## horizontal line



Computer Graphics and Visualization

A Block Shooting Game - The Bull’s Eye

24.11.2017

**Professor**

V Badri Prasad

**Team Members**

**SANJUKTHA GEERAVANI B P**  - 01FB15ECS266

**MOHAMMED SALAMUDDIN** - 01FB15ECS175

# Table of Contents

**Chapter 1 Introduction**

1.1 OpenGL

1.2 GLUT Library

1.3 Game Development

**Chapter 2 Literature Survey**

2.1 Other games available

2.2 Tools used

**Chapter 3 Methodology**

3.1 Architecture design

3.2 Implementation

**Chapter 4 Results**

4.1 conclusion

**Chapter 5 Bibliography**

# Abstract

The aim of this project is to implement a game which we call as **Block Shooting Game**. This game contains **rectangular objects** ( boxes ) as targets, which has to be shot down using available number of **arrows**. We have made use of **OpenGL API, GLUT Library** and the C Programming Language to build our game the Block Shooting Game.

# Objective

1. Development of OpenGL API.
2. Develop a simple game and explore the OpenGL functions.

# Chapter 1 Introduction

## OpenGL

Open Graphics Library (OpenGL) is a cross-language, cross-platform application programming interface (API) for rendering 2D and 3D vector graphics. The API is typically used to interact with a graphics processing unit (GPU), to achieve hardware-accelerated rendering. The API is defined as a set of [functions](https://en.wikipedia.org/wiki/Subroutine) which may be called by the client program, alongside a set of [named integer constants](https://en.wikipedia.org/wiki/Enumerated_type) ( for example, the constant GL\_TEXTURE\_2D, which corresponds to the [decimal](https://en.wikipedia.org/wiki/Decimal) number 3553 ). Although the function definitions are superficially similar to those of the programming language [C](https://en.wikipedia.org/wiki/C_(programming_language)), they are language-independent.

Most of our application will be designed to access OpenGL directly through functions in three libraries. Functions in the main GL (or OpenGL in windows) library have names that begin with the letters gl and are stored in a library usually referred to as GL (or OpenGL in windows).

The second is the OpenGL Utility Library (GLU). This library uses only GL functions but contains code for creating common objects and simplifying viewing. All functions in GLU can be created from the core GL library but application programmers prefer not to write the code repeatedly. The GLU library is available in all OpenGL implementations; functions in the GLU library begin with letters glu.

To interface with the window system and to get input from external devices into our programs, we need at least one more system-specific library that provides the “glue” between the window system and OpenGL. For the Window System, this library is functionality that should be expected in any modern windowing system.

OpenGL is successful platform Independent API

1. Easy to use
2. Close enough to the hardware to get excellent results
3. Focus on rendering
4. Omitted windowing and input to Window System Dependencies.

GLUT

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.

GLUT is designed for constructing small to medium sized OpenGL programs. While GLUT is well-suited to learning OpenGL and developing simple OpenGL applications, GLUT is not a full-featured tool kit so large applications requiring sophisticated user interfaces are better off using native window system toolkits. GLUT is simple, easy, and small.

The toolkit supports:

1. Multiple windows for OpenGL rendering
2. Callback driven event processing
3. Sophisticated input devices
4. An 'idle' routine and timers
5. A simple, cascading pop-up menu facility
6. Utility routines to generate various solid and wireframe objects
7. Support for bitmap and stroke fonts
8. Miscellaneous window management functions

Game Development

Video game development is the process of creating a video game.The first video games were noncommercial, and were developed in the 1960s. They required mainframe computers to run and were not available to the general public for play. Commercial game development began in the 1970s with the advent of first-generation video game consoles and early home computers like the Apple I. Due to low costs and low capabilities of computers, a lone programmer could develop a full game. However, approaching the 21st century, ever-increasing computer processing power and heightened consumer expectations made it difficult for a single person to produce a mainstream console or PC game.

Development Process:

1. Prototyping
2. Game design
3. Production
4. Testing
5. Nearing Completion
6. Maintenance
7. Duration

Tools Required:

1. Programming Language
2. APIs and Libraries
3. Graphics APIs

# Chapter 2 Literature survey

Other games available

There are other games available with the same concept of shooting the object by pressing a key. One major difference which can be noticed is here in this game the objects are stationary and the arrows move upwards where as in the other games the arrows remain in a stationary position and the objects move. The objects vary from game to game. Some have also used balloons in place of objects.

Tools used

OpenGL tool is used to implement the game along with the GLU library. This library uses only GL functions but contains code for creating common objects and simplifying viewing.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.

Advantages of OpenGL is,It is by nature, not restricted to a single operating system. Offers quite a bit of low-level control for graphics. Supports a lot of programming languages.Pretty stable. Is extensible.Graphics based software tend to use OpenGL for their 3D back end.

Disadvantages is that It's only a graphics API, meaning that it does not handle anything more than graphics. Audio, controls, logic, etc. must be programmed in manually.

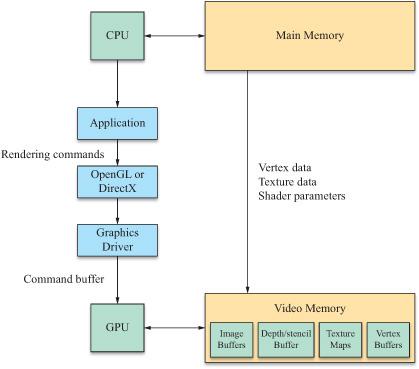
# 

# 

# 

# Chapter 3 Methodology

Architecture design



Application is layered on top of the OpenGL which interacts with the graphics .At the lowest level there's graphics device. These are GPUs which provide a set of registers controlling their operation (which registers exactly is device dependent) have some program memory for shaders, bulk memory for input data (vertices, textures, etc.) and an I/O channel to the rest of the system over which it recieve / sends data and command streams.

The graphics driver keeps track of the GPUs state and all the resources application programs that make use of the GPU.Then there's the driver dependent OpenGL client library/driver. On Windows this gets loaded by proxy through opengl32.dll, on Unix systems this resides in two places: X11 GLX module and driver dependent GLX driver and /usr/lib/libGL.so may contain some driver dependent stuff for direct rendering On Mac OS X this happens to be the "OpenGL Framework".

Finally the actual OpenGL API library, opengl32.dll in Windows, and on Unix /usr/lib/libGL.so; this mostly just passes down the commands to the OpenGL implementation proper.

Implementation

The GL/glut.h library is included first. Then we define the size of the screen by setting max-x and max-y values to 500 and 600 respectively. We have 10 counter variables which keeps the status of the objects,that is if the first object is hit than the counter corresponding to that object is set and the object size is changed to distinguish between the hit objects and objects which are not hit by the arrow.

The bitmap\_output() function displays a string to the screen using the glu library functio ns glRasterPos2f(x,y) and glutBitmapCharacter(font,string[i]) with the selected font. These functions are used to display the necessary comments onto the screen.

The counting() function checks if all the counters corresponding to the objects are set (meaning all the objects are hit by the arrow) and prints you won! on the screen and ends the game.

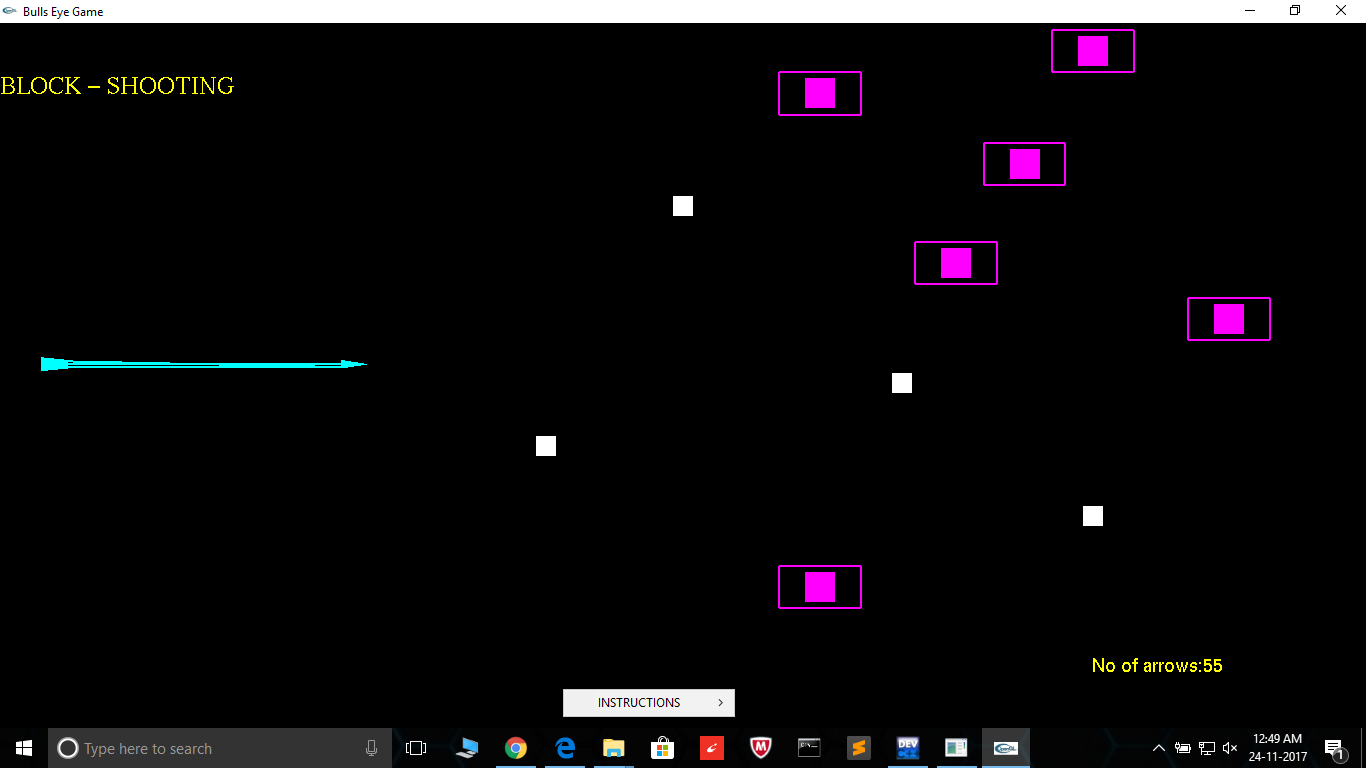
The hit\_target() changes the shape of the object when hit and the disp() function creates the arrow.

In the main function we first initialize the library,glutInitWindowSize(1000,1000) function is used to specify the size of the window,glutCreateWindow("Bulls Eye Game") function gives the title to the window,glutCreateMenu(demo\_menu) function creates the drop down menu when cleft click on the screen.glutKeyboardFunc(keys) function is used to fetch input from the keyboard.

**To play the game**

* Compile and run the C-code on the Dev C++ software.
* Right click on the window to see the instruction menu.
* Press R on the keyboard to shoot.
* Once all the objects are hit,the game is over.
* And if all the arrows are finished,the game is over.

# Chapter 4 Result



The arrow on the left hand side is shot by pressing letter ‘r’ on the keyboard. Pink objects are yet to be hit by the arrow,the while objects are already hit by the arrow. Bottom right corner shows the number of arrows used.

Conclusion

This report introduced to the Creative platform of OpenGL and told us about the GLUT libraries and implementation of Simple Games.OpenGL is evolving API.It’s not just awesome Platform for game development but pool of Innovation and Creativity.

The development of computer graphics has made computers easier to interact with and better for understanding and interpreting many types of data.Developments in computer graphics have had a profound impact on many types of media and have revolutionized the animation and Video Game Industry.

We started with the modest aim with no experience of OpenGL,but ended up in learning many things,fine tuning the programming and getting into real world of software development with exposure to corporate environment.During the development of any software of significant utility we face with some of the trade-offs between speed of execution and amount of memory consumed.This is simple interactive application.It has extremely user friendly features which makes simple graphics project.It is an open source code and no security features been included.The user is free to alter the code for free enhancement.Care has been taken to avoid bugs.

Wonderful learning experience and and we came to know how to develop simple games using OpenGL.

# Chapter 5 Bibliography

1. Wikipedia
2. <https://www.3dgep.com/introduction-opengl/>
3. [www.opengl-tutorial.org](http://www.opengl-tutorial.org) for tutorials
4. <https://open.gl/>