CHAPTER 1

INTRODUCTION

1.1 Computer Graphics

Computer graphics is a sub-field of computer science and is concerned with digitally synthesizing and manipulating visual content. Although the term refers to three-dimensional computer graphics, it also encompasses two-dimensional graphics and image processing. Computer graphics is often differentiated from field of visualization, although two have some similarities. Graphics are visual presentation on some surface like wall, canvas, computer screen. Graphics often combine text, illustration and colour.

Computer graphics started with the display of data on hardcopy plotters and cathode ray tube (CRT) screens soon after the introduction of computers. These models come from a diverse and expanding set of fields, and include physical, mathematical, engineering, architectural and even conceptual structures. Computer graphics today is largely interactive. The user controls the contents, structure and appearance of objects and of their displayed images by using input devices such as keyboard, mouse or touch-sensitive panel on screen. Figure 1.1 shows the library organization of OpenGL.

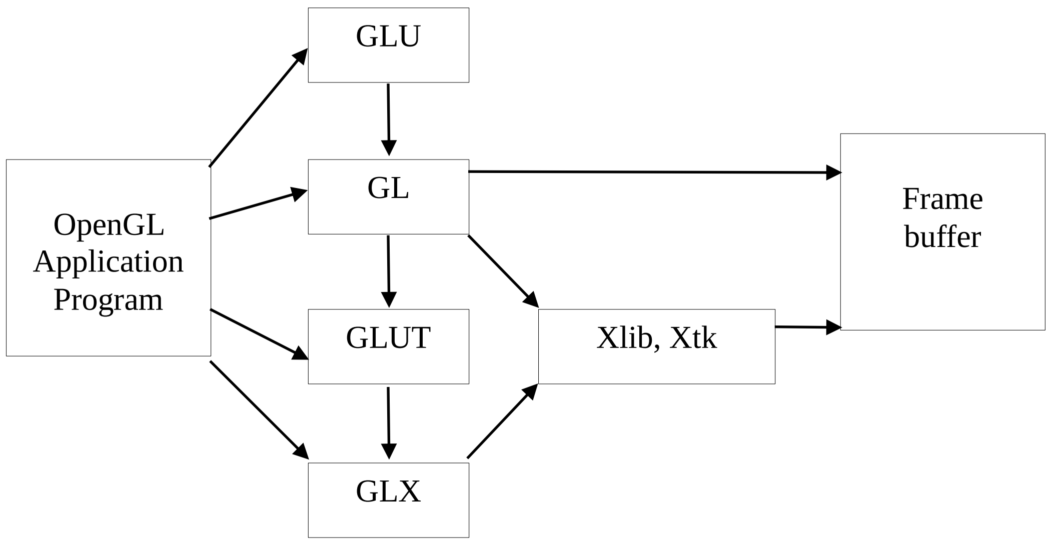


Figure 1.1: OpenGL Library Organization

Computer graphics is no more a rarity. Even people who do not use computers in their daily work encounter computer graphics in television commercials and as cinematic special effects. Computer graphics is a part of all user interfaces and is indispensable for visualizing objects. Graphical interfaces have replaced textual interfaces as the standard means for user-computer interaction. Graphics has also become a key technology for communication ideas, data and trends in most areas of commerce, science, engineering and education. Much of the task of creating effective graphic communication lies in modelling the objects whose image we want to produce.

1.2 OpenGL

Open Graphics Library (OpenGL) 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.

Silicon Graphics Inc., (SGI) started developing OpenGL in 1991 and released it in January 1992; applications use it extensively in the fields of computer-aided design (CAD), virtual reality, scientific visualization, information visualization, flight simulation, and video games. Since 2006 OpenGL has been managed by the non-profit technology consortium Khronos Group.

One aspect of OpenGL that suits it so well for use in computer graphics course is its device independence or portability. A student can develop and run program on any available computer. OpenGL offers rich and highly usable API for 2D graphics and image manipulation, but its real power emerges with 3D graphics.

OpenGL provides a powerful but primitive set of rendering commands, and all higher-level drawing must be done in terms of these commands. Also, OpenGL programs have to use the underlying mechanisms of the windowing system. A number of libraries exist to simplify the programming tasks, including the following:

The OpenGL Utility Library(GLU)contains several routines that use lower-level OpenGL commands to perform such tasks as setting up matrices for specific viewing orientations and projections, performing polygon tessellation, and rendering surfaces. The library is provided as part of every OpenGL implementation.

Function names in the OpenGL basic library are prefixed with gl , and each component word within a function name has its first letter capitalized **glBegin,glClear, glCopyPixels, glPolygonMode** .Component words within a constant name are written in capital letters, and the underscore (_) is used as a separator between all component words in the name. GL_2D, GL_RGB, GL_CCW, GL_POLYGON, GL_AMBIENT_AND_DIFFUSE OpenGL data-types GLbyte, GLshort, GLint, GLfloat, GLdouble, Glboolean.

The **OpenGL Utility Library (GLU)** contains several routines that use lower-level OpenGL commands to perform such tasks as setting up matrices for specific viewing orientations and projections, performing polygon tessellation, and rendering surfaces. The library is provided as part of every OpenGL implementation [2].

OpenGL Utility Toolkit (GLUT)

Provides a library of functions for interacting with any screen-windowing system. The GLUT library functions are prefixed with glut contains methods for describing and rendering quadric curves and surfaces. GLUT is an interface to other device-specific window systems, so the programs will be device-independent [2].

1.3 About Project

In this project we use OpenGL library to demonstrate the working of nuclear power plant. We implement it in C. Initially we construct the schematic diagram of the nuclear power plant using different OpenGL primitives. We also give different colours, textures and thickness to various parts. Once the schematics is completed we'll show how the coolant flows through the entire power plant, moving of control rods and also how the turbine rotates.

The project will contain different options displayed in menu. The user will land in the project name page when he executes the project from where he can navigate to the simulation of the nuclear power plant and also there's one more option provided in the menu which gives the detail of how the power plant work.

The project also contains the option to change the view. We have provided two views, one is closed view which displays the external structure of the project and the other is internal view which displays the internal view of the project.

CHAPTER 2
REQUIREMENT SPECIFICATIONS

The basic purpose of software requirement specification (SRS) is to bridge the communication between the parties involved in the development project. SRS is the medium through which the user's needs are accurately specified; indeed, SRS forms the basis of software development. Another important purpose of developing an SRS is helping the users understand their own needs.

Now we will be discussing the requirement analysis of the project. This report gives the description of the roles of users, the functional overviews of the project, input and output characteristics and also the hardware and software for the project.

2.1 Hardware Requirements

- Processor-Intel or AMD (Advance Micro Devices)
- RAM-512MB(minimum)
- Hard Disk -1MB(minimum)
- Mouse
- Keyboard
- Monitor

2.2 Software Requirements

- C language compliers.
- OpenGL libraries.
- WINDOWS or LINUX based platform OS.
- Gedit.

CHAPTER 3
SYSTEM DESIGN

3.1 FLOWCHART

A flowchart is a common type of chart that represents an algorithms or process showing the steps as boxes of various kinds and their order by connecting these with arrows. The figure 3.1 shows the flow of control in our project

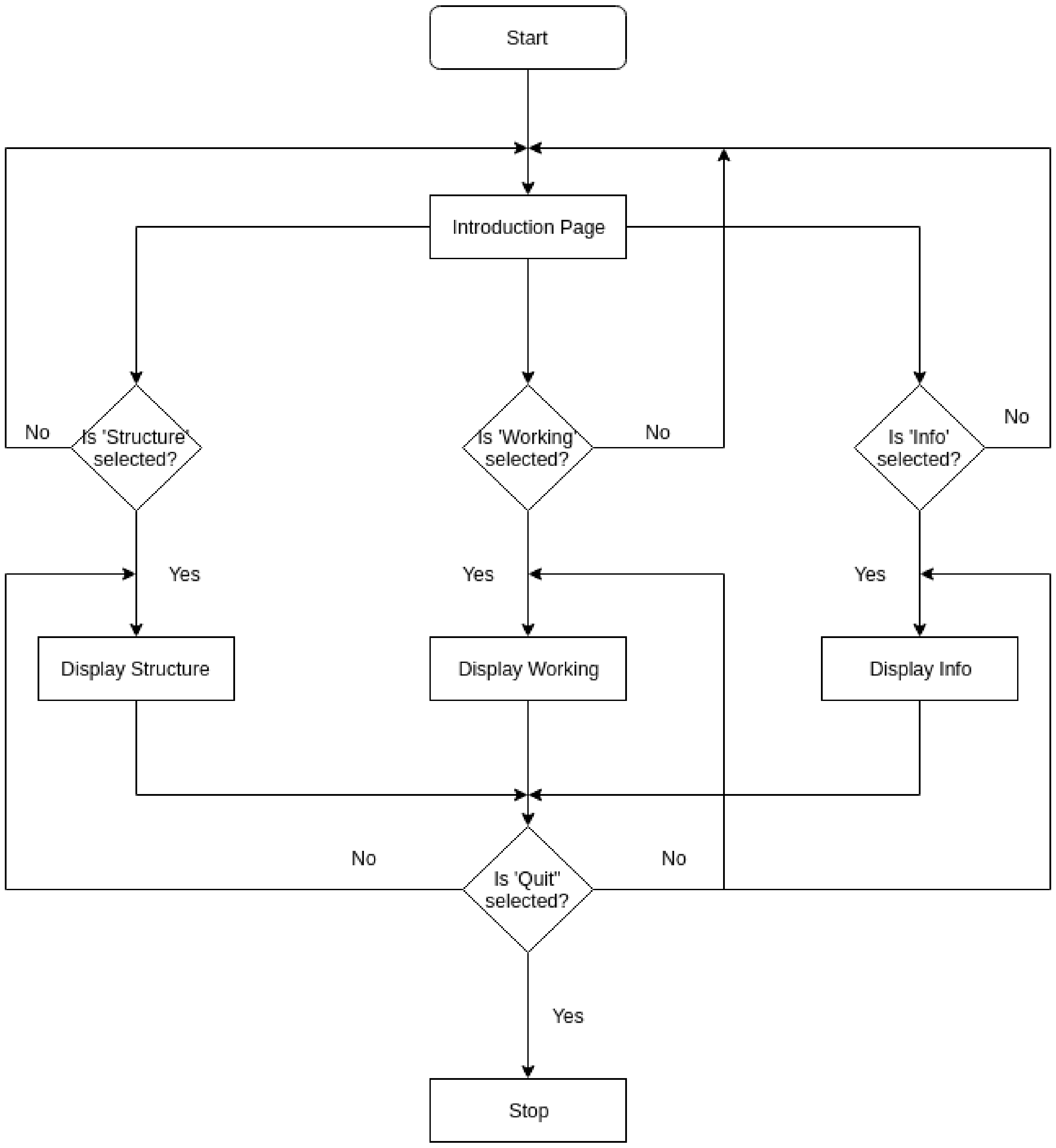


Figure 3.1: Flow chart of project.

CHAPTER 4
IMPLEMENTATION

This project demonstrates the working of a nuclear power plant. Initially when the user executes the code he'll land in introduction page where the title of the project is displayed. The user will be provided with a menu to interact with the program which will take him to different pages.

The user can select the 'view' option where he can select either the closed view of the power plant and also the internal structure of the power plant is provided. The user can get more information about the project by clicking on the 'Info' option in which he'll be able to see the reactions taking place in power plant and also how the power is generated.

If the user wishes to see how the power plant works he can click on the 'Working' option which displays the working in which the flowing of liquid between different parts of the power plant is shown. The liquid starts filling from the reactor vessel and carried to the steam generator where the steam gets generated. The steam produced at the steam generator is carried through steam pipe to the turbine which rotates the turbine and generates power. This is a continuous process. At the same time the water is cooled down using the condenser cooling water and the cooled water is supplied to the steam generator through the pump. Also it displays the rotation of the turbine which generates the power.

Finally, if the user wants to exit from the project he can click on the 'Quit' option.We have implemented the above using following functions.

4.1 OpenGL Functions

4.1.1 Specifying Simple Geometry

void glBegin (glEnum mode)

Initiates a new primitive of type mode and starts the collection of vertices. Values of mode include GL_POLYGON, GL_POINTS and GL_LINES.

void glEnd()

Terminates a list of vertices.

4.1.2 Attributes

void glClearColor(GLclampf r, GLclampf g, GLclampf b, Glclampf a)

Sets the present RGBA clear color used when clearing the color buffer. Variables of GLclampf floating–point numbers between 0.0 and 1.0.

void glPointSize(GLfloat size)

Sets the point size attribute in pixels.

void glLineWidth(GLfloat size)

Sets the line width attribute. Assignment of floating-point value to parameter size, is rounded to the nearest nonnegative integer.

4.1.3 Working with the window

void glFlush()

Forces any buffered OpenGL commands to execute.

void glutInit(int *argc,char **argv)

Initializes GLUT. The arguments from main are passed in and can be used by the application.

int glutCreateWindow(char *title)

Creates a window on the display. The string title can be used to label the window.

void glutInitDisplayMode(unsigned int mode)

Requests a display with properties in mode. The value of mode is determined by logical OR of options including the colour model (GLUT_RGB, GLUT_INDEX) and buffering (GLUT_SINGLE,GLUT_DOUBLE).

void glutInitWindowSize(int width,int height)

Specifies the initial height and width of the window in pixel.

void glutInitWindowPosition(int x,int y)

Specifies the initial position of the top-left corner of the window in pixel.

void glutMainLoop()

Cause the program to enter an event-processing loop. It should be the last statement in main.

void glutDisplayFunc(void (*func)(void))

Registers the display function func that is executed after the current call back returns.

void glutPostRedisplay()

Requests that the display call back be executed after the current call back returns.

4.1.4 Interactions

void glutKeyboardFunc(void *f(char key, int width, int height)

Registers the keyboard call back function f. The call back function returns the ASCII code of the key pressed and the position of the mouse.

4.1.5 Enabling Features

void glEnable(GLenum feature)

Enables an openGL feature. Features that can be enabled include GL_DEPTH_TEST, GL_LIGHTING, GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, GL_LINE_SMOOTH, GL_POLYGON_SMOOTH, GL_POINT_SMOOTH, GL_BLEND, GL_LINE_STIPPLE, GL_POLYGON_STIPPLE, GL_NORMALIZE.

void glDisable(GLenum feature)

Disables an openGL feature.

4.1.6 Transformations

void glMatrixMode(GLenum mode)

Specifies which matrix will be affected by subsequent transformations. Mode can be GL_MODELVIEW, GL_PROJECTION or GL_TEXTURE.

void glLoadIdentity()

Sets the current transformation matrix to an identity matrix

void glPushMatrix() and void glPopMatrix()

Pushes to and pops from the matrix stack corresponding to the current matrix mode.

void glRotate[fd](TYPE angle, TYPE dx, TYPE dy, TYPE dz)

Alters the current matrix by a rotation of angle degrees about the axis(dx, dy, dz).

void glTranslate[fd](TYPE x, TYPE y, TYPE z)

Alters the current matrix by a displacement of(x, y, z).

void glScale[fd](TYPE sx, TYPE sy, TYPE sz)

Alters the current matrix by a scaling of(sx, sy, sz).

4.1.7 Viewing

void gluOrtho2D(GLdouble left, GLdouble right, GLdouble bottom, GLdouble top)

Defines an orthographic viewing volume with all parameters measured from the centre of projection plane.

4.2 Working with the window system

void glFlush();

Forces any buffered OpenGL command to execute.

void glutInit(intargc,char **argv);

glutInit() should be called before any other glut routine, because it initializes the GLUT library. glutInit() will also process command line options, but he specific options are window system dependent.

int glutCreateWindow(char *title);

Creates a window on the display. The string title can be used to label the window. The return value provides a reference to the window that can be used when there are multiple windows.

4.3 User defined functions

We have used many users defined functions for the convenience in translating and using other basic functions.

void fan1(), void fan2(), void fan3(), void fan4()

used to create fans of the turbine.

void rotateFan()

used to rotate fans of the turbine.

void turbine ()

used to create outline of program

void generator ()

used to create the generator of the power plant.

void connecting_pipe()

used to create connecting pipes of the pump

void turbinePoints()

used to create scattered points near the turbine.

void steam()

used to fill steam in the steam line.

void coolant_pipe()

used to create flow of liquid in coolant pipes.

void reactorPoints()

used to create scattered points in the reactor vessel.

void generatorPoints()

used to create scattered points in the steam generator.

void working()

shows the working of the nuclear power plant.

void drawstring(float x,float y,float z,char *string)

used to write something on the screen.

void display_nuclear_power_plant()

used to display the power plant structure.

void display_about(void)

used to create the introduction scene**.**

void reactions(void)

used to show the reaction scene.

void display_operations(void)

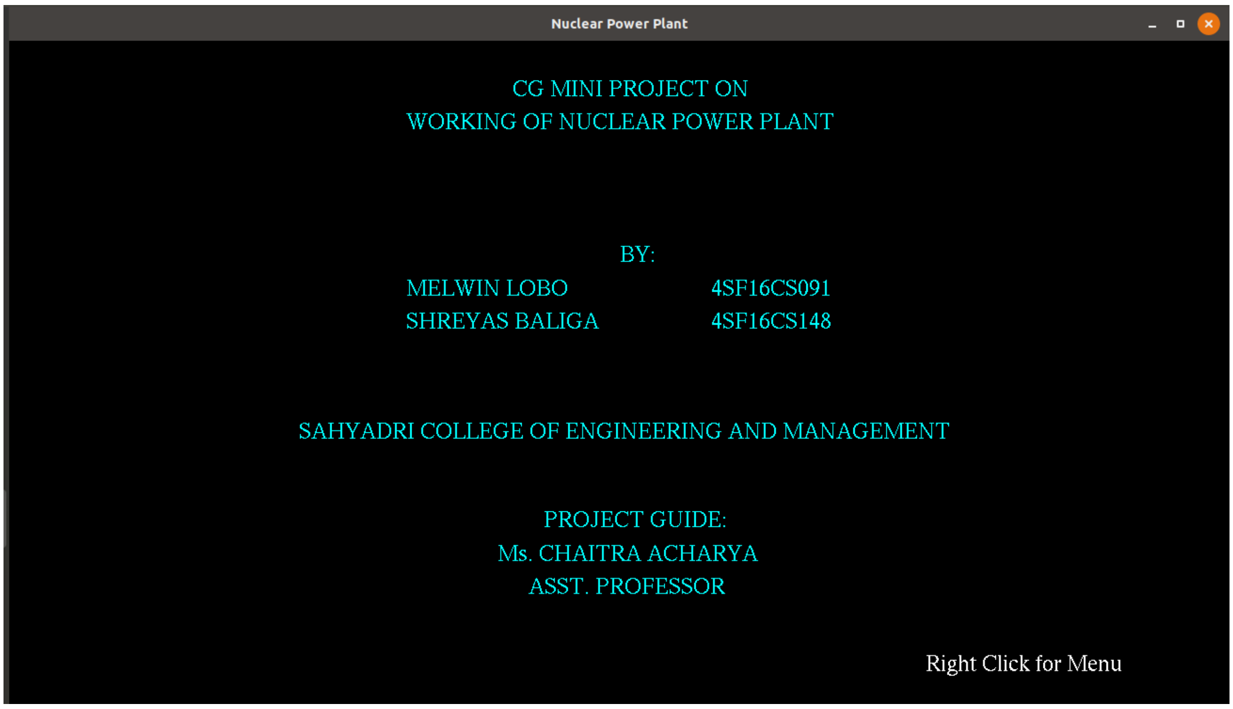
used to show the scene containing the information about the operation of power plant.

void options(int id)

menu function**.**

CHAPTER 5

RESULTS



Nuclear Power Plant

CG MINI PROJECT ON
WORKING OF NUCLEAR POWER PLANT

BY:

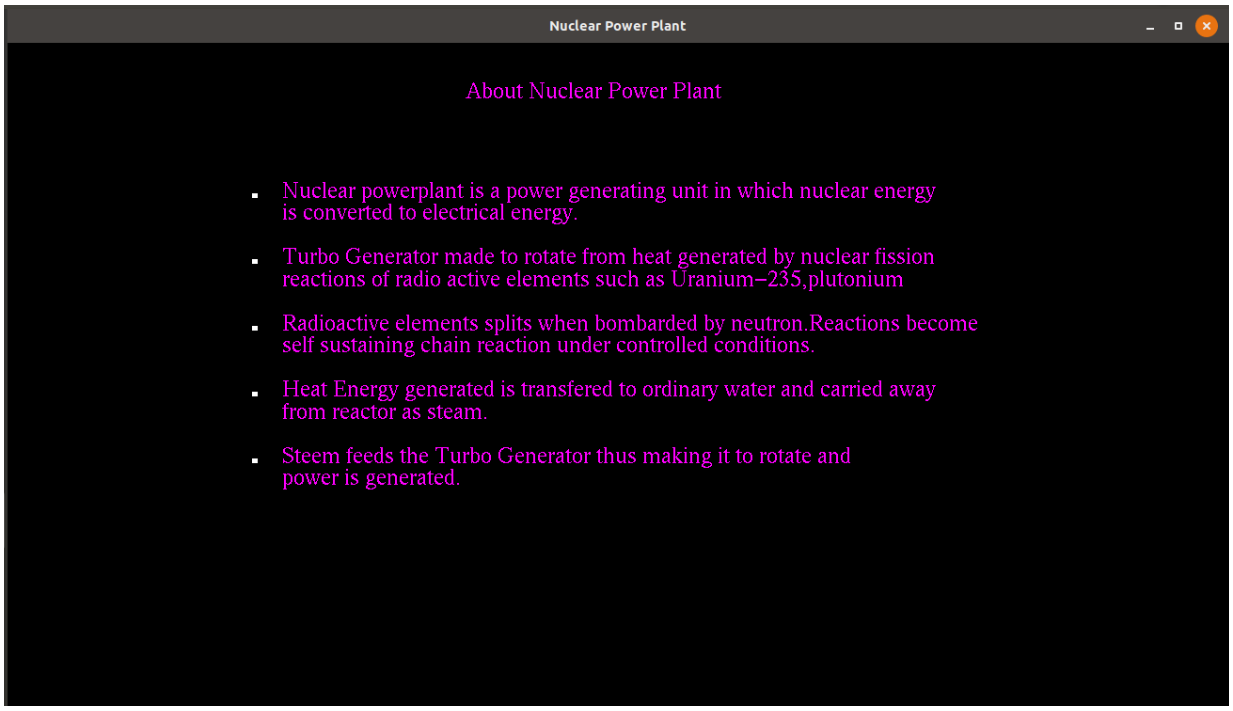
MELWIN LOBO	4SF16CS091
SHREYAS BALIGA	4SF16CS148

SAHYADRI COLLEGE OF ENGINEERING AND MANAGEMENT

PROJECT GUIDE:
Ms. CHAITRA ACHARYA
ASST. PROFESSOR

Right Click for Menu

Figure 5.1: Shows the Introduction about project members and project guides.

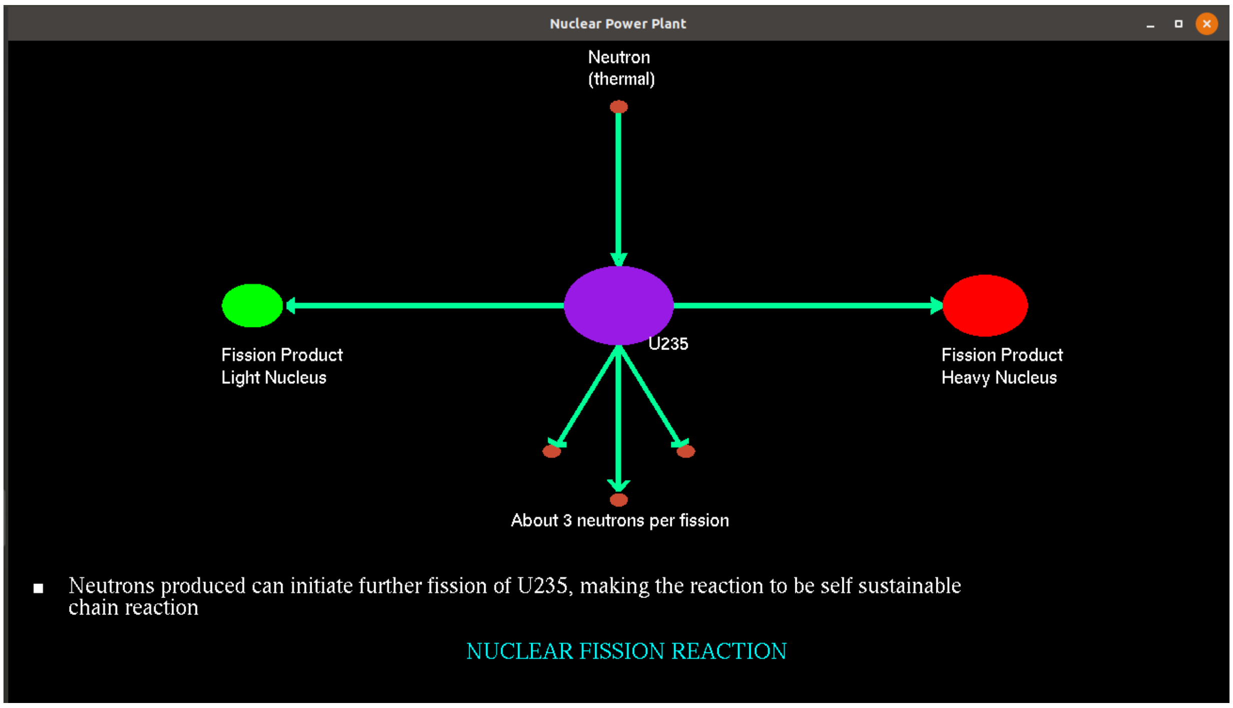


Nuclear Power Plant

About Nuclear Power Plant

- Nuclear powerplant is a power generating unit in which nuclear energy is converted to electrical energy.
- Turbo Generator made to rotate from heat generated by nuclear fission reactions of radio active elements such as Uranium–235,plutonium
- Radioactive elements splits when bombarded by neutron.Reactions become self sustaining chain reaction under controlled conditions.
- Heat Energy generated is transfered to ordinary water and carried away from reactor as steam.
- Steem feeds the Turbo Generator thus making it to rotate and power is generated.

Figure 5.2: Shows the information about nuclear power plant


Nuclear Power Plant
Neutron (thermal)
U235
Fission Product Light Nucleus
Fission Product Heavy Nucleus
About 3 neutrons per fission

- Neutrons produced can initiate further fission of U235, making the reaction to be self sustainable chain reaction

NUCLEAR FISSION REACTION

Figure 5.3: Shows the information about the reactions in a nuclear power plant.

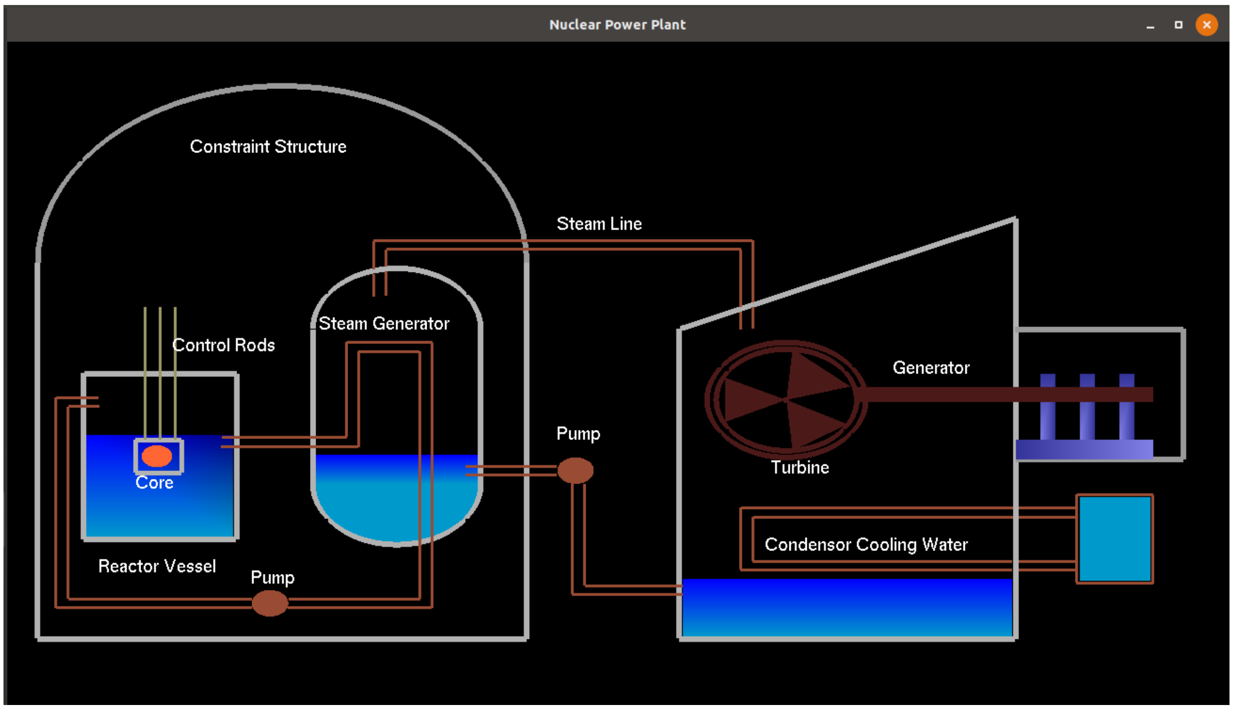


Figure 5.4: Shows the internal structure of nuclear power plant.

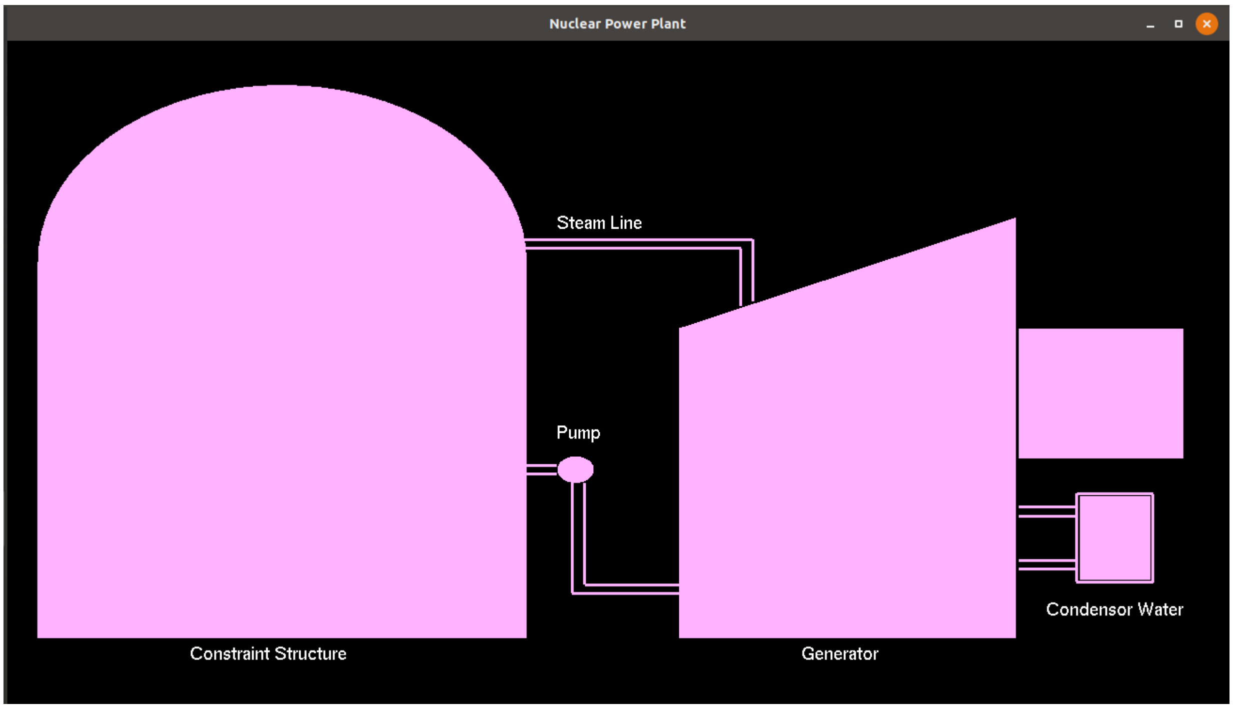


Figure 5.5: Shows the closed view structure of nuclear power plant.

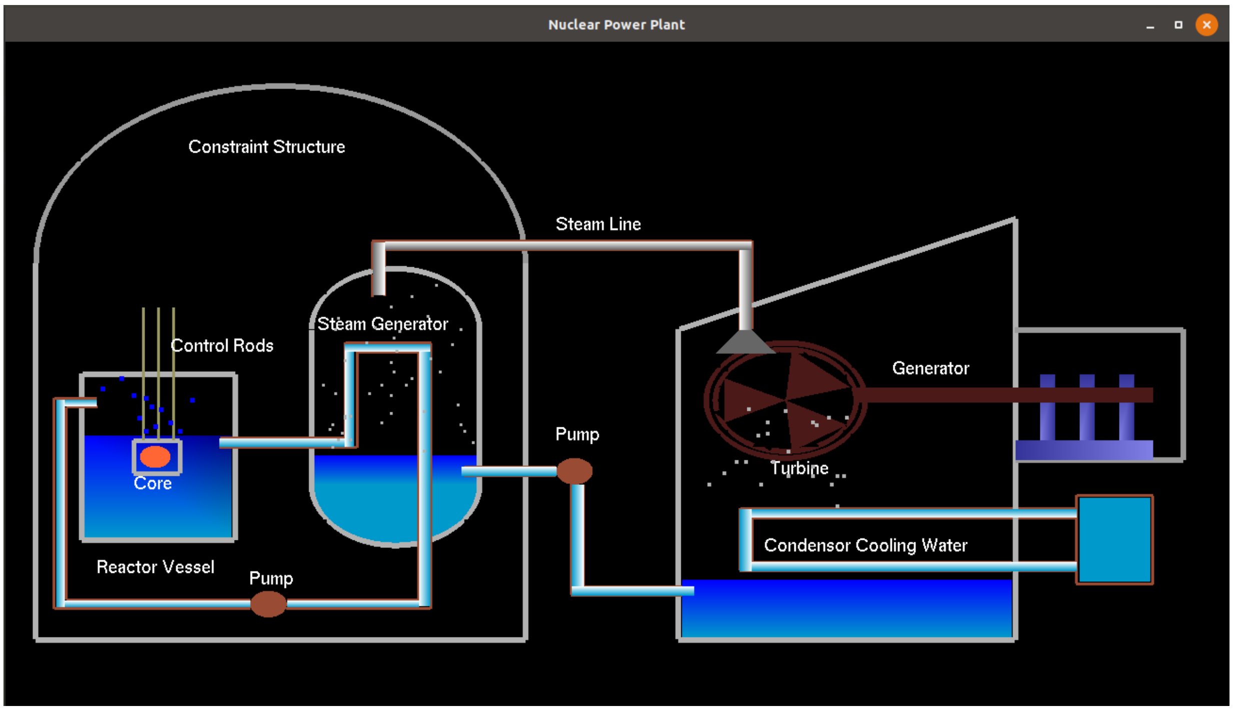


Figure 5.6: Shows the working of nuclear power plant.

CHAPTER 6
CONCLUSION

Our project **“Visualization of Nuclear Power Plant”** has helped us in understanding about computer graphics and OpenGL basic functions which can be used to manipulate the data and provide some animations and various concepts and methodologies used in computer graphics.

The project helped us learn more about the subject practically and implement it in this project to demonstrate some simple movements of the objects, sudden appearance of objects, and many other functions which is useful in depicting some of the basic concepts in our day today life.

Developing this project helped us to learn more about the computer graphics practically and to add realistic animations required for the project. The project is user friendly and has the features which can be easily understood by any user. It demonstrates the OpenGL applications in 2D with animation. The program has been written in C language with OpenGL libraries.

The project is used as an informative animation depicting the working of nuclear power plant. The project is used to demonstrate the uses of computer graphics and OpenGL functions in a very informative and colourful way. The project can be extended further by adding all the animations required for showing what happens if the power plant reactor core melts down and the harm caused.

REFERENCES

[**1**].https://www.glprogramming.com

[**2**].https://energyeducation.ca

[3] Interactive Computer Graphics A Top-Down Approach with OpenGL -Edward Angel, 5th Edition, Addison-Wesley, 2008.

[4] James D Foley, Andries Van Dam, Steven K Feiner, John F Hughes, **Computer Graphics**, Pearson Education 1997.

[5] OpenGL Super Bible: Comprehensive Tutorial and Reference (5th Edition).

[6] F.S. Hill Jr**.: Computer Graphics using OpenGL**, 3rd Edition, PHI, 2009.