VISVESVARAYA TECHNOLOGICAL UNIVERSITY
BELGAUM-590014

A Computer Graphics and Visualization

Mini-Project Report

On

"Jumping Box"

A Mini-project report submitted in partial fulfilment of the requirements for the award of the degree of Bachelor of Engineering in Computer Science and Engineering of Visvesvaraya Technological University, Belgaum.

Submitted by:

Amogh S Bharadwaj (1DT19CS013)
AND
Chandrakanth N Murthy (1DT19CS031)

Under the Guidance of:
Mrs.Apoorva Busad
Asst. Prof. Dept of CSE
Mrs.Nivetha K
Asst. Prof. Dept of CSE

DAYANANDA SAGAR ACADEMY OF TECHNOLOGY AND MANAGEMENT

Department of Computer Science and Engineering
(Accredited by NBA, NAAC A+, New Delhi)
Kanakpura Road,Udayapura, Bangalore

2021-2022

DAYANANDA SAGAR ACADEMY OF TECHNOLOGY AND MANAGEMENT

Department of Computer Science and Engineering

(Accredited by NBA, NAAC A+, New Delhi)

Kanakpura Road,Udayapura, Bangalore

CERTIFICATE

This is to certify that the Mini-Project on Computer Graphics (CG) entitled **"Jumping Box"** has been successfully carried out by **AMOGH S BHARADWAJ(1DT19CS013) and CHANDRAKANTH N MURTHY (1DT19CS031)** a bonafide students of **Dayananda Sagar Academy of Technology and Management** in partial fulfillment of the requirements for the award of degree in **Bachelor of Engineering in Computer Science and Engineering** of **Visvesvaraya Technological University, Belgaum** during academic year 2021-22. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library. The mini project report has been approved as it satisfies the academic requirements in respect of project work for the said degree.

GUIDE:

Mrs. Apoorva Busad	**Mrs. Nivetha K**
Asst. Prof. Dept. of CSE	Asst. Prof. Dept. of CSE

Dr.C.Nandini

Vice Principal & HOD, Dept. of CSE

Examiners: **Signature with Date**

1:

2:

ACKNOWLEDGEMENT

It gives us immense pleasure to present before you our project titled **"Jumping Box".** The joy and satisfaction that accompany the successful completion of any task would be incomplete without the mention of those who made it possible. We are glad to express our gratitude towards our prestigious institution **DAYANANDA SAGAR ACADEMY OF TECHNOLOGY AND MANAGEMENT** for providing us with utmost knowledge, encouragement and the maximum facilities in undertaking this project.

We wish to express a sincere thanks to our respected principal **Dr. Ravishankar M** for all their support.

We express our deepest gratitude and special thanks to **Dr.C.Nandini, Vice Principal & H.O.D, Dept.Of Computer Science Engineering**, for all her guidance and encouragement.

We sincerely acknowledge the guidance and constant encouragement of our project guides

Mrs.Apoorva Busad(Asst. Prof. Dept of CSE) and Mrs.Nivetha K (Asst. Prof. Dept of CSE)

Amogh S Bharadwaj(1DT19CS013)

Chandrakanth N Murthy (1DT19CS031)

TABLE OF CONTENTS

CHAPTER NO.	CHAPTER NAME	PAGE NO.
1.	INTRODUCTION	6
1.1	Introduction to Open GL	6
1.2	About Open GL	7
1.3	Features of Open GL	7
1.4	Open GL Architecture	8
1.5	About the Project	8
2.	REQUIREMENT SPECIFICATION	9
2.1	Hardware Requirements	9
2.2	Software Requirements	9
3.	SYSTEM DESIGN	10
3.1	Design	10
4.	IMPLEMENTATION	11
4.1	Psuedo Code	11
4.2	OpenGL Function	13
4.3	Implementation of UserDefined Functions	14
5.	SOURCE CODE	15
6.	SNAPSHOT	21
6.1	Start Screen	21
6.2	Game During Morning	22
6.3	Game During Night	22
6.4	SplashScreen	23
6.5	ScoreBoard	23
7.	REQUIREMENT SPECIFICATION	24
7.1	Conclusion	24
7.2	Future Enhancement	24
	REFRENCES	25

LIST OF FIGURES

FIGURE NO.	FIGURE TOPIC	PAGE NO.
1.1	Graphic System	6
1.2	Open GL Architecture	8
3.1	Design of Project	10
6.1	Start Page	21
6.2.1	Game Start	21
6.2.2	Box Jump	22
6.3.1	Change to Night	22
6.3.2	Box Jump at Night	22
6.4	Splash Screen	23
6.5	ScoreBoard	23

CHAPTER 1:INTRODUCTION

1.1 Introduction to Open GL

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.

In computer graphics, pictures or graphics objects are presented as a collection of discrete picture elements called **pixels**. The pixel is the smallest addressable screen element

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 a keyboard, mouse, or touch-sensitive panel on the screen.

Computer Graphics relies on an internal model of the scene, that is, mathematical representation suitable for graphical computations. The model describes the 3D shapes, layout and materials of the scene. This 3D representation then has to be projected to compute a 2D image from a given viewpoint, this is rendering step. Rendering involves projecting the objects, handling visibility (which parts of objects are hidden) and computing their appearance and lighting interactions. Finally, for animated sequence, the motion of objects has to be specified.

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 a keyboard, mouse, or touch-sensitive panel on the screen.

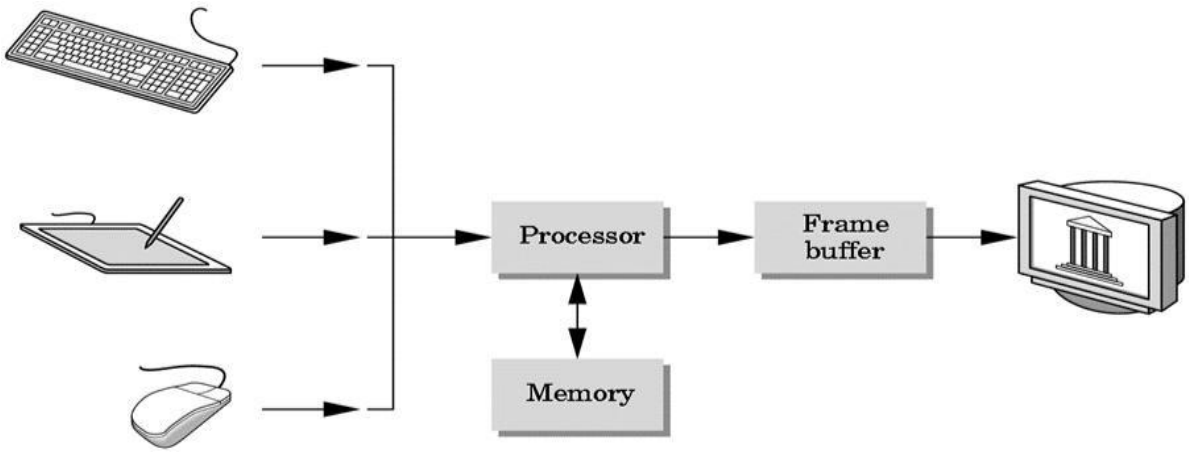

Fig 1.1: Graphic System

The image processing can be classified as

- Image enhancement.
- Pattern detection and recognition
- Scene analysis and computer vision.

The image enhancement deals with the improvement in the image quality by eliminating noise or by increasing image contrast. Pattern detection and recognition deals with the detection and clarification of standard patterns

And finding deviations from these patterns .The optical character recognition (OCR) technology is a practical example for pattern detection & recognition. Scene analysis deals with the recognition and reconstruction of 3D model of scene from several 2D images.

1.2 About OpenGL

OpenGL (**O**pen **G**raphics **L**ibrary) 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.

OpenGL's basic operation is to accept primitives such as points, lines and polygons, and convert them into pixels. This is done by a graphics pipeline known as the OpenGL state machine.

1.3 Features of OpenGL:

- Geometric Primitives allow you to construct mathematical descriptions of objects.
- Viewing and Modelling permits arranging objects in a 3-dimensional scene, move our camera around space and select the desired vantage point for viewing the scene to be rendered.
- Materials lighting OpenGL provides commands to compute the colour of any point given the properties of the material and the sources of light in the room.
- Transformations: rotation, scaling, translations, perspectives in 3D, etc.

1.4 OpenGL Architecture

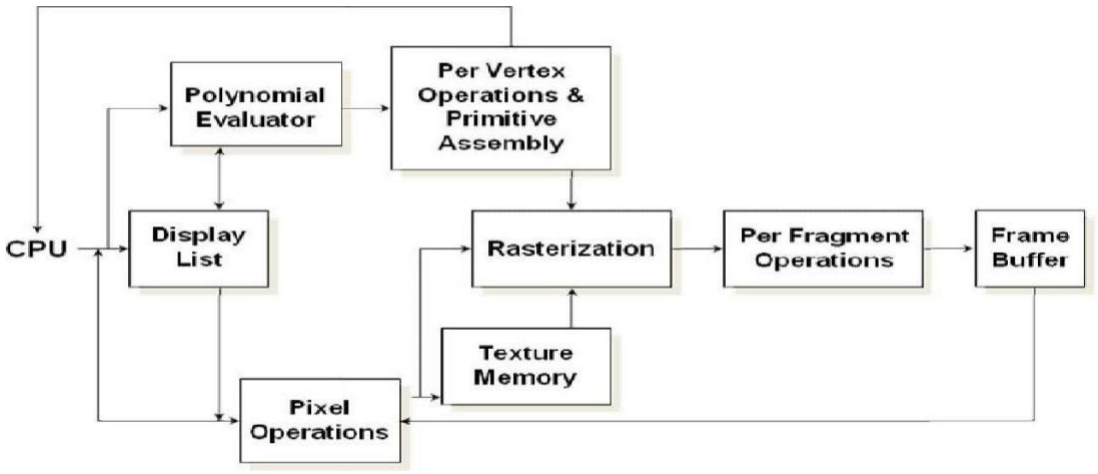

Fig 1.2: OpenGL Architecture

This is the most important diagram you will see today, representing the flow of graphical information, as it is processed d from CPU to the frame buffer.

There are two pipelines of data flow. The upper pipeline is for geometric, vertex-based primitives. The lower pipeline ne is for pixel-based, image primitives. Texturing combines the two types of primitives together.

1.5 Overview Of Project

Computer graphics involves the designing of objects in different forms which are regular and irregular in shape. The mini project named “Jumping Box Game” is a game which involves the jumping of box against the obstacle. The obstacle created is a green cactus. The jumping of the box can be controlled by the player by pressing the keys ‘w’ or ‘W’ or the spacebar. Trees have been drawn on the side of the track in which the box is jumping. The weather keeps changing from morning to night and vice versa on particular period of time which is specified by the player. Each time the box jumps the single cactus the score gets increased by two hundred points. When the box colloids with the cactus the game is over.

CHAPTER 2:REQUIREMENTS SPECIFICATION

2.1 Hardware Requirements:

- Processor - Intel® Pentium 4 CPU or higher versions
- RAM - 128 MB or more RAM
- Hard Disk – Approx 2GB
- Storage Space – Approx 2MB
- A standard 101 keyboard, and Microsoft compatible mouse
- VGA monitor

2.2 Software requirements:

- Operating System: Windows XP, Windows 2000 or Higher
- Language Tools: OpenGL
- Compiler: GNU GCC Compiler/C++ Compiler
- IDE :Code::Blocks: cross-platform Integrated Development Environment (IDE)
- GLUT libraries, Glut utility toolkit must be available.

CHAPTER 3:SYSTEM DESIGN

3.1 Design

Design of any software depends on the architecture of the machine on which that software runs, for which the designer needs to know the system architecture. Design process involves design of suitable algorithms, modules, subsystems, interfaces etc.

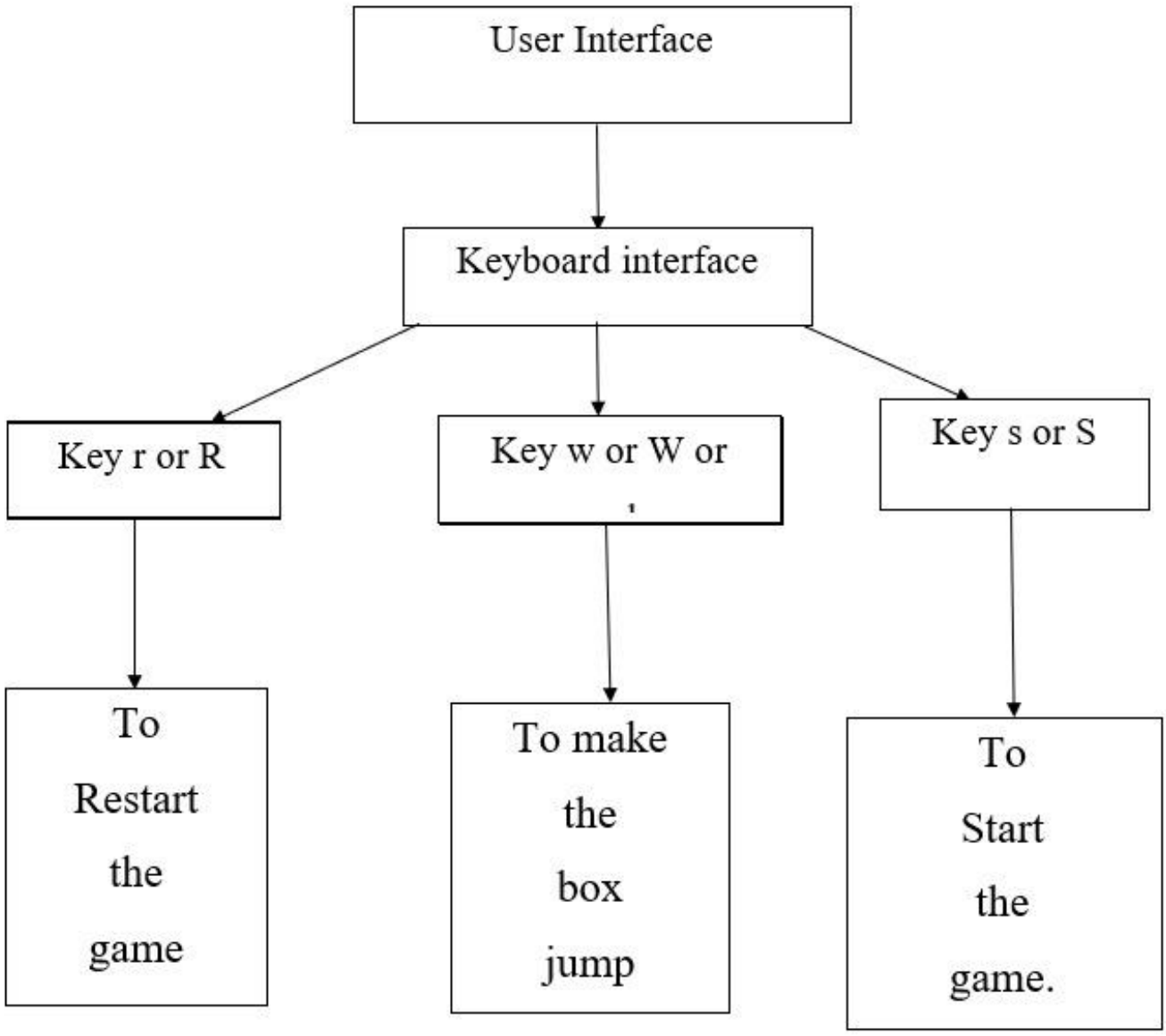

Fig. 3.1: Design Of the Project

CHAPTER 4:IMPLEMENTATION

4.1 Psuedo Code

Void init ()

{

This function is used to initialize the viewing of the output.

}

Void drawtext ()

{

This function is used to write the text wherever necessary.

}

Void drawsun ()

{

This function is used to draw the sun.

}

Void drawbox ()

{

This function is used to draw the main object which is the box.

}

Void drawsand ()

{

This function is used to draw the sand on which the box moves.

}

Void drawdesert()

{

This function is used to draw the desert.

}

Void drawkillscrnsquare ()

{

This function is used to draw the square which appears after the game is over.

}

Void drawcoverpage ()

{

This function is used to draw the cover page of the object.

}

Void drawscnfreeze()

{

This function is used to calculate the scores.

}

Void display (){

Color buffer is cleared by using glClear (),

This function displays the bit map characters. glFlush () guarantees the execution of commands.

4.2 OpenGL Functions

The various functions used in implementing this project are:

- **glutInit(&argc,argv)**

  This function is used to initialize glut.

- **glutInitDisplayMode()**

  This function is used to initialize a display mode for the screen. It can choose one of the following constant values as it's parameters:

  ▪ GLUT_SINGLE ▪ GLUT_RGB
  ▪ GLUT_DOUBLE

- **glutInitWindowPosition()**

  This function is used to set the position of the top left corner of the window. It takes in 2 integer values as input parameters which are the coordinates specified for the position for the top left corner of the window.

- **glutInitWindowSize()**

  This function is used to specify the size of the created window, inside which the scene will be generated. It takes in two integer parameters, the width and height of the window.

- **glutCreateWindow()**

  This function creates the window with the specified dimensions and position and assigns the string parameter passed to it as the name of the window.

- **glutMainLoop()**

  This function is called at most once in a GLUT program. Once called, this routine will never return. It will call as necessary, any callbacks required for the program.

- **glBegin()**

  This function is used in drawing the basic primitives like lines, points, polygons and quads. It takes in a constant value which specifies the primitive to be drawn. Some of the parameters it can have are-

▪ GL_POLYGON

▪ GL_LINES CG Mini-project – Archery Game ▪ GL_LINE_LOOP

▪ GL_QUADS

▪ GL_QUAD_STRIP

- **glClear()**

  This function is used to clear the contents of the buffer passed as the parameters to this function, onto the screen. It takes a constant value as a parameter like:

  ▪ GL_CLEAR_BUFFER_BIT ▪ GL_DEPTH_BUFFER_BIT

- **glClearColor()**

  This function takes in four floating values:

  ▪ Red▪ Green ▪ Blue ▪ Alpha

  Based on RGB values, the colour is decided. The Alpha value gives the degree of transparency of the colour. The values for RGBA will range from 0.0 to 1.0.

- **glEnd()**

  This function works with the glBegin() function. It marks the end of all the vertices to be used for drawing the primitive.

4.3 Implementation of User Defined Functions:

- **void drawText(char ch[],int xpos, int ypos,int z=0)**

  Used to Print the Text for the score and Game over screen once the screen was in screen 3

- **void drawTextNum(char ch[],int numtext,int xpos, int ypos)**

  This method was used in order to display the number of scores high score dodged blocks etc

- **void drawPath()**

  This Method was used to draw the obstacles in the initial phase of the game in screen2

- **void drawCircle(int tileX, int tileY, int radius)**

  This method is used to draw circles for out flying object

- **void drawUFO()**

  Here the drawcircle method is called one by one in order to make to UFO design

- **void drawObject()**

  This method used to Determine the object colour and its spawning position on the screen

- **void Specialkey(int key, int x, int y)**

  This method allows us to navigate the ufo by either pressing right or left arrows on the keyboard

- **void reinit()**

  This method is used in order to reinitialize all the variables after the game ends

- **void restart()**

  This method constitutes the screen 3 or the final screen of the game it displays the score and dodged blocks with other relevant information

- **void Normalkey(unsigned char key, int x, int y)**

  This is a keyboard input function in order to select the difficulty of the game i.e. B/E/I

- **void init()**

  Here the initialization function init() is used to set the OpenGL state variables dealing with viewing and attributes-parameters that is prefered to set once, independently of the display function.

- **void display()**

  Here the screen to display is determined either of the screen zero, one ,two or three is selected in order to display according to the input or actions by the user

- **int main(int argc, char **argv)**

  In This function we have certain “gl” functions used to initialize the basic glut window with desired properties. After initialization the display function is called with respect to the window and finally the last statement of This section is the glutMainLoop(). Which is an event processing function that enters into an infinite loop.

CHAPTER 5:SOURCE CODE

```
#include<stdio.h>
#include<stdlib.h>
#include<GL/glut.h>
#include<string.h>
#define collisioncheck 1
int texchangeid,terchangeid,speedchangeid,sunshowid,sunsizeid,sunspinid,settimeid,daylenid;
int COLLISIONCHECKTIME=1;
int JUMPSPEED=8;
int MAX_HEIGHT=150;
int NUM_CACT=25;
int DASH_CACT_SAND=3;
int SCREEN_CENTERX=1000;
int SCREEN_CENTERY=250;
int ANGULAR_SPEED=5;
int ZOOMING=1;
float SUN_SCALE=0.5;
int SUN_EDGE=10;
int SUN_X=1900;
int SUN_Y=400;
int SUN_ASPEED=1;
int SPINSUN=1;
int NUM_SAND_GRANULE=40;
int NUM_STAR=63;
int NUM_SKY_COLOR=1000;
int day=1;
int skyindex=1000;
int DAYCYCLE=10;
float STAR_SHOW=0.4;
CoordinateGLdouble dinox[]={
```

```
20.0,40.0,40.0,20.0,
70.0,90.0,90.0,70.0,
10.0,100.0,100.0,10.0,
0.0,00.0,0.0,0.0,
0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0,
0.0,0.0,0.0,0.0
};
GLdouble dinoy[]={
200.0,200.0,210.0,210.0,200.0,200.0,210.0,
210.0,210.0,210.0,250.0,250.0,250.0,250.0,
00.0,00.0,0.0,0.0,0.0,0.0,0.0,0.0,
0.0,0.0,00.0,0.0,0.0,0.0,0.0,0.0
};
GLfloat dinomorfer[][3]={{1.0,1.0,1.0},{0.491176,0.275098,0.1058824},{.0,0.0,1.0}};
int boxcolor=2;
GLfloat desertx[]={0.0,1999.0,1999.0,0.0};
GLfloat deserty[]={0.0,0.0,200.0,200.0};
GLfloat landmorfer[]={
0.639216,0.431373,0.250980,0.121,0.267,0.251,
0.168627,0.219608,0.137255,-0.035294,+0.325490,-0.003922
};
int landcolor=0;
GLfloat cactusx[]={
320.0,340.0,340.0,337.5,335.0,325.0,322.5,320.0,
340.0,350.0,360.0,340.0,
350.0,360.0,360.0,358.75,357.5,352.5,351.25,350.0,
300.0,310.0,320.0,320.0,
300.0,310.0,310.0,308.75,307.5,302.5,301.25,300.0
};
GLfloat cactusy[]={
185.0,185.0,280.0,283.5,285.0,285.0,283.5,280.0,
230.0,230.0,240.0,240.0,
```

```
240.0,240.0,280.0,281.66,282.5,282.5,281.66,280.0,
240.0,230.0,230.0,240.0,
240.0,240.0,280.0,281.66,282.5,282.5,281.66,280.0
};
GLfloatcactposmsg[]={2000.0,2830.0,3600.0,4450.0,
5320.0,6210.0,6980.0,7790.0,8630.0,
9500.0,10300.0,11180.0,12070.0,
12880.0,13750.0,14580.0,15430.0,
16200.0,16990.0,17880.0,18680.0,
19550.0,20450.0,21200.0,22000.0};
GLfloatcactpos[]={2000.0,2830.0,3600.0,4450.0,
5320.0,6210.0,6980.0,7790.0,
8630.0,9500.0,10300.0,11180.0,
12070.0,12880.0,13750.0,14580.0,
15430.0,16200.0,16990.0,17880.0,
18680.0,19550.0,20450.0,21200.0,22000.0};
GLfloatsandx[]={1620.000000,267.000000,45.000000,387.000000,
1236.000000,52.000000,468.000000,179.000000,
998.000000,61.000000,1935.000000,920.000000,
1305.000000,1253.000000,1584.000000,1142.000000,
1459.000000,1328.000000,801.000000,1912.000000,
1198.000000,1302.000000,1977.000000,557.000000,
1562.000000,1697.000000,257.000000,427.000000,
1832.000000,1784.000000,1305.000000,1453.000000,
125.000000,1423.000000,1913.000000,1362.000000,
1548.000000,382.000000,1614.000000,620.000000};
GLfloatsandy[]={116.000000,9.000000,45.000000,166.000000,
92.000000,60.000000,26.000000,19.000000,
143.000000,120.000000,89.000000,54.000000,
171.000000,59.000000,150.000000,73.000000,
149.000000,60.000000,107.000000,92.000000,
134.000000,73.000000,83.000000,126.000000,
27.000000,176.000000,84.000000,181.000000,
```

```
32.000000,16.000000,143.000000,100.000000,
177.000000,188.000000,18.000000,69.000000,
0.000000,44.000000,41.000000,144.000000};
GLdoublesunxmsg[]={2000.000000,1980.901733,1930.901733,1869.098267,
1819.098267,1800.000000,1819.098267,1869.098267,
1930.901611,1980.901611};
GLdoublesunymsg[]={400.000000,458.778534,495.105652,495.105652,458.778534,
400.000000,341.221466,304.894348,304.894348,341.221436};
GLdoublesunx[]={2000.000000,1980.901733,1930.901733,1869.098267,1819.098267,1800.000000,181
9.098267,1869.098267,1930.901611,1980.901611};
GLdoublesuny[]={400.000000,458.778534,495.105652,495.105652,458.778534,400.000000,341.22146
6,304.894348,304.894348,341.221436};
int SHOWSUN=1;
int showsunmenu=1;
GLdouble sunangle=0;
GLfloatstarx[]={85.0,44.0,110.0,1482.0,1579.0,146.0,716.0,125.0,793.0,801.0,1524.0,1706.0,1432.0,19
92.0,700.0,1038.0,1889.0,858.0,1845.0,666.0,748.0,1403.0,
236.0,208.0,1904.0,1507.0,1589.0,1319.0,760.0,1475.0,915.0,265.0,1324.0,1921.0,1647.0,903.0,1967.0,
363.0,928.0,1613.0,1917.0,1304.0,271.0,301.0,249.0,871.0,1240.0,138.0,582.0,1937.0,1556.0,182.0,292
.0,645.0,290.0,197.0,1004.0,731.0,1416.0,1664.0,207.0,1183.0,1829.0};
GLfloatstary[]={469.0,458.0,300.0,383.0,205.0,228.0,286.0,224.0,444.0,467.0,237.0,361.0,223.0,461.0,
262.0,472.0,284.0,454.0,263.0,466.0,343.0,371.0,248.0,436.0,468.0,390.0,333.0,373.0,422.0,401.0,489.
0,329.0,384.0,419.0,212.0,389.0,499.0,350.0,466.0,443.0,369.0,255.0,356.0,393.0,216.0,471.0,417.0,30
1.0,477.0,232.0,319.0,321.0,404.0,368.0,309.0,372.0,310.0,494.0,245.0,232.0,395.0,235.0,413.0};
GLfloat killscrnxmsg[]={-3,3,3,-3};
GLfloat killscrnymsg[]={-3,-3,3,3};
GLfloat killscrnx[]={-3,3,3,-3};
GLfloat killscrny[]={-3,-3,3,3};
GLfloat killscrnangle=0;
GLint killscrnover=0;
GLfloat height=0.0;
GLint jflag=0;
GLint jcount=0;
```

```
int gameover=0;
int score=0;
char textb[]={"Computer Graphics Mini Project By Amogh S Bharadwaj & Chandrakanth N Murthy"};
char textc[]={"Department Of Computers Science and Engg, DSATM                    "};
char textd[]={"Jumping Box                                                    "};
char texte[]={"       |---------------------- INSTRUCTIONS ----------------------|"};
char textf[]={"1).Press 'W' or <Spacebar Key> to make Box jump"};
char textg[]={"2).Press 'R' to Restart the game"};
char texth[]={"3).Press 'S' to start the game"};
char textj[]={"Press <ESC Key> to quit the game"};
GLint gameStart=0;
GLuint displayID1,displayID2;
void drawText(GLint x,GLint y,char* str)
{
 int i;
glRasterPos2f(x,y);
for(i=0;i<strlen(str);i++)
glutBitmapCharacter(GLUT_BITMAP_9_BY_15,str[i]);
}
void drawSun()
{
 int i;
glBegin(GL_POLYGON);
glColor3f(246,190.0,0.0);
for(i=0;i<SUN_EDGE;i++)
glVertex2f(sunx[i]-SUN_X,suny[i]-SUN_Y);
glEnd();
}
```

CHAPTER 6:SNAPSHOT

A Snapshot is the state of the system at a particular point in time. It can refer to the actual copy of a state of a system or to a capability provide by systems.

6.1 Start Screen

Fig 6.1 Start Page which display the project authors and instructions

6.2 Game During Morning

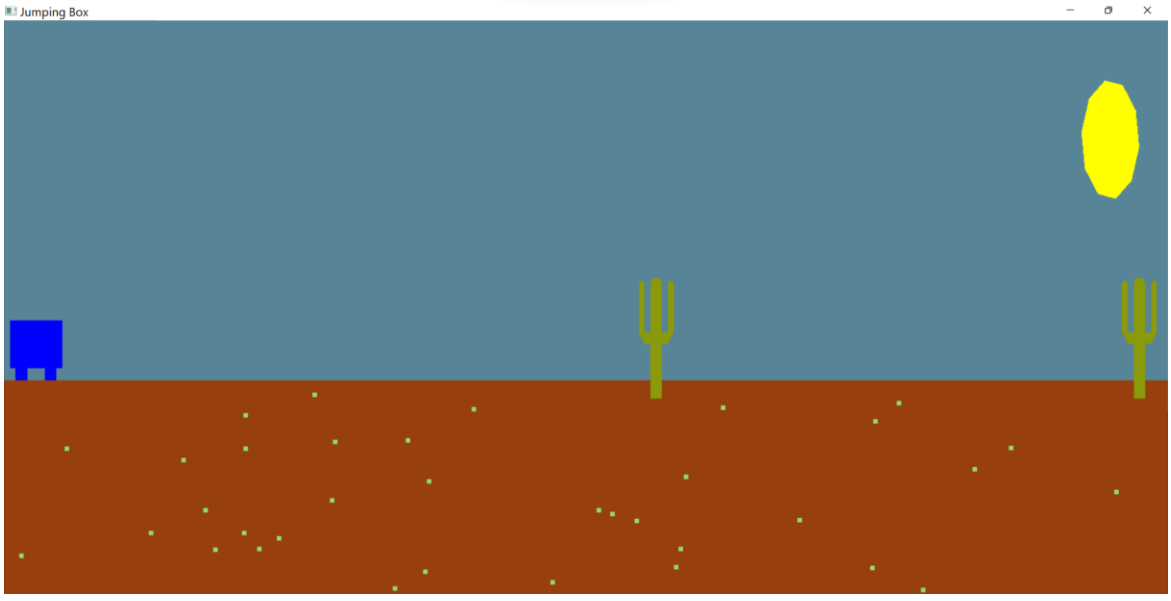

Fig 6.2.1 Game Starts once S is pressed

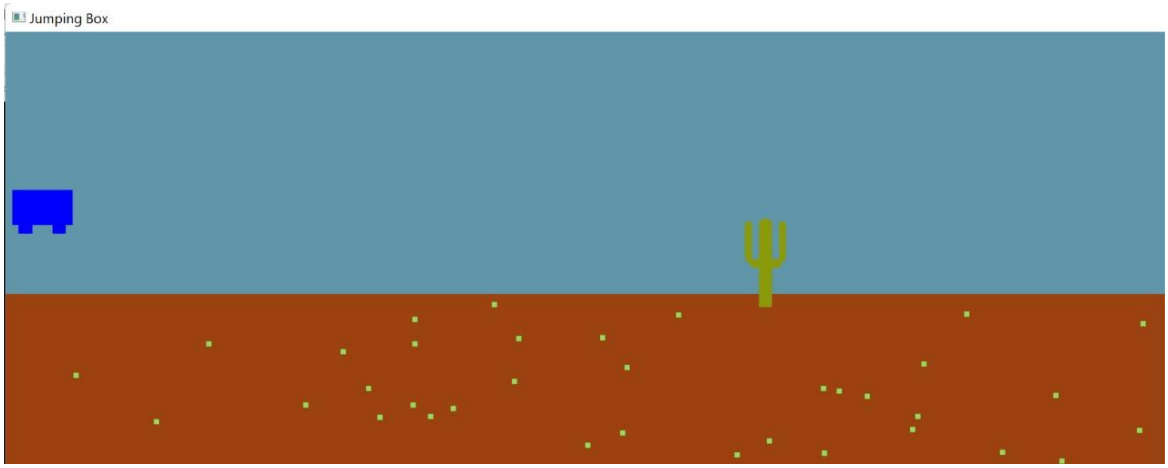

Fig 6.2.2 Box Jumps when Spacebar is pressed

6.3 Game During Night

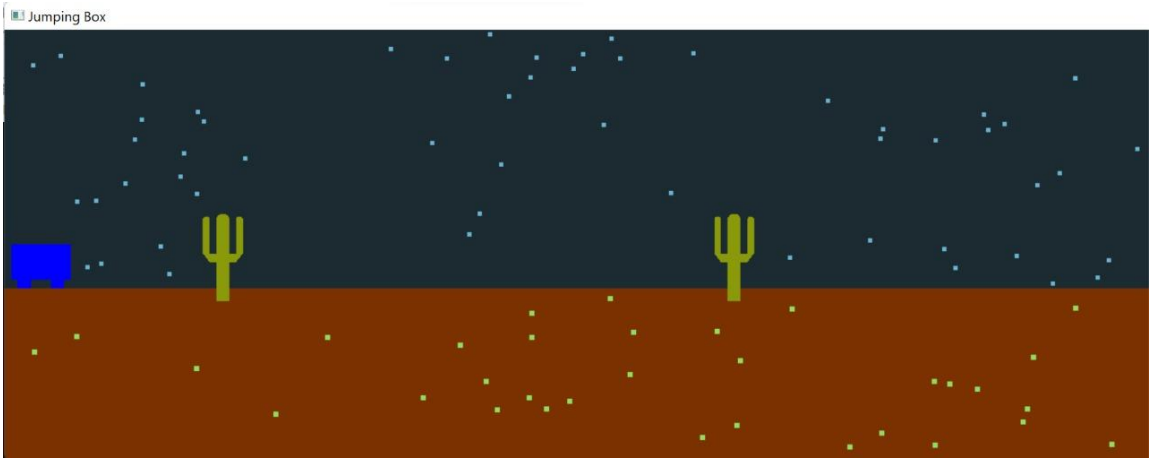

Fig 6.3.1 Background turns into night as the game advances

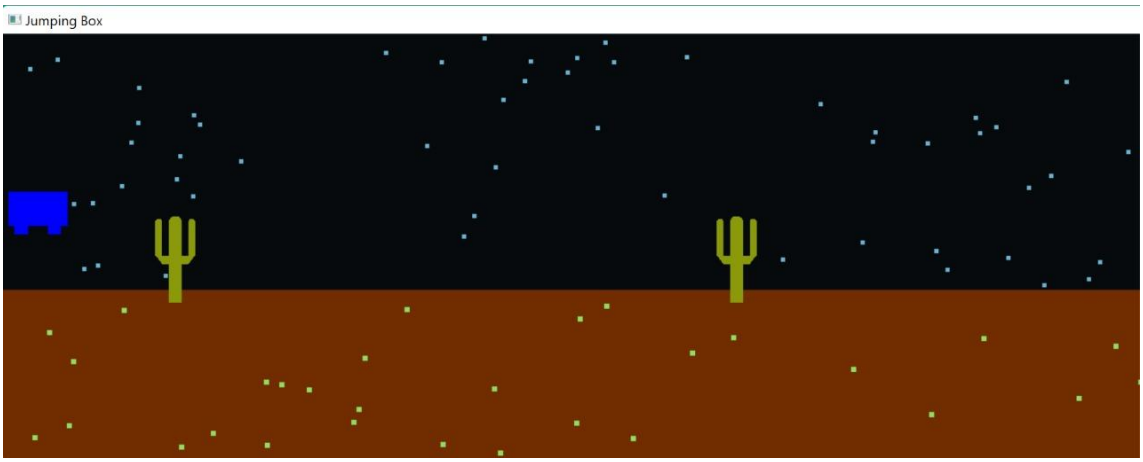

Fig 6.3.2 Game gets over once the Box touches the Cactus

6.4 Splash Screen

Fig 6.4 Splash Screen is displayed once the box hits the cactus

6.5 ScoreBoard

Fig 6.5 Game Over with Score are displayed

CHAPTER 6: CONCLUSION AND FUTURE ENHANCEMENT

6.1 CONCLUSION

The Jumping Box Graphics package has been developed Using OpenGL. The main idea of the program is entertainment. The game is an infinite runner game which can be played by anyone.

6.2 FUTURE ENHANCEMENTS

The project has graphical usage where in different functions which can be applicable in graphical packages are used. The applications developed so far is in its basic working stage. There can be more enhancements like

- Adding the different themes for background.
- Adding more obstacles such as pebbles, birds etc
- Adding more lives for the box.
- We can also add the levels of difficulty.

REFERENCES

Books:

1. Computer Graphics with OpenGL Version,3rd Edition, Donald Hearn & Pauline Baker
2. Interactive Computer Graphics-A Top Down approach with OpenGL,5rd Edition,Edward Angel
3. Computer Graphics Using OpenGL, 3rd Edition, F.S Hill Jr

Websites:

http://en.wikipedia.org/wiki/Computer_graphics

http://stackoverflow.com/questionsquickly

http://www.opengl-tutorial.org/intermediate-tutorials/

http://www.opengl-tutorial.org/

https://open.gl/

http://www.cs.uccs.edu/~ssemwal/indexGLTutorial.html

http://www.videotutorialsrock.com/

http://ogldev.atspace.co.uk/

https://www.opengl.org/sdk/docs/tutorials/

http://learnopengl.com/

http://lazyfoo.net/tutorials/OpenGL/