Chapter 1

Preamble

1.1 Introduction

Pictorial synthesis of real/imaginary objects from computer-based models is Computer Graphics. 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.

1.1.1 History of Computer Graphics

Computer Graphics is the creation, manipulation, and storage of models and images of picture objects by the aid of computers. This was started with the display of data on plotters and CRT. Computer Graphics is also defined as the study of techniques to improve the communication between user and machine, thus Computer Graphics is one of the most effective medium of communication between machine and user.

William fetter was credited with coning the term Computer Graphics in 1960, to describe his work at Boeng. One of the first displays of computer animation was future world (1976), which included an animation of a human face and hand-produced by Carmull and Fred Parkle at the University of Utah.

There are several international conferences and journals where the most significant results in computer-graphics are published. Among them are the SIGGRAPH and Euro graphics conferences and the association for computing machinery (ACM) transaction on Graphics journals.

1.1.2 Introduction to Open GL

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.

OpenGL draws primitives—points, line segments, or polygons—subject to several selectable modes. Primitives are defined by a group of one or more vertices. A vertex defines a point, an endpoint of a line, or a corner of a polygon where two edges meet. Data (consisting of vertex coordinates, colors, normal, texture coordinates, and edge flags) is associated with a vertex, and each vertex and its associated data are processed independently, in order, and in the same way.

The figure 1.1. 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.

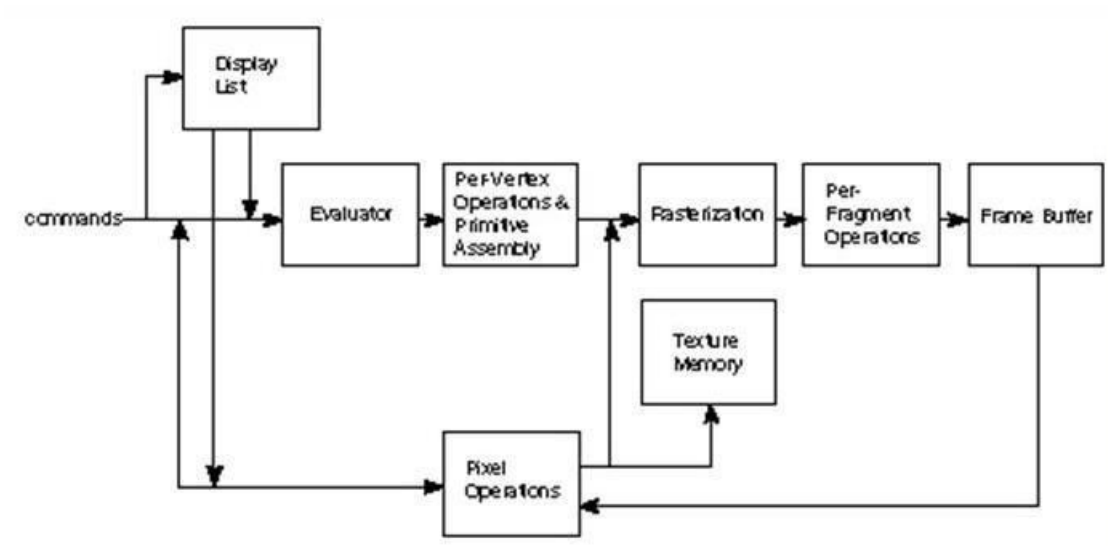


Figure 1.1: OpenGL Block Diagram

We can choose to accumulate some of commands in a display list for processing at a later time. The evaluator stage of processing provides an efficient means for approximating curve and surface geometry by evaluating polynomial commands of input values. 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 perform the final operations on the data before it's stored as pixels in the frame buffer.

1.2 Objectives

Objectives are

- To design a model using C++ graphical functions to demonstrate the interactions of alpha, beta and gamma rays with matter.
- To demonstrate and compare the penetrating powers of alpha, beta and gamma rays for different objects.

1.3 Organization of the Report

Chapter 1 provides the information about the basics of OpenGL. In Chapter 2, all the OpenGL functions used in our program is described. In Chapter 3, system requirements and specifications are specified. Chapter 4 gives the idea of the project and its actual implementation. In Chapter 5 discuss about the results and include the screenshots. Chapter 6 concludes by giving the direction for future enhancement and references are specified.

1.4 Summary

The chapter discussed before is an overview about the computer graphics, its history and OpenGL interface. It even includes the OpenGL block diagram. The scope of study and objectives of the project are mentioned clearly. The organization of the report is been pictured to increase the readability. Further, coming up chapter depicts the OpenGL built-in functions used in project source code.

Chapter 2

A Preview of OpenGL Functions

When we start drawing any graphics in OpenGL using C language we need a header file called <GL/glut.h>. The header file contains definitions and explanations of all functions and constants we'll need, whereas the graphics functions are kept in the library file. Both these files are provided as a part of TURBO C compliers.

- **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 etc.

- **stdlib.h:**

  This is also one the standard library header file which contains the entire standard library functions like exit, free, alloc, mallocetc.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.

- **string.h:**

  This header file is used to manipulate different functions defined for strings such as strlen-used to find the length of the strings. This file defines the structures, macros, values and functions used by standard input and output routine.

The different OpenGL functions used in our project are described as follows:

- **Name:** glBegin ()

  C specification: glBegin (glEnum mode)

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

.

- **Name:** glPushMatrix ()

  C specification: void glPushMatrix ( )

  Description: Pushes and pops from the attributes stack.

- **Name :**glPopMatrix ( )

  C specification: void glPopMatrix ( )

  Description: All attributes are popped from the stack.

- **Name:** glutInit( )

  C specification: void glutInit ( )

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

- **Name:** glutCreateWindow ( )

  C specification: void glutCreateWindow ( )

  Description: creates a window on the display. The string title can be used to label the window.

- **Name:** glutDisplayMode ()

  C specification: void glutDisplayMode ()

  Description: Requests a display with the properties in mode. The mode is determined by the logical OR of options including the color model.

- **Name:** glutWindowSize ( )

  C specification: void glutWindowSize ( )

  Description: Specifies the initial height and width of the window in pixels.

- **Name:** glutInitWindowPosition ()

  C specification: void glutInitWindowPosition (int x, int y)

  Description: Specifies the initial position of the top-left corner of the window in pixels.

- **Name:** void glVertex2i ()

  C specification: void glVertex2i (GLint x, GLint y);

  Description: This is used within glBegin/glEnd pairs to specify point, line and polygon vertices. The current color, normal, texture coordinates and fog coordinates are associated with the vertex when glVertex is called.

- **Name:** glutMainLoop ()

  C specification: Void glutMainLoop ()

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

- **Name:** glutDisplayFunc ()

  C specification: Void glutDisplayFunc (void (*func) (void))

  Description: Registers the display functions that is executed when the window must to be redrawn.

- **Name:** glMatrixMode ( )

  C specification: void glMatrixMode (GLenum mode)

  Description: Specify matrix will be affected by subsequent transformation mode can be GL_MODELVIEW, GL_PROJECTION.

- **Name:** glTranslatef ( )

  C specification: void glTranslatef (GLfloat *x*, GLfloat *y*, GLfloat *z*)

  Description: glTranslatef produces a translation by (x, y, z).

- **Name:** glutKeyboardFunc ( )

  C specification: void key (unsigned char keys, int x, int y)

  Description: sets the keyboard interaction with the current window.

- **Name:** glClear ()

  C specification: glClear (GL_COLOR_BUFFER_BIT)**;**

  Description: to refresh the color buffer and depth buffer so that ifalgorithm stores information in the depth buffer, we must clear this buffer whenever we wish to redraw the display.

- **Name:** glutSwapBuffers ()

  C specification: glutSwapBuffers ()

  Description: Swaps the buffers of the current window if double buffer performs a buffer swap on the layer in use for the current window. Specifically, glutSwapBuffers promotes the contents of the back buffer of the layer in use of the current window to become the contents of the front buffer. The contents of the back buffer then become undefined.

Chapter 3

About the project

2.1 Overview

When an atom undergoes radioactivity, it emits particles like alpha, beta and gamma rays. Radioactivity basically occurs because the unstable atom tries to attain stability. Hence, when they are unstable, they eventually decay by emitting a particle transforming the nucleus into another nucleus, or into a lower energy state. This chain of decays continues till the nucleus attains the stage of stability.

There are basically three types of radiations that are emitted by radioactive particles. These three are called the alpha, beta and gamma rays. All these radiations are released from the nucleus of the atom. Though all three cause some ionization and have some penetration power, but their behavior differs from the others.

2.2 Objective

- The aim of this project is to develop a graphics package which supports operations which include creating objects and transformation operations like translation, rotation etc. This project demonstrates the behavior of different radiation with different materials.
- The package must have an user friendly interface.
- Creation of primitives i.e. points, lines and polygons.
- Providing human interaction through Mouse and Keyboard.

2.3 Description:

➢ Penetration of alpha particles:

- Alpha particles can be absorbed by a thin sheet of paper or by a few centimeters of air. As alpha particles travel through air they collide with nitrogen and oxygen molecules.
- With each collision they lose some of their energy in ionising the air molecule until eventually they give up all of their energy and are absorbed. In a sheet of paper the molecules are much close together so the penetration of alpha particles is much less than in air.

➢ Penetration of beta particles:

- Beta particles travel faster than alpha particles and carry less charge (one electron compared to the 2 protons of an alpha particle) and so interact less readily with the atoms and molecules of the material through which they pass. Beta particles can be stopped by a few millimeters of aluminium.

➢ Penetration of gamma particles:

- Gamma rays are the most penetrating of the radiations. Gamma rays are highly energetic waves and are poor at ionizing other atoms or molecules.
- It cannot be said that a particular thickness of a material can absorb all gamma radiation. Many centimeters of lead or many meters of concrete are required to absorb high levels of gamma rays.

2.4 Software Requirements

- ❖ Programming language: C++ using OpenGL
- ❖ Operating system: Ubuntu
- ❖ Compiler: G++ Compiler
- ❖ Graphics library: GL/glut.h
- ❖ OpenGL 2.0 and above

2.5 Hardware Requirements

- ❖ Processor: Pentium or higher processor.
- ❖ Main memory: 512 MB or more RAM.
- ❖ Keyboard: A standard QWERTY keyboard
- ❖ Mouse: 2 or 3 button mouse
- ❖ Monitor: Standard VGA monitor.4
- ❖ GPU: Intel HD Graphics 5000 or better

Chapter 4

System Design and Implementation

4.1 Introduction

Systems design is the process or art of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. One could see it as the application of systems theory to product development.

This Project is implemented using OpenGL, which is proven to be a very efficient tool in the field of computer graphics, programming is done on Ubuntu platform. GL/glut.h library is used to create the objects and to translate them. C++ programming language is used to implement the entire code. Interface to the program is provided with the help of input device keyboard and mouse.

One of the most important aspects of wave properties. This project demonstrates the wave Nature of alpha, beta and gamma rays. It also deals with the penetration power of these Waves depending upon the intensity and frequency of these waves. The penetration test Of these waves is done using objects such as paper, aluminium and lead.

4.2 Overall Design Process

There are few graphical functions used in our project. The arrow keys are used to move the objects down on the rays and stop the corresponding rays according to their property. The various components used in this project are explained below.

4.2.1 MOLECULES

The alpha, beta and gamma molecules are drawn using the glutSolidSphere() Method with different ambient, light and color values. The glutSolidSphere() has the Following parameters:

- GLdouble radius → The radius of the sphere.
- GLint slices → The number of sub divisions around the Z axis.
- GLint stacks → The number of sub divisions along the Z axis.

4.2.2 USER INTERFACE

A set of keys are used to change the following

- User can view the functions of the project by using Keyboard and Mouse function.
- Keys are
    - Key ‘Left-arrow’ to move page wall downwards.
    - Key ‘Down-arrow’ to move the aluminium wall downwards.
    - Key ‘Right-arrow’ to move lead wall downwards.
    - Key ‘Pg up’ to move page wall upwards.
    - Key ‘Pg dn’ to move the aluminium wall upwards.
    - Key ‘end’ to move lead wall upwards.
- Menu Items
    - Penetration directs to the testing page.
    - Page to move page wall downwards.
    - Aluminium to move the aluminium wall downwards.
    - Lead to move lead wall downwards.
    - Page back to move page wall upwards.
    - Aluminium back to move the aluminium wall upwards.
    - Lead back to move lead wall upwards.
    - Quit to exit the screen.

4.2.1 User defined functions:

- **Name:** rwall()
  C specification: void rwall()
  Description: To draw the lead wall which is in the last position.
- **Name:** mwall()
  C specification: void mwall()
  Description: To draw the aluminium wall which is in the middle position.
- **Name:** lwall()
  C specification: void lwall()
  Description: To draw the paper wall which is in the first position.

- **Name:** floor ()

  C specification: void floor ()

  Description: To provide camera angle and view parameter to display floor area.

- **Name:** gamma ()

  C specification: void gamma ()

  Description: To draw the gamma rays on window.

- **Name:** alpha ()

  C specification: void alpha ()

  Description: To draw the alpha rays on window.

- **Name:** beta ()

  C specification: void beta ()

  Description: To draw the beta rays on window.

- **Name:** molecules ()

  C specification: void molecules ()

  Description: To draw the molecules of alpha,beta and gamma who emit radiation.

- **Name:** specialkeyboard ()

  C specification: void specialkeyboard ()

  Description: To take the input from the keyboard.

- **Name:** display ()

  C specification: void display ()

  Description: To display the window with penetration test.

- **Name:** display1 ()

  C specification: void display1 ()

  Description: To display the front page with details.

- **Name:** menu ()

  C specification: void menu ()

  Description: To create menu of the items listed.

4.3.1 Algorithm for Alpha rays Testing

Step 1: BEGIN

Step 2: Initially, the alpha rays are displayed which are emitting from the alpha particle.

Step 3: If _Left_Key_ or Paper from menu is pressed paper wall moves downward to block the alpha rays.

Step 4: When the paper block is placed in path of alpha rays it blocks alpha rays.

Step 5: Alpha rays are only blocked by paper material.

Step 6: if Page_Up_ or Paper_back from is pressed paper wall moves upwards.

Step 6: END

4.3.2 Algorithm for Beta rays Testing

Step 1: BEGIN

Step 2: Initially, the beta rays are displayed which are emitting from the beta particle.

Step 3: If _Down_Key_ or Aluminium from menu is pressed aluminium wall moves downward to block the beta rays.

Step 4: When the aluminium block is placed in path of beta rays it blocks beta rays.

Step 5: Beta rays are only blocked by aluminium material.

Step 6: if Page_Down_ or Aluminium_back from is pressed aluminium wall moves upwards.

Step 6: END

4.3.3 Algorithm for Gamma Rays Testing

Step 1: BEGIN

Step 2: Initially, the gamma rays are displayed which are emitting from the gamma particle.

Step 3: If _Right _Key_ or Lead from menu is pressed lead wall moves downward to block the gamma rays.

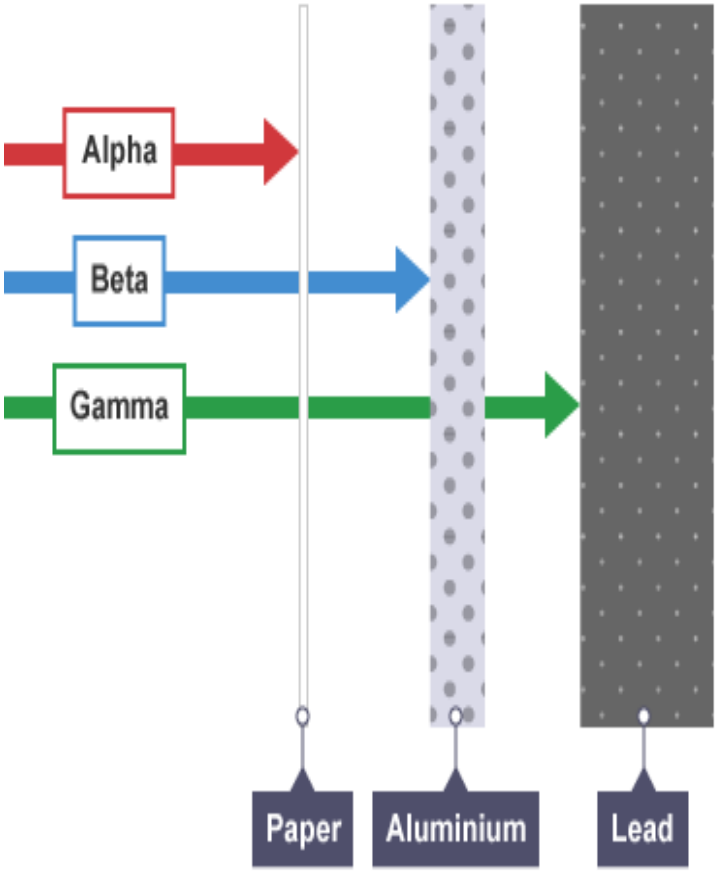
Step 4: When the lead block is placed in path of gamma rays it blocks gamma rays.

Step 5: Gamma rays are only blocked by thick Lead material.

Step 6: if _End _ or Lead_back from is pressed Lead wall moves upwards.

Step 6: END

4.4 Diagram representation of Testing Performed



Particle	*Symbol*	*Mass*	*Penetrating Power*	*Ionizing Power*	*Shielding*
Alpha	α	4 amu	Very Low	Very High	Paper Skin
Beta	β	1/2000 amu	Intermediate	Intermediate	Aluminum
Gamma	γ	0 (energy only)	Very High	Very Low	2 inches lead

Chapter 5

Results and Discussions

The project is compiled and executed on OpenGL and C++ language. This chapter shows few screen shots that illustrates the behavior of rays with different materials.

Snapshots:

Initially, when program is executed and run, a display window is displayed with the objects in their initial positions.

- This is the first window of this project which includes the title of the project, project guider name and group members name as shown in figure 5.1. Use Menu to move to testing page.

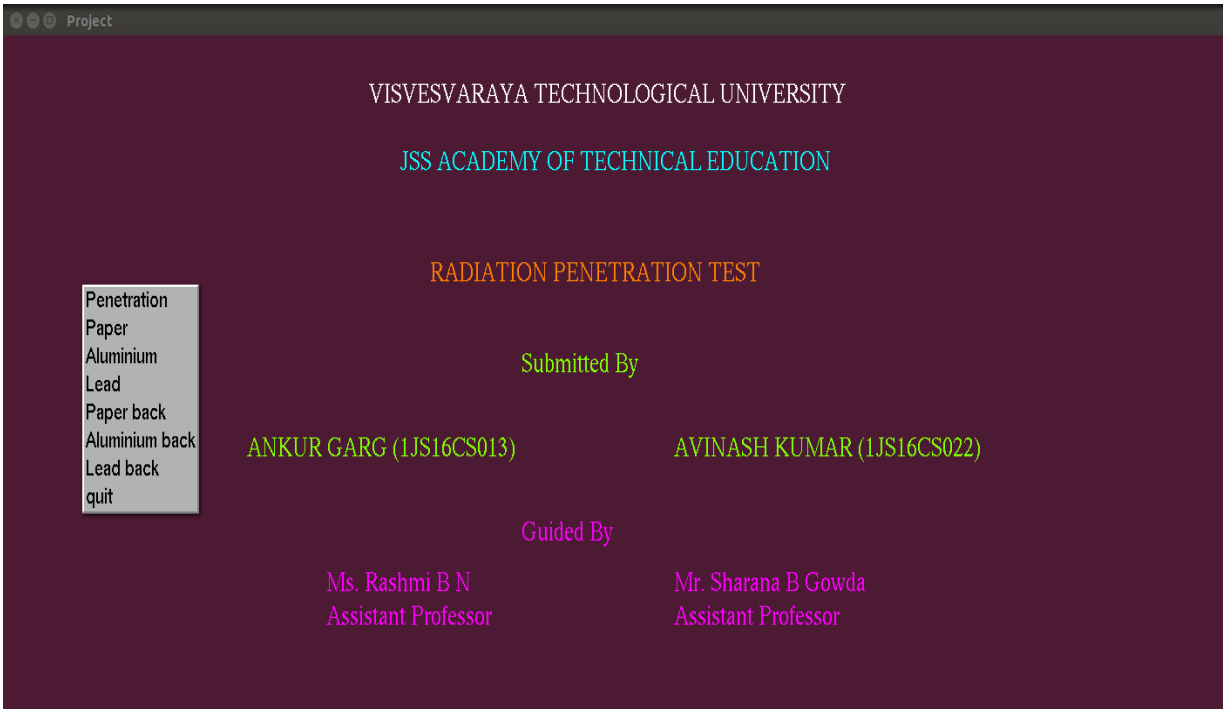


Figure 5.1: shows the home page of the program

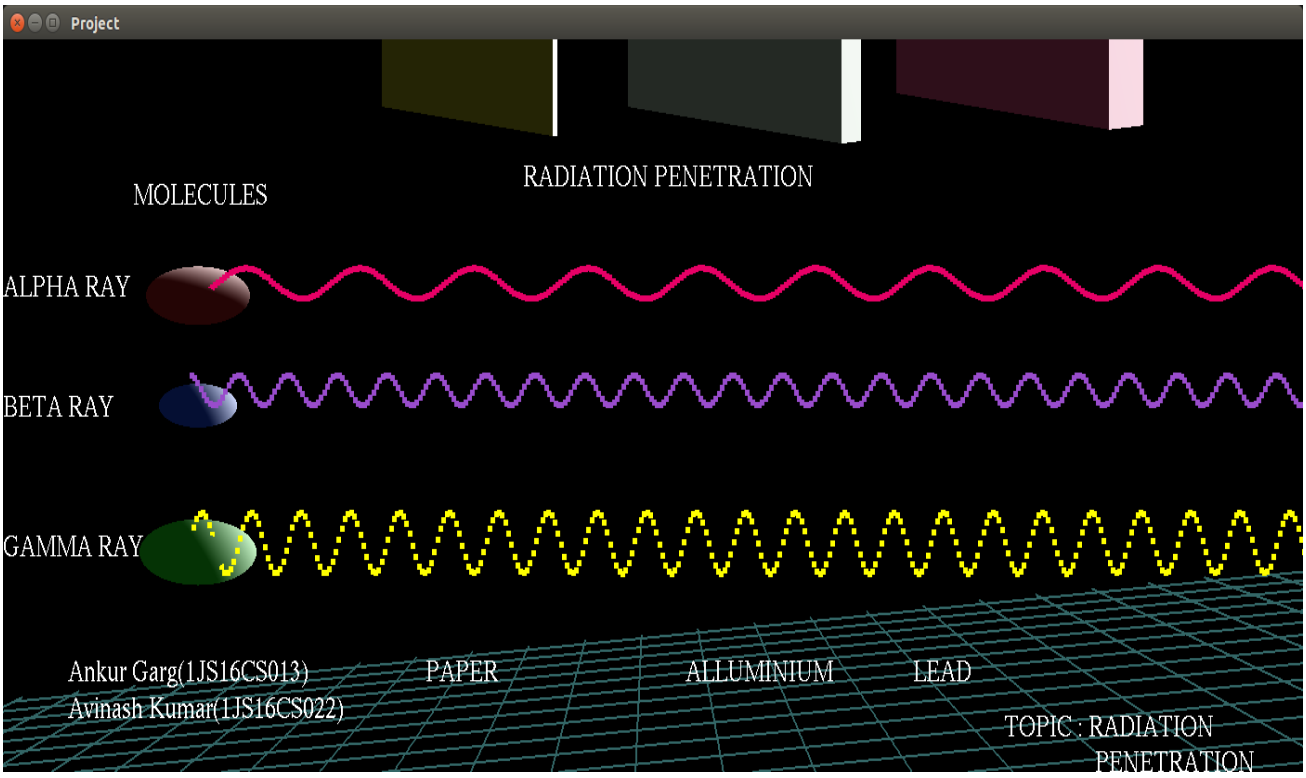


Figure 5.2: shows the initial phase where different particles are emitting different radiations.

- ✓ Alpha Particle emits alpha rays.
- ✓ Beta Particle emits beta rays.
- ✓ Gamma Particle emits gamma rays.

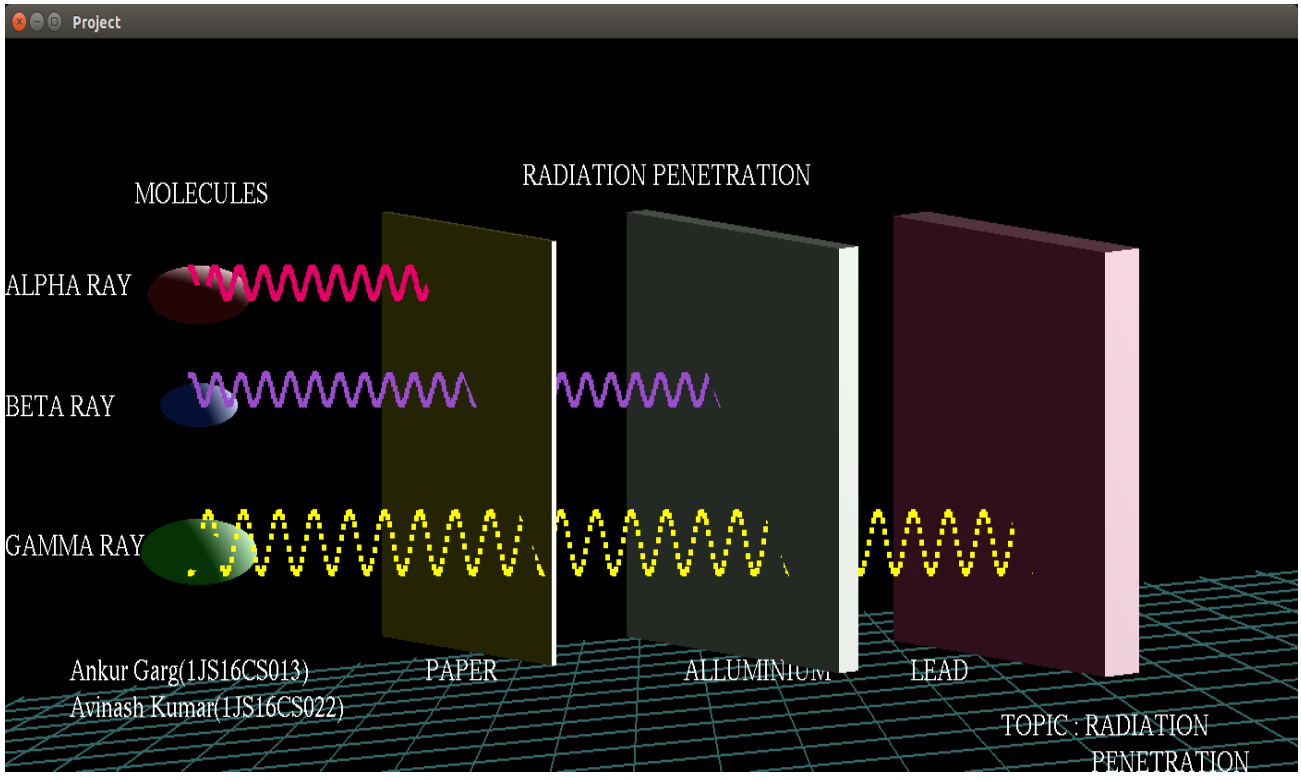


Figure 5.3: shows the blocking of alpha, beta and gamma radiations by different materials.

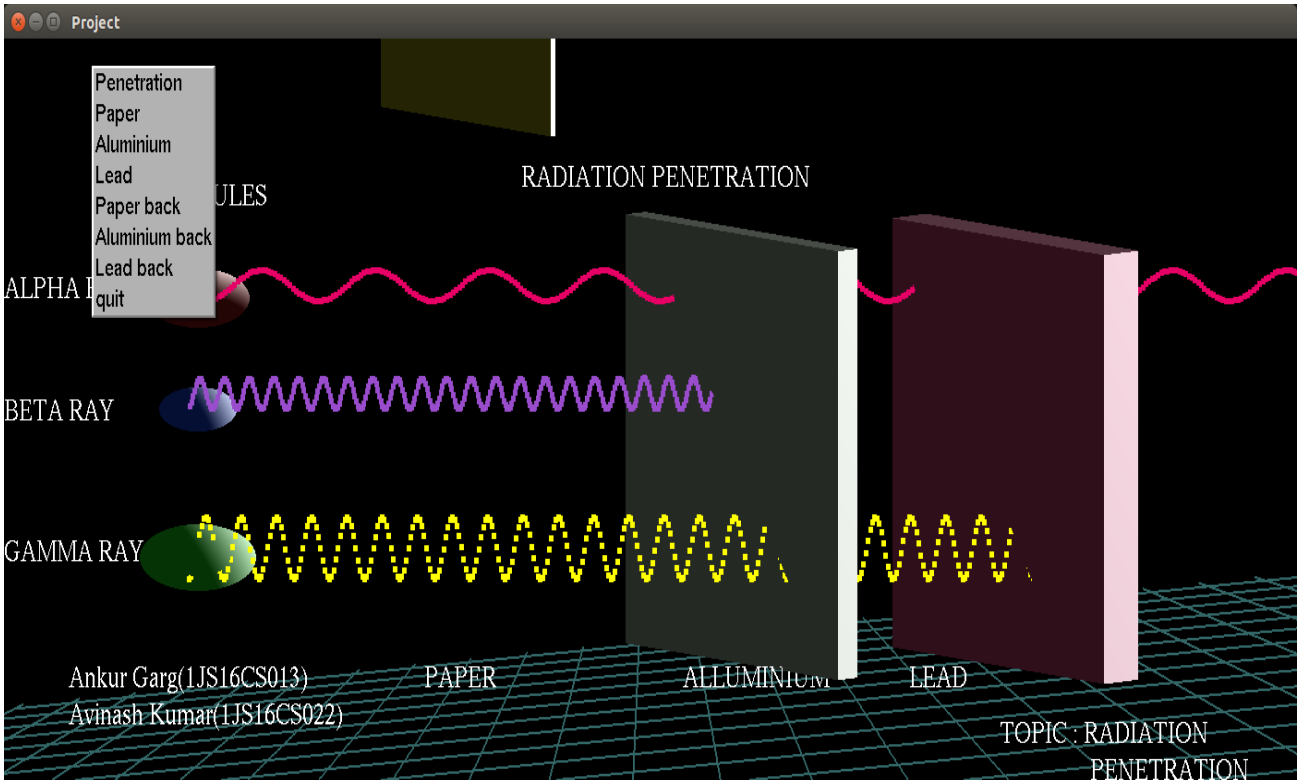


Figure 5.4: shows the that alpha radiations are passed through aluminium and lead and aluminium blocks beta radiations and lead blocks gamma radiations.

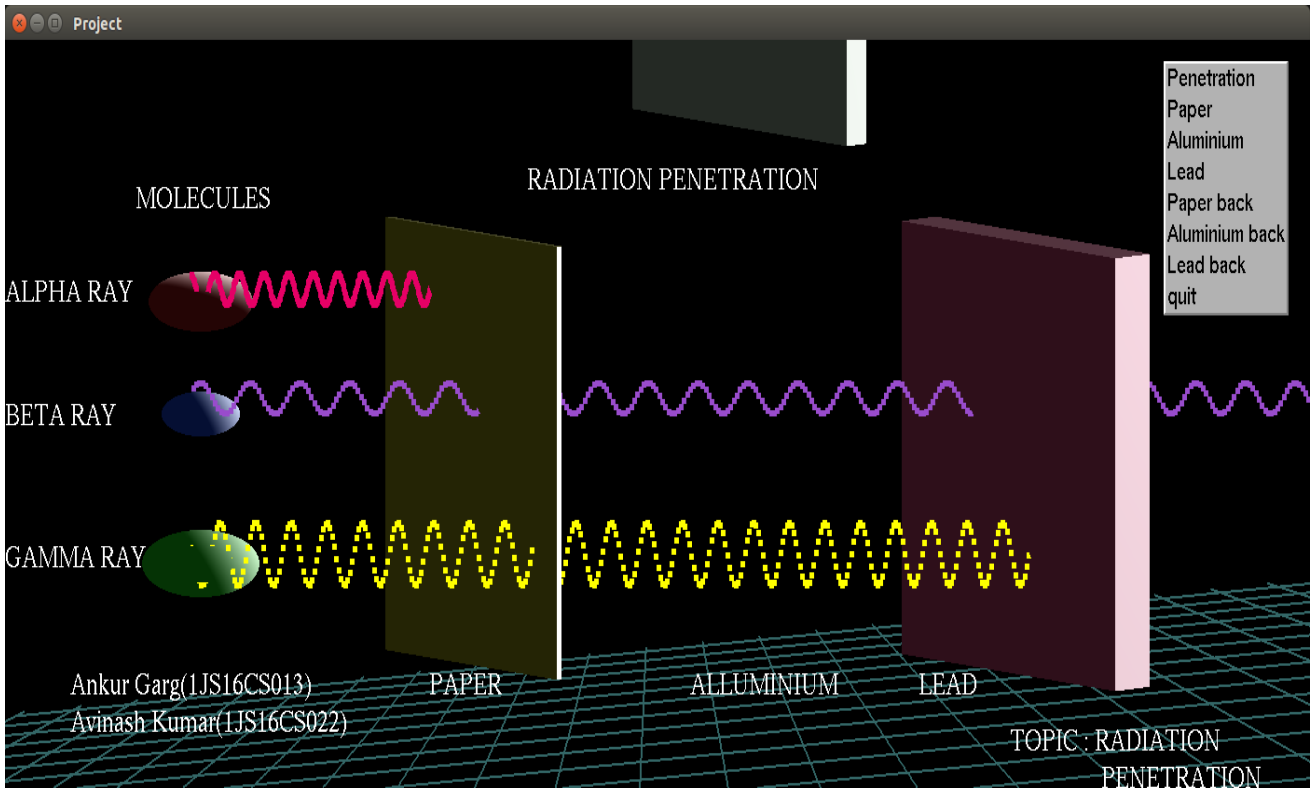


Figure 5.5: shows the that beta radiation are passed through paper and lead and paper blocks alpha radiations and lead blocks gamma radiations.

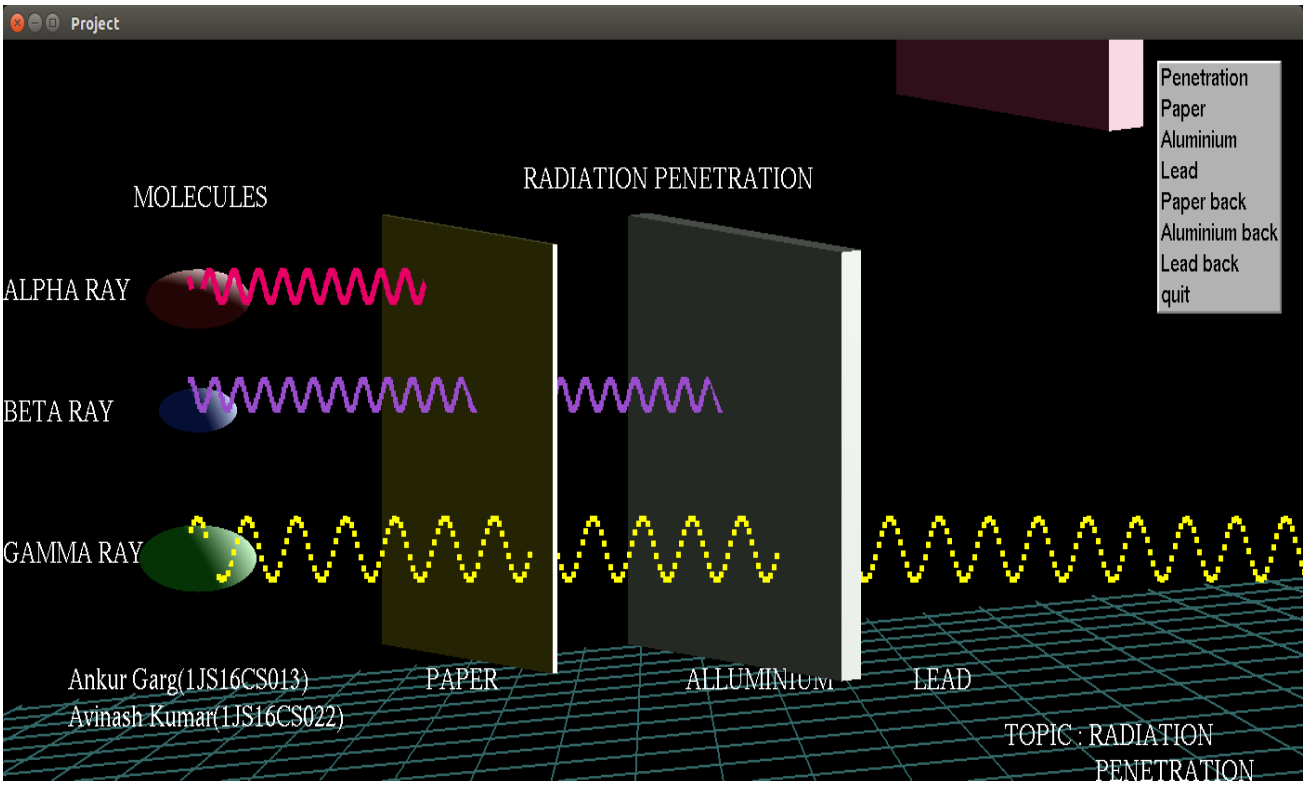


Figure 5.6: shows the that gamma particle are passed through paper and aluminium and

- ✓ Paper blocks alpha radiations
- ✓ Aluminium blocks beta radiations.

- ➢ Keyboard keys are used to move the paper, aluminium and lead wall up and down.
- ➢ Menu function also contains items to move the walls up and down to block and unblock the radiations respectively.

Chapter 6

Conclusion and Future Scope

6.1 Conclusion

This mini project on Radiation penetration testing using OpenGL is a reliable and simple project that provides the user with simulation of different properties of Alpha, Beta, Gamma radiations with respect to different materials. It provides a visual representation of the blocking of radiations by materials. The user-friendly interface allows the user to interact with it very effectively. The user can interact with the interface and can easily simulate and checks the behavior of radiations under different conditions.

It is very efficient in terms of quality of the graphics and memory utilization. It also provides scope for further improvement.

6.2 Future Enhancements

The future scope of our project is vast and can be used in extensive ways:

- ✓ It can be used in simulation to display working of radiations
- ✓ It can be used to provide training to different field workers.
- ✓ Useful in understanding behavior of radiations.
- ✓ Can be used in understanding shielding required specially in nuclear power plants.

References

Web Sources

[1]: https://www.thomasnet.com/articles/custom-manufacturing-fabricating/radiation shielding-materials

[2]: https://www.nrc.gov/about-nrc/radiation/health-effects/radiation-basics.html

[3]: https://www.bbc.com/bitesize/guides/zt9s2nb/revision/3

[4]: www.opengl.org/resources

[5]: http://nuclearconnect.org/know-nuclear/science/protecting

Books

[6] Donald Hearn & Pauline Baker: Computer Graphics with OpenGL Version,3rd / 4th Edition, Pearson Education,2011

[7] Edward Angel: Interactive Computer Graphics- A Top Down approach with OpenGL,5th edition. Pearson Education, 2008.