CHAPTER - 1

INTRODUCTION

1.1 Introduction to Computer Graphics

COMPUTER GRAPHICS is concerned with all aspects of producing pictures or images using a computer. The field began humbly almost 50 years ago, with the display of a few lines on a cathode-ray tube (CRT); now, we can create images by computer that are indistinguishable from photographs of real objects. We routinely train pilots with simulated airplanes, generating graphical displays of a virtual environment in real time. Feature-length movies made entirely by computer have been successful, both critically and financially. Massive multiplayer games can involve tens of thousands of concurrent participants.

VISUALIZATION is any technique for creating images, diagrams or animations to communicate a message.

1.2. Image Types

➢ 2D computer graphics:

2D computer graphics are the computer-based generation of digital images mostly from two-dimensional models, such as 2D geometric models, text, and digital images, and by techniques specific to them.2D computer graphics are mainly used in applications that were originally developed upon traditional printing and drawing technologies, such as typography, cartography, technical drawing, and advertising. Two-dimensional models are preferred, because they give more direct control of the image than 3D computer graphics, whose approach is more akin to photography than to typography.

There are two approaches to 2D graphics: vector and raster graphics.

- Pixel art: Pixel art is a form of digital art, created through the use of raster graphics software, where images are edited on the pixel level.
- Vector graphics: Vector graphics formats are complementary to raster graphics, which is the representation of images as an array of pixels, as it is typically used for the representation of photographic images.

- **3D computer graphics:**

With the birth of the workstation computers(like LISP machines, paint box computers and Silicon Graphics workstations)came the 3D computer graphics.3D computer graphics in contrast to 2D computer graphics are graphics that use a three-dimensional representation of geometric data that is stored in the computer for the purposes0 of performing calculations and rendering 2D images.

1.3 Applications of Computer Graphics

Some of the applications of computer graphics are listed below:

- Computational biology
- Computational physics
- Computer-aided design
- Computer simulation
- Digital art
- Education
- Graphic design
- Video Games
- Virtual reality
- Web design

1.4 Introduction to OpenGL

As a software interface for graphics hardware, OpenGL's main purpose is to render two- and three-dimensional objects into a frame buffer. These objects are described as sequences of vertices (which define geometric objects) or pixels (which define images). OpenGL performs several processing steps on this data to convert it to pixels to form the final desired image in the frame buffer.

- **Basic OpenGL Operation**

The figure shown below gives an abstract, high-level block diagram of how OpenGL processes data. In the diagram, commands enter from the left and proceed through what can

be thought of as a processing pipeline. Some commands specify geometric objects to be drawn, and others control how the objects are handled during the various processing stages.

1.5 Introduction to 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 on both Win32 PCs and X11 workstations.

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 toolkit so large applications requiring sophisticated user interfaces are better off using native window system toolkits like Motif. GLUT is simple, easy, and small. My intent is to keep GLUT that way.

The GLUT library supports the following functionality:

- Multiple windows for OpenGL rendering.
- Call back driven event processing.
- An `idle' routine and timers.
- Utility routines to generate various solid and wire frame objects.
- Support for bitmap and stroke fonts.
- Miscellaneous window management functions.

1.6 Overview of the project

- OpenGL Glut program can be mainly divided into 4 parts. First part includes initialization with library and necessary files. Second part include other necessary functions and third is the display functions which draw the objects on screen. Final part is the main functions. In this OpenGL animation example, we will see 4 parts separately.

- This program will teach us how to draw an objects on the screen and event functions. The events can be generated using keyboard and mouse.
- In this program there are four functions – main, display, keyboard, and mouse. Here we need to understand more about the display, keyboard, and mouse functions.
- The display function has all the code about the objects that is going to display on the screen. The various objects drawn are – Ship, Iceberg, and Sea. Each of these are coded separately and placed in the display function. A global variable are declared which helps in control the animation as per the user presses the particular buttons on keyboard and mouse.

1.7 Aim of the project

The aim of the project is to develop a suitable graphic animation using various functions present in OpenGL GLUT libraries and to implement the skills learned in Interactive Computer Graphics and Visualization theory, using OpenGL.

CHAPTER - 2

SOFTWARE REQUIREMENT SPECIFICATION

2.1 Software Requirements

- Operating System : Windows 98/XP or Higher
- Programming Language : C,C++
- Microsoft Visual Studio 2005 or higher: This Software package containing visual basics in C++ language is required.
- Toolkit : GLUT Toolkit, VC++

2.2 Hardware Requirements

This package has been developed on:

- Processor : Pentium Processor
- Processor Speed : 333 MHz
- RAM : 32 MB or Higher
- Graphics Card : 512MB
- Monitor : Color
- Keyboard : Low Profile, Dispatch able Type
- I/O Parts : Mouse, Monitor

CHAPTER - 3

DESIGN

When first run, the project opens with a window size of 1366 by 768 pixels. The animation starts as soon as you run the project. You can pause and play again the animation using mouse left and right button respectively. You also can control the speed of the ship. ‘s’ or ‘S’ key to decrease the speed and ‘f’ or ‘F’ key to increase the speed of the ship. And finally by pressing ‘e’ or ‘E’ key you can exit the animation.

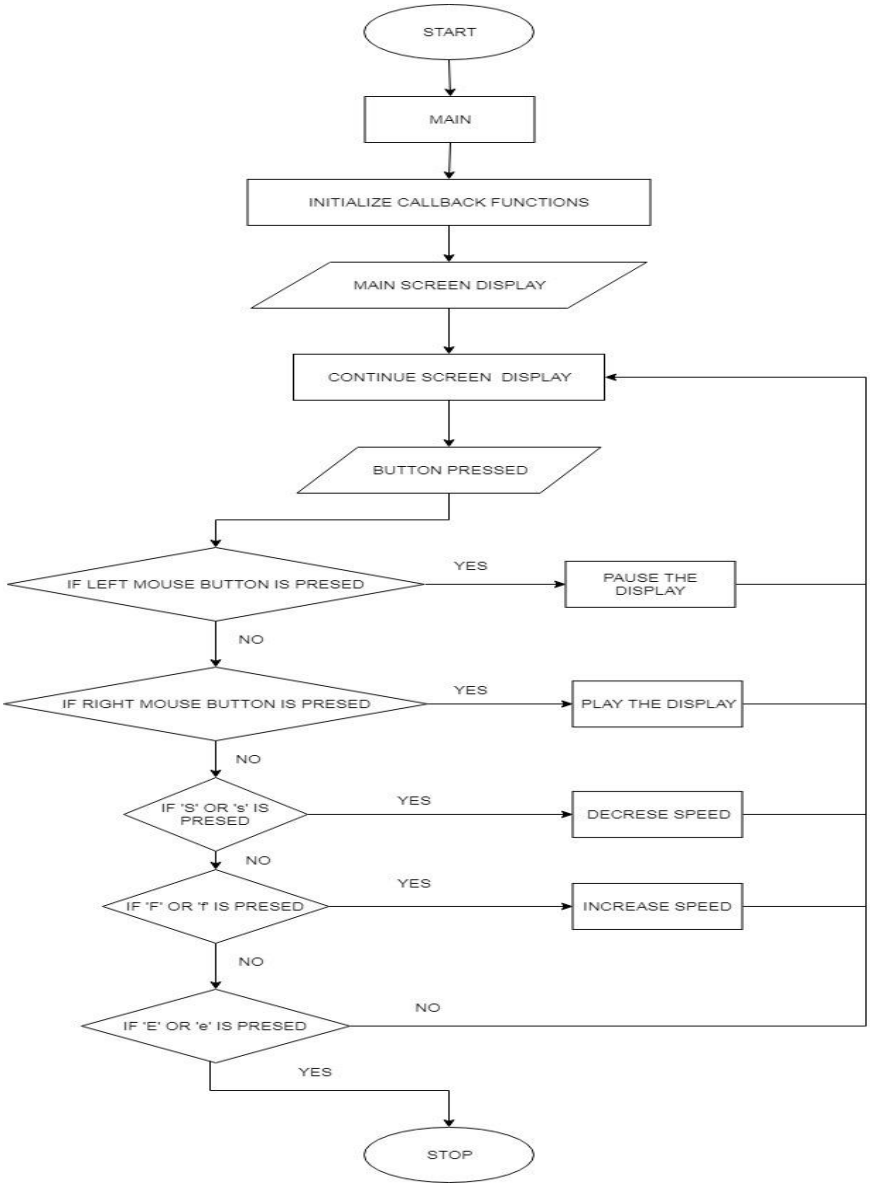

Fig 3.1: Flowchart of Ship and Iceberg Collision

CHAPTER - 4

IMPLEMENTATION

4.1 OpenGL Functions

The various functions used in implementing this project are:

- **glutInit(&argc,argv)**
  - This function is used to initialize glut.

- **glutInitDisplayMode()**
  - This function is used to initialize a display mode for the screen. It can choose one of the following constant values as it's parameters:
    - GLUT_SINGLE
    - GLUT_RGB

- **glutInitWindowPosition()**
  - This function is used to set the position of the top left corner of the window. It takes in 2 integer values as input parameters which are the coordinates specified for the position for the top left corner of the window.

- **glutInitWindowSize()**
  - This function is used to specify the size of the created window, inside which the scene will be generated. It takes in two integer parameters, the width and height of the window.

- **glutCreateWindow()**
  - This function creates the window with the specified dimensions and position and assigns the string parameter passed to it as the name of the window.
- **glutMainLoop()**
  - This function is called at most once in a GLUT program. Once called, this routine will never return. It will call as necessary, any callbacks required for the program.

- **glBegin()**
    - This function is used in drawing the basic primitives like lines, points, polygons and quads. It takes in a constant value which specifies the primitive to be drawn. Some of the parameters it can have are:
        - GL_POLYGON
        - GL_LINES
        - GL_LINE_LOOP

- **glClear()**
    - This function is used to clear the contents of the buffer passed as the parameter to this function, onto the screen.
    - It takes a constant value as a parameter like:
        - GL_CLEAR_BUFFER_BIT
        - GL_DEPTH_BUFFER_BIT

- **glClearColor()**
    - This function takes in four floating values:
        - Red
        - Green
        - Blue
        - Alpha
    - Based on RGB values, the color is decided. The Alpha value gives the degree of transparency of the color.
    - The values for RGBA will range from 0.0 to 1.0

- **glEnd()**
    - This function works with the glBegin() function. It marks the end of all the vertices to be used for drawing the primitive.

- **glVertex2f()**
    - This function is used to take in the coordinates of a vertex in a geometric primitive. It takes floating point values as parameters.

4.2 Mouse events:

When mouse event occurs, the ASCII code for the corresponding coordinates that generate the event and the location of mouse are returned. Mouse callback function is:

```
glutMouseFunc (mymouse);
void mymouse (int btn, int state, int x, int y) {
if(button == GLUT_LEFT_BUTTON &&
state == GLUT_DOWN) begin = x;
}
```

4.3 Keyboard events:

When keyboard event occurs, the ASCII code for the corresponding button pressed generate the event and the location of mouse are returned. Keyboard callback function is:

```
glutKeyboardFunc (mykeyboard);
void mykeyboard (unsigned char key, int x, int y)
 {
       if(key ==’e’ ||  key ==’E’) begin = x;
 }
```

CHAPTER - 5

TESTING

Insert for Test Case	Expected Output	Observed Output	Remarks
LEFT MOUSE BUTTON	Pause the animation or stop the movement of ship.	Pause the animation or stop the movement of ship.	Pass
RIGHT MOUSE BUTTON	Pause the animation or stop the movement of ship.	Pause the animation or stop the movement of ship.	Pass
‘s’	Decrease in the speed of the ship.	Decrease in the speed of the ship.	Pass
‘S’	Decrease in the speed of the ship.	Decrease in the speed of the ship.	Pass
‘f’	Increase in the speed of the ship.	Increase in the speed of the ship.	Pass
‘F’	Increase in the speed of the ship.	Increase in the speed of the ship.	Pass
‘e’	Stops the animation and terminates the program.	Stops the animation and terminates the program.	Pass

'E'	Stops the animation and terminates the program.	Stops the animation and terminates the program.	Pass
Space bar	The animation should get affected by the key.	If animation does not gets affected by key.	Fail

Table 5.1: Various test cases for the project

CHAPTER 6

RESULTS

In this animation, initially the ship starts moving from left side to right side of the screen. The speed of ship can be controlled using ‘F’ or ‘f’ and ‘S’ or ‘s’ keys to increase and decrease the speed respectively. The animation can be paused and resumed using left and right mouse button. Here, the left mouse button is used to pause the animation and right mouse button is used to resume the animation. As the ship moves further, it collides with an iceberg and slowly the ship starts drowning into the sea. The animation can be stopped by pressing the ‘E’ or ‘e’ key.

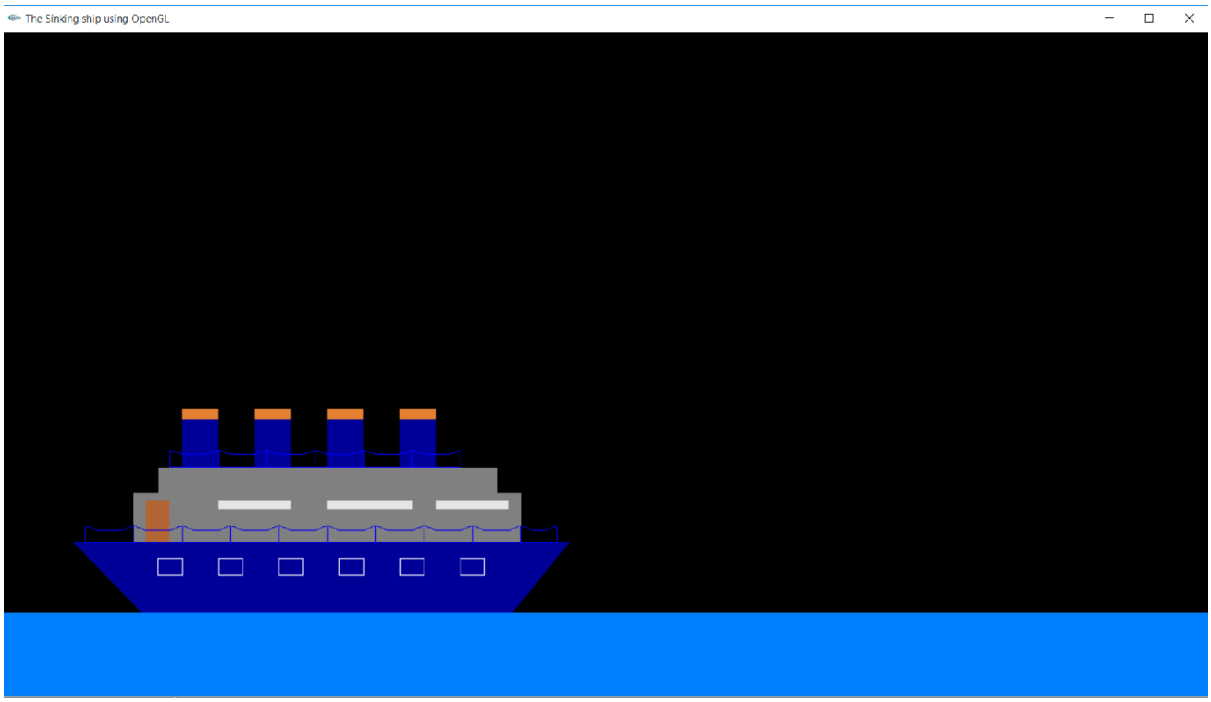

Fig 6.1: Initial Window

This is first scene of the animation project. In this scene the Ship is moving in forward direction (left to right). The speed of ship can be adjusted to either increase or decrease the speed.

Fig 6.2: Ship moving from left to right

In this scene the Ship is moving in forward direction (left to right). The speed of ship can be adjusted to either increase or decrease the speed.

Fig 6.3: Ship colliding with iceberg

In this scene the Ship collides with the Iceberg. After collision, the ship slowly starts sinking into the sea.

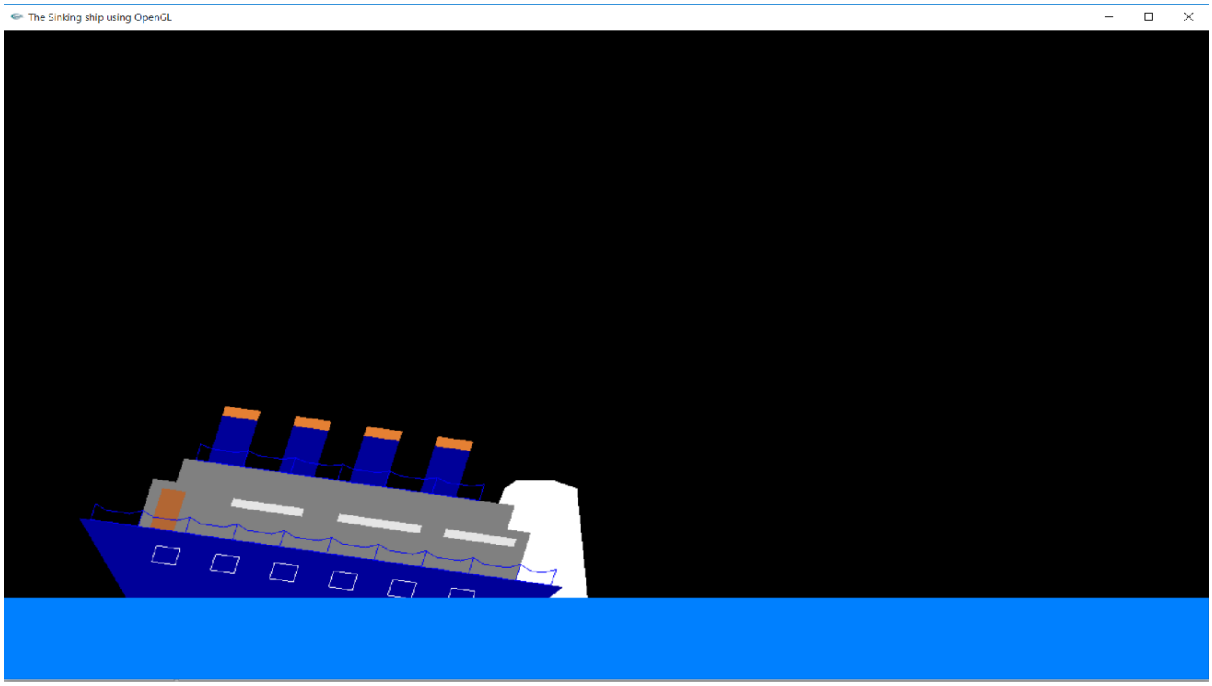

Fig 6.4: Ship drowning into sea

Fig 6.5: Ship drowning into sea

After the collision, the ship sinks into the sea and the ship disappears from the animation screen.

Conclusion

This project implements many functions which are available in OpenGL like translate, rotation, polygon, color etc. The project can be controlled through the mouse to pause and play the scene. The keys can also be used from the keyboard where key pressed defines the speed of the ship movement.

References

[1]. Edward Angel *"Interactive Computer Graphics"* Pearson Education, 5th Edition.2008.

[2]. Donald Hearn & Pauline baker *"Computer Graphics with OpenGL"* 3rd Edition. 2011.

[3]. www.opengl.org

[4]. www.stackoverflow.com

[5]. www.freeglut.sourcefourge.net

[6]. www.github.com

[7]. www.openglcg.blogspot.in

[8]. www.licidchart.com