**COMPUTER GRAPHICS AND VISUALIZATION**

**(UE15CS321)**

**PROJECT REPORT**

**SPACE SHOOTING GAME**

**Mini Project Group No: 8**

**Mini Project Group Members :**

|  |  |  |  |
| --- | --- | --- | --- |
| **SRN** | **Name** | **Mobile** | **Email** |
| **01FB15ECS069** | **Bhoomika R Rao** | **+91 9449356644** | **bhoomikarao19@gmail.com** |
| **01FB15ECS328** | **Tvarita Jain** | **+91 94483504977** | **tvarita1297@gmail.com** |

**TABLE OF CONTENTS**

* 1. Introduction
  2. Literature Survey
  3. Methodology
  4. Game Architecture
  5. Results
  6. Bibliography

**Project Title:**

**Space Shooting Game using OpenGL and C++**

* 1. **INTRODUCTION:**
* What is OpenGL?

Open Graphics Library (OpenGL)[3][4] 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.

* Significance of the Gaming Industry :

The computer game industry (sometimes referred to[by whom?] as the interactive entertainment industry) is the economic sector involved in the development, marketing and monetizing of video games. It encompasses dozens of job disciplines and its component parts employ thousands of people worldwide.

* 1. **LITERATURE SURVEY:**

There have already been a lot of implementations of this game and pushed to the open source,

Some examples are :

* <http://oechup.com/getfile/NQwcnpj9b0s/simple-2d-space-shooter-game-with-opengl-and-c++-still-in-development->
* <https://github.com/channguyen/solar-system-opengl>
* <https://cboard.cprogramming.com/game-programming/87572-opengl-space-shooter-problems.html>

The methodology of these codes more or less is based on obstacle destruction , this gave us a lot of background in knowing how to structure our game and how to implement the common goal.

These changes were added to make it more efficient :

* Our code is more object oriented and has definite classes as entities for each component we design, this also increases the readability of our code for fellow developers.
* Moving on from a monotonous game , we have implemented level difficulties and have also included motion backgrounds

**1.3 METHODOLOGY**

**1.3.1 Game Overview :**

A simple 2-D space shooter moves only across x-axis. Obstacles move across y-axis, objective of the game is to dodge and shoot the obstacles with the help of the space-shooter.

**1.3.2 Implementation :**

OpenGL and C/C++.

**1.3.3 Controls**

* Mouse

moves the shooter along x-axis

* Keyboard

z shoot

x special feature to kill many obstacles at a time

**1.3.4 Gameplay**

* The goal is to survive and kill maximum number of obstacles as possible without run out of health
* The game flow would be

Open the application

Select 1 to play or 2 for instruction

If 1 then play ,Either kill or be killed or Play again

**1.4 GAME ARCHITECTURE**

**1.4.1 Modification**

Special feature made use is press button x when maximum number of obstacles come around as and when we press x the obstacle speed increases gradually.

**1.4.2 Effects**

Explosion -when enemy is killed

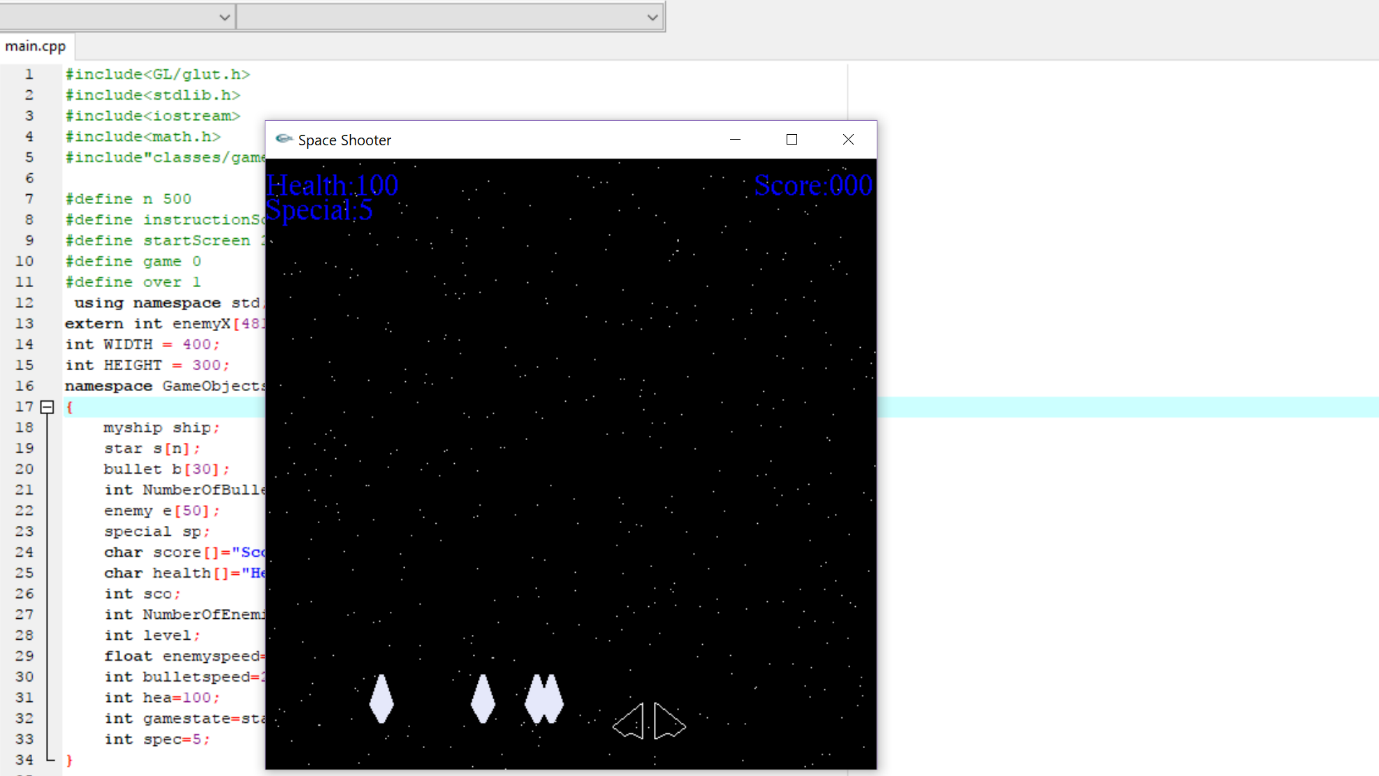
Stars - comes toward the players direction

**1.4.3 Classes and Responsibilities**

* Classes
  + bullet.h- on press of ‘x’ on keyboard it draws a line along that x point and shoots any obstacle coming on the way
  + enemy.h- draws enemy objects using opengl
  + myship.h- draws the shooter using opengl
  + special.h- to implement the special feature i.e on press of ‘z’ on keyboard clears all the enemy objects when multiple enemies are coming
  + star.h- makes an effect on the background (dotted pattern) which comes towards the player’s direction

**1.5 RESULTS :**

**Snapshots**



1. **Snapshot when ‘x’ was used as a weapon**



**(b)Snapshot when ‘z’ was used as a weapon**

**1.6 BIBLIOGRAPHY**

* + <https://github.com/kaushikj/OpenGL/blob/Public098/spaceship.cpp>
  + <https://www.youtube.com/watch?v=f1C3GjWMQNA>
  + [**http://www.openglprojects.in/#gsc.tab=0**](http://www.openglprojects.in/#gsc.tab=0)