

CHAPTER 1

INTRODUCTION

1.1 Computer Graphics

Computer graphics is an art of drawing pictures, lines, charts, using computers with the help of programming. Computer graphics is made up of number of pixels. Pixel is the smallest graphical picture or unit represented on the computer screen. Basically, there are 2 types of computer graphics namely,

Interactive Computer Graphics involves a two-way communication between computer and user. The observer is given some control over the image by providing him with an input device. This helps him to signal his request to the computer.

Non-Interactive Computer Graphics otherwise known as passive computer graphics it is the computer graphics in which user does not have any kind of control over the image. Image is merely the product of static stored program and will work according to the instructions given in the program linearly. The image is totally under the control of program instructions not under the user. Example: screen savers.

1.2 Applications of Computer Graphics

Scientific Visualization: Scientific visualization is a branch of science, concerned with the visualization of three-dimensional phenomena, such as architectural, meteorological, medical, biological systems.

Graphic Design: The term graphic design can refer to a number of artistic and professional disciplines which focus on visual communication and presentation.

Computer-aided Design: Computer-aided design (CAD) is the use of computer technology for the design of objects, real or virtual. The design of geometric models for object shapes, in particular, is often called computer-aided geometric design (CAGD). The manufacturing process is tied in to the computer description of the designed objects so that the fabrication of a product can be automated using methods that are referred to as CAM, computer-aided manufacturing.

Web Design: Web design is the skill of designing presentations of content usually hypertext or hypermedia that is delivered to an end-user through the World Wide Web, by way of a Web browser.

Digital Art: Digital art most commonly refers to art created on a computer in digital form.

Video Games: A video game is an electronic game that involves interaction with a user interface to generate visual feedback on a raster display device.

Virtual Reality: Virtual reality (VR) is a technology which allows a user to interact with a computer simulated environment. The simulated environment can be similar to the real world. This allows the designer to explore various positions of an object. Animations in virtual reality environments are used to train heavy equipment operators or to analyse the effectiveness of various cabin configurations and control placements.

Computer Simulation: A computer simulation, a computer model or a computational model is a computer program, or network of computers, that attempts to simulate an abstract model of a particular system.

Education and Training: Computer simulations have become a useful part of mathematical modelling of many natural systems in physics, chemistry and biology, human systems in economics, psychology, and social science and in the process of engineering new technology, to gain insight into the operation of those systems, or to observe their behaviour. Most simulators provide screens for visual display of the external environment with multiple panels is mounted in front of the simulator.

Image Processing: The modification or interpretation of existing pictures such as photographs and TV scans, is called image processing. In computer graphics, a computer is used to create a picture. Image processing techniques, on the other hand, are used to improve picture quality, analyse images, or recognize visual patterns for robotics applications.

1.3 Aim

The aim of this project is to create a visually appealing and realistic representation of a smart village. It focuses on using computer graphics techniques to bring the village to life, showcasing its various elements, such as buildings, infrastructure, and natural surroundings.

1.4 Introduction to OpenGL

OpenGL is an open specification for an applications program interface for defining 2D and 3D objects. The specification is cross-language, cross-platform API for writing applications that produce 2D and 3D computer graphics. It renders 3D objects to the screen, providing the same set of instructions on different computers and graphics adapters. Thus it allows us to write an application that can create the same effects in any operating system using any OpenGL adhering graphics adapter.

Computer graphics, a 3-dimensional primitive can be anything from a single point to an n-sided polygon. From the software standpoint, primitives utilize the basic 3dimensional rasterization algorithms such as Bresenham's line drawing algorithm, polygon scan line fill, texture mapping and so forth. OpenGL's basic operation is to accept primitives such as points, lines and polygons, and convert them into pixels. This is done by a graphics pipeline known as the OpenGL state machine. Most OpenGL commands either issue primitives to the graphics pipeline, or configure how the pipeline processes these primitives. OpenGL is a low-level, procedural API, requiring the programmer to dictate the exact steps required to render a scene. OpenGL's low-level design requires programmers to have a good knowledge of the graphics pipeline, but also gives a certain amount of freedom to implement novel rendering algorithms.

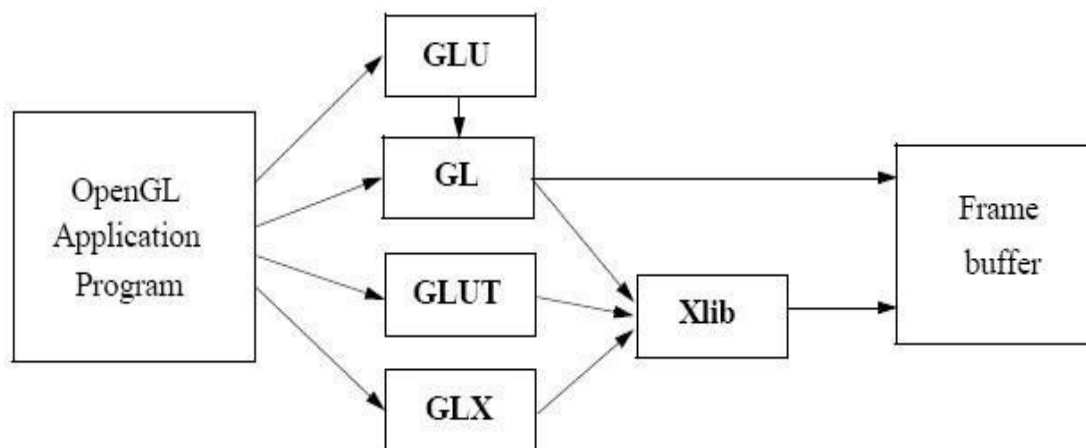


Figure 1.1: Basic block diagram of OpenGL

GLUT

GLUT, short for OpenGL Utility Toolkit, is a set of support libraries available on every major platform. OpenGL does not directly support any form of windowing, menus, or

input. That's where GLUT comes in. It provides basic functionality in all of those areas, while remaining platform independent, so that you can easily move GLUT-based applications from, for example, Windows to UNIX with few, if any, changes.

1.5 Project Related Concepts

The objective of the 3D game "Egg Catch" in computer graphics is to create an entertaining and interactive gaming experience for players. The game involves controlling a character or a basket to catch falling eggs within a virtual environment. The primary goal is to catch as many eggs as possible within a specified time limit or until a certain number of eggs have been caught.

The basic **requirements of the smart village are analyzed to be:**

- **Interactive Controls:** The player interacts with the game using input devices like a keyboard, mouse, or touch controls, depending on the platform. The controls should be intuitive and responsive, allowing the player to move the catching object smoothly and accurately.
- **Catching Eggs:** The main objective is to catch as many eggs as possible before they hit the ground. The player controls a virtual character or object and moves it around to catch the falling eggs.
- **High Score Tracking:** The game could keep track of the highest scores achieved by different players. The objective is to encourage competition among players and motivate them to improve their skills and beat existing records.

1.6 Interfaces

The Project which we have done uses OpenGL functions and is implemented using C++. Our Project is to demonstrate 3D Egg Catch. User can perform operations using Mouse.

Mouse Interface: The functionality are implemented using the mouse function.

- Firstly, after compiling we get directly Main window Page.
- Then we click the maximize button to display the Main window here the 3D Egg Catch view is presented.
- If we right click on the Screen we get options in which user has to specify his choices:
- We need to double click on the options to see the proper changes.
- Game start option to start the game.
- Score option to view the score.
- QUIT option to exit the screen.

CHAPTER 2

REQUIREMENTS SPECIFICATION

Visual Studio 2005 delivers on Microsoft's vision of smart client applications by letting developers quickly create connected applications that deliver the highest quality rich user experiences. This new version lets any size organization create more secure, more manageable, and more reliable applications that take advantage of Windows Vista, windows7, 2007 Office System and the Web. By building these new types of applications, organizations will find it easier than ever to capture and analyze information so that they can make effective business decisions.

2.1 Software Requirements

- An MS-DOS based operating system like Windows 98, Windows 2000 or Windows XP, vista, windows 7 is the platform required to develop the 2D and 3D graphics applications.
- A Visual C/C++ compiler is required for compiling the source code to make the executable file which can then be directly executed.
- A built in graphics library like glut and glut32, and header file like GL\glut.h and also dynamic link libraries like glut and glut32 are required for creating the 3D layout.

2.2 Hardware Requirements

The hardware requirements are very minimal and software can run on most of the machines

- Processor Speed – 500MHz or above
- RAM – 64MB or above Storage Space – 2 MB or above, hard disk – 10MB □
Monitor resolution – A color monitor with a minimum resolution of 1000*700
- Support both single and double buffering.

CHAPTER 3

DESIGN

3.1 Window design

3D Egg Catch uses only one window. This window contains all the contents that is menu bar and simulation display. This is window used for all the events and functions in this project. In this window we display simulation of catching the eggs. And all mouse events triggered in this window. All the labels and Information about the model will be displayed on this window..

3.2 Menu bar

Menu bar is designed so that one can easily access the various options like starting the game or viewing the score or quit.

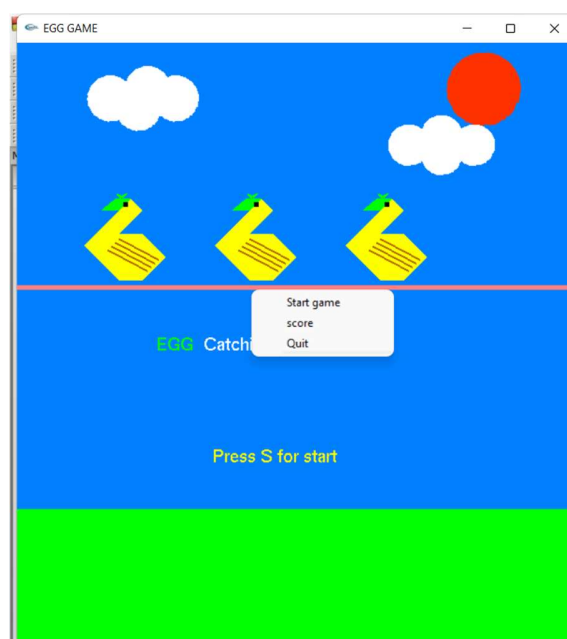


Figure 3.2: Menu Bar

3.3 Simulation display

As soon as the user clicks the right button in the mouse simulation begins.

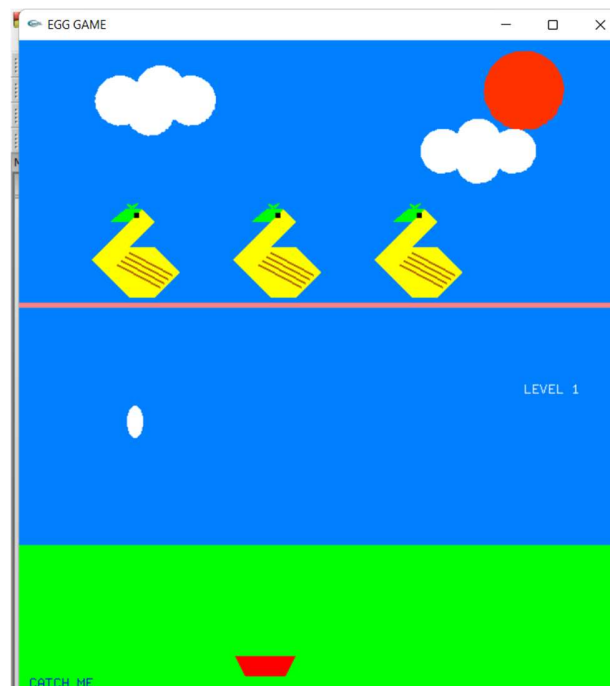


Figure 3.3: Simulation Display

CHAPTER 4

IMPLEMENTATION

4.1 Functions used **glRasterPos(x,y,z):**

OpenGL maintains a 3-D position in window coordinates. This position, called the raster position, is maintained with subpixel accuracy. The current raster position consists of three window coordinates (x, y, z), a clip coordinate w value, an eye coordinate distance, a valid bit, and associated color data and texture coordinates.

glutPassiveMotionFunc(): glutPassiveMotionFunc() function is part of the GLUT (OpenGL Utility Toolkit) library and is used to register a callback function that is called whenever the mouse is moved within the window, regardless of whether any mouse buttons are pressed. This function allows you to track the mouse movement and perform certain actions based on the mouse's position.

glutAddMenuEntry(args): glutAddMenuEntry() adds a menu entry to the bottom of the current menu. The string name will be displayed for the newly added menu entry. If the menu entry is selected by the user, the menu's callback will be called passing value as the callback's parameter.

glutAttachMenu(button): glutAttachMenu() attaches a mouse button for the current window to the identifier of the current menu; glutDetachMenu detaches an attached mouse button from the current window. By attaching a menu identifier to a button, the named menu will be popped up when the user presses the specified button. Button should be one of

GLUT_LEFT_BUTTON, GLUT_MIDDLE_BUTTON and GLUT_RIGHT_BUTTON.

Note that the menu is attached to the button by identifier, not by reference.

glutKeyboardFunc(args): glutKeyboardFunc() sets the keyboard callback for the current window. When a user types into the window, each key press generating an ASCII character will generate a keyboard callback. The key callback parameter is the generated ASCII character. The state of modifier keys such as Shift cannot be determined directly;

their only effect will be on the returned ASCII data. The x and y callback parameters indicate the mouse location in window relative coordinates when the key was pressed. When a new window is created, no keyboard callback is initially registered, and ASCII key strokes in the window are ignored. Passing NULL to glutKeyboardFunc disables the generation of keyboard callbacks. During a keyboard callback, glutGetModifiers may be called to determine the state of modifier keys when the keystroke generating the callback occurred.

CHAPTER 5**TESTING**

Test Case ID	Steps to execute the Test Case	Expected Result	Actual Result	Remarks
1.	Mouse right click	It show the menu bar to the user	It show the menu bar to the user	Pass
2.	Selecting the options 1.Game start. 2.View score. 3.Quit.	It shows the list of option to user from which They can select an option. Starts the simulation. Score board. Stop the simulation	It shows the list of option to user from which They can select an option. Starts the simulation. Score board. Stop the simulation	Pass

Table 5.1 Test cases for Mouse Interface

CHAPTER 6

SNAPSHOTS

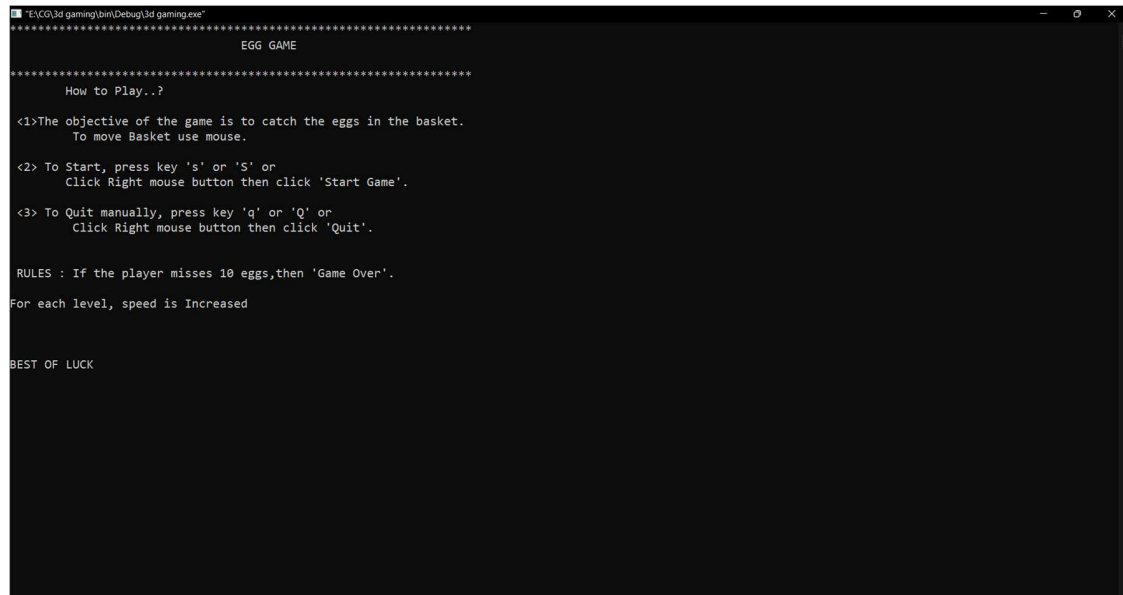


Figure 6.1: Front Page

Click the right button of the mouse to move to the next state.

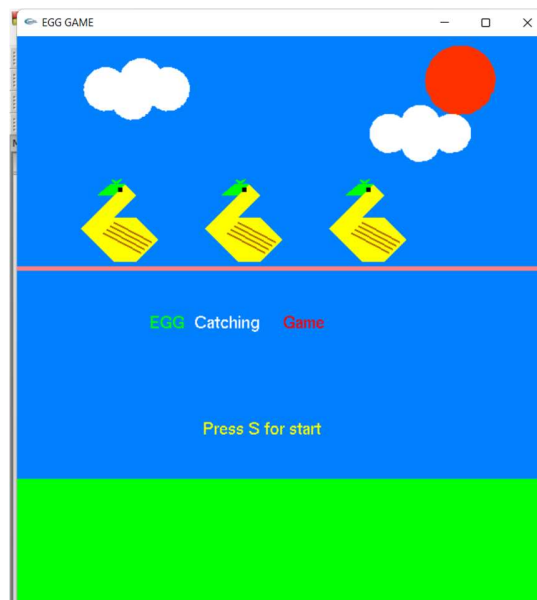


Figure 6.2: Home Screen

Home Screen contains ducks and the key required to start the game.



Figure 6.3: Menu Interface

Menu interface contains the options required for the simulation like start, score and quit.

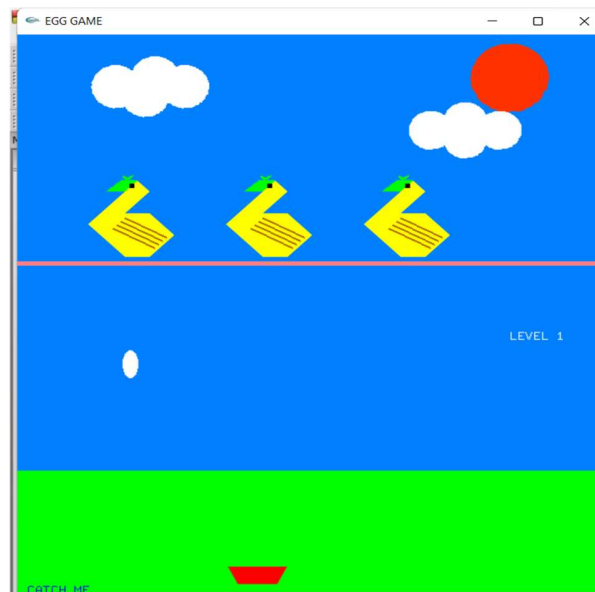
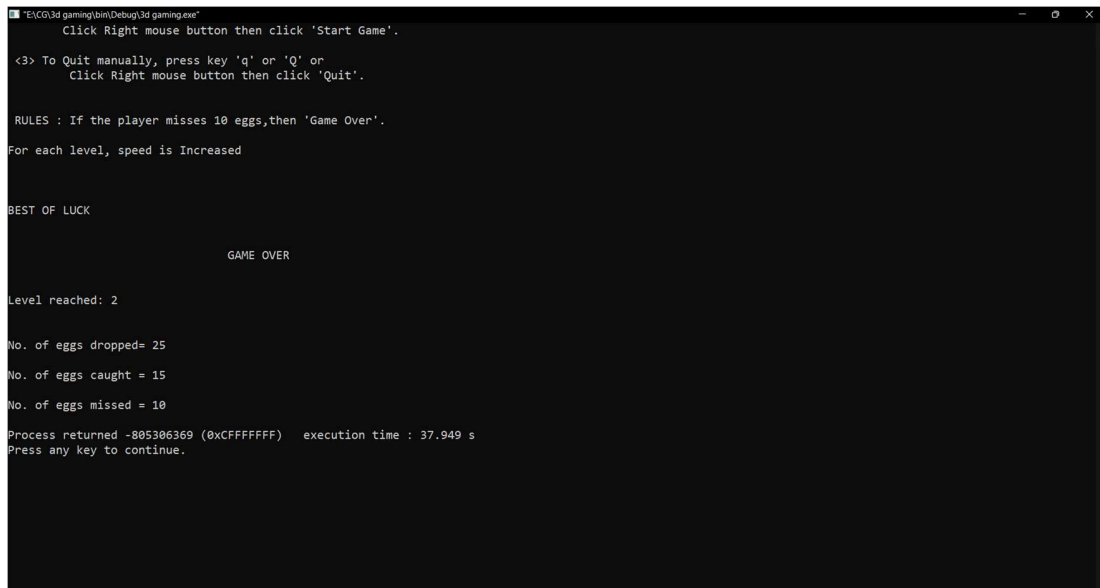


Figure 6.4: Simulation

This picture shows the simulation of 3D Egg catch.



```
E:\CG\3d gaming\bin\Debug\3d gaming.exe
Click Right mouse button then click 'Start Game'.

<3> To Quit manually, press key 'q' or 'Q' or
Click Right mouse button then click 'Quit'.

RULES : If the player misses 10 eggs, then 'Game Over'.
For each level, speed is Increased

BEST OF LUCK

GAME OVER

Level reached: 2

No. of eggs dropped= 25
No. of eggs caught = 15
No. of eggs missed = 10

Process returned -805306369 (0xCFFFFFFF)   execution time : 37.949 s
Press any key to continue.
```

Figure 6.5: Simulation Window

This screen contains the record of no. of eggs dropped, no. of eggs in baskets.

CHAPTER 7**CONCLUSION**

The project was started with the designing phase in which I figured the requirements needed, the layout design, then comes the detail designing of each function after which, was the testing and debugging stage. 2 We have tried to implement the project making it as user-friendly and error free as possible. We regret any errors that may have inadvertently crept in.

FUTURE ENHANCEMENTS

The overall trend we are seeing is the blurring of the boundaries between the game world and everything else,”

“In the future, games will cross boundaries of genres, cultures, technologies, and platforms. The future of game design will be some new hybrid thing that we can't even imagine yet:

BIBLIOGRAPHY

- Interactive Computer Graphics A Top-Down Approach with OpenGL -Edward Angel, 5th Edition, Addison-Wesley, 2008.
- Computer Graphics Using OpenGL – F.S. Hill Jr. 2nd Edition, Pearson Education, 2001.
- Computer Graphics – James D Foley, Andries Van Dam, Steven K Feiner, John F Hughes, Addison-Wesley 1997.
- Computer Graphics - OpenGL Version – Donald Hearn and Pauline Baker, 2nd Edition, Pearson Education, 2003.
- Fundamentals of Computer graphics- Steve Marschner, Peter Shirley, 4th Edition.

FURTHER LINKS:

- www.google.com
- www.openglforum.org