## UNIVERSITY VISVESVARAYA COLLEGE OF ENGINEERING BANGALORE UNIVERSITY



**K.R CIRCLE, BANGALORE-560001**

**COMPUTER GRAPHICS MINI PROJECT ON** **MINESWEEPER**

**GAME SUBMITTED BY**

**VISHAL VINAY RAM (20GANSE064)**

**UNDER THE GUIDANCE OF**

**Dr.**

**ARUNALATHA JS**

## Department of Computer Science and Engineering UVCE, Bangalore

**UNIVERSITY VISVESVARAYA COLLEGE OF ENGINEERING**

**BANGALORE UNIVERSITY,**



**K.R CIRCLE, BANGALORE-560001**

**CERTIFICATE**

This is to certify that **VISHAL VINAY RAM** have successfully completed the mini project work on **‘MINESWEEPER”**and submitted in partial fulfilment of the requirements of 5th semester Bachelor of Technology, Information Science and Engineering, prescribed by the **BANGALORE UNIVERSTY** during the academic year 2020-2021.

Dr. H S VIMALA Dr.ARUNALATHA JS

Chairman Professor

Department of Computer Science Department of Computer Science

and Engineering . and Engineering.

**Signature of the examiners: Signature of the examiners:**

**……………………….. ………………………..**

We take this opportunity to thank our institution **UNIVERSITY VISVESVARAYACOLLEGE OF ENGINEERING** for giving us an opportunity to carry out this project.

**ACKNOWLEDGEMENT**

We would like to thank **Dr. VENUGOPAL K R, Vice-Chancellor** and Dr. **RAMESH N Principal, UVCE,** for providing us all the facilities to work on this project. We indebted to him for being our pillar of strength and inspiration.

We wish to place our grateful thanks to **Dr. H. S. Vimala, Chairperson and Professor, Department of Computer Science and Engineering, UVCE,** who helped us to make our project a great success.

We are grateful to our Guide **Dr. Arunalatha JS, Assistant Professor, Department of Computer Science and Engineering, UVCE,** for his valuable suggestions and support, which has sustained us throughout the course of the project. We express our sincere thanks for her support. We also thank **Mr. Suman G, Research Scholar, Department of Computer Science and Engineering, UVCE** for his suggestions and support. We thank the lab staff and office staff for their extensive support throughout the CG laboratory.

We also thank our parents and friends for their help and encouragement.

- VISHAL VINAY RAM

20GASNE064

ABSTRACT

This mini-project demonstrates a simple game of **Mine-Sweeper**. Minesweeper is a single-player puzzle video game. The objective of the game is to clear a square board containing hidden "mines" without detonating any of them, with help from clues about the number of neighboring mines in each field. The game is played by revealing squares of the grid by clicking each square. If a square containing a mine is revealed, the player loses the game. If no mine is revealed, a digit is instead displayed in the square, Indicating how many adjacent squares contain mines; if no mines are adjacent, the square becomes blank. The player uses this information to deduce the contents of other squares, and may either safely reveal each square or mark the square as containing a mine by placing a flag.

Our implementation of this game has a small twist which we call “**Wrap- Around**” where clicking on tiles on the corners of the map opens up fields on the opposite corners of the game area, hence making the game more challenging.

# CONTENTS

|  |  |
| --- | --- |
| **Chapters** | **Page No.** |
| **1. Introduction** | **1-6** |
| 1.1 Motivation and OpenGL | 1 |
| 1.2 Basic | 5 |
| 1.2Problem Section Statement | 5 |
| 1.3 Existing Statement | 5 |
| 1.4 Proposed System | 5 |
| 1.5 Objectives of The Project | 6 |
| **2**. **Requirement Specification** | **7** |
| **Specification** |  |
| 2.1 User Requirements | 7 |
| 2.2 Hardware Requirements | 7 |
| 2.3 Software Requirements | 7 |
| **3. Design and Implementation** | **8-10** |
| **4. Screen Shots** | **10-11** |
| **5. Conclusion & Future Work** | **12** |
| **6. References** | **13** |

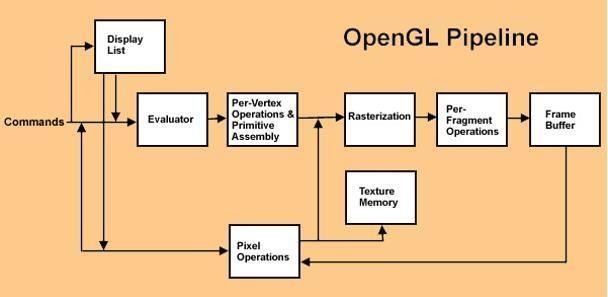
**LIST OF FIGURES**

|  |  |
| --- | --- |
| **Figure Name** | **Page No.** |
| Fig 4.1 OpenGL Pipeline | 8 |
| Fig 6.1 Initial Screen of the game | 10 |
| Fig 6.2 Gameplay | 10 |
| Fig 6.3 Game-Over Screen | 11 |
| Fig 6.3 Game-Won Screen | 11 |

**CHAPTER 1**

# INTRODUCTION

This report contains implementation of “Minesweeper” using a set of OpenGL functions. The objects are drawn by using GLUT functions. This project has been developed using windows 10 (codeblocks) with OpenGL package. It is a game in which a few blocks have objects with colors hidden inside. Move the focus on to an object and press enter to uncover it. As soon as you move to non-bomb block this they reveal the score they have and when u click on a bomb it explores and the game ends. You have to uncover all blocks with object hidden behind to clear both the blocks. This is a luck or strategical game.

**functions that can be used to manipulate graphics and images. OpenGL draws primitives: points, line segments, or polygons—subject to several selectable modes and can control modes independently of each other.**

## 1.1 Motivation

The main aim is to implement the skills learnt in computer graphics theory and computer graphics lab programs using OpenGL software which includes the concepts like: Lighting, Projection, Rotation, Motion, Clipping, Transformation, Viewing, Visible surface detection, Input and interaction, Curves and Computer Animation. The application of this particular project motivates to learn lots of concepts which demonstrate the motion of the ball which can be observed in the animation along with the camera effect that makes the viewer control the camera as they please to see the animation.

## 1.2 Basic Idea

Project’s main goal is to design a game using OpenGL. This project displays a 2D array of square blocks which display colored objects when selected. The user has to determine the location of each every grid which contains the points . The game ends when a grid containing a bomb is clicked or when you reveal all the non bomb squares. The score is calculated based on the the points behind the non-bomb grids. With the help of various OpenGL functions, it’s managed

* + - To create a 2D array of rectangular boxes.
    - To create colored objects that pop up when the boxes are selected.
    - To create a window for the game using OpenGL create window.

**1.2.1 OpenGL** (Open Graphics Library)**:**

OpenGL has become a widely accepted standard for developing graphics application. OpenGL is easy to learn, and it possesses most of the characteristics of other popular graphics system. It is top-down approach. OpenGL[8] is a standard specification defining a cross-language, cross-platform API for writing applications that produce 2D and 3D computer graphics. The interface consists of over 250 different function calls which can be used to draw complex three-dimensional scenes from simple primitives.

OpenGL was developed by Silicon Graphics Inc. (SGI) in 1992and is widely used in CAD, virtual reality, scientific visualization, information visualization, and flight simulation. It is also used in video games, where it competes with Direct3D on Microsoft Windows platforms.

The interface between the application program and the graphics system can be specified through that set of function that resides in graphics library. The specification is called the APPLICATION PROGRAM INTERFACE (API). The application program sees only the API and is thus shielded from the details both the hardware and software implementation of graphics library. The software driver is responsible for interpreting the output of an API and converting these data to a form that is understood by the particular hardware.

Most of our applications will be designed to access openGL directly through functions in three libraries. Function in the main GL library have name that begin with the letter gl and stored in the library. The second is the openGL utility Library (GLU). This library uses only GL function but contains codes for creating common object and viewing. Rather than using a different library for each system we used available library called openGL utility toolkit (GLUT). It used as #include <glut.h>

A graphics editor is a computer program that allows users to compose and edit pictures interactively on the computer screen and save them in one of many popular “bitmap” or “raster” a format such as TIFF, JPEG, PNG and GIF.

Graphics Editors can normally be classified as:

* 2D Graphics Editors.
* 3D Graphics Editors.

A 3D Graphics Editor is used to draw 3D primitives Rectangles, Circle, polygons, etc. and alter those with operations like cut, copy, paste. These may also contain features like layers and object precision etc.

3D Graphics Editor should include the following features:

* Facilities: Cursor Movement, Editing picture objects.
* Good User Interface: GUI / Toolbars / Icon based User Interface.

Computer Graphics is concerned with all aspects of producing pictures or images using a computer. A particular graphics software system called OpenGL, which has become a widely accepted standard for developing graphics applications.

The applications of computer graphics in some of the major areas are as follows

* + 1. Display of information.
    2. Design.
    3. Simulation and Animation.
    4. User interfaces.

Our mini-project titled “” uses OpenGL software interface and develops 2D images.

This project uses certain techniques like Transformation, sounds, display list, etc.

## 1.2.2 Existing Systems

The existing system involves computer graphics. Computer graphics started with the display of data on hardcopy plotters and cathode ray tube screens soon after the introduction of computer themselves. It includes the creation, storage and manipulation of models and images of objects.[3]

These models include physical, mathematical, engineering, architectural and so on Computer graphics today is largely interactive –the user controls the contents, structure and appearance of objects and their displayed images by using input devices, such as keyboard, mouse or touch-sensitive panel on the screen. Interactive computer graphics is the most important means of producing pictures since the invention of photography and television.

## 1.3 Proposed System

In proposed system, the OpenGL is a 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.[2]

**1.4 Problem Section Statement**

Computer graphics is no longer a rarity. It is an integral part of all computer user interfaces, and is indispensable for visualizing 2D, 3D and higher dimensional objects. Creating 3D objects, rotations and any other manipulations are laborious process with graphics implementation using text editor. OpenGL provides more features for developing 3D objects with few lines by built in functions.

The geometric objects are the building blocks of any individual. Thereby developing, manipulating, applying any transformation, rotation, scaling on them is the major task of any image development.

Thereby we have put our tiny effort to develop 2D objects and perform different operations on them by using OpenGL utilities.

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.

## 1.5 Objectives Of The Project

* Developing a package using computer graphics with OpenGL.
* To show that implementation of Depth is easier with OpenGL.
* Implementing certain technical concept like Translation, motion, and use of Idle Function.
* How to use Buffer algorithms to remove hidden surfaces.

## 1.6 Scope

## The player starts the game & can traverse 2-dimentionally from one box to another and select the boxes using the keyboard. The score is calculated based on the least number of selections made to match all the objects with their pair.

## Summary

## This chapter introduces the given problem and its solution and the introduction to required software OpenGl and its utility toolkit. This also explains the motivation, basic idea, proposed model, scope and objectives of the solution.

## 

**CHAPTER 2**

# SYSTEM REQUIREMENT SPECIFICATION

## 2.1 User Requirements

* + - Easy to understand and should be simple.
    - The built-in functions should be utilized to the maximum extent.
    - OpenGL library facilities should be used.

## 2.2 Hardware Requirements

* + - Intel Pentium CPU 2.6 GHz or AMD Athlon 64 (K8) 2.6 GHz or higher
    - 1 GB RAM or more
    - Mouse
    - Keyboard 108 standard
    - Monitor resolution 800x600

## 2.3 Software Requirements

* + - OpenGL Tools (FreeGLUT for Linux systems)
    - OpenGL Extension Wrangler Library (GLEW)
    - Mesa OpenGL Tools
    - Linux 32-Bit Operating System
    - G++ compiler for the Linux platform

**2.4 Functional Requirements**

**OpenGL APIs**

To have a control on the flow of program and to interact with the window system then OpenGL API’S are used.

**GL/glut.h**

Readily available library called the OpenGL Utility Toolkit (GLUT), which provides the minimum functionality that should be expected in any modern windowing system is used.

**IDE**: Emacs (with C-Make enabled for Linux)

**CHAPTER 3**

# DESIGN AND IMPLEMENTATION

To design the ‘Wrap-around Minesweeper Game’ using the glut library, we need to understand various concepts, components and utility functions that are essential to implement/integrate the required visual (and audio) effects. The function containing the main algorithm is **grid\_sweeper** which checks the if the mains are present that grid.

## 3.1 Design and algorithm

START

THE PLAYER STARTS OPENING ALL THE GRIDS

AND CHECKS IF A BOMB

1. **NO**

NO

IF SELECTED GRID

CONTAINS BOMB

IF ALL NON BOMB IS GRID IS NOT EMPTY

YES

DISPLAYS WINNING SCREEN

DISPLAYS LOSING

SCREEN

STOP

1. **Algorithm 3.1: grid\_sweeper**

.

**Input**: Mouse is used for clicking on the grid .

**Output**: Win if all non-mine tiles are reveled or lose.

**Steps:**

**Step 1:** All the mines are placed n random positions.

**Step 2:** When the user starts selecting the based on if it a mine or not results are obtained.

**Step 3:** If the selected grid is a mine then end the game.

**Step 4:** If the selected grid is not a mine then search around the selected tile and increase the points on the tiles based on the closeness to the mine.

**Step 5:** If multiple bombs are present near a tile then make the points on the equal to 3.

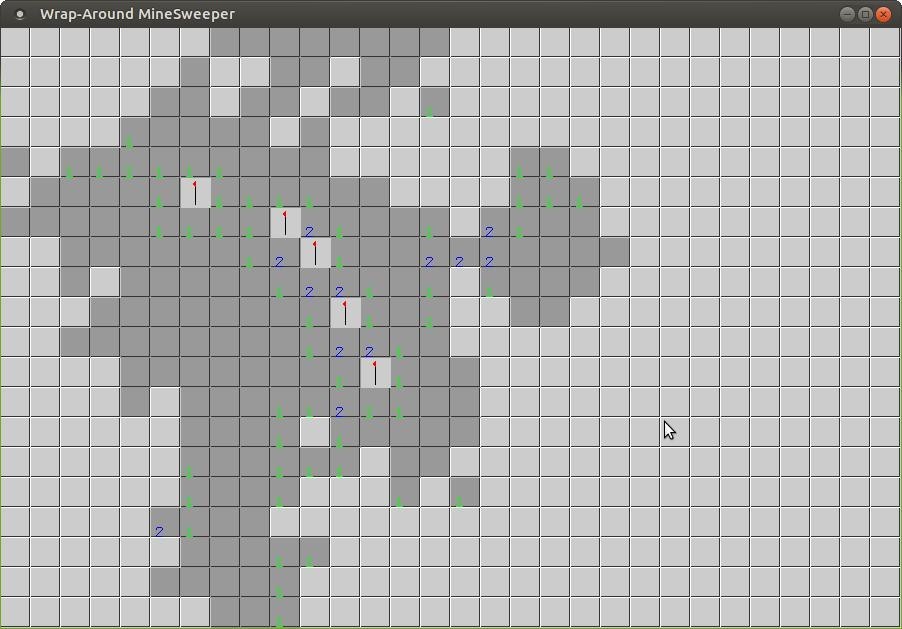
**Step 6:** If the user has selected all the non-mine tiles then end the game and declare him as the winner.

**Step 7:** Repeat step 4 to 5 until step 6 or step 3 condition is not satisfied .

**Summary**

This chapter mainly describes the algorithm used for implementation of the proposed project with flow of control and also the inputs and outputs of the project. Main part of the project is the display function whose working is designed above.

**CHAPTER 4**



# SNAPSHOTS

DEPT. OF CSE,UVCE

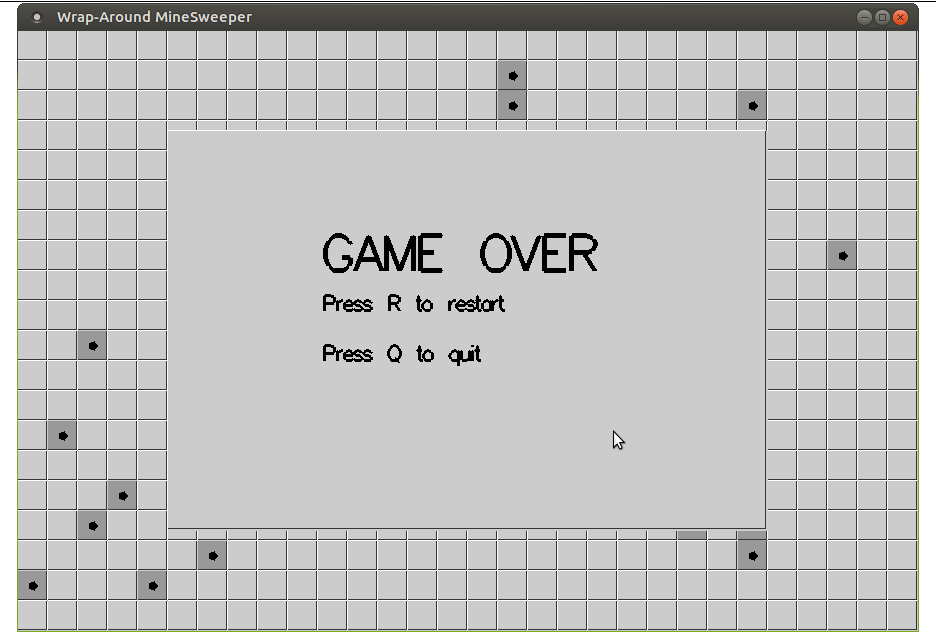
**4.1 Starting screen**

This the starting screen that gets booted when the code is built.

**FIG 4.2 GAMEPLAY**

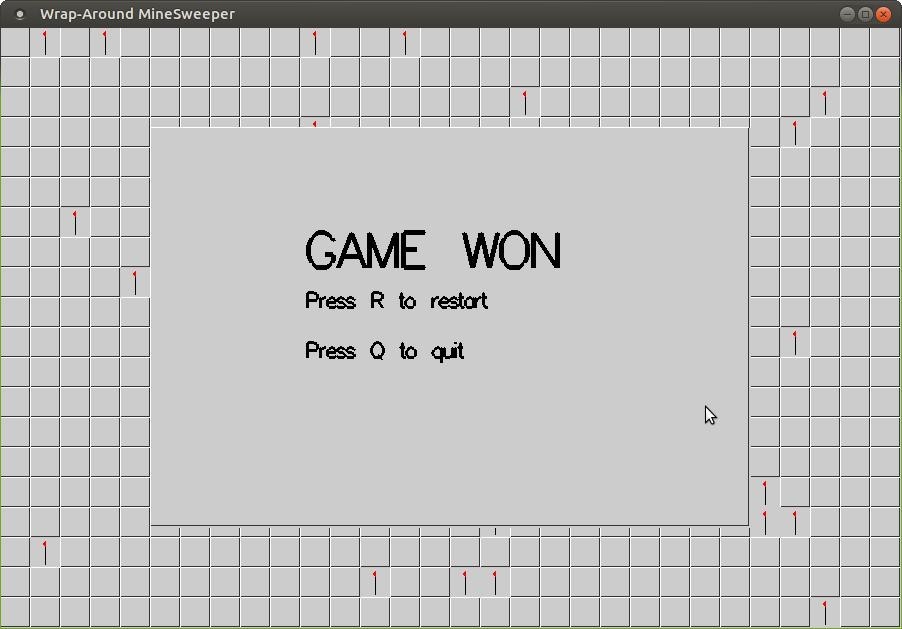
This is the actual playing screen where you get points and even bombs

**`**



#### **FIG 4.3 GAME-OVER SCREEN**

#### This is the screen that appears when a bomb is clicked

**FIG 4.4 GAME-WON SCREEN**

This the screen that gets displayed when you solve the entire the

**CHAPTER 5**

# CONCLUSION

An attempt has been made to develop an OpenGL graphics package, which meets necessary requirements of the users successfully. It enables us to learn about the basic concept in OPENGL graphics and graphics and know standard library graphics function and to explore some other function. OpenGL graphics is a huge library which consists of numerous functions.

The various shapes at lower level or to simulate any real thing animation etc. at high level. This project has given us an insight into the use of Computer graphics. As we had to use many built-in and user defined functions, we have managed to get a certain degree of familiarity with these functions and have now understood the power of these functions. We were able to comprehend the true nature of the most powerful tool graphics in OpenGL and have understood the reason why graphics is so powerful for programmers.

We can now converse with the certain degree of confidence about graphics in openGL. Finally, we have implemented this mini projection " Game" using openGL package. We would like to end by saying that doing this graphics project has been a memorable experience in which we have learned a lot, although there is a scope for further improvement. We got to know a lot of different applications of OPENGL while doing this project.

**FUTURE WORK**

Scope of further improvement:

* Various lightening effects can be implemented show shadows.
* The game can include 3D graphics and support for material properties.
* Explosion effects can be shown for the bombs.
* Celebration effects by showing fireworks can be done for winning.

**CHAPTER 5**

# REFERENCES

Books:

The books that helped us in implementing this project are as follows:

## [1] OpenGL Game Development by Example

Robert Madsen and Stephen Madsen, Packt Publishing Limited, 2016

* **[2] Computer Graphics** (OpenGL Version)

Donald Hearn and Pauline Baker, 2nd edition, Pearson Education, 2003

## [3] Interactive Computer Graphics

Edward Angel, 5th edition, Addison Wesley, 2009

Reference Websites:

1. freeglut.sourceforge.net/docs/api.php
2. [www.opengl.com](http://www.opengl.com/)
3. [www.learnopengl.com](http://www.learnopengl.com/)
4. [www.tutorialspoint.com](http://www.tutorialspoint.com/)
5. https://en.wikipedia.org/wiki/OpenGL

START

THE PLAYER STARTS OPENING ALL THE GRIDS

AND CHECKS IF A BOMB

**NO**

NO

IF SELECTED GRID

CONTAINS BOMB

IF ALL NON BOMB IS GRID IS NOT EMPTY

YES

DISPLAYS WINNING SCREEN

DISPLAYS LOSING

SCREEN

STOP