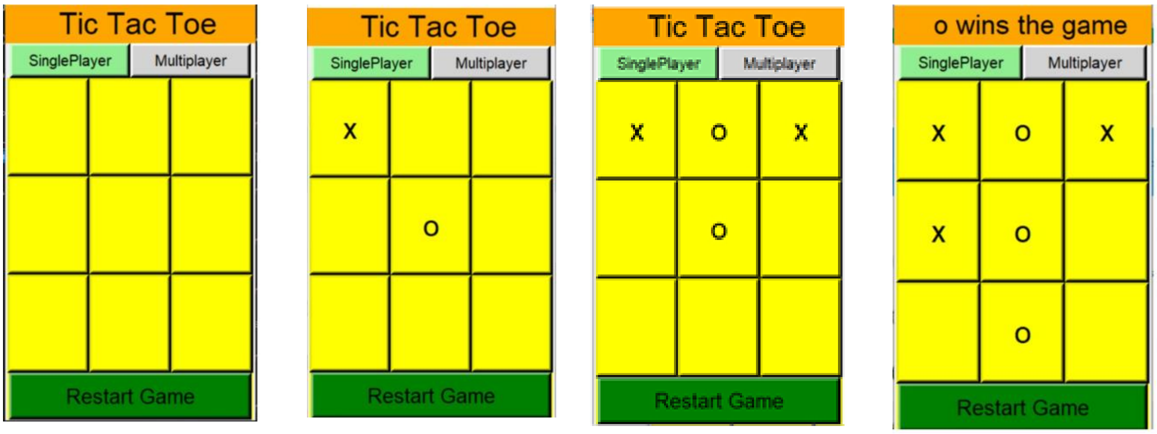

Fig. 3 Tic Tac Toe game where 'O' wins

Although the remaining cells on the board exhibit a combination of 'X' marks and empty cells, the 'X' player failed to establish a winning line before the 'O' player achieved victory. As soon as a player manages to align three of their marks horizontally, vertically, or diagonally, the game promptly concludes, and that player is declared the winner. In this instance, the player utilizing 'O' has effectively outplayed their opponent, strategically positioning their marks to secure the decisive diagonal line, thereby clinching victory in the game.

***Case 2:* Following is the condition where ‘X’ wins:**

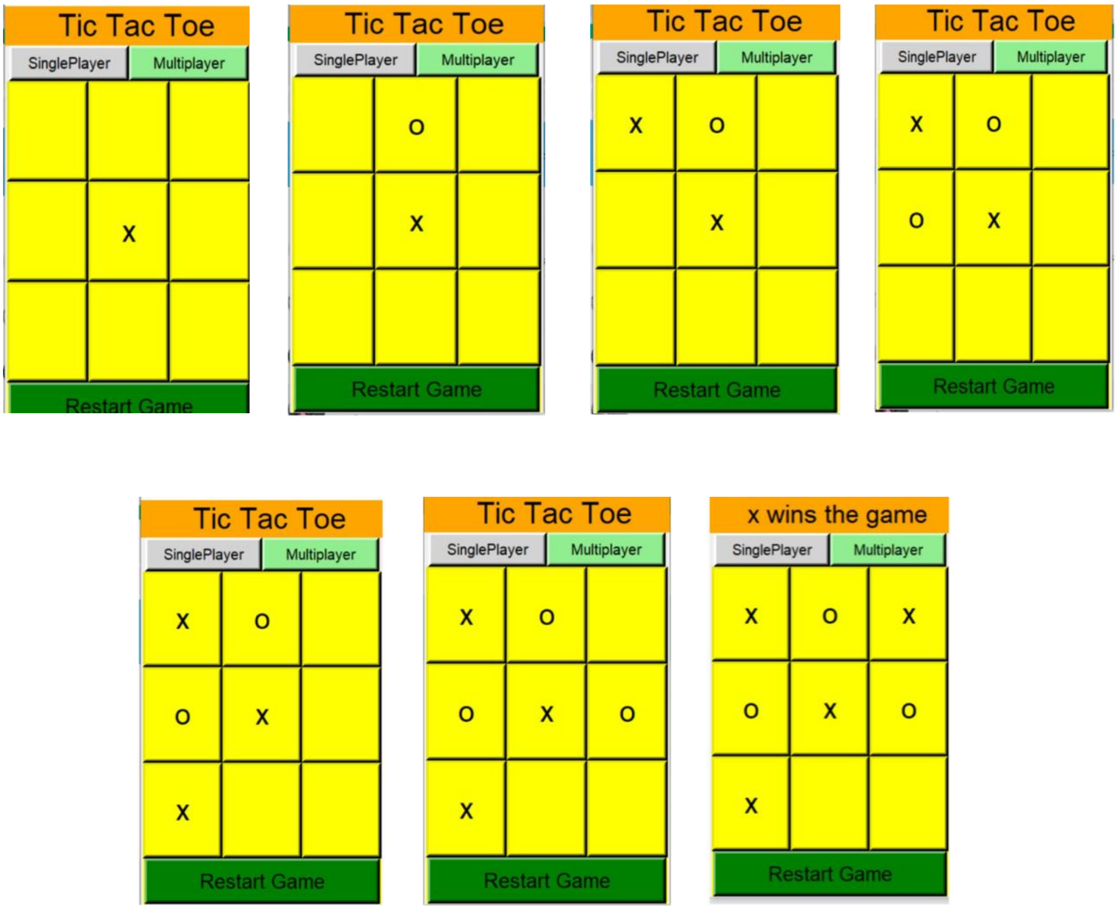

Fig. 4.Tic Tac Toe game where ‘X’ wins

Figure 4 image portrays the board configuration at the juncture where the player employing the 'X' mark has secured a winning line. Here, the 'X' marks are aligned horizontally across the central row of the 3x3 grid, forming a consecutive sequence of three 'X' marks. Similar to Case 1, the remaining cells on the board exhibit a mixture of 'O' marks and empty cells. However, the 'O' player failed to establish a winning line before the 'X' player accomplished the requisite horizontal alignment for victory. By successfully positioning their 'X' marks in a straight line, the 'X' player has fulfilled the winning condition of the game, resulting in their triumph over the opponent. Once a winning line is formed, the game concludes, and the player who achieves the line is proclaimed the winner.

Case 3: **Following is the condition when the game draws:**

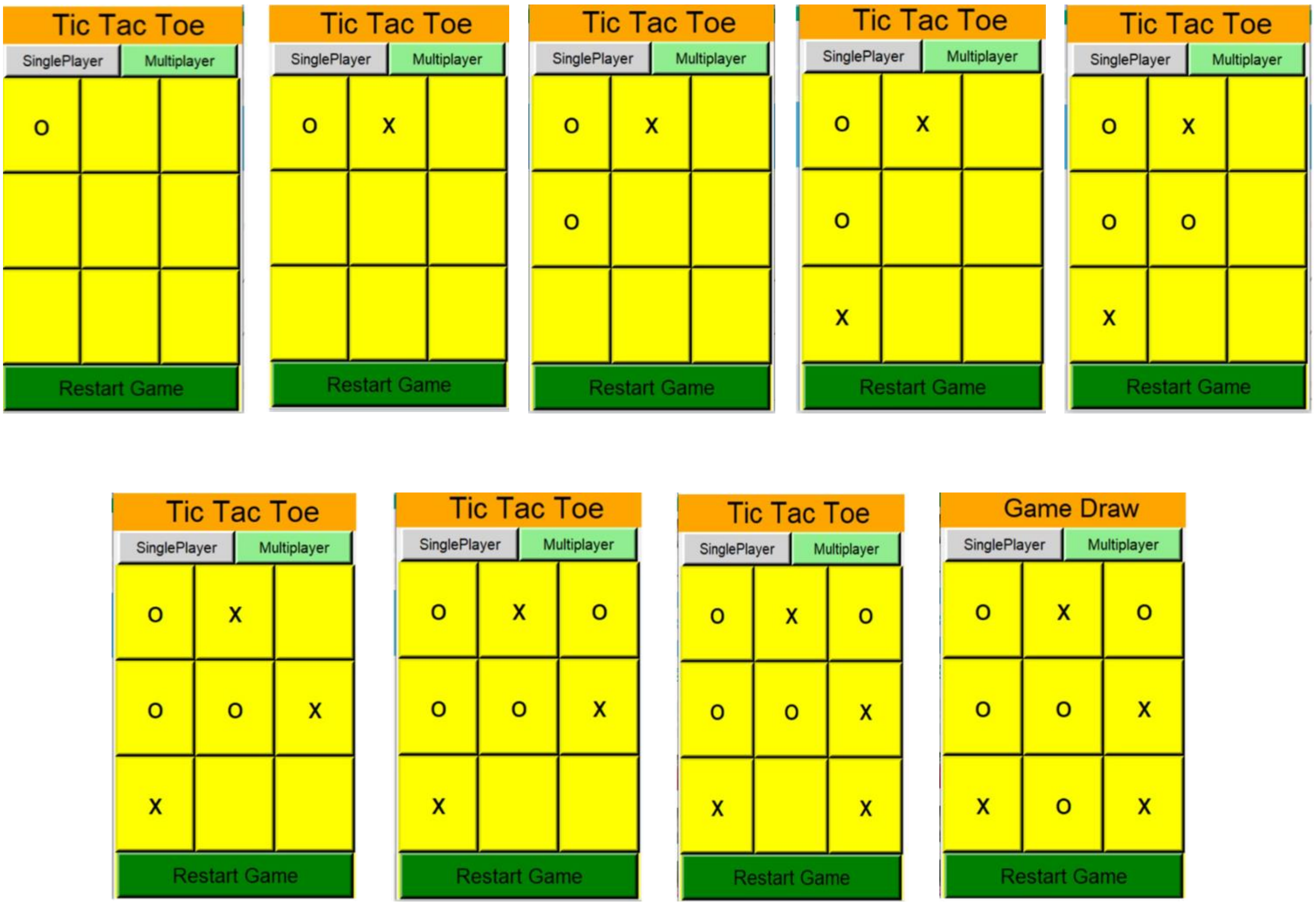

Fig. 5.Tic Tac Toe game when the game draws

In this scenario, figure 5 the image represents the final state of the game board, where all nine cells are occupied by a combination of 'X' and 'O' marks. However, despite the board being fully occupied, neither player has succeeded in aligning three of their marks horizontally, vertically, or diagonally. Both players have strategically maneuvered throughout the game, endeavoring to create winning lines or thwart their opponent's potential winning sequences. Nonetheless, their endeavors have culminated in a deadlock situation, where neither player has achieved the necessary alignment for victory. When all cells on the board are filled, and no player has formed a winning line, the game is deemed a draw. In this scenario, neither player emerges as the victor, and the game concludes in a tie. The image likely illustrates a intricate arrangement of 'X' and 'O' marks, with both players employing strategic placements to impede their opponent from securing a winning line. Nevertheless, their efforts have led to a stalemate, resulting in the draw condition.

The Tic Tac Toe game implemented using Python's Tkinter GUI is a success. it effectively manages win conditions for 'X' and 'O', along with draws. the code reliably detects wins and draws, offering players a visually pleasing interface. overall, it's a skillful implementation that follows game rules, improving user interaction.

The classic game of Tic-Tac-Toe has been implemented by researchers using different programming languages, techniques, and algorithms. We have conducted a thorough review of various reference papers on Tic-Tac-Toe implementations. To compare them effectively, an analysis is carried out in the table I. This allows for a clear understanding of each implementation's unique features and contributions

Table I. Comparative analysis of proposed work with previous research

SN	Features	Proposed research	Previous research
1.	Implementation Language	Python	Python, C
2.	GUI Library	Tkinter	Mostly console-based implementations
3.	Game Modes	Single-player, Multiplayer	Mostly focused on multiplayer
4.	AI Opponent	Minimax algorithm for single-player mode	AI opponents not explicitly mentioned or different algorithms used
5.	Mode Selection	Dedicated buttons for choosing modes	Mode selection not clearly specified
6.	Game Board	3x3 grid with clickable buttons	Various grid sizes (3x3, 4x4)
7.	Win/Draw Conditions	Comprehensive checks for rows, columns, diagonals, and full board	Similar win/draw condition checks
8.	Restart Game	Button for restarting the game	Restart functionality not explicitly mentioned
9.	Visual Elements	Labels and text for game title, status, winner/draw announcement	Less emphasis on visual elements
10.	User Experience	Intuitive GUI with interactive buttons and feedback	Some implementations may lack a user-friendly interface
11.	Simplicity	Emphasis on code simplicity for learning	-

6.CONCLUSION

In summary, the code demonstrates a practical implementation of the Tic Tac Toe game using Python's Tkinter library for the graphical interface. It constructs a user-friendly interface where players can interact with the game, selecting modes and making moves. The game logic handles player turns, checks for win or draws conditions, and allows for restarting the game. Notably, the single-player mode incorporates a basic AI using the Minimax algorithm, providing a challenging opponent.

This code serves as a valuable reference for understanding GUI development in Python and basic game programming concepts. Its simplicity makes it accessible for beginners, while the inclusion of AI adds depth for more advanced users. Overall, it stands as a solid foundation for learning and exploring further advancements in game development with Python and Tkinter.

The study employed game theory to analyze and devise an optimal strategy for near-endgame scenarios in tic-tac-toe. Each possible endgame circumstance was assigned a quantitative score, aiding in the identification of the ideal move and overall strategy. Utilizing the Minimax algorithm and the tic-tac-toe game tree, optimal movements and tactics were developed for each situation. Additionally, the concept of proof by fatigue was applied to demonstrate the effectiveness of the strategy, providing a comprehensive framework for decision-making in critical game positions.

7. FUTURE WORK

Tic Tac Toe could evolve into a surprisingly strategic game with a variety of features. Boards could expand beyond the classic 3x3 grid, transforming into n x n squares where "n" can be any positive number. This opens up a world of strategic possibilities.

AI can also be used to play with the user instead of just another person; the user could challenge a computer opponent that gets tougher as you play better. Beginners can practice against an easy computer, while experts can face a real brain teaser.

Features can also be added so that the game could keep a record of wins, losses, and ties. This way, the player can see how he/she is improving and maybe even compete with friends to be the Tic-Tac-Toe champion.

The game could offer friendly advice if you're having trouble deciding where to place your next X or O. This would be especially helpful for new players learning the ropes.

REFERENCES

[1] Abubakarsidiq Makame Rajab, Nadhira Haji Hassan and Abdul-rahimsidiq Makame Raja, “ Designing and Implementation of Tic-Tac-Toe-4X4 based Artificial Intelligence using Python Programming”, International Journal of Engineering and Information Systems (IJEAIS) issue 11,Vol.3 2019

[2] Sneha Garg, Dalpat Songara; Saurabh Maheshwari , “The winning strategy of Tic Tac Toe Game model by using Theoretical Computer Science” ,2017 International Conference on Computer, Communications and Electronics (Comptelix)- Manipal University Jaipur, Malaviya National Institute of Technology Jaipur & IRISWORLD, July 01-02, 2017

[3] Anurag Bhatt, Pratul Varshney, Kalyanmoy Deb - KanGAL Report Number 2007002 – “Evolution of No-loss Strategies for the Game of Tic-Tac-Toe”.

[4] Prof. Sumit S Shevtekar, Mugdha Malpe, Mohammed Bhaila, "Analysis of Game Tree Search Algorithms Using Minimax Algorithm and Alpha-Beta Pruning", International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT), ISSN : 2456-3307, Volume 8 Issue 6, pp. 328-333, November-December 2022.

[5] Karan, Chitaraj, et al. , “Implementation of a Tic-Tac-Toe Game Using Python Environment for Gaming Application”. Nov. 2022, https://doi.org/10.46647/ijetms.2022.v06i06.056.

[6] Pavitha N, Tanushri Bhuruk , “TIC-TAC-TOE IN C”,International Journal of Engineering and technology ,Volume: 09 Issue: 02 | Feb 2022

[7] James Shah, “The Classic Tic-Tac-Toe Game in Python 3” ,2019

[8] Shahd H. Alkaraz, Essam El-Seidy, Neveen S. Morcos , “Tic-Tac-Toe: Understanding the Minimax Algorithm”,Journal of Game Theory p-ISSN: 2325-0046 e-ISSN: 2325-0054 ,2020

[9] APPLICATION OF GAME THEORY IN TIC-TAC-TOE - https://prezi.com/nwmtgddhbfim/application-of-game-theory-in-tic-tac-toe/

[10] Garg, R.,et al., “Tic-Tac-Toe Game: Simulation using Min-Max Algorithm”, International Journal of Advanced Research in Computer Science, Vol. 8, No. 7, pp.1074-1077, 2017.

[11] Elvis Kobina Donkoh, et al “Application of Combinatorial Techniques to the Ghanaian Board Game Zaminamina Draft”, Eur. J. Pure Appl. Math., vol. 12, no. 1, pp. 159–175, Jan. 2019, doi: 10.29020/nybg.ejpam.v12i1.3308.

[12] B. Brendan O’Donoghue, "Min-Max Approximate Dynamic Programming", IEEE Multi-Conference on Systems and Control, pp. 28-30, 2011.

[13] T. G. Daly, "Game and gaming machine having tic-tac-toe type feature," ed: Google Patents, 2016.