CHAPTER 1

INTRODUCTION

1.1 COMPUTER GRAPHICS:

Computer Graphics is concerned with all aspect of producing pictures or image using computer. The field began humble almost 50 years ago, with the display of few lines on the cathode-ray tube(CRT); now, we can create image using computer that are indistinguishable from photographs from the real objects. We routinely train pilots with simulated airplane, generating graphical display of the virtual environment in the real time. Feature length movies made entirely by computer have been successful, both critically and financially; massive multiplayer game can involve tens of thousands of concurrent participants.

Graphics is created using computers and, more generally, the representation and manipulation of pictorial data by a computer. The development of computer graphics has made computers easier to interact with and better for understanding and interpreting many types of data. Developments in computer graphics have had a profound impact on many types of media and have revolutionized the animation and video game industry. The phrase "Computer Graphics" was coined in 1960 by William Fetter, a graphic designer for Boeing.

In today's world advanced technology, interactive computer graphics has become a powerful tool for the production of realistic features. Today's we find computer graphics used in various areas that include science, engineering, medicine, business, industry, art, entertainment etc. The main reason for effectiveness of the interactive computer graphics is the speed with which the user can understand the displayed information.

The graphics in openGL provides a wide variety of built-in function. The computer graphics remains one of the most exciting and rapidly growing computer fields. It has become a common element in user interface, data visualization, TV commercials, motion picture and many other applications. The current trend of computer graphics is to incorporate more physics principles into

3D graphics algorithm to better simulate the complex interactions between objects and lighting environment.

1.2 OPEN GL (Open Graphics Library):

OpenGL has become a widely accepted standard for developing graphics application. OpenGL is easy to learn, and it possesses most of the characteristics of other popular graphics system. It is top-down approach. OpenGL is a standard specification defining a cross-language, cross-platform API for writing applications that produce 2D and 3D computer graphics. The interface consists of over 250 different function calls which can be used to draw complex three-dimensional scenes from simple primitives.

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, where it competes with Direct3D on Microsoft Windows platforms.

The interface between the application program and the graphics system can be specified through that set of function that resides in graphics library. The specification is called the APPLICATION PROGRAM INTERFACE (API). The application program sees only the API and is thus shielded from the details both the hardware and software implementation of graphics library. The software driver is responsible for interpreting the output of an API and converting these data to a form that is understood by the particular hardware.

Most of our applications will be designed to access openGL directly through functions in three libraries. Function in the main GL library have name that begin with the letter gl and stored in the library. The second is the openGL utility Library (GLU). This

library uses only GL function but contains codes for creating common object and viewing. Rather then using an different library for each system we used available library called openGL utility toolkit (GLUT). It used as #include<glut.h>

A graphics editor is a computer program that allows users to compose and edit pictures interactively on the computer screen and save them in one of many popular "bitmap" or "raster" a format such as TIFF, JPEG, PNG and GIF.

Graphics Editors can normally be classified as:

- 2D Graphics Editors.
- 3D Graphics Editors.

A 3D Graphics Editor is used to draw 3D primitives Rectangles, Circle, polygons, etc and alter those with operations like cut, copy, paste. These may also contain features like layers and object precision etc.

3D Graphics Editor should include the following features:

- Facilities: Cursor Movement, Editing picture objects.
- Good User Interface: GUI / Toolbars / Icon based User Interface.

Computer Graphics is concerned with all aspects of producing pictures or images using a computer. A particular graphics software system called OpenGL, which has become a widely accepted standard for developing graphics applications .

The applications of computer graphics in some of the major areas are as follows

1. Display of information.
2. Design.
3. Simulation and Animation.
4. User interfaces.

OpenGL is a software interface to graphics hardware. This interface consists of about 150 distinct commands that you use to specify the objects and operations needed to produce interactive three-dimensional applications.

My project named "FLOWING FOUNTAIN MODEL" uses OpenGL software interface and develops 2D images. This project uses the techniques like Translation, motion, display list, transformation techniques, etc.

1.3 PROBLEM SECTION STATEMENT

Computer graphics is no longer a rarity. It is an integral part of all computer user interfaces, and is indispensable for visualizing 2D; 3D and higher dimensional objects .Creating 3D objects, rotations and any other manipulations are laborious process with graphics implementation using text editor. OpenGL provides more features for developing 3D objects with few lines by built in functions.

The geometric objects are the building blocks of any individual .Thereby developing, manipulating, applying any transformation, rotation, scaling on them is the major task of any image development.

Thereby we have put our tiny effort to develop 3D objects and perform different operations on them by using OpenGL utilities.

1.4 EXISTING SYSTEM

The existing system involves computer graphics. Computer graphics started with the display of data on hardcopy plotters and cathode ray tube screens soon after the introduction of computer themselves. It includes the creation, storage and manipulation of models and images of objects.

These models include physical , mathematical , engineering , architectural and so on Computer graphics today is largely interactive –the user controls the contents ,structure and appearance of objects and their displayed images by using input devices , such as keyboard, mouse or touch-sensitive panel on the screen.

Interactive computer graphics is the most important means of producing pictures since the invention of photography and television.

1.5 PROPOSED SYSTEM

In proposed system, the OpenGL is a graphic software system designed as a streamlined, hardware-independent interface to be implemented on many different hardware platforms. To achieve these qualities, no commands for performing windowing tasks or obtaining user input are included in OpenGL; instead, you must work through whatever windowing system controls the particular hardware you're using.

OpenGL doesn't provide high-level commands for describing models of three-dimensional objects. Such commands might allow you to specify relatively complicated shapes such as automobiles, parts of the body, airplanes, or molecules. With OpenGL, you must build up your desired model from a small set of *geometric primitives* - points, lines, and polygons.

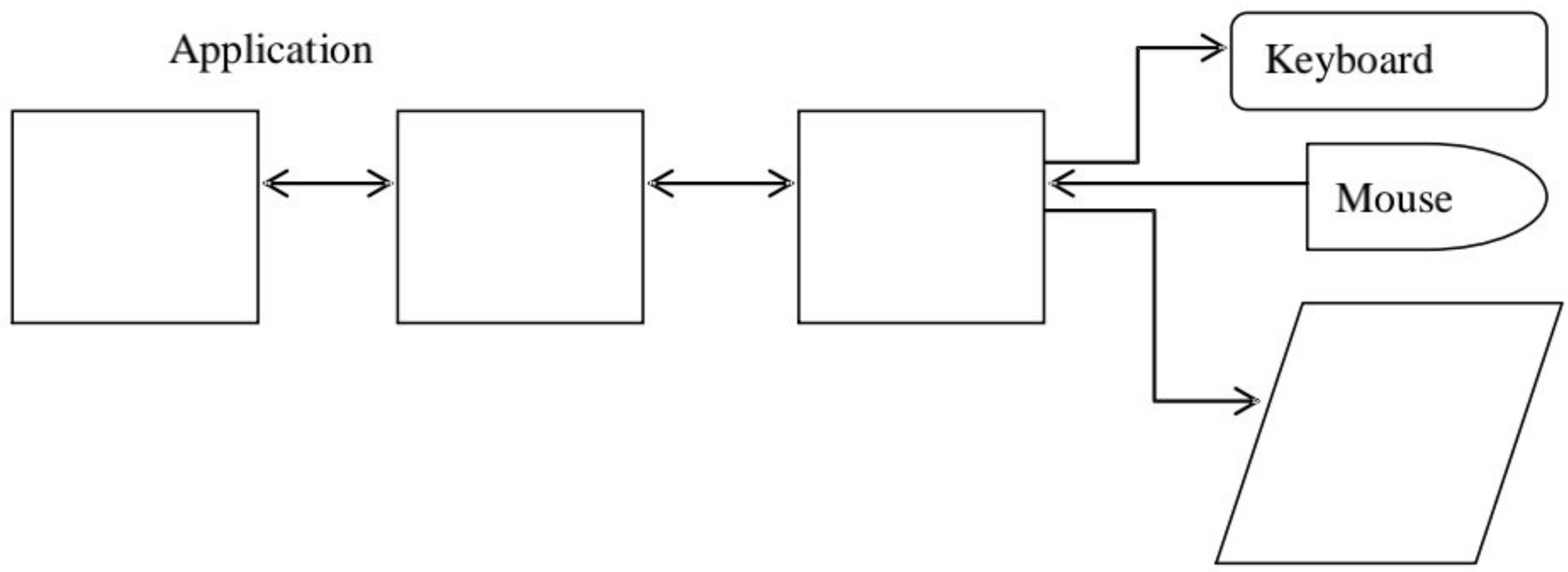

Fig1.1: Application programmers model of graphics system

The interface between an application program and a graphics system can be specified through a set of functions the resides in a graphics library .These specification are called the application programmer's interface (API).The application programmer see only the API and is thus shielded from the details of both the hardware and the

software implementation of the graphics library. The software drivers are responsible for interpreting the output of the API and converting this data to a form that is understood by the particular hardware.

1.6 OBJECTIVES OF THE PROJECT

- Developing a package using computer graphics with OpenGL.
- Migration from text editor to OpenGL.
- To show that implementation of Translation is easier with OpenGL.
- Implementing certain technical concept like Translation, motion, and use of Idle Function.
- How to use Lightning effects used to produce computer animation.

CHAPTER 2

LITERATURE SURVEY

In late 1960's the development of Free-form curves and surfaces for computer graphics begins. Free-form curves and surfaces where developed to describe curved 3-D objects without using polyhedral representations which are bulky and intractable. To get a precise curve with polygons might require thousands of faces, whereas a curved surface requires much less calculations. The UNISURF CAD system was created for designing cars which utilized the curve theories.

A research was made, but was never published so designers get most of the credit. The men were pioneers in Computer Aided Geometric Design (CAGD) for the auto industry, which replaced the use of hand drawn French-curve templates in design of auto bodies. The curves were based on Berstein Polynomials which had been developed by the mathematician Berstein much earlier. Another kind of basic curve predated that was the Hermite Curve developed by the mathematician C. Hermite. Also in the same era as, Schonenberg, a mathematician at the University of Wisconsin was working on Mathematical Splines, which would influence the work of S. Coons at MIT in Splines, Bicubic Surface Patches, Rational Polynomials around 1968. A surface patch is freeform curved surface defined by two or more curves.

In 1973, designers based their research into Parametric B-Splines on Coon's work. The main difference between B-Splines and Bezier curves is the former allows for local control of key control points and the later has more of a global control system. B-Splines are also faster to calculate for a computer than cubic polynomial based curves like the Hermite and Bezier.

Riesenfeld's pioneering development of B-Splines later influenced the E. Catmull's research at Utah.

The "FLOWING FOUNTAIN MODEL". It depicts a 3 Dimensional model of a fountain through which water is continuously flowing out in its idle state. The water gets stored into a small reservoir. The water flows out through different levels in the fountain, giving it a realistic look. User can specify these levels as three, four or five at the beginning.

The program starts with a menu on the screen giving you the options as mention below:

- Proceed.
- Help.
- Exit.

The user is provided with an option to change the color of the fountain using the RIGHT MOUSE BUTTON. The user can view the fountain from different angles including a Top-view and can also zoom in or zoom out. This can be controlled using a set of specified keys on keyboard such as ‘N’ and ‘A’ for ZOOM IN and ZOOM OUT, buttons ‘T’ and ‘F’ for TOP and FRONT VIEW etc. Clicking on the RIGHT MOUSE BUTTON shows a sub menu -'Help' which displays the keyboard shortcuts for various controls. The third option ‘Exit’ pops out of the program.

The project is based on Simple window coordinates and using recursive techniques in OpenGL.

CHAPTER 3

SPECIFICATIONS & REQUIREMENTS

3.1 Hardware requirements:

- Pentium or higher processor.
- 16 MB or more RAM.
- A standard keyboard, and Microsoft compatible mouse
- VGA monitor.
- If the user wants to save the Created files a secondary storage medium can be Used.

3.2 Software requirements:

- The graphics package has been designed for OpenGL; hence the machine must have Eclipse.
- Software installed preferably 6.0 or later versions with mouse driver installed.
- Turbo c Libraries are used and hence a TC version 2 or later is required.

3.3 Development platform: Ubuntu 10.10.

CHAPTER 4

SOFTWARE DESIGN

Movement of a drop:

The movement of a drop contains two factors.

The direction, how the drop gets out of the fountain and the gravity. The position of a drop is pretty easy to compute if we know, how much time has passed since the drop has leaved the fountain.

We have to multiplicate the vector of the constant moving (how the drop leaves the fountain) with the time and then subtract the squared time multiplicated with an acceleration factor. This acceleration factor contains the weight of a drop and the power of gravity. We now have to know the direction, how the drop comes out of the fountain, but this is just a bit calculating with sine and cosine.

Blending means that a pixel on the screen isn't replaced by another one, but they are "mixed". Therefore you can use the alpha value of colors, it indicates how much of the color of the consisting pixel is used for the new color - for antialiasing of points, OpenGL computes this alpha value.

After calling *glEnable(GL_BLEND);* you have to tell OpenGL how to use the alpha values. It isn't specified, that a higher alpha-value means more transparency or something like that. You can use them as you want. To tell OpenGL _what_ you want, you must use *glBlendFunc()*. It takes two parameters, one for the source factor and the second for the destination factor. I used GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA as parameters. This is quite an often used combination and affects, that the higher the alpha value, the less transparency of the incoming fragment

CHAPTER 5

IMPLEMENTATION

Step 1: [To create a fountain]

Declare a class called CDrop.

In GetNewPosition (), we calculate the position and delay of each drop with respect to the co-ordinate axes.

Step 2: [function createlist ()]

Dynamically allocate memory for the required vertices.

Function glGenLists used to generate a contiguous set of empty display lists.

Then specify a series of ‘for ‘loops to construct the top and bottom of the stone.

Then create a qaudrilateral to represent the ground.

To create water, use the following functions:

GlTranslatef () – is to calculate water and stone height.

rand () function is used to generate random unique numbers, for each time it is executed.

Step 3: [function InitFountain ()]

Create fountain drops and vertices. Declare StepAngle, the angle which the ray gets out of the fountain and RayAngle, the angle you see when you look down on the fountain. Use sine () and

cosine () functions inside for loops, to calculate the speed of each step in the fountain, how many steps are required, that a drop comes out and falls down again.

Step 4: [Displaying]

[Keyboard function]

Manages operations by various keys pressed on the key board

[Display function]

Renders the program on to the screen.

Uses the following functions:

GlClear (GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT)- Indicates the buffers currently enabled for color writing and also indicates the depth buffer.

GlPushMatrix (), glPopMatrix () — to push and pop the current matrix stack.

DrawTextXY () — used to set the text of the program.

glFlush () — force execution of GL commands in finite time.

GlutSwapBuffers ()-Swap the buffers ->make the result of rendering visible.

[reshape function]

glMatrixMode (GL_PROJECTION) -applies subsequent matrix operations to the projection matrix stack.

glMatrix

Mode (GL_MODELVIEW)-applies subsequent matrix operations to the model view matrix stack.

We adjust the viewing volume. We use the whole window for rendering and adjust point size to window size.

Step 5: [main function]

Here we specify the initial display mode, window size and position. Create a new window where the output is rendered. Create menus to move near, move away, move down, move up and sub-menus for color, flow, level, and help.

CHAPTER 6

SNAPSHOTS

Fig 6.1 Start Screen

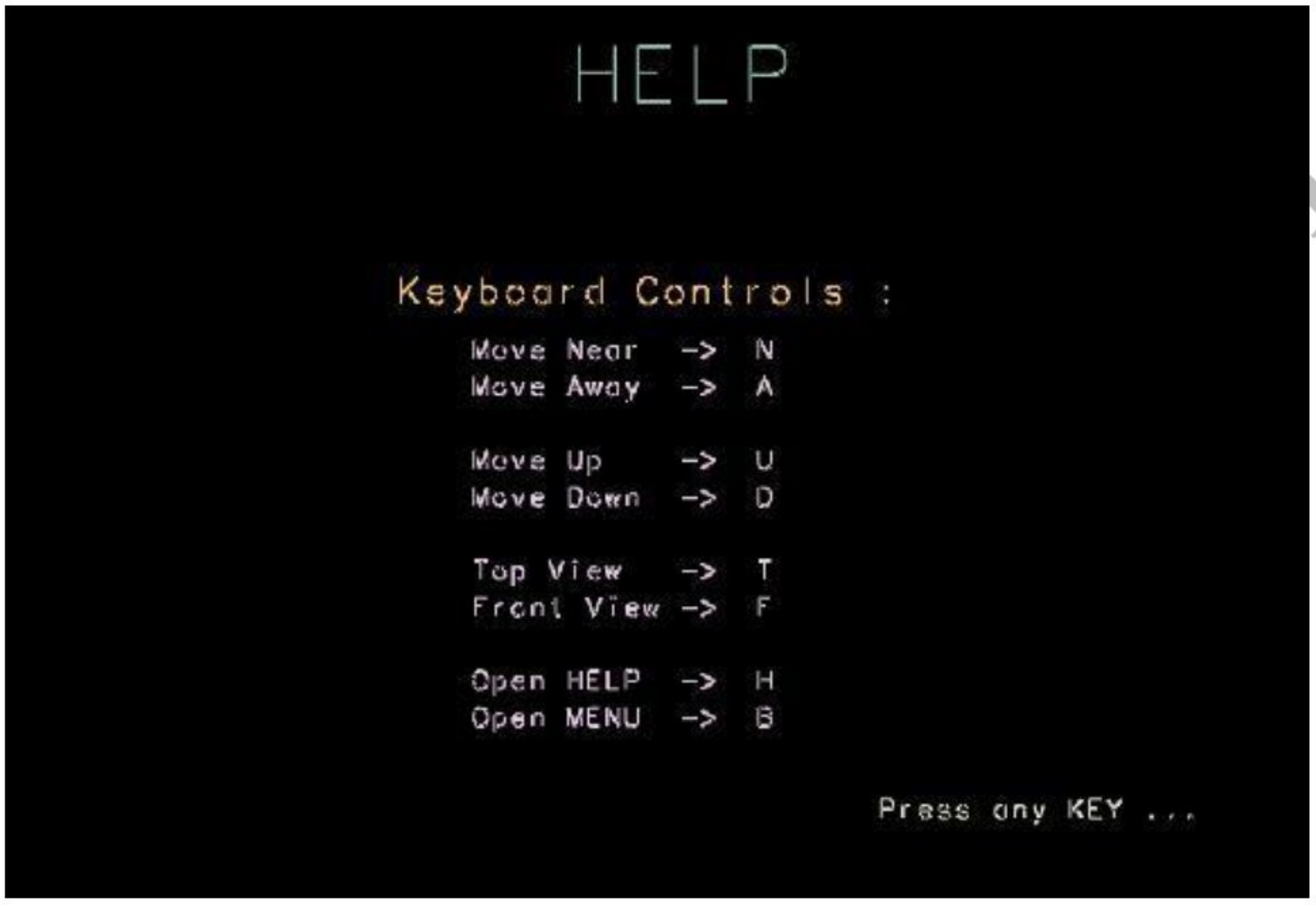

Fig 6.2 Help Menu

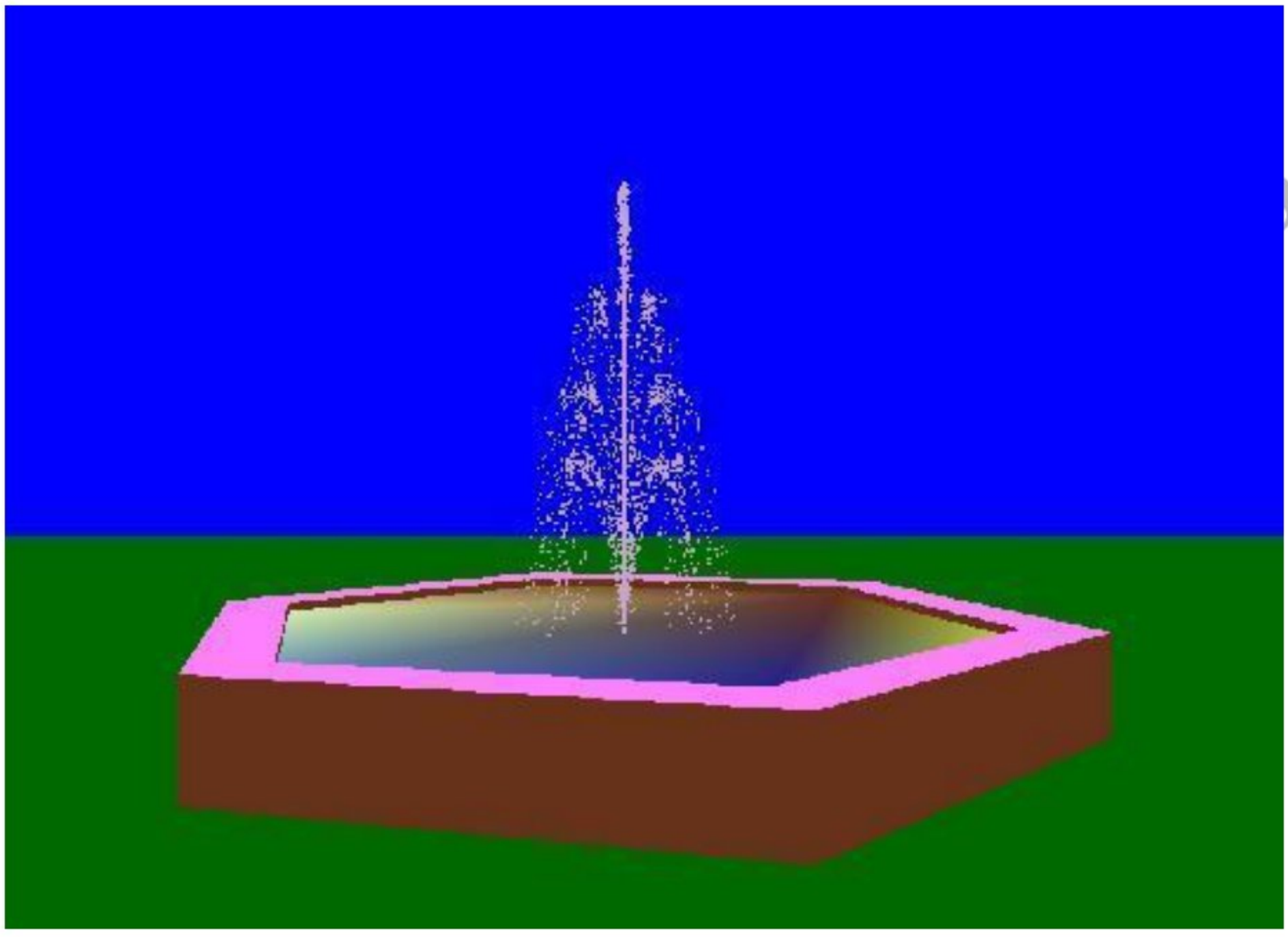

Fig 6.3 Flowing Fountain

CHAPTER 7

CONCLUSION AND FUTURE SCOPE

An attempt has been made to develop an OpenGL package which meets necessary requirements of the user successfully. Since it is user friendly, it enables the user to interact efficiently and easily.

The development of the mini project has given us a good exposure to OpenGL by which we have learnt some of the technique which help in development of animated pictures, gaming.

Hence it is helpful for us even to take up this field as our career too and develop some other features in OpenGL and provide as a token of contribution to the graphics world.

BIBLIOGRAPHY

- Edward Angel's Interactive Computer Graphics Pearson Education 5th Edition

WEBSITES

- www.OpenGL Redbook.
- www.OpenGL simple examples.
- www.OpenGL programming guide.