VISVESVARAYA TECHNOLOGICAL UNIVERSITY
BELAGAVI

Mini Project Report on

"TIC TAC GAME"

Submitted in the partial fulfillment for the requirements of Computer Graphics & Visualization Laboratory of 6th semester CSE requirement in the form of the Mini Project work

Submitted By

RISHI RAJ **USN: 1BY19CS119**

SANKALP SHARMA **USN: 1BY19CS136**

Under the guidance of

Prof. Muneshwara M S
Assistant Professor
Dept. of CSE

Prof. Chethana C
Assistant Professor
Dept. of CSE

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BMS INSTITUTE OF TECHNOLOGY & MANAGEMENT

YELAHANKA, BENGALURU - 560064.

2021-2022

BMS INSTITUTE OF TECHNOLOGY &MANAGEMENT

YELAHANKA, BENGALURU – 560064

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the Project work entitled **"TIC TAC GAME"** is a bonafide work carried out by **Rishi Raj (1BY19CS119) and Sankalp Sharma (1BY19CS136)** in partial fulfillment for *Mini Project* during the year 2021-2022. It is hereby certified that this project covers the concepts of *Computer Graphics & Visualization*. It is also certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in this report.

Signature of the Guide
Prof. Muneshwara M S/
Prof. Chethana C
Assistant Professor
CSE, BMSIT&M

Signature of the HOD
Dr. Thippeswamy G
Professor & HOD
CSE, BMSIT&M

Name and Signature of the Examiner

Internal Examiner

External Examiner

INSTITUTE VISION

To emerge as one of the finest technical institutions of higher learning, to develop engineering professionals who are technically competent, ethical and environment friendly for betterment of the society.

INSTITUTE MISSION

Accomplish stimulating learning environment through high quality academic instruction, innovation and industry-institute interface.

DEPARTMENT VISION

To develop technical professionals acquainted with recent trends and technologies of computer science to serve as valuable resource for the nation/society.

DEPARTMENT MISSION

Facilitating and exposing the students to various learning opportunities through dedicated academic teaching, guidance and monitoring.

PROGRAM EDUCATIONAL OBJECTIVES

1. Lead a successful career by designing, analyzing and solving various problems in the field of Computer Science & Engineering.
2. Pursue higher studies for enduring edification.
3. Exhibit professional and team building attitude along with effective communication.
4. Identify and provide solutions for sustainable environmental development.

PROGRAM SPECIFIC OUTCOMES

1. Analyze the problem and identify computing requirements appropriate to its solution.
2. Apply design and development principles in the construction of software systems of varying complexity.

ACKNOWLEDGEMENT

We are happy to present this project after completing it successfully. This project would not have been possible without the guidance, assistance and suggestions of many individuals.

We would like to express our deep sense of gratitude and indebtedness to each and every one who has helped us make this project a success.

We heartily thank our Principal, **Dr. MOHAN BABU G N,** BMS Institute of Technology & Management, for his constant encouragement and inspiration in taking up this project.

We heartily thank our Professor and Head of the Department, **Dr. Thippeswamy G**, Department of Computer Science and Engineering, BMS Institute of Technology and Management, for his constant encouragement and inspiration in taking up this project.

We gracefully thank our Project Guide, **Prof. Muneshwara M S and Prof. Chethana C**, Assistant Professors, Department of Computer Science and Engineering for their intangible support and for being constant backbone for our project.

Special thanks to all the staff members of Computer Science Department for their help and kind co-operation.

Lastly, we thank our parents and friends for the support and encouragement given throughout in completing this precious work successfully.

RISHI RAJ (1BY19CS119)

SANKALP SHARMA (1BY19CS136)

ABSTRACT

Tic Tac Toe game, is a very famous game played by almost everyone. With the basis of the minimax algorithm for mathematical analysis, alongside speeding up the computation by alpha-beta-pruning concept and optimizing the utility function using the heuristic function, the 4X4 AI- based tic tac toe game is developed by us to defeat a human player. There are high chances that the AI will defeat the human. There are special functions made to ensure the smart play by the AI, instead of just random moves. These calculated moves ensure the smart play of the computer, instead of just randomly picking spots to plot the move. Although this doesn't ensure the winning of the AI. The player can outsmart the computer by critical thinking. When the player first plots the move on the corner, then the AI has higher chances of winning, but when the center box is chosen by the player before, then it becomes difficult for the AI to plot a winning move, because it is programmed to look for empty boxes on the corners, so when all the corners are empty, then it becomes a bit tricky for the computer to compute a plot a winning move. So the Player has higher chances of winning when the first move is in the center and the AI will be tricked.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION:

Tic Tac Toe game is one of the most famous games available on all types of device. **Tic Tac** is a video game genre where the player maneuvers a growing line that becomes a primary obstacle to itself. The concept originated in the 1976 two-player arcade game Blockade from Gremlin Industries, and the ease of implementation has led to hundreds of versions (some of which have the word tic tac or worm in the title) for many platforms. 1982's Tron arcade game, based on the film, includes tic tac gameplay for the single-player Light Cycles segment. The Tic tac design dates back to the arcade game Blockade, developed and published by Gremlin in 1976. It was cloned as Bigfoot Bonkers the same year. In 1977, Atari, Inc. released two Blockade-inspired titles: the arcade game Dominos and Atari VCS game Surround.] Surround was one of the nine Atari VCS launch titles in the US and was sold by Sears under the name Chase. That same year, a similar game was launched for the Bally Astrocade as Checkmate.

The first known home computer version, titled Worm, was programmed in 1978 by Peter Trefonas for the TRS-80, and published by CLOAD magazine in the same year. This was followed shortly afterwards with versions from the same author for the Commodore PET and Apple II. A clone of the Hustle arcade game, itself a clone of Blockade, was written by Peter Trefonas in 1979 and published by CLOAD. An authorized version of Hustle was published by Milton Bradley for the TI-99/4A in 1980. The single-player Tic tac Byte was published in 1982 for Atari 8-bit computers, Apple II, and VIC-20; a tic tac eats apples to complete a level, growing longer in the process. In Tic tac for the BBC Micro (1982), by Dave Bresnen, the tic tac is controlled using the left and right arrow keys relative to the direction it is heading in. The tic tac increases in speed as it gets longer, and there's only one life.

Nibbler (1982) is a single-player arcade game where the tic tac fits tightly into a maze, and the gameplay is faster than most tic tac designs. Another single-player version is part of the 1982 Tron arcade game, themed with light cycles. It reinvigorated the tic tac concept, and many subsequent games borrowed the light cycle theme.

Starting in 1991, Nibbles was included with MS-DOS for a period of time as a QBasic sample program. In 1992, Rattler Race was released as part of the second Microsoft Entertainment Pack. It adds enemy tic tacs to the familiar apple-eating gameplay. Nokia puts Tic tac on the majority of their phones:

- Tic Tac – The first published by Nokia, for monochrome phones. It was programmed in 1997 by Taneli Armanto of Nokia and introduced on the Nokia 6110.
- Tic Tac II – Included on monochrome phones such as the Nokia 3310 from 2000.
- Tic Tac Xenzia – Included on later-model monochrome phones (and most cheaper colour phones, such as the Series 30 and Series 30+ budget mobile devices).
- Tic Tac EX – Included on colour phones. First introduced with the Nokia 9290 Communicator in 2002. It supports multiplayer through Bluetooth and Infra-Red.
- Tic Tac EX2 – Introduced with the Nokia 3100 in 2003 and included in several Series 40 handsets.
- Tic Tac – A 3D version designed for the N-Gage in 2005. It featured multiplayer through Bluetooth. Later Nokia started preinstalling it (without multiplayer) on some Nseries smartphones like N70, N73, N80, etc. It can be downloaded from Nokia support pages and played on any S60 device.
- Tic Tac III – A 3D version, different from Tic tacs. Tic tac III takes a more living tic tac approach, rather than the abstract feel of Tic tacs. An example of a phone with it installed is the Nokia 3250 from 2005, and it supports multiplayer modes via Bluetooth.
- Tic Tac Subsonic - Sequel to Tic tacs, released on May 22, 2008 for N-Gage 2.0.
- Tic Tac Xenzia (2017) - First released on the Nokia 3310.
- Tic Tac (2017) - Released with Facebook Messenger (2017)

1.2 MOTIVATION:

In today's day and age games have become a big part of the society. Many young people are indulging in gaming as a way of relaxing and a method of recreation. Other have been able to transform gaming into a career path and been able to earn money thus leading to job opportunities for gaming companies and the players themselves. Science has shown that gaming is also good for the brain it helps increase cognitive abilities and help you think differently so as to win the game due to this there has been the rise of gamification. Gamification is where video game designs and game elements in learning environments so as to motivate students. This helps to maximize enjoyment and engagement through capturing the interest of learners. It helps one learn while enjoy the game at the same time is able to learn something new from the topic at hand. Gamification has been seen to have good results in that it increases the engagement of students, user retention and knowledge with this method of learning being available I have come up with the idea genius tic tac game. As the name suggests this old age tic tac game with a few modifications to help students learn. The objective of the game is to get the highest score without hitting a wall. As the game progresses the tic tac moves at a faster pace. Here's the caveat, when the tic tac hits the wall the game prompts you to answer a maths question for you to be able to continue. When answered correctly the game continues from where you left off and your score continues to grow. The tic tac's speed is reduced so as to get the highest score possible.

1.3 SCOPE:

The application program developed can be used in various fields as follows:

- Entertainment World like in the form of gaming.
- Business World like selling it.
- Online World like making people play it online.

1.4 PROBLEM STATEMENT:

Due to the fact that games get addictive over time and can cause students not to study and focus on those games. This leads to poor grade in school and poor attendance. Genius tic tac game will help students gain knowledge as they play the game. As the brain is stimulated to play they can answer question and progressively get good at maths. This in turn will help them become better students and will give the the urge to study more so as to be able to answer questions and proceed in the game.

1.5 PROPOSED SYSTEM:

In the proposed system, the OpenGL is an graphic software system designed as a streamlined, hardware-independent interface to be implemented on many different hardware platforms. To achieve these qualities, no commands for performing windowing tasks or obtaining user input are included in OpenGL; instead, you must work through whatever windowing system controls the particular hardware you're using.

OpenGL doesn't provide high-level commands for describing models of three-dimensional objects. Such commands might allow you to specify relatively complicated shapes such as automobiles, parts of the body, airplanes, or molecules. With OpenGL, you must build up your desired model from a small set of geometric primitives - points, lines, and polygons. The interface between an application program and a graphics system can be specified through a set of functions the resides in a graphics library These specification are called the application programmer’s interface (API).The application programmer see only the API and is thus shielded from the details of both the hardware and the software implementation of the graphics library. The software drivers are responsible for interpreting the output of the API and converting this data to a form that is understood by the particular hardware.

1.6 LIMITATIONS:

- There are very rare systems available in the market which uses Open Gl functions.
- It can be Addictive.
- Playing it for longer period of time will hurt your eyes.

CHAPTER 2

LITERATURE SURVEY

Most of our applications will be designed to access OpenGL directly through functions in three libraries. Firstly functions in the main GL library have names that begin with the letters gl and are stored in the library usually referred to as GL .

The second is the OpenGL utility library (GLU).Third is the OpenGL utility toolkit (GLUT), which provides the minimum functionality that should be expected in any modern windowing system. With GLUT, our application structures its event handling to use callback functions. (This method is similar to using the Xt Toolkit, also known as the X Intrinsics, with a widget set.) For example, first you open a window and register callback routines for specific events. Then, you create a main loop without an exit. In that loop, if an event occurs, its registered callback functions are executed. Upon completion of the callback functions, flow of control is returned to the main loop.

Geometric data (vertices, lines, and polygons) follows the path through the row of boxes that include evaluators and per-vertex operations, while pixel data (pixels, images, and bitmaps) is treated differently for part of the process. Both types of data undergo the rasterization and per-fragment operations before the final pixel data is written into the frame buffer.

All data, whether it describes geometry or pixels, can be saved in a display list or processed immediately. When a display list is executed, the data is sent from the display list just as if it were sent by the application.

All geometric primitives are eventually described by vertices. If evaluators are used, that data is converted to vertices and treated as vertices from then on. Vertex data may also be stored in and used from specialized vertex arrays. Per-vertex calculations are performed on each vertex, followed by rasterization to fragments. For pixel data, pixel operations are performed, and the results are either stored in the texture memory, used for polygon stippling, or rasterized to fragments.

Finally, the fragments are subjected to a series of per-fragment operations, after which the final pixel values are drawn into the frame buffer.

CHAPTER 3

SOFTWARE REQUIREMENTS SPECIFICATION

A software requirement definition is an abstract description of the services which the system should provide, and the constraints under which the system must operate. It should only specify the external behaviors of the system. In software engineering, a functional requirement defines a function of a software system or its component. A function is described as a set of inputs, the behavior, and outputs (see also software). Functional requirements may be calculations, technical details, data manipulation and processing and other specific functionality that define what a system is supposed to accomplish. Behavioral requirements describing all the cases where the system uses the functional requirements are captured in use cases.

The various methods used in this project are as follows:- **myDisplay** The module draws the output on the screen and the functions in it. **myIdle** This module is used to modify the structure variables when the system is idle **myReshape** This module will reshape the window if the window is resized. **main_menu** This module specifies the action corresponding to menu entry. **Myinit** This module is used to initialize the structure variables.

Here, the coding of our project is done in Microsoft Visual C++ which is a commercial integrated development environment (IDE) with OpenGL (Open Graphics Library) which is a standard specification to produce 2D and 3D computer graphics. We use, the OpenGL Utility Toolkit called GLUT which is a library of utilities for OpenGL programs. OpenGL (Open Graphics Library) is a standard specification defining a cross-language, cross-platform API for writing applications that produce 2D and 3D computer graphics, describing a set of functions and the precise behaviors that they must perform. From this specification, hardware vendors create implementations - libraries of functions created to match the functions stated in the OpenGL specification, making use of hardware acceleration where possible. Hardware vendors have to meet specific tests to be able to qualify their implementation as an OpenGL implementation.

GLUT is the OpenGL Utility Toolkit, a window system independent toolkit for

writing OpenGL programs. It implements a simple windowing application programming interface (API) for OpenGL. GLUT makes it considerably easier to learn about and explore OpenGL programming. GLUT provides a portable API so you can write a single OpenGL program that works across all PC and workstation OS platforms.

3.2 Non Functional Requirements

The non-functional requirement elaborates a performance characteristic of the system.Non-functional requirements impose constraints on the design or implementation (such as performance engineering requirements, quality standards, or design constraints).The following are the functional requirements for our application as listed below:

 Performance- The response time is short for the requested service, the application

provides user friendly interface.

 Reliability- The application is highly realistic and generates all update information

in correct order.

 Security- The application is secured and authentication is provided by proper login.

 Availability- The application is available all the time.

 Portability- The application is portable on any android device**.**

3.3 Hardware requirements

Hardware requirements is most common set of requirements defined by any operating system or software application is the physical computer resources, also known as hardware, A hardware requirements list is often accompanied by a hardware compatibility list (HCL), especially in case of operating systems. An HCL lists tested, compatible, and sometimes incompatible hardware devices for a particular operating system or application.

The following sub-sections discuss the various aspects of hardware requirements. Hardware that is required for this project can be listed as follows:

 Laptop/PC for Android Application Development.

 Server (Windows 7/8/10 (32-bit or 64-bit)).

3.4 Software requirements

Software requirements deal with defining software resource requirements and prerequisites that need to be installed on a computer to provide optimal functioning of an application. These requirements or prerequisites are generally not included in the software installation package and need to be installed separately before the software is installed. Software that are required for this project can be listed as follows:

 2 GB RAM minimum and 8 GB RAM recommended.

 2 GB of available disk space minimum.

 4 GB recommended (500 MB for IDE + 1.5 GB for Android SDK and emulator system image)

 1200x800 minimum screen resolution.

 Open Gl software.

 free Glut software.

 For accelerated emulator: 64-bit operating system and Intel® processor with support for Intel® VT-x, Intel® EM64T (Intel® 64), and Execute Disable (XD) Bit functionality

CHAPTER 4

SYSTEM ANALYSIS

System Analysis is a detailed study of the various operations performed by a system and their relationships within and outside of the system. Here the key question is- why all problems exist in the present system? What must be done to solve the problem? Analysis begins when a user or manager begins a study of the program using existing system. During analysis, data collected on the various files, decision points and transactions handled by the present system. The commonly used tools in the system are Data Flow Diagram etc. Training, experience and common sense are required for collection of relevant information needed to develop the system. The success of the system depends largely on how clearly the problem is defined, thoroughly investigated and properly carried out through the choice of solution. A good analysis model should provide not only the mechanisms of problem understanding but also the frame work of the solution. Thus, it should be studied thoroughly by collecting data about the system. Then the proposed system should be analyzed thoroughly in accordance with the needs. System analysis can be categorized into four parts.

- System planning and initial investigation
- Information Gathering
- Applying analysis tools for structured analysis
- Feasibility study
- Cost/ Benefit analysis.

All geometric primitives are eventually described by vertices. If evaluators are used, that data is converted to vertices and treated as vertices from then on. Vertex data may also be stored in and used from specialized vertex arrays. Per-vertex calculations are performed on each vertex, followed by rasterization to fragments. For pixel data, pixel operations are performed, and the results are either stored in the texture memory, used for polygon stippling, or rasterized to fragments. The second is the OpenGL utility library (GLU).Third is the OpenGL utility toolkit (GLUT), which provides the minimum functionality that should be expected in any modern windowing system. With GLUT, our application

structures its event handling to use callback functions. (This method is similar to using the Xt Toolkit, also known as the X Intrinsics, with a widget set).

CHAPTER 5
SYSTEM IMPLEMENTATION

void initLight(): This function will provide the lighting to the game plate.

void myInit(void): This function will provide plate.

void resize(int w, int h): This function will re size the tic tac.

void Write(char *string): This function will make us write on the screen.

void ManipulateViewAngle(): This function will help us in changing the viewing angles od the plate.

void Reset(): This function will help us in resenting the game back to start mode.

void WelcomeScreen(): This function will help us to create the welcome screen of our project.

CHAPTER 6

INTREPRETATION OF RESULTS

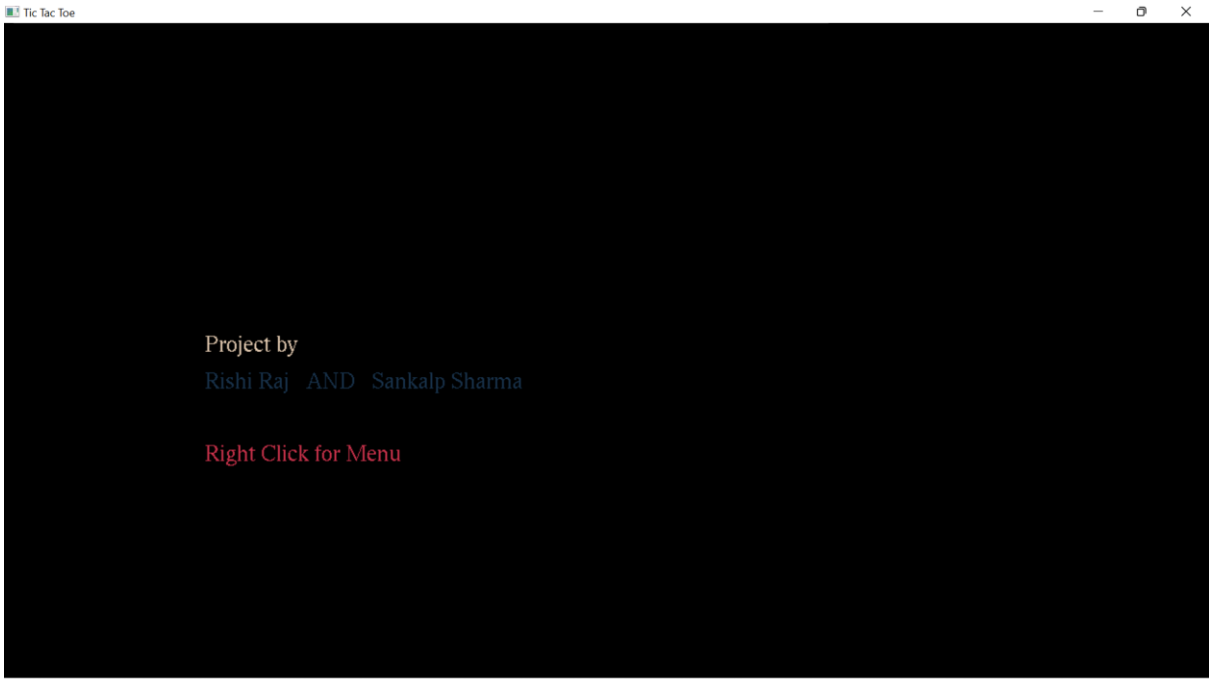

Figure 1. Home Page

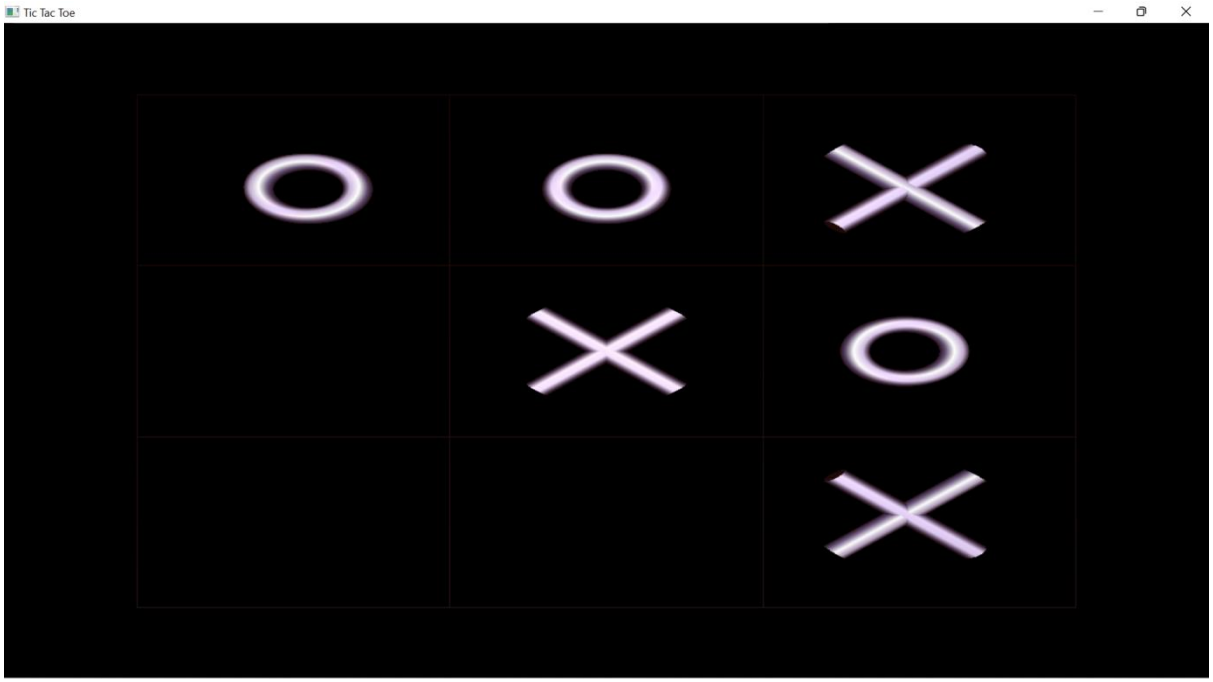

Figure 2. Game page

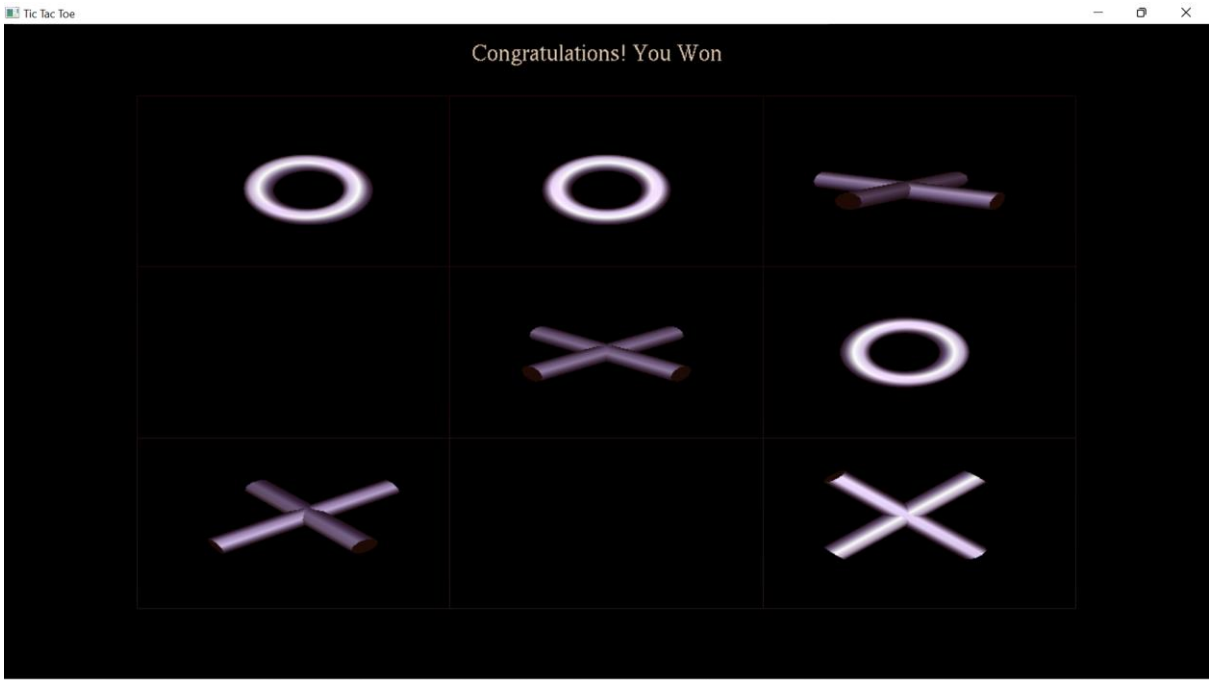

Figure 3. Player Wins

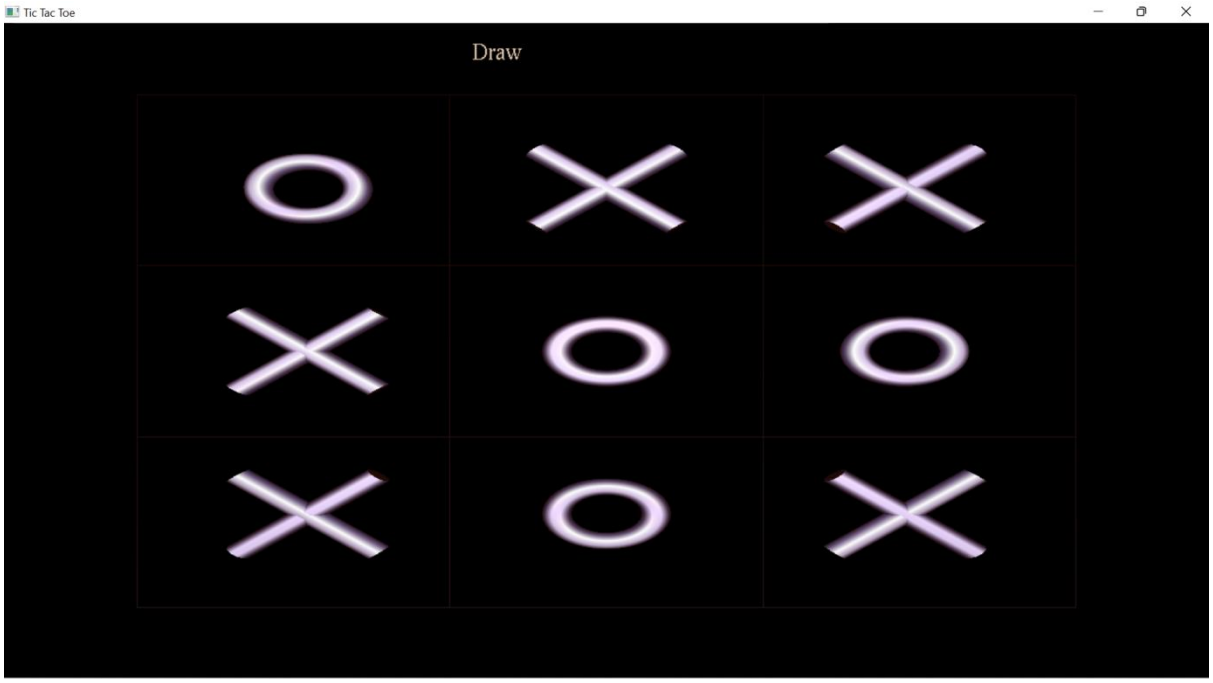

Figure 4. Draaw

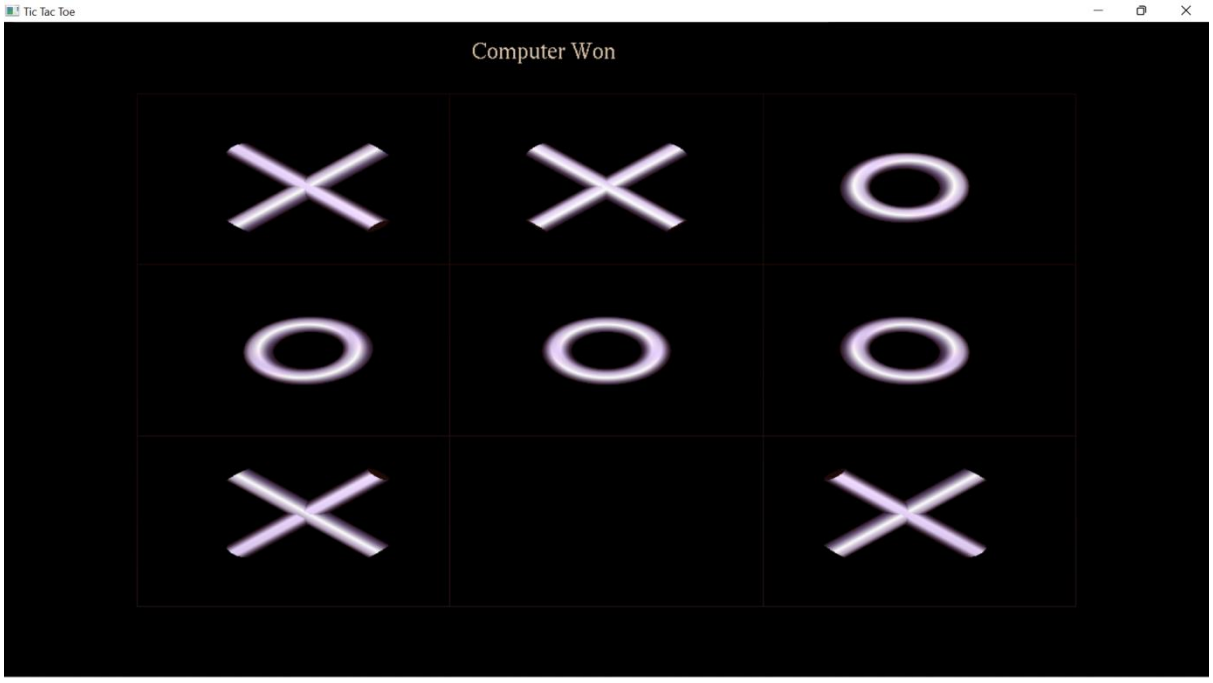

Figure 5. Computer Wins

CHAPTER 7

CONCLUSION

The coding of Tic tac was extremely difficult with many errors arising. Many systems had to be written numerous ways before a final working solution was found. For example, two different movement methods were used prior to final version; however, even the final version is flawed as vertical movement causes the tic tac to change scale. There were also issues with the food – tic tac collision detection. While the final version resulted in a tic tac that could eat food, the movement glitch caused the food to cause further size issues.

Despite the fact that the game could not truly be played due to the fact no score could be given, the game is still satisfying. With the exception of the size glitch when turning, the tic tac responds to user input and moves around the screen as directed. Given longer to work on this, the collision detection with the movement would be the first thing fixed. By fixing this, all other sections of code that are currently not working would run. The leaderboard would work as there would be correct scores input, and the tic tac would grow as the food would cause it to only increase by one and not varying numbers based on direction. In addition, fixing the movement would allow for the tic tac to die when colliding with itself. In the current state, the tic tac moves as a matrix so it can not kill itself as it would be impossible to move in any direction. This failure to establish a perfect movement system was the biggest disappointment of the game as all other problems stemmed from it.

For these reasons, it is recommended that anyone who wishes to recreate this game starts simply when writing the code. It is advisable that they first perfect the tic tacs movement controls before messing with the food generation. By taking the code in small sections, it is easier to get individual features to work. Building off this, use functions to contain each aspect of the game. Using functions made it easier to determine where errors were occurring when debugging the code. It also kept the code more organized.

CHAPTER 7

FUTURE ENHANCEMENT

In the future we are planning to add several new functionalities to our game such as different difficulty levels which will help to make our game more challenging and fun to play. We are also planning to implement the players profile menu so that a player can change his personal settings from the profile section. This game is currently a single player game which we are planning to make a multiplayer in near future for better gaming experience. In the current implementation the game gets over as soon as the tic tac hits itself in near future we are planning to add features where the tic tac will die once it hits the walls of the map this will also improve the challenging levels of the game. We will also planning to add the kill effects and effects when the tic tac eats the food this will improve the user experience as it will improve the user-interaction of the game.

BIBLIOGRAPHY

[1] IEEE: *https://www.cse.iitd.ac.in/~cs5110283/projects.html*

[2] Tic tac Game: *https://en.m.wikipedia.org/wiki/Tic tac _(video_game_genre)*

[3] Visual Studio : *https://visualstudio.microsoft.com/*

[4] Code Blocks : *https://www.codeblocks.org/*

[5] Free Glut : *http://freeglut.sourceforge.net/*