# Chapter-1 INTRODUCTION

## Computer Graphics

Computer graphics is an art of drawing pictures, lines, charts, using computers with the help of programming. Computer graphics is made up of number of pixels. Pixel is the smallest graphical picture or unit represented on the computer screen. Basically, there are 2 types of computer graphics namely,

**Interactive Computer Graphics** involves a two-way communication between computer and user. The observer is given some control over the image by providing him with an input device. This helps him to signal his request to the computer.

**Non-Interactive Computer Graphics** otherwise known as passive computer graphics it is the computer graphics in which user does not have any kind of control over the image. Image is merely the product of static stored program and will work according to the instructions given in the program linearly. The image is totally under the control of program instructions not under the user. Example: screen savers.

## Applications of Computer Graphics

#### Scientific Visualization

Scientific visualization is a branch of science, concerned with the visualization of threedimensional phenomena, such as architectural, meteorological, medical, biological systems.

#### Graphic Design

The term graphic design can refer to a number of artistic and professional disciplines which focus on visual communication and presentation

#### Computer-aided Design

Computer-aided design (CAD) is the use of computer technology for the design of objects, real or virtual. The design of geometric models for object shapes, in particular, is often called computer-aided geometric design (CAGD). The manufacturing process is tied in to the

computer description of the designed objects so that the fabrication of a product can be automated using methods that are referred to as CAM, computer-aided manufacturing.

#### Web Design

Web design is the skill of designing presentations of content usually hypertext or hypermedia that is delivered to an end-user through the World Wide Web, by way of a Web browser.

#### Digital Art

Digital art most commonly refers to art created on a computer in digital form.

#### Video Games

A video game is an electronic game that involves interaction with a user interface to generate visual feedback on a raster display device.

#### Virtual Reality

Virtual reality (VR) is a technology which allows a user to interact with a computersimulated environment. The simulated environment can be similar to the real world. This allows the designer to explore various positions of an object. Animations in virtual reality environments are used to train heavy equipment operators or to analyse the effectiveness of various cabin configurations and control placements.

#### Computer Simulation

A computer simulation, a computer model or a computational model is a computer program, or network of computers, that attempts to simulate an abstract model of a particular system.

#### Education and Training

Computer simulations have become a useful part of mathematical modelling of many natural systems in physics, chemistry and biology, human systems in economics, psychology, and social science and in the process of engineering new technology, to gain insight into the

operation of those systems, or to observe their behaviour. Most simulators provide screens for visual display of the external environment with multiple panels is mounted in front of the simulator.

#### Image Processing

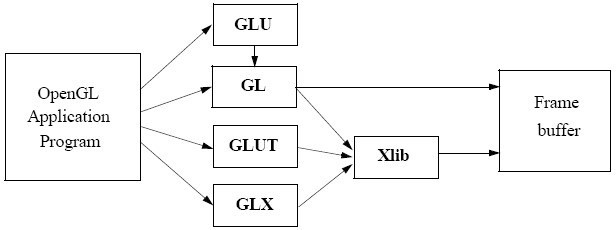
The modification or interpretation of existing pictures such as photographs and TV scans, is called image processing. In computer graphics, a computer is used to create a picture. Image processing techniques, on the other hand, are used to improve picture quality, analyse images, or recognize visual patterns for robotics applications.

## OpenGL

OpenGL has become a widely accepted standard for developing graphics applications. Most of our applications will be designed to access OpenGL directly through functions in the three libraries. Functions in main GL libraries have names that begin with the letters gl and are stored in a library usually referred to as GL.

The second is the OpenGL Utility Library (GLU). This library uses only GL functions but contains code for creating common objects and simplifying viewing. All functions in GLU can be created from the core GL library. The GLU library is available in all OpenGL implementations. Functions in the GLU library starts with the letters glu.

The third is the OpenGL Utility Toolkit (GLUT). It provides the minimum functionality that should be formulated in modern windowing systems.



## Objective of the project

The aim of this project entitled “Rocket launching Simulation“ is to build a graphics package using OpenGL technology & Microsoft Visual C++ as the platform for compilation. It deals with various implementations of OpenGL functions to render objects using primitives. It also demonstrates animating objects using transformation functions in OpenGL.

The scene consists of:

* A area which is suitable to build an rocket
* A space vehicle which is rendered using Basic primitives & it can be moved on pressing key manually using inbuilt timer.

## Organisation of the project

The project was organised in a systematic way. First we analysed what are the basic features to be included in the project to make it acceptable. As it is a graphics oriented project, we made the sketches prior, so as to have an idea like how our output must look like. After all these, the source code was formulated as a paper work. All the required software were downloaded. Finally, the successful implementation of the project.

# Chapter -2 SYSTEM REQUIREMENTS

## Hardware Requirements

* + - Main Processor: PENTIUM III
    - Processor Speed: 800 MHz

|  |  |  |
| --- | --- | --- |
| • | RAM Size: | 128 MB DDR |
| • | Keyboard: | Standard qwerty serial or PS/2 keyboard |
| • | Mouse: | Standard serial or PS/2 mouse |
| • | Compatibility: | AT/T Compatible |
| • | Cache memory: | 256 KB |
| • | Diskette drive: | 1,44MB,3.5 inches |

## Software Requirements

* + - Operating System: Windows XP and Linux(Fedora)
    - Hypervisor used: VMware workstation
    - Compiler used: gcc or g++
    - Language used: C++ language
    - Editor: gedit Text Editor
    - Toolkit: GLUT Toolkit

# Chapter -3

# Control Flow Diagram

Working ();

Mouse function

Keyboard function

Main function

Display function

# Chapter -4 IMPLEMENTATION

## Built in Functions

### 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
    - glColor3fv() : used to render color to faces
    - 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.
  1. **User-Defined Functions**

#### void starting():

#### To display rocket by calling body().

#### void internal():

In this section fuel tank, oxygen tank are constructed by using the values which are earlier defined. In this body**,** fueltankpipe ,oxygentankpipe, cumbustionchamberpipe1, cumbustionchamberpipe2, combustion chamber, nozzle, pump\_pipe1, pump\_pipe2, pump hexagon, satellite callbacks are set and they are defined on the screen using **glutBitmapCharacter** function

#### void working():

#### In this section first body() is called. The working is carried out by executing the function that is called and the control is returned back to required position.

#### void ignite():

#### Define the vertices for flow from combustion chamber,external chamber

#### void starting():

It provides the initial scene of a rocket supported with tower lines.

## Pseudocode

#include<GL/glut.h

#include<math.h>

#include<string.h>

void display();

void myinit();

void mymouse(int button,int state,int x,int y);

void starting();

void internal();

void idle();

void mykey(unsigned char key,int x,int y);

void circle\_draw(int h,int k,int r);

void plotpixels(int h,int k,int x,int y);

void draw\_pixel(int cx,int cy);

void working();

void flow1();

void flow2();

void rotate();

void ignite();

void triangle(float a[2],float b[2],float c[2]);

void divide\_triangle(float a[2],float b[2],float c[2],int k);

void launch();

void body();

void flag();

void oxygentankpipe();

void fueltankpipe();

void pump\_pipe1();

void pump\_pipe2();

void pumphexagon();

void satellite();

void cumbustionchamberpipe1();

void cumbustionchamberpipe2();

void cumbustionchamber();

void nozzel();

void main(int argc,char\*\* argv)

{

glutInit(&argc,argv);

glutInitDisplayMode(GLUT\_RGB| GLUT\_DOUBLE);

glutInitWindowSize(1017,705);

glutInitWindowPosition(0,0);

glutCreateWindow("rocket");

myinit();

glutDisplayFunc(display);

glutIdleFunc(idle);

glutKeyboardFunc(mykey);

glutMouseFunc(mymouse);

glutMainLoop();

}

void body()

{

int i;

if(key2=='s'||key2=='S')

{

glBegin(GL\_POLYGON);

glColor3f(0.95,0.95,0.95);

for(i=0;i<=3;i++)

glVertex2fv(body1[i]);

glEnd();

glBegin(GL\_LINE\_LOOP);

glColor3f(0.0,0.0,0.0);

for(i=0;i<=3;i++)

glVertex2fv(body1[i]);

glEnd();

}

if(key2=='i'||key2=='I'||key2=='w'||key2=='W')

{

glBegin(GL\_POLYGON);

glColor3f(0.5,0.5,0.5);

for(i=0;i<=3;i++)

glVertex2fv(body1[i]);

glEnd();

glBegin(GL\_LINE\_LOOP);

glColor3f(0.0,0.0,0.0);

for(i=0;i<=3;i++)

glVertex2fv(body1[i]);

glEnd();

}

if(key2=='s'||key2=='S'||key2=='i'||key2=='I'||key2=='w'||key2=='W')

{

glBegin(GL\_POLYGON);

glColor3f(0.7,0.7,0.7);

for(i=0;i<=3;i++)

glVertex2fv(body2[i]);

glEnd();

glBegin(GL\_LINE\_LOOP);

glColor3f(0.0,0.0,0.0);

for(i=0;i<=3;i++)

glVertex2fv(body2[i]);

glEnd();

}

}

void pump\_pipe2()

{

glBegin(GL\_POLYGON);

glColor3f(0.3,0.3,0.3);

glVertex3fv(pumppipe2[0]);

glVertex3fv(pumppipe2[1]);

glVertex3fv(pumppipe2[2]);

glVertex3fv(pumppipe2[3]);

glEnd();

glBegin(GL\_POLYGON);

glColor3f(0.3,0.3,0.3);

glVertex3fv(pumppipe2[4]);

glVertex3fv(pumppipe2[5]);

glVertex3fv(pumppipe2[6]);

glVertex3fv(pumppipe2[7]);

glEnd();

glBegin(GL\_LINE\_LOOP);

glColor3f(0.0,0.0,0.0);

glVertex3fv(pumppipe2[0]);

glVertex3fv(pumppipe2[1]);

glVertex3fv(pumppipe2[2]);

glVertex3fv(pumppipe2[3]);

glEnd();

glBegin(GL\_LINE\_LOOP);

glColor3f(0.0,0.0,0.0);

glVertex3fv(pumppipe2[4]);

glVertex3fv(pumppipe2[5]);

glVertex3fv(pumppipe2[6]);

glVertex3fv(pumppipe2[7]);

glEnd();

}

void draw\_pixel(int cx,int cy)

{

glPointSize(2);

glColor3f(0.0,0.0,0.0);

glBegin(GL\_POINTS);

glVertex2i(cx,cy);

glEnd();

}

void mymouse(int button,int state,int x,int y)

{

if(button==GLUT\_LEFT\_BUTTON && state==GLUT\_DOWN)

count=count+1;

}

void idle()

{

theta=theta+325.0;

if(theta>360.0)

theta=theta-360.0;

x=18.0\*cos(radian\*theta);

y=18.0\*sin(radian\*theta);

glutPostRedisplay();

}

void internal()

{

int i;

/\*---------body and tower--------\*/

body();

/\*---------fuel tank-------------\*/

glBegin(GL\_POLYGON);

glColor3f(0.5,0.4,0.3);

for(i=0;i<=3;i++)

glVertex2fv(fuel[i]);

glEnd();

glBegin(GL\_LINE\_LOOP);

glColor3f(0.0,0.0,0.0);

for(i=0;i<=3;i++)

glVertex2fv(fuel[i]);

glEnd();

glBegin(GL\_LINE\_LOOP);

glColor3f(0.0,0.0,0.0);

glVertex2f(-32.0,240.0);

glVertex2f(32.0,240.0);

glVertex2f(32.0,220.0);

glVertex2f(-32.0,220.0);

glVertex2f(-32.0,200);

glVertex2f(32.0,200);

glVertex2f(32.0,180.0);

glVertex2f(-32.0,180.0);

glEnd();

char feul[]="Fuel Tank";

glBegin(GL\_LINES);

glColor3f(0.0,0.0,1.0);

glVertex2f(-20.0,220.0);

glVertex2f(-120.0,220.0);

glEnd();

raster[0]=-120;

raster[1]=220;

for(i=8;i>=0;i--)

{

raster[0]=raster[0]-7;

glColor3f(1.0,0.0,0.0);

glRasterPos2iv(raster);

glutBitmapCharacter(GLUT\_BITMAP\_8\_BY\_13,feul[i]);

}

/\*---------oxygen tank-----------\*/

glBegin(GL\_POLYGON);

glColor3f(0.5,0.4,0.3);

for(i=0;i<=3;i++)

glVertex2fv(oxygen[i]);

glEnd();

glBegin(GL\_LINE\_LOOP);

glColor3f(0.0,0.0,0.0);

for(i=0;i<=3;i++)

glVertex2fv(oxygen[i]);

glEnd();

char oxy[]="Oxygen Tank";

glBegin(GL\_LINES);

glColor3f(0.0,0.0,1.0);

glVertex2f(-20.0,140.0);

glVertex2f(-120.0,140.0);

glEnd();

raster[0]=-120;

raster[1]=140;

for(i=10;i>=0;i--)

{

raster[0]=raster[0]-7;

glColor3f(1.0,0.0,0.0);

glRasterPos2iv(raster);

glutBitmapCharacter(GLUT\_BITMAP\_8\_BY\_13,oxy[i]);

}

/\*---------cumbustion chamber----------------\*/

/\*---------cumbustion chamber pipe1----------\*/

cumbustionchamberpipe1();

/\*---------cumbustion chamber pipe2----------\*/

cumbustionchamberpipe2();

/\*---------cumbustion chamber----------------\*/

cumbustionchamber();

glBegin(GL\_POLYGON);

glColor3f(0.3,0.3,0.3);

glVertex2f(-36.0,-96.0);

glVertex2f(-36.0,-104.0);

glVertex2f(36.0,-96.0);

glVertex2f(36.0,-104.0);

glEnd();

glBegin(GL\_LINE\_LOOP);

glColor3f(0.0,0.0,0.0);

glVertex2f(-36.0,-96.0);

glVertex2f(-36.0,-104.0);

glVertex2f(36.0,-104.0);

glVertex2f(36.0,-96.0);

glEnd();

glBegin(GL\_POLYGON);

glColor3f(0.3,0.3,0.3);

glVertex2f(-36.0,-136.0);

glVertex2f(-36.0,-144.0);

glVertex2f(36.0,-144.0);

glVertex2f(36.0,-136.0);

glEnd();

glBegin(GL\_LINE\_LOOP);

glColor3f(0.0,0.0,0.0);

glVertex2f(-36.0,-136.0);

glVertex2f(-36.0,-144.0);

glVertex2f(36.0,-144.0);

glVertex2f(36.0,-136.0);

glEnd();

char ccum[]="Combustion Chamber";

glBegin(GL\_LINES);

glColor3f(0.0,0.0,1.0);

glVertex2f(-20.0,-120.0);

glVertex2f(-120.0,-120.0);

glEnd();

raster[0]=-120;

raster[1]=-120;

for(i=17;i>=0;i--)

{

raster[0]=raster[0]-7;

glColor3f(1.0,0.0,0.0);

glRasterPos2iv(raster);

glutBitmapCharacter(GLUT\_BITMAP\_8\_BY\_13,ccum[i]);

}

/\*----------nozzle---------------------------\*/

nozzel();

glBegin(GL\_POLYGON);

glColor3f(0.3,0.3,0.3);

glVertex2f(-16.0,-156.0);

glVertex2f(-16.0,-164.0);g

lVertex2f(16.0,-164.0);

glVertex2f(16.0,-156.0);

glEnd();

glBegin(GL\_LINE\_LOOP);

glColor3f(0.0,0.0,0.0);

glVertex2f(-16.0,-156.0);

glVertex2f(-16.0,-164.0);

glVertex2f(16.0,-164.0);

glVertex2f(16.0,-156.0);

glEnd();

glBegin(GL\_POLYGON);

glColor3f(0.3,0.3,0.3);

glVertex2f(-28.0,-196.0);  
glVertex2f(-28.0,-204.0);

glVertex2f(28.0,-204.0);

glEnd();

glBegin(GL\_LINE\_LOOP);

glColor3f(0.0,0.0,0.0);

glVertex2f(-28.0,-196.0);

glVertex2f(-28.0,-204.0);

glVertex2f(28.0,-204.0);

glVertex2f(28.0,-196.0);

glEnd();

char nozzel[]="Nozzel";

glBegin(GL\_LINES);

glColor3f(0.0,0.0,1.0);

glVertex2f(-20.0,-200.0);

glVertex2f(-120.0,-200.0);

glEnd();

raster[0]=-120;

raster[1]=-200;

for(i=5;i>=0;i--)

{

raster[0]=raster[0]-7;

glColor3f(1.0,0.0,0.0);

glRasterPos2iv(raster);

glutBitmapCharacter(GLUT\_BITMAP\_8\_BY\_13,nozzel[i]);

}

/\*---------pump----------------------------------\*/

/\*---------pump pipe1-------------------\*/

pump\_pipe1();

/\*----------pump pipe2-------------------\*/

pump\_pipe2();

/\*----------pump hexagon-----------------\*/

pumphexagon();

/\*----------pump circle------------------\*/

int xc=0,yc=0,ra=28;

circle\_draw(xc,yc,ra);

char pump[]="Pump";

glBegin(GL\_LINES);

glColor3f(0.0,0.0,1.0);

glVertex2f(-20.0,-0.0);

glVertex2f(-120.0,-0.0);

glEnd();

raster[0]=-120;

raster[1]=-0;

for(i=3;i>=0;i--)

{

raster[0]=raster[0]-7;

glColor3f(1.0,0.0,0.0);

glRasterPos2iv(raster);

glutBitmapCharacter(GLUT\_BITMAP\_8\_BY\_13,pump[i]);

}

/\*----------satellite-------------------\*/

satellite();

char sat[]="Satellite";

glBegin(GL\_LINES);

glColor3f(0.0,0.0,1.0);

glVertex2f(-0.0,268.0);

glVertex2f(-120.0,268.0);

glEnd();

raster[0]=-120;

raster[1]=268;

for(i=8;i>=0;i--)

{

raster[0]=raster[0]-7;

glColor3f(1.0,0.0,0.0);

glRasterPos2iv(raster);

glutBitmapCharacter(GLUT\_BITMAP\_8\_BY\_13,sat[i]);

}

glFlush();

}

void launch()

{

int i;

/\*------------remove LAUNCH button--------------------\*/

glBegin(GL\_POLYGON);

glColor3f(0.0,0.0,0.0);

glVertex2f(-292.0,104.0);

glVertex2f(-292.0,80.0);

glVertex2f(-244.0,80.0);

glVertex2f(-244.0,104.0);

glEnd();

/\*----------timer countdown remove-------------------\*/

glBegin(GL\_POLYGON);

glColor3f(0,0,0);

for(i=0;i<=7;i++)

glVertex2fv(time0[i]);

glEnd();

glBegin(GL\_LINE\_LOOP);

glColor3f(0,0,0);

for(i=0;i<=7;i++)

glVertex2fv(time0[i]);

glEnd();

/\*----------tower lines remove-------------\*/

glBegin(GL\_LINES);

glColor3f(0,0,0);

glVertex2f(12.0,340.0);

glVertex2f(160.0,340.0);

glEnd();

glBegin(GL\_LINES);

glColor3f(0,0,0);

glVertex2f(12.0,320.0);

glVertex2f(160.0,320.0);

glEnd();

/\*-----------ground translate--------------\*/

float shift=40.0;

for(i=3;i>=0;i--)

{

ground[i][1]=ground[i][1]-shift;

}

/\*-----------tower translate---------------\*/

for(i=10;i>=0;i--)

{

tower[i][1]=tower[i][1]-shift;

}

/\*-----------hills translate---------------\*/

for(i=9;i>=0;i--)

{

hill1[i][1]=hill1[i][1]-shift;

}

for(i=9;i>=0;i--)

{

hill2[i][1]=hill2[i][1]-shift;

}

for(i=7;i>=0;i--)

{

hill3[i][1]=hill3[i][1]-shift;

}

/\*-----------stars translate---------------\*/

for(i=23;i>=0;i--)

{

stars[i][1]=stars[i][1]-40;

if(stars[i][1]==-320.0)

{

stars[i][1]=400;

}

}

/\*-----------body------------------\*/

glBegin(GL\_POLYGON);

glColor3f(0.95,0.95,0.95);

for(i=0;i<=3;i++)

glVertex2fv(body1[i]);

glEnd();

glBegin(GL\_LINE\_LOOP);

glColor3f(0.0,0.0,0.0);

for(i=0;i<=3;i++)

glVertex2fv(body1[i]);

glEnd();

glBegin(GL\_POLYGON);

glColor3f(0.7,0.7,0.7);

for(i=0;i<=3;i++)

glVertex2fv(body2[i]);

glEnd();

glBegin(GL\_LINE\_LOOP);

glColor3f(0.0,0.0,0.0);

for(i=0;i<=3;i++)

glVertex2fv(body2[i]);

glEnd();

glBegin(GL\_POLYGON);

glColor3f(0.95,0.95,0.95);

for(i=0;i<=5;i++)

glVertex2fv(body3[i]);

glEnd();

glBegin(GL\_LINE\_LOOP);

glColor3f(0.0,0.0,0.0);

for(i=0;i<=3;i++)

glVertex2fv(body3[i]);

glEnd();

glBegin(GL\_POLYGON);

glColor3f(0.7,0.7,0.7);

for(i=0;i<=6;i++)

glVertex2fv(body4[i]);

glEnd();

glBegin(GL\_LINE\_LOOP);

glColor3f(0.0,0.0,0.0);

for(i=0;i<=6;i++)

glVertex2fv(body4[i]);

glEnd();

glBegin(GL\_POLYGON);

glColor3f(0.7,0.7,0.7);

for(i=0;i<=6;i++)

glVertex2fv(body5[i]);

glEnd();

glBegin(GL\_LINE\_LOOP);

glColor3f(0.0,0.0,0.0);

for(i=0;i<=6;i++)

glVertex2fv(body5[i]);

glEnd();

glBegin(GL\_POLYGON);

glColor3f(0.0,0.0,0.0);

for(i=0;i<=3;i++)

glVertex2fv(body9[i]);

glEnd();

glBegin(GL\_LINES);

glColor3f(0.0,0.0,0.0);

glVertex2f(-40.0,-80.0);

glVertex2f(40.0,-80.0);

glEnd();

glBegin(GL\_POLYGON);

glColor3f(0.0,0.0,0.0);

for(i=0;i<=3;i++)

glVertex2fv(body10[i]);

glEnd();

glBegin(GL\_LINES);

glColor3f(0.0,0.0,0.0);

glVertex2f(-40.0,160.0);

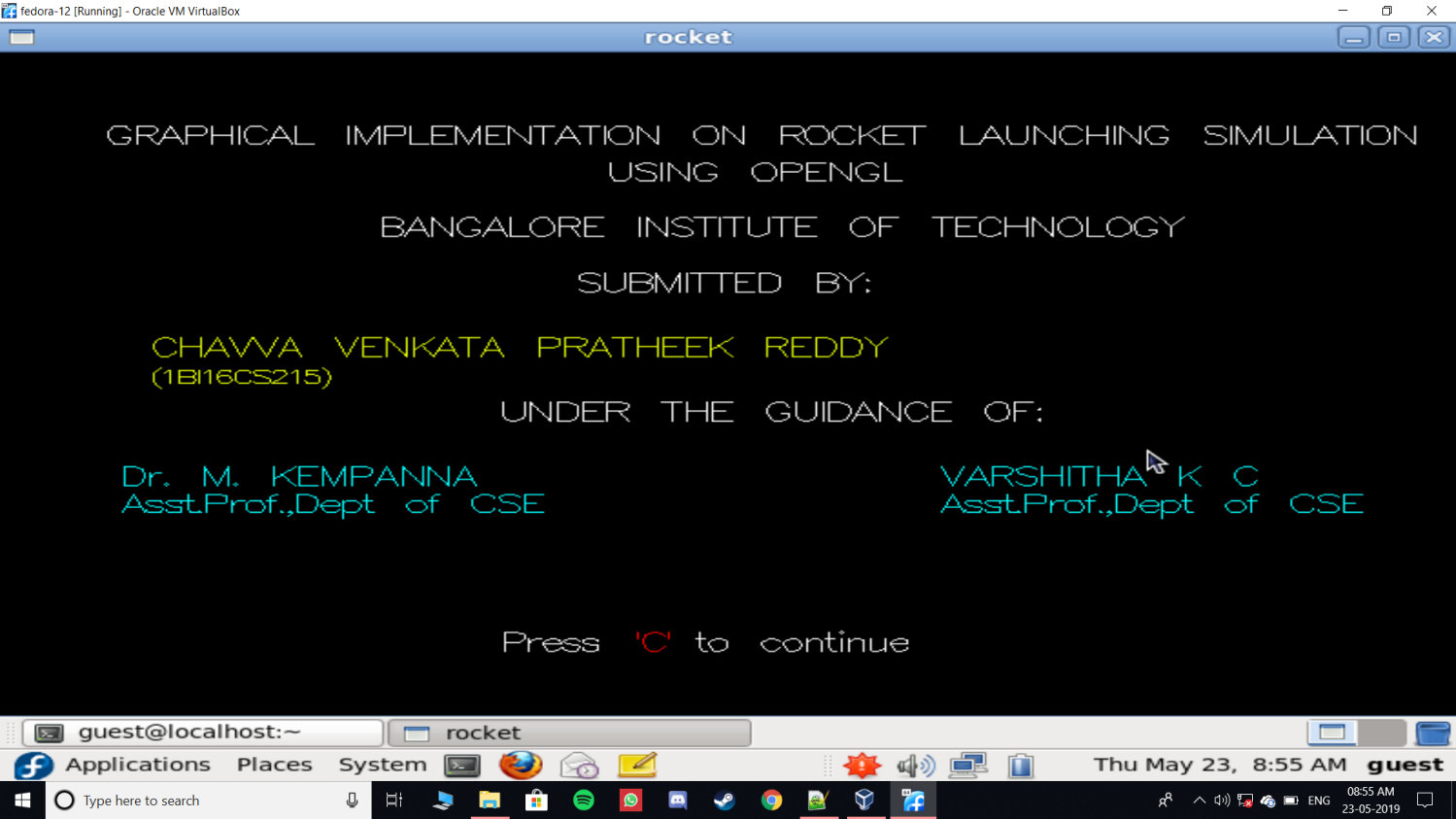
glVertex2f(40.0,160.0);

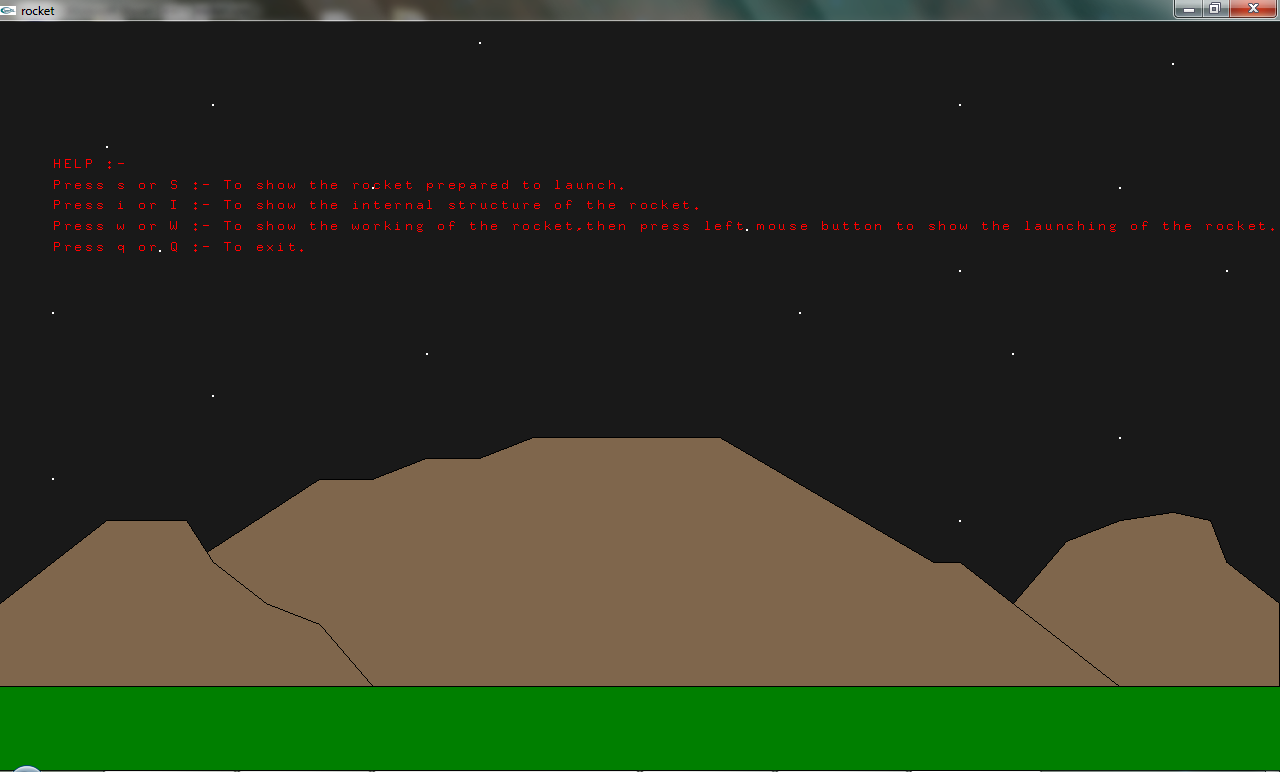
glEnd();

flag();

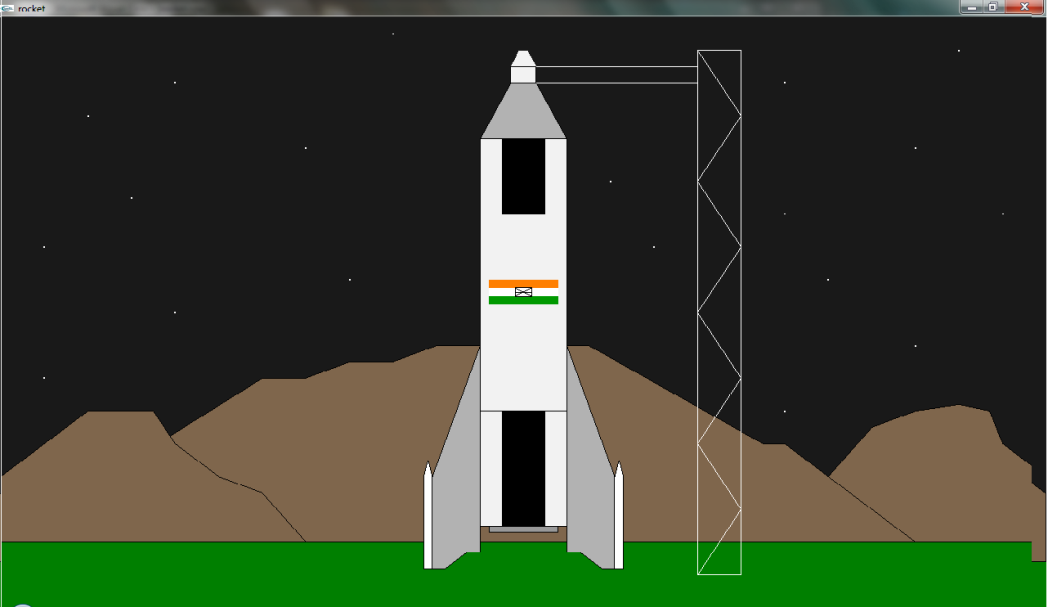
}

# Chapter -5 SCREENSHOTS

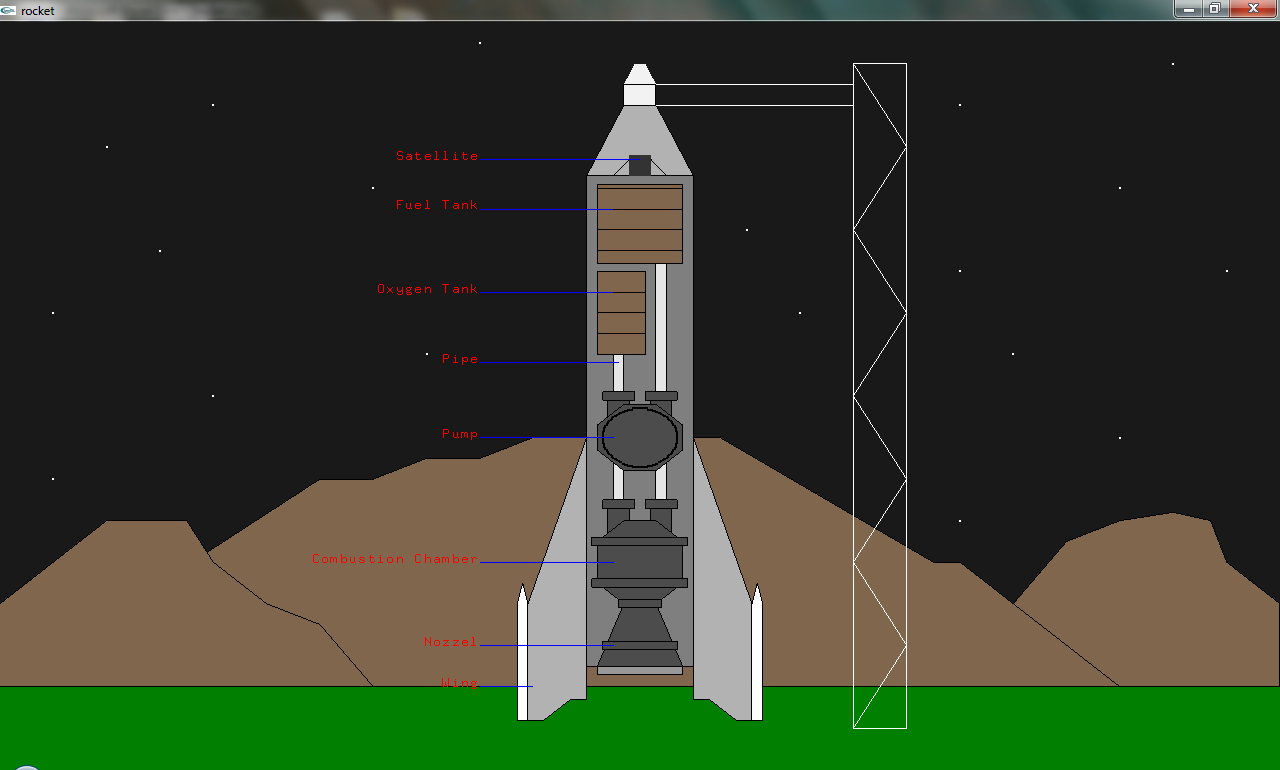
****

**Fig 5.0: Intial view of area for rocket launching along with different options**

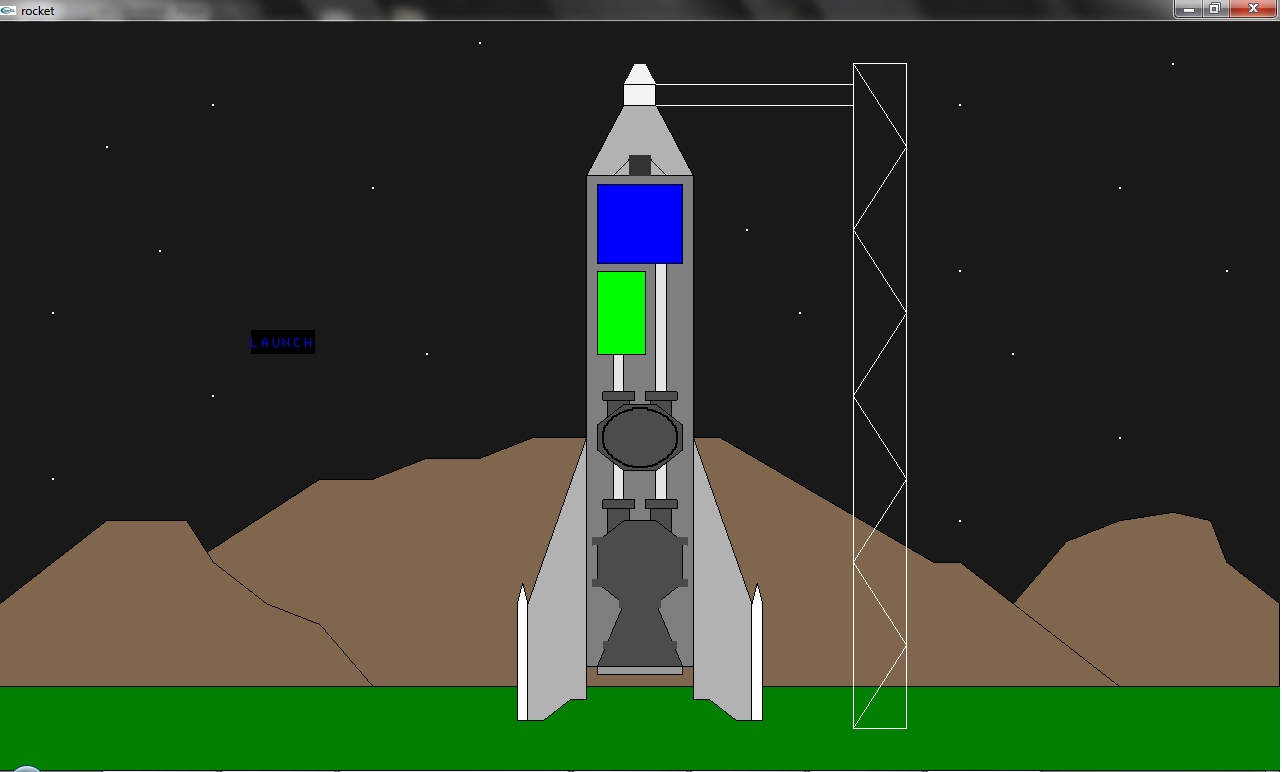
**Fig 5. 1:** Initial scene of a rocket



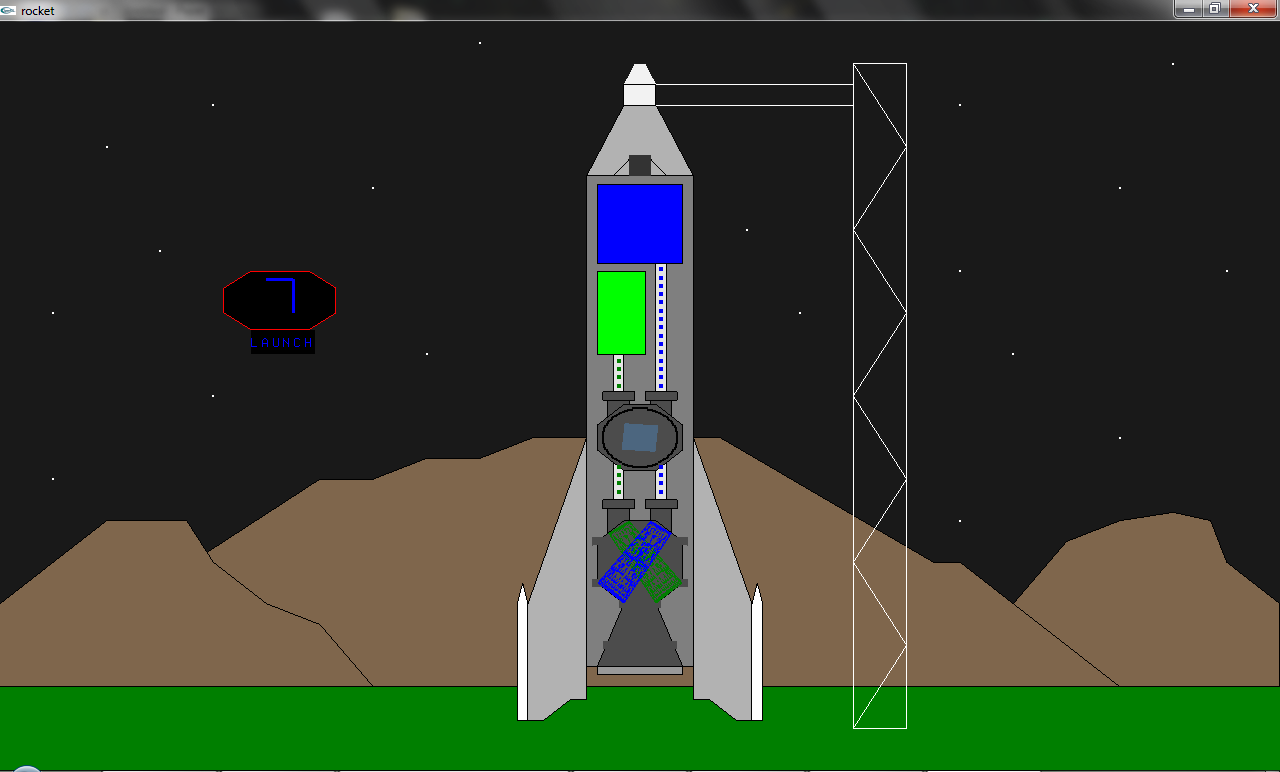
**Fig 5.2:** Internal view of a rocket

****

**Fig 5.3 :** Scene of a rocket which is ready to get launched

****

**Fig 5.4:** View of a rocket internal working at countdown number 7.



**5.5:** Final view of a launched rocket



**Chapter -6 CONCLUSION**

Thus, in this project we have acquired a lot of knowledge about various techniques in OpenGL programming. We have explored many new concepts on the World Wide Web, such as Texture, mapping, randomizing, colour swapping etc.

We thus would like to emphasize of the importance of this project that opened our eyes to many other perspectives of Technical, mathematical, graphical and software concepts which we were unaware of.

Thus, we thank the Principal, the HOD, the faculty and all our classmates for their continuous support and belief in us.

## Future Enhancement

In future the same project can be enhanced in such a way that we can interact more with the project. As of now, we can only simulate the launch of rocket in 2D model.

A vast amount of future work can be possible by following investigations and strategies.

More features can be included and can be modified in a more versatile way.

# BIBILOGRAPHY

## Reference Books

* Donald Hearn & Pauline Baker: Computer Graphics with OpenGL Version, 3rd/4th Edition, Pearson Education, 2011
* Edward Angel: Interactive Computer Graphics- A Top Down approach with OpenGL, 5th

Edition, Pearson Education

* James D Foley, Andries Van Dam, Steven K Feiner, John F Huges Computer Graphics with OpenGL, Pearson Education
* Macro Cantu: Mastering Delphi

## Websites

* [www.opengl.org](http://www.opengl.org/)
* [www.google.co.in](http://www.google.co.in/)
* [www.wikipedia.com](http://www.wikipedia.com/)
* [www.tutorialspoint.com](http://www.tutorialspoint.com/)
* [www.geeksforgeeks.com](http://www.geeksforgeeks.com/)

## VISVESVARAYA TECHNOLOGICAL UNIVERSITY

**“Jnana Sangama”, Belagavi-590018, Karnataka**



**BANGALORE INSTITUTE OF TECHNOLOGY**

**K.R. Road, V.V. Puram, Bengaluru-560 004**



**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

#### COMPUTER GRAPHICS & VISULIZATION MINI PROJECT (15CSL68)

**“ROCKET LAUNCHING SIMULATION”**

**Submitted By**

**1BI16CS215 CHAVVA VENKATA PRATHEEK REDDY**

**for the academic year 2018-19**

Under the guidance of

**Dr. M Kempanna Varshitha K C**

Associate Professor Assistant Professor

## VISVESVARAYA TECHNOLOGICAL UNIVERSITY

**“Jnana Sangama”, Belagavi-590018, Karnataka**

**BANGALORE INSTITUTE OF TECHNOLOGY**

#### K.R. Road, V.V. Puram, Bengaluru-560 004



**Department of Computer Science & Engineering**

***Certificate***

This is to certify that the implementation of **Computer Graphics and Visualization MINI PROJECT (15CSL68)** entitled **“ROCKET LAUNCHING SIMULATION”** has

been successfully completed By **CHAVVA VENKATA PRATHEEK REDDY** bearing the USN **1BI16CS086** of VI Semester B.E. for the partial fulfillment of the requirements for the Bachelor's degree in **Computer Science & Engineering** of the **Visvesvaraya Technological University** during the academic year **2018-2019**.

#### Lab In charges:

|  |  |  |
| --- | --- | --- |
| **Dr. M Kempanna**  Associate Prof. Dept. of CS&E | **Varshitha K C** Assistant Prof. Dept. of CS&E | **Dr. ASHA T.**  Professor and Head Department of CS&E  Bangalore Institute of Technology |
| Examiners: 1) |  | 2) |

**ACKNOWLEDGEMENT**

The knowledge & satisfaction that accompany the successful completion of any task would be incomplete without mention of people who made it possible, whose guidance and encouragement crowned my effort with success. I would like to thank all and acknowledge the help I have received to carry out this Mini Project.

I would like to convey my thanks to Head of Department **Dr. ASHA T.** for being kind enough to provide the necessary support to carry out the mini project.

I am most humbled to mention the enthusiastic influence provided by the lab in-charges **Prof. M KEMPANNA**  **a**nd **Prof. Dr VARSHITHA K C**

, on the project for their ideas, time to time suggestions for being a constant guide and cooperation showed during the venture and making this project a great success.

I would also take this opportunity to thank my friends and family for their constant support and help. I'm very much pleasured to express my sincere gratitude to the friendly co-operation showed by all the **staff members** of Computer Science Department, BIT.

CHAVVA VENKATA PRATHEEK REDDY

1BI16CS215

**Table of contents**

1. [Introduction 1](#_bookmark0)
   1. [Computer Graphics 1](#_bookmark1)
   2. [Application of Computer Graphics 1](#_bookmark2)
   3. [OpenGL](#_TOC_250005) 3
   4. [Objective Of The Project](#_TOC_250004) 4
   5. Organization Of The Report. 4
2. System Specification 5
   1. Software Requirements… 5
   2. Hardware Requirements 5
3. Design… 6
   1. [Flow Diagram…](#_TOC_250003) 6
4. Implementation… 7
   1. [Built In Functions…](#_TOC_250001) 7
   2. User Defined Functions With Modules… 8
5. Snapshots… 25
6. Conclusion… 29

[Future Enhancement… 29](#_TOC_250000)

Bibliography 30