A Mini-Project Report on

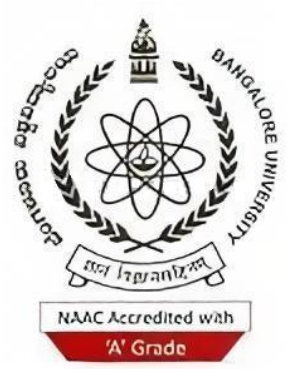
TOWER OF HANOI

Submitted by

Rachana S Shetty(19GACSE045)

Shubham N J(19GACSE057)

5th Semester, CSE



Under the guidance of

Dr. Kumaraswamy S
Assistant Professor
Dept. of CSE
UVCE

Mrs. Usha Rani R E
Guest Faculty
Dept. of CSE
UVCE

Department of Computer Science and Engineering

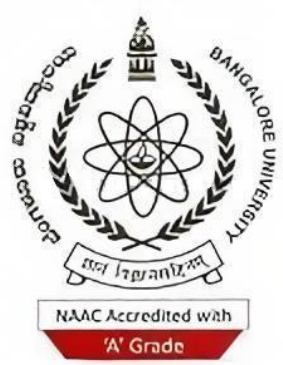
UNIVERSITY VISVESVARAYA COLLEGE OF ENGINEERING
K.R. Circle, Bengaluru- 560001

February 2021-2022

Bangalore University

UNIVERSITY VISVESVARAYA COLLEGE OF ENGINEERING

K.R. Circle, Bengaluru- 560001



Department of Computer Science and Engineering

CERTIFICATE

This is to certify that **Shubham N J(19GACSE057)** has successfully completed the Mini-Project work entitled **Tower Of Hanoi,** in partial fulfillment for the requirement of the **Computer Graphics Laboratory** of 5th semester Computer Science and Engineering during the academic year 2021- 2022.

Dr. Kumaraswamy S	**Mrs. Usha Rani R E**	**Dr.H.S Vimala**
Assistant Professor	**Guest Faculty**	**Chairperson**
Dept. of CSE	**Dept. of CSE**	**Dept. of CSE**
UVCE	**UVCE**	**UVCE**

Examiners:

1. …… **2. ……**

ACKNOWLEGMENT

We, the students of the fifth semester of Computer Science and Engineering, University Visvesvaraya college of Engineering declare that the work entitled “Tower of Hanoi” has been successfully completed under the guidance of Mrs.Usha Rani, Computer Science and Engineering Department. This dissertation work is submitted to Bangalore University in partial fulfillment of the requirements for the award of Degree of Bachelor of Engineering in Computer Science during the academic year 2021- 2022.

Team members:

1)Shubham N J(19GACSE057)

2)Rachana S Shetty(19GACSE045)

ABSTRACT

Project shows the implementation of simulation of tower of hanoi using a recursive approach. The design allows for the user to choose the number of discs and to see the movements of the discs based on the recusrive algorithm as the simulation advances.The project simulates the optimal solution of the Tower of Hanoi problem given a number of disk. The project implements various geometric transformations like Translation, Rotation, and Scaling. The basic model consists of three poles, source, auxiliary and destination. The disks initially resting on the source pole reaches the destination.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

This report contains implementation of 'TOWER OF HANOI '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.

1.1 Computer Graphics

Graphics provides one of the most natural means of communicating within a computer, since our highly developed 2D and 3D pattern-recognition abilities allow us to perceive and process pictorial data rapidly and effectively. Interactive computer graphics is the most important means of producing pictures since the invention of photography and television. It has the added advantage that, with the computer, we can make pictures not only of concrete real world objects but also of abstract, synthetic objects, such as mathematical surfaces and of data that have no inherent geometry, such as survey results. Computer graphics started with the display of data on hardcopy plotters and cathode ray tube screens soon after the introduction of computers themselves. It has grown to include the creation, storage, and manipulation of models and images of objects. These models come from a diverse and expanding set of fields, and include physical, mathematical, engineering, architectural, and even conceptual structures, natural phenomena, and so on. Computer graphics today is largely interactive. The user controls the contents, structure, and appearance of the objects and of their displayed images by using input devices, such as keyboard, mouse, or touch-screen. Due to close relationships between the input devices and the display, the handling of such devices is included in the study of computer graphics. The advantages of the interactive graphics are many in number. Graphics provides one of the most natural means of communicating with a computer, since our highly developed 2D and 3D patter-recognition abilities allow us to perceive and process data rapidly and efficiently. In many design, implementation, and construction processes today, the information pictures can give is virtually indispensable. Scientific visualization became an important field in the 1980s when the scientists and engineers realized that they could not interpret the prodigious

quantities of data produced in supercomputer runs without summarizing the data and highlighting trends and phenomena in various kinds of graphical representations.

1.2 OpenGL Interface

OpenGL is an application program interface (API) offering various functions to implement primitives, models and images. This offers functions to create and manipulate render lighting, coloring, viewing the models. OpenGL offers different coordinate system and frames. OpenGL offers translation, rotation and scaling of objects. Most of our applications will be designed to access OpenGL directly through functions in three libraries.

They are:

1. Main GL: Library has names that begin with the letter gl and are stored in a library usually referred to as GL.

2. OpenGL Utility Library (GLU): This library uses only GL functions but contains code for creating common objects and simplifying viewing.

3. OpenGL Utility Toolkit(GLUT): This provides the minimum functionality that should be accepted in any modern windowing system.

1.3 OpenGL Overview

• OpenGL (Open Graphics Library) is the interface between a graphic program and graphics hardware. It is streamlined. In other words, it provides low-level functionality. For example, all objects are built from points, lines and convex polygons. Higher level objects like cubes are implemented as six four-sided polygons.

• OpenGL supports features like 3-dimensions, lighting, anti-aliasing, shadows, textures, depth effects, etc.

• It is system-independent. It does not assume anything about hardware or operating system and is only concerned with efficiently rendering mathematically described scenes. As a result, it does not provide any windowing capabilities.

• It is a state machine. At any moment during the execution of a program there is a current model transformation .

• It is a rendering pipeline. The rendering pipeline consists of the following steps:

  o Defines objects mathematically.

o Arranges objects in space relative to a viewpoint.

o Calculates the color of the objects.

o Rasterizes the objects.

Graphics provides one of the most natural means of communicating with a computer, since our highly developed 2D and 3D pattern-recognition abilities allow us to perceive and process pictorial data rapidly and efficiently. Interactive computer graphics is the most important means of producing pictures since the invention of photography and television. It has the added advantage that, with the computer, we can make pictures not only of concrete real world objects but also of abstract, synthetic objects, such as mathematical surfaces and of data that have no inherent geometry, such as survey results. OpenGL (open graphics library) is a standard specification defining a cross language cross platform API for writing applications that produce 2D and 3D computer graphics. OpenGL was developed by silicon graphics Inc. (SGI) in 1992 and is widely used in CAD, virtual reality, scientific visualization, information visualization and flight simulation. It is also used in video games. OpenGL serves two main purpose:

- To hide the complexities of interfacing with different 3D accelerators, by presenting programmer with a single, uniform API

- To hide the differing capabilities of hardware platforms, by requiring that all Implementations support the full openGL, feature set. OpenGL has historically been influential on the development of 3D accelerator, promoting a base level of functionality that is now common in consumer level hardware:

    - Rasterized points, lines and polygons are basic primitives.
    - A transform and lighting pipeline.
        - Z buffering.
        - Texture Mapping.
        - Alpha
        - Blending

CHAPTER 2

SYSTEM REQUIREMENTS SPECIFICATION

2.1 HARDWARE REQUIREMENTS

♣ **Microprocessor**: 1.0 GHz and above CPU based on either AMD or INTEL Microprocessor Architecture

♣ **Main memory** : 2 GB RAM

♣ **Hard Disk** : 40 GB

♣ **Hard disk speed in RPM**:5400 RPM

♣ **Keyboard**: QWERTY Keyboard

♣ **Mouse** :2 or 3 Button mouse

♣ **Monitor** : 1024 x 768 display resolution

2.2 SOFTWARE REQUIREMENTS

- **Programming language** – C/C++ using OpenGL
- **Operating system** – windows
- **Compiler** – C++ Compiler
- **Graphics library** – GL/glut.h
- OpenGL 2.

2.3:FUNCTIONAL REQUIREMENTS:

OpenGL APIs: If we want to have a control on the flow of program and if we want to interact with the window system then we use OpenGL API'S. Vertices are represented in the same manner internally, whether they are specified as two-dimensional or three-dimensional entities, everything that we do are here will be equally valid in three dimensions. Although OpenGL is easy to learn, compared with other APIs, it is nevertheless powerful. It supports the simple three dimensional programs and also supports the advanced rendering techniques.

GL/glut.h: We use a readily available library called the OpenGL Utility Toolkit (GLUT), which provides the minimum functionality that should be expected in any modern windowing system. The application program uses only GLUT functions and can be recompiled with the GLUT library for other window system. OpenGL makes a heavy use of macros to increase

code readability and avoid the use of magic numbers. In most implementation, one of the include lines

CHAPTER 3

ABOUT THE PROJECT

3.1Overview

Program to demonstrate the simulation of Tower of Hanoi.

The Tower of Hanoi is a mathematical game or puzzle. It consists of three rods, and a number of disks of different sizes which can slide onto any rod. The puzzle starts with the disks in a neat stack in ascending order of size on one rod, the smallest at the top, thus making a conical shape.

The objective of the puzzle is to move the entire stack to another rod, obeying the following simple rules:

1. Only one disk can be moved at a time.
2. Each move consists of taking the upper disk from one of the stacks and placing it on top of another stack i.e. a disk can only be moved if it is the uppermost disk on a stack.
3. No disk may be placed on top of a smaller disk.

With three disks, the puzzle can be solved in seven moves. The minimum number of moves required to solve a Tower of Hanoi puzzle is 2n - 1, where n is the number of disks.

Recursive Solution:

A key to solving this puzzle is to recognize that it can be solved by breaking the problem down into a collection of smaller problems and further breaking those problems down into even smaller problems until a solution is reached. For example:

- label the pegs A, B, C
- let n be the total number of discs
- number the discs from 1 (smallest, topmost) to n (largest, bottommost) To move n discs from peg A to peg C:

1. move n−1 discs from A to B. This leaves disc n alone on peg A
2. move disc n from A to C
3. move n−1 discs from B to C so they sit on disc n

The above is a recursive algorithm, to carry out steps 1 and 3, apply the same algorithm again for n−1. The entire procedure is a finite number of steps, since at some point the algorithm will be required for n = 1. This step, moving a single disc from peg A to peg C, is trivial. This approach can be given a rigorous mathematical formalism with the theory of dynamic programming and is often used as an example of recursion when teaching programming.

3.2 Explanation about the package

This is a mini project on Tower of Hanoi based on OPENGL API for Computer graphics laboratory. The project simulates the optimal solution of the Tower of Hanoi problem given a number of disk. The project implements various geometric transformations like Translation, Rotation, and Scaling. The basic model consists of three poles, source, auxiliary and destination. The disks initially resting on the source pole reaches the destination. The poles are drawn using the GLUT library function glutSolidCone(), and the disks are drawn using glutSolidTorus(). Initially the user is presented with an introduction scene which has a menu to choose the number of disks.

On pressing the Enter key the actual simulation scene is loaded. The user can use the mouse wheel to advance through the simulation. An optional animation has been implemented, which shows the movement of the disks from pole to pole. The viewing model used is an Orthogonal Projection. The moves to be performed as per the optimal solution are displayed on the top left of the screen is a raster text using the function glutBitmapCharacter(). Lighting has been implemented by the inbuilt OPENGL lighting functions. Menus have been provided to modify various features such as Lighting, Move camera, animation, change background color, to automatically simulate the complete solution, restart the simulation and to exit the program. The movement of the camera is only along the Y axis, and is implemented using the gluLookAt() function.

3.3 User interface

Keyword Control is allotted which are listed below

Start up: Press Enter to go to the simulation screen.

Navigation: Use menu options for choosing the various options available including start and restart of simulation .

Mouse Wheel: Use mouse wheel for advancing the simulation.

3.4 OBJECTIVE:

¬ Aim is to demonstrate the simulation of Tower of Hanoi, a recusrsive approach to solve the puzzle.

¬ Use windows to provide graphics editor

¬ It should be easy to understand, user interactive interface.

¬ Providing human interaction through Mouse and keyboard

CHAPTER 4

IMPLEMENTATION

4.1 EXISTING SYSTEM

Existing system for a graphics is the TC++. This system will support only the 2D graphics. 2D graphics package being designed should be easy to use and understand. It should provide various options such as free hand drawing, line drawing, polygon drawing, filled polygons, flood fill, translation, rotation, scaling, clipping etc. Even though these properties were supported, it was difficult to render 2D graphics cannot be very difficult to get a 3 Dimensional object. Even the effects like lighting, shading cannot be provided. So we go for Codeblocks software.

4.2 PROPOSED SYSTEM

To achieve three dimensional effects, open GL software is proposed. It is software which provides a graphical interface. It is an interface between application program and graphics hardware. The advantages are:

1. Open GL is designed as a streamlined.

2. It's a hardware independent interface i.e it can be implemented on many different hardware platforms.

3. With Open GL we can draw a small set of geometric primitives such as points, lines and polygons etc.

4. It provides double buffering which is vital in providing transformations.

5. It is event driven software.

6. It provides call back functions.

4.3 USER DEFINED FUNCTIONS

void push() - This function is used to push discs on a peg.

void pop() - This function is used to pop discs from a peg.

void tower() - This function is used to recursively solve the problem.

void drawPegs() - This function is used to draw pegs.

void drawText() - This function is used to draw text that displays the status.

void drawSolved() - This function is used to draw Solved in a rectangle

void lighting() - This function is used to change the lighting.

void animate() - This function is used to begin animation.

void restart() - This function is used to restart animation.

void display() - this is used to display the function

void processMenuCamera(int option) - This function is used to adjust the camera.

void processMenuExit() – This function is used to exit the system.

void processMenuBgColor(int option) – This function is used to change the background color.

main( ) - The main function is used for creating the window for display of the model of the atom. Here, we create menu for ease of use for the user. The callback functions, i.e., mouse callback, keyboard callback, display callback, idle callback, are written in main. The callback functions registered in main ( ) are:

glutDisplayFunc(display);

glutKeyboardFunc(Keyboard);

4.4 OPEN GL FUNCTIONS

glColor3f (float, float, float):-This function will set the current drawing color

glClear( ):-Takes a single argument that is the bitwise OR of several values indicating which buffer is to be cleared.

glClearColor ():-Specifies the red, green, blue, and alpha values used by glClear to clear the color buffers.

void glutInit (int *argc, charargv)**:-Initializes GLUT, the arguments from main are passed in and can be used by the application.

void glutInitDisplayMode (unsigned int mode):-Requests a display with the properties in mode. The value of mode is determined by the logical OR of options including the color model and buffering.

void glutInitWindowSize (int width, int height):- Specifies the initial position of the topleft corner of the window in pixels

glutInitCreateWindow (char *title):-A window on the display.The string title can be used to label the window. The return value provides references to the window that can be used when there are multiple windows.

void glutMouseFunc(void *f(int button, int state, int x, int y):-Register the mouse callback function f. The callback function returns the button,the state of button after the event and the position of the mouse relative to the top-left corner of the window.

void glutKeyboardFunc(void(*func) (void)):-This function is called every time when you press enter key to resume the game or when you press ‘b’ or ‘B’ key to go back to the initial screen or when you press esc key to exit from the application.

void glutDisplayFunc (void (*func) (void)):-Register the display function func that is executed when the window needs to be redrawn.

glutPostReDisplay ( ) :-which requests that the display callback be executed after the current callback returns.

void glutMainLoop () :-Cause the program to enter an event-processing loop.It should be the last statement in main function.

CHAPTER 5

TESTING

Testing process started with the testing of individual program units such as functions or objects. These were then integrated into sub-systems and systems, and interactions of these units were tested.

Testing involves verification and validation.

Validation: “Are we building right product?”

Verification: “Are we building the product right?”

The ultimate goal of the verification and validation process is to establish confidence that the software system is ‘fit for purpose’. The level of required confidence depends on the system’s purpose, the expectations of the system users and the current marketing environment for the system. With the verification and validation process, there are two complementary approaches to the system checking and analysis: Software inspections or peer reviews analyses and check system representations such as the requirements document, design diagrams, and the program source code. Software testing involves running an implementation of the software with test data.

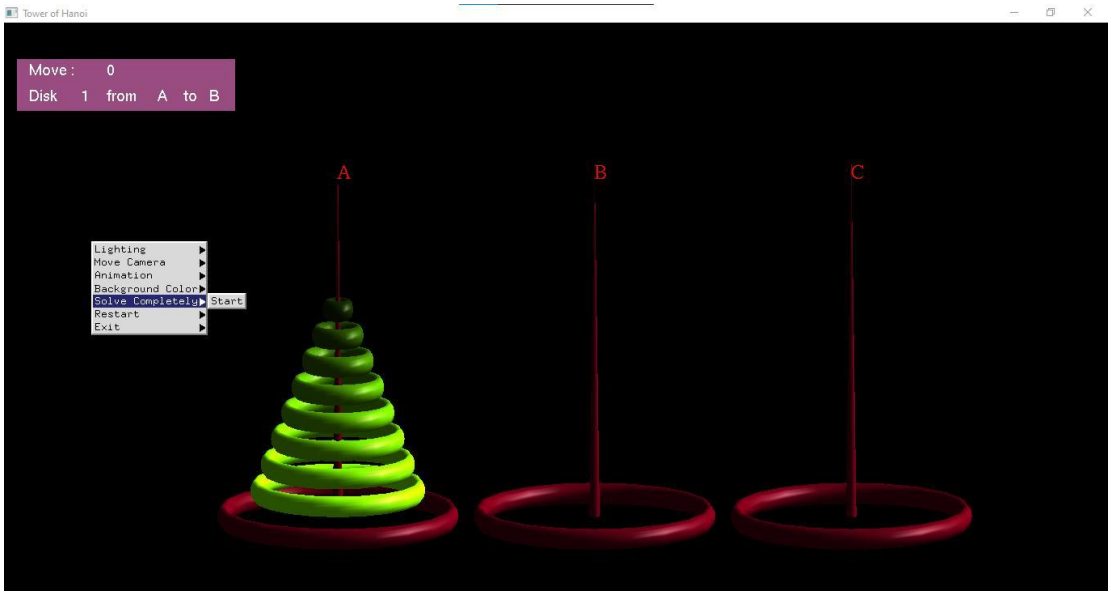
CHAPTER 6

SNAPSHOTS

Output screen at the beginning of the game.

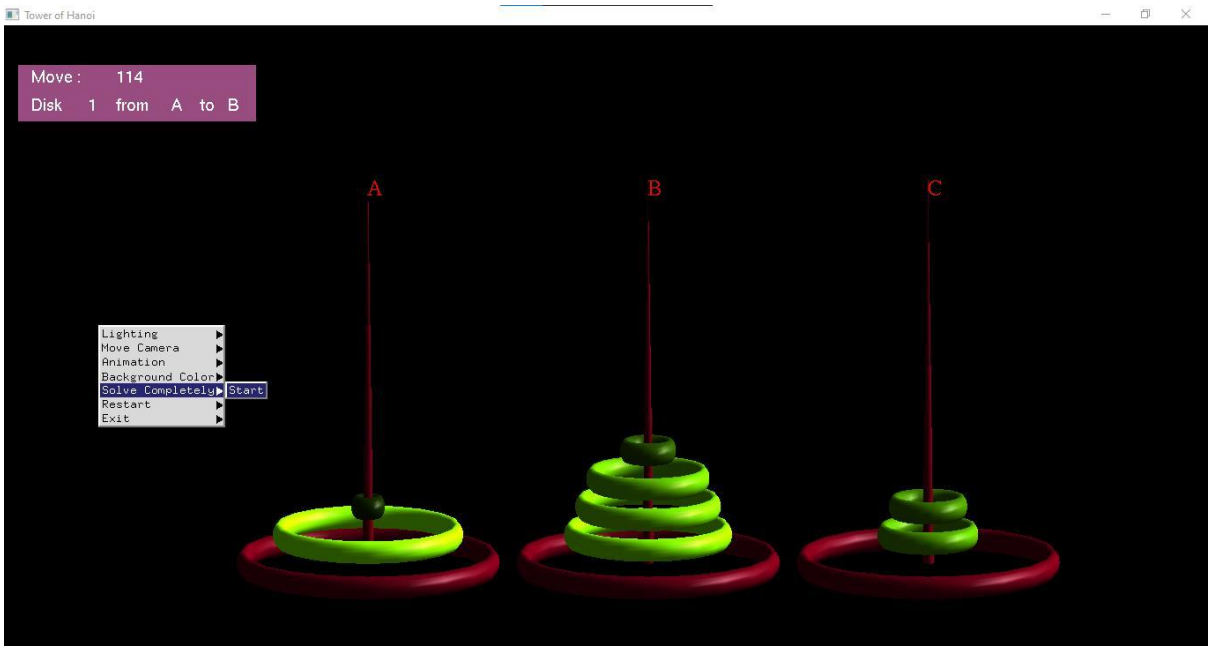
SS.1:WELCOME SCREEN

Screen before the simulation of the game



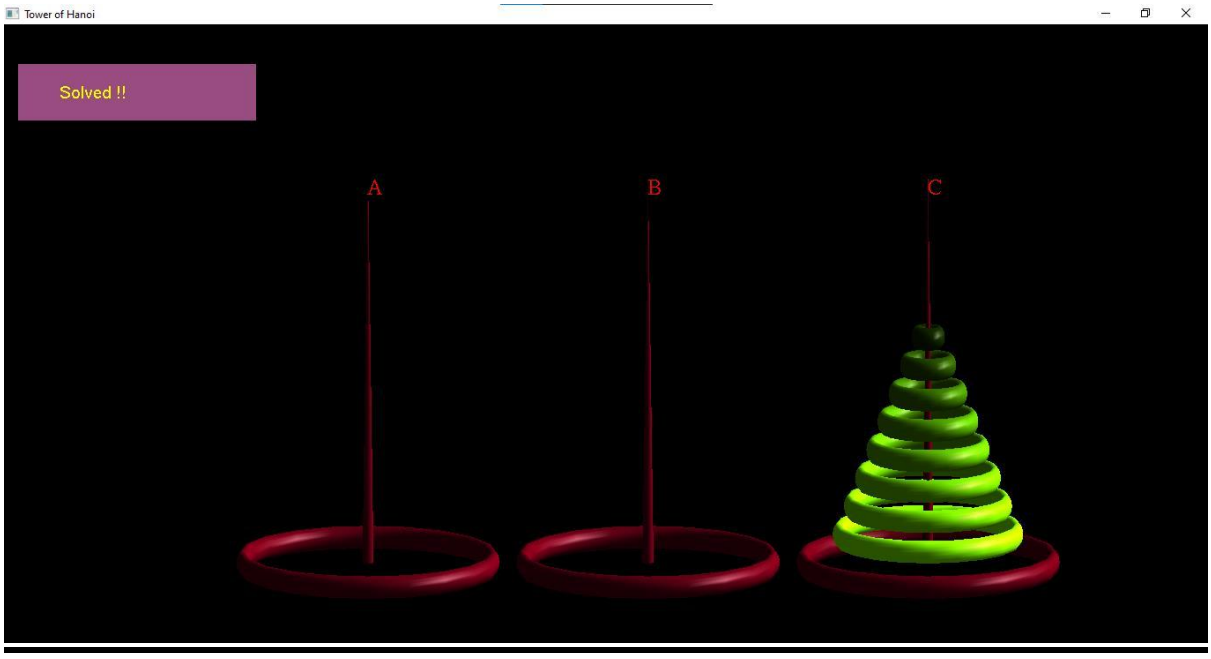
SS.2 BEFORE SIMULATION

Screen during the simulation of the game.



SS.3 :DURING THE SIMULATION

screen after solving the problem



SS.4:END OF SIMULATION

Conclusions and Future Enhancements

The project was started with a modest aim with no prior experience in any programming projects like this but ended up learning many things, fine-tuning the programming skills, and getting into the real world of software development with exposure to the corporate environment.

During the development of any software of significant utility, we are forced with the tradeoff between speed of execution and amount of memory consumed. This is a simple interactive application.

It is extremely user-friendly and has features, which makes simple graphics project. It has open-source and no security features have been included. The user is free to alter the code for feature enhancement. Checking and verification of all possible types of functions are taken care of. Care was taken to avoid bugs. Bugs may be reported to the creator as the need.

BIBLIOGRAPHY

[1] Edward Angel's Interactive Computer Graphics Pearson Education 5th Edition .

[2] Interactive computer Graphics --A top down approach using open GL--by Edward Angle.

[3] Jackie.L.Neider,Mark Warhol,Tom.R.Davis,"OpenGL Red Book",Second Revised Edition,2005.

[4] Donald DHearn and M.Pauline Baker,"Computer Graphics with OpenGL", 3rd Edition.

[5] Portion of the code for implementing GEAR has been borrowed from Brian Paul's MESA.