CHAPTER-1

INTRODUCTION

The Main aim of this project is to show the simulation of Towers of Hanoi using OpenGL which include changing visual properties, redisplay functionality and mouse button press actions. The package must also have a user friendly interface. The input can be given by the user and hence the project is dynamic in nature. Few parameters can be changed in the #define constructs. Towers of Hanoi Simulation is designed and implemented using a graphics software system called Open GL which became a widely accepted standard for developing graphics application. Usage of Open GL functions and primitives are well understood and henceforth can be applied for real time applications. This project is both informative and entertaining. This project provided an opportunity to learn the various concepts of the subject in detail and provided a platform to express creativity and imagination come true. Further animation can be included to enhance the project's look and feel. One of the key characteristics of the Tower of Hanoi puzzle is its recursive nature. By observing the problem closely, one can notice that moving a stack of size 'n' disks involves moving a stack of 'n-1' disks, which, in turn, involves moving a stack of 'n-2' disks. The Tower of Hanoi puzzle begins with a stack of disks of different sizes, arranged in decreasing order of size, on one peg. The other two pegs are initially empty. The objective of the puzzle is to move all the disks from one peg to another, using the third peg as an auxiliary, while following specific rules.

1.1 History

The phrase "Computer Graphics" was coined in 1960 by William Fetter, a graphic designer for Boeing. The field of computer graphics developed with the emergence of graphics hardware. The first major advance in computer graphics was the development of the Sketchpad by Ivan Sutherland. Further advances in computing led to greater advancement in interactive computer graphics. In 1959, the TX2 computer was developed at MIT's Lincoln Laboratory.

1.2 Application of Computer Graphics

- Computational biology
- Computational physics
- Computer-aided design
- Computer simulation
- Digital art
- Interfaces
- Display of information

1.3 OpenGL Graphics Architecture

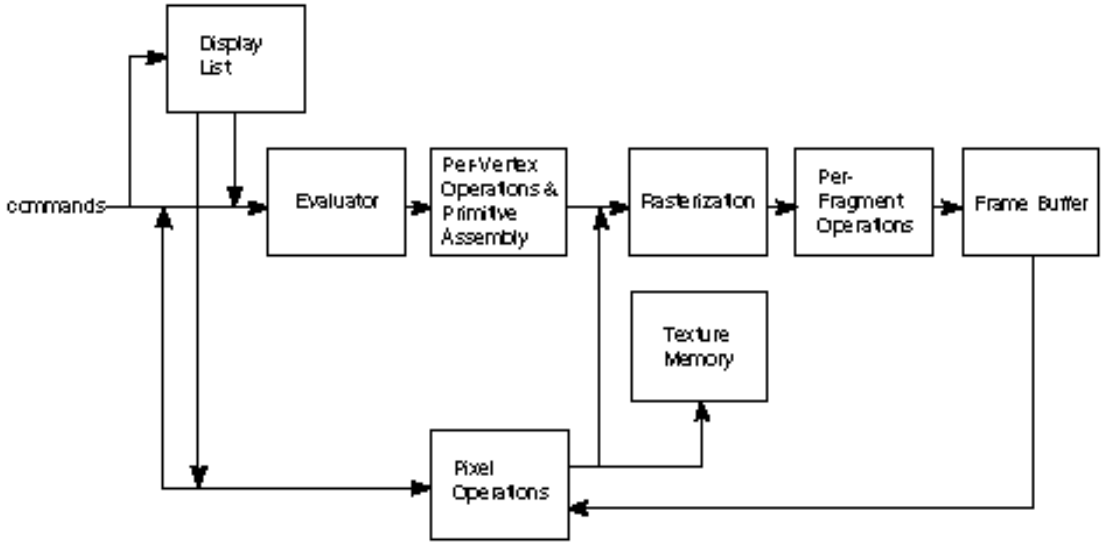

Fig 1.1 : OpenGL Block Diagram

As shown by the first block in the diagram, rather than having all commands proceed immediately through the pipeline, you can choose to accumulate some of them in a display list for processing at a later time. Rasterization produces a series of frame buffer addresses and associated values using a two-dimensional description of a point, line segment, or polygon. Each fragment so produced is fed into the last stage, per-fragment operations, which performs the final operations on the data before it's stored as pixels in the frame buffer. These operations include conditional updates to the frame buffer based on incoming and previously stored z-value s (for z-buffering) and blending of incoming pixel colors with stored colors, as well as masking and other logical operations on pixel values. The Computer Graphics is one of the most effective and commonly used methods to communicate the processed information to the user. It displays the information in the form of graphics objects such as pictures, charts, graphs and diagram instead of simple text.

OpenGL is a low-level graphics library specification. It makes available to the Programmer a small set of geometric primitive – points, lines, polygons, images, and bitmaps. OpenGL provides a set of commands that allow the specification of geometric objects in two or three dimensions, using the provided primitives, together with commands that control how these objects are rendered. Since OpenGL drawing commands are limited to those that generate simple geometric primitive (points, lines and polygons), the OpenGL utility(GLUT)has been created to aid in the development of more minimum.

CHAPTER-2

SYSTEM REQUIRMENTS

2.1 Software Requirements

- Operating system : Ubuntu,MAC,Windows
- Language Tool : OpenGL
- Compiler : GNU GCC Compiler / C++ Compiler.
- Libraries : GL/gl.h, GL/glut.h, GL/glu.h
- Documentation Tools : vi-editor, geditor, visual studio

2.2 Hardware Requirement

- Processor : Intel CORE i3,i5
- RAM : 16Mb RAM.
- Monitor : Graphics Compatible.
- Keyboard : Normal keyboard(QWERTY).
- Backup Media : Hard disk.

CHAPTER-3

SYSTEM DESIGN AND ARCHITECTURE

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 or pixels. OpenGL performs several processing steps on this data to convert it to pixels to form the final desired image in the frame buffer. The Graphics Package is designed using the in-built graphics library. The objects, which are drawn, are stored as functions that can be used according to the requirements. Ease of understanding and speed of working are two main requirements for it, which should be kept in mind during phase of design and implementation. The 2-D transformation project has been developed in C that also provides an in-built graphics library through "GL/glut.h".

3.1 DATA FLOW DIAGRAM

The interaction with the windows is initialized using glutInit() OpenGL API. The display mode-double buffer and depth buffer is, various callback functions for drawing and redrawing, for mouse and keyboard interfaces, input and calculate functions for various mathematical calculations, the window position and size are also initialized and create the window to display the output.

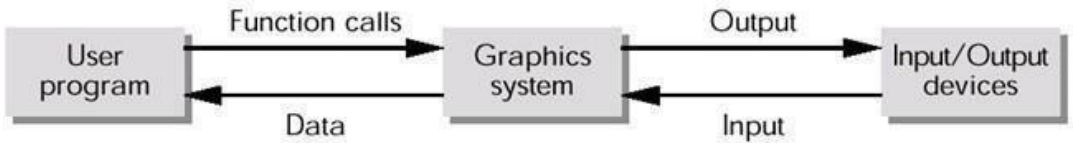

Fig 3.1 : OpenGL Block Diagram

As shown by the first block in the diagram, rather than having all commands proceed immediately through the pipeline, you can choose to accumulate some of them in a display list for processing at a later time. Rasterization produces a series of frame buffer addresses and associated values using a two dimensional description of a point, line segment, or polygon. Each fragment so produced is fed into the last stage, per-fragment operations, which performs the final operations on the data before it's stored as pixels in the frame buffer. These operations include conditional updates to the frame buffer based on incoming and previously stored z-value s (for z buffering) and blending of incoming pixel colors with stored colors, as well as masking and other logical operations on pixel values. All elements of OpenGL state, including the contents of the texture memory and even of the frame buffer, can be obtained by an OpenGL application. Pictorial synthesis of real/imaginary objects from computer-based models is Computer Graphics.

3.2 FLOWCHART

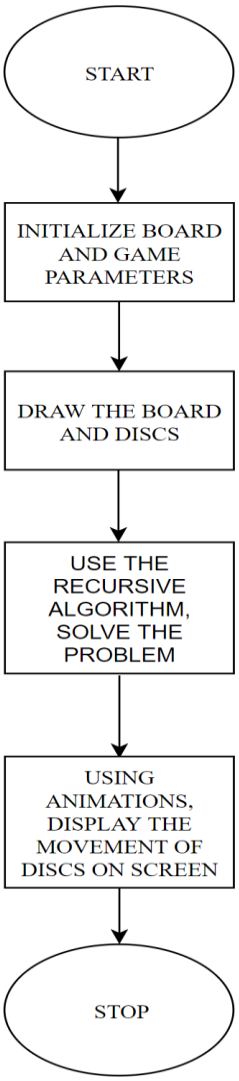

Fig 3.2 : Flow chart of the application

A flowchart is a graphical representation of a process or algorithm, using various symbols and arrows to illustrate the flow of control. The flowchart begins with the "Start" symbol, indicating the start of the algorithm. Next, there is an input symbol where the number of disks in the Tower of Hanoi puzzle is entered. This value determines the size of the problem. After the input, there is an "Initialize" symbol where the variables or data structures needed for the algorithm are initialized. In this case, you would typically initialize the three towers (A, B, and C) and place the disks on tower A in descending order of size. . If the termination condition is not met, the flowchart proceeds to the next step, which involves moving the top disk from the source tower (A) to the destination tower (C), using the auxiliary tower (B) as a temporary storage. Finally, when all the disks have been moved to the destination tower, the flowchart reaches the "Stop" symbol, indicating the end of the algorithm.

CHAPTER-4

IMPLEMENTATION

This program is implemented using various openGL functions and header files which are shown below.

4.1 Header files and their descriptions

stdio.h: This is a standard input header file which is used in any program. This file contains all the built-in functions like printf(),scanf(),fopen(),fclose() etc. It also contains data types and global variables. Some of the examples are BUFSIZ, EOF, and NULL.

stdlib.h: This is also one the standard library header file which contains the entire standard library functions like exit, free, alloc, malloc etc. Some of the constants and data types are NULL, size_t etc.

Gl/glut.h : This is very familiar library function of visual basic graphics library. This header file contains so many numbers of built in functions of a graphics library.

4.2 Various functions used in program

- glutInit (): interaction between the windowing system and OPENGL is initiated.
- glutInitDisplayMode() : used when double buffering is required and depth information is required.
- glutCreateWindow() : this opens the OPENGL window and displays the title at top of the window.
- glutInitWindowSize() : specifies the size of the window.
- glutInitWindowPosition() : specifies the position of the window in screen co ordinates.
- glutKeyboardFunc() : handles normal ASCII symbols.
- glutSpecialFunc() : handles special keyboard keys.
- glutReshapeFunc() : sets up the callback function for reshaping the window.
- glutIdleFunc() : this handles the processing of the background.
- glutDisplayFunc() : this handles redrawing of the window.
- glutMainLoop (): this starts the main loop, it never returns.
- glViewport() : used to set up the viewport.
- glVertex3fv() : used to set up the points or vertices in three dimensions.
- glFlush() : used to flush the pipeline.
- glutPostRedisplay() : used to trigger an automatic redrawal of the object.
- glMatrixMode() : used to set up the required mode of the matrix.
- glLoadIdentity() : used to load or initialize to the identity matrix.
- glTranslatef() : used to translate or move the rotation centre from one point to another in three dimensions.

4.3 Interaction with program

This program includes interaction through keyboard.

- Both mouse and keyboard are used to interact with the program.
- Press Enter to Start the project.
- Right mouse button can be used to get a menu.

4.4 ALGORITHM:

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 as A, B and 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.

CHAPTER-5

RESULT

Fig 5.1 : Setup page

In fig 5.1 we can asks the user to enter the number of disks that as to be displayed.

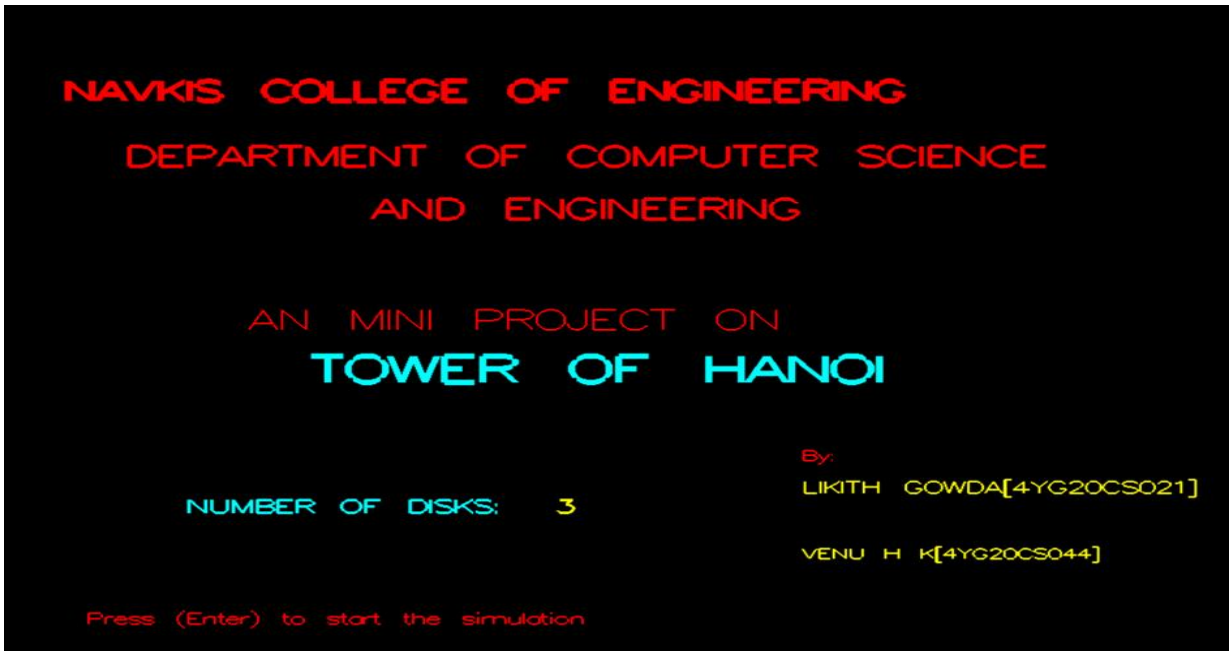

Fig 5.2 : Main page

In fig 5.2 we can observe the information about Project Title , Teammates and Number of Disks and tells the users to Press Enter to start the simulation.

Fig 5.3 : Move Camera Effect

In Fig 5.3 we can observe the rotation of the camera angle. The user can rotate in clockwise or anticlockwise .

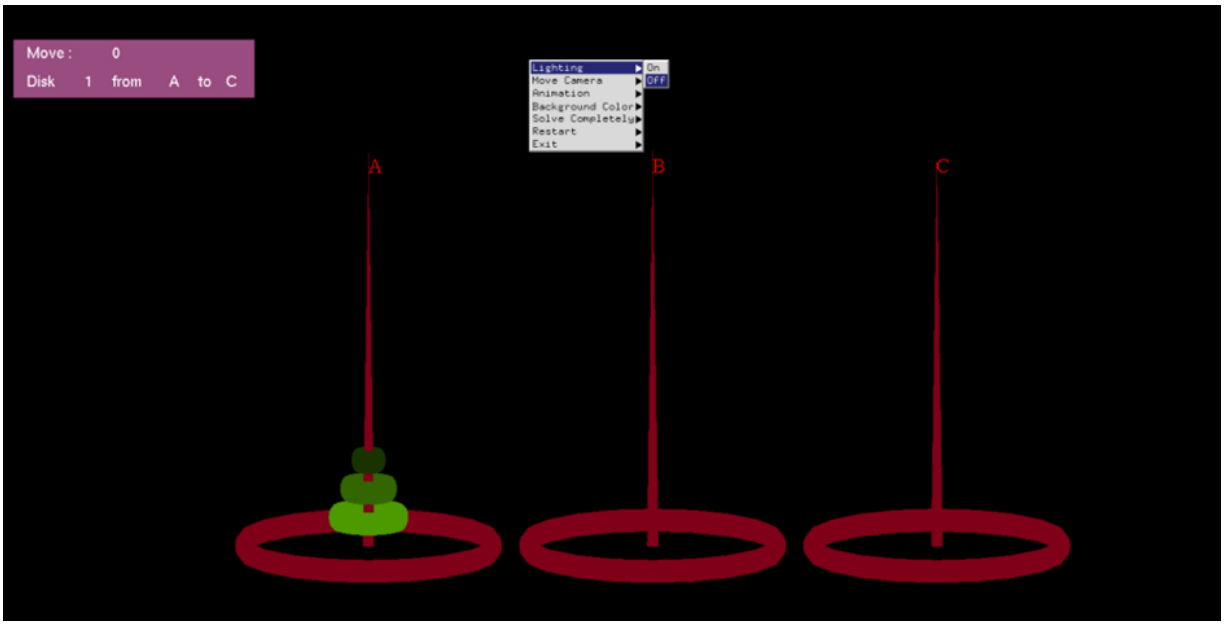

Fig 5.4 : Lighting Effect

In Fig 5.4 we can observe the lighting effect. The user can On or Off these effect .

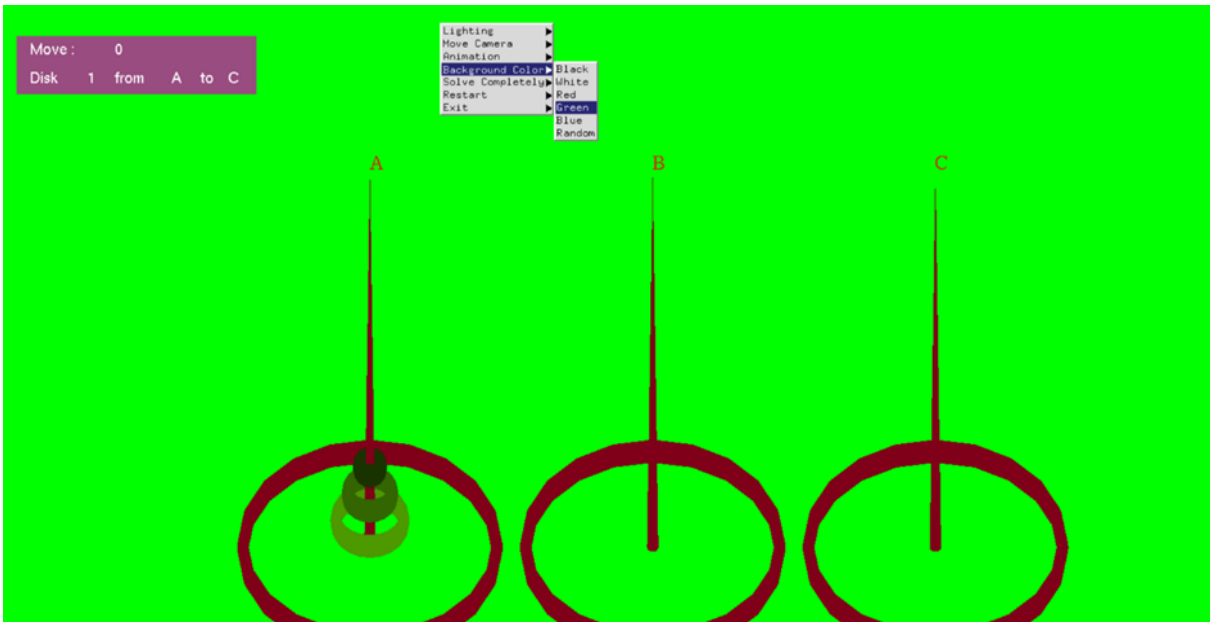

Fig 5.5 : Background Change Effect

In Fig 5.5 we can observe the Background Color Change. The user can also select Random colors other than RGB.

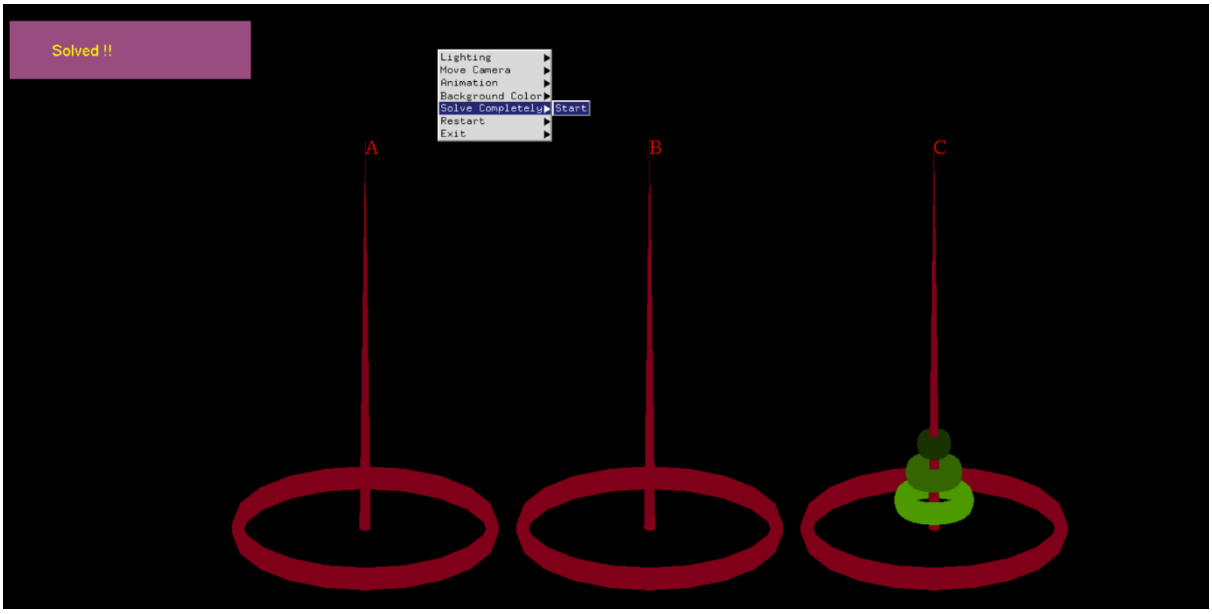

Fig 5.6 : Final output

In Fig 5.6 we can observe the complete movement of disks .

CONCLUSION

Towers of Hanoi Simulation is designed and implemented using a graphics software system called Open GL which became a widely accepted standard for developing graphics application.

Usage of Open GL functions and primitives are well understood and henceforth can be applied for real time applications. This project is both informative and entertaining. This project provided an opportunity to learn the various concepts of the subject in detail and provided a platform to express creativity and imagination come true. Further animation can be included to enhance the project's look and feel. Through careful analysis and observation, it has been proven that the Tower of Hanoi puzzle can always be solved efficiently using recursive algorithms. The recursive approach breaks down the problem into smaller subproblems, gradually solving each one until the entire puzzle is solved. This elegant solution demonstrates the power of recursion in problem-solving.

REFERENCES

[1] http://jerome.jouvie.free.fr/OpenGl/Lessons/Lesson3.php

[2] http://google.com

[3] http://opengl.org

[4] www.opengl.org/documentation/specs/gut/spec3/spec3.html

[5] Edward Angel, “Interactive Computer Graphics”, Pearson education, Fifth Edition.

[6] Computer Graphics Using OpenGL - F.S.Hill, Jr. 2nd Edition, Pearson Education,2001

[7] www.OpenGL.org/recources/code/samples