**CHAPTER 1**

# INTRODUCTION

## **1.1 The OpenGL Interface**

**Computer graphics** are graphics created using computers and more generally, the representation and manipulation of image data by a computer. The term computer graphics has been used in a broad sense to describe "almost everything on computers that is not text or sound".

The development of computer graphics has made computers easier to interact with, 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 animation, movies and the video game industry.

## **1.2 Applications of computer graphics**

**The various applications of computer graphics are**

* Graphs and charts
* Computer-Aided design
* Virtual-Reality environment
* Data Visualization
* Education and Training
* Computer Art
* Entertainment
* Image Processing
* Graphical User interfaces

**Graphs and Charts:**

An early application for computer graphics is the display of simple data graphs, usually plotted on a character printer. Data plotting is still one of the most common graphics applications, but today one can easily generate graphs showing highly complex data relationships for printed reports or for presentations using 35 mm slides, transparencies, or animated videos. Graphs and charts are commonly used to summarize financial, statistical, mathematical, scientific, engineering, and economic data for research reports, managerial summaries, consumer information bulletins, and other types of publications.

**Computer Aided Design**:

A major use of computer graphics is in design processes—particularly for engineering and architectural systems, although most products are now computer designed. Generally referred to as CAD, computer-aided design, or CADD, computer-aided drafting and design, these methods are now routinely used in the design of buildings, automobiles, aircraft, watercraft, spacecraft, computers, textiles, home appliances, and a multitude of other products. The manufacturing process is also tied in to the computer description of designed objects so that the fabrication of a product can be automated, using methods that are referred to as CAM, computer-aided manufacturing.

**Virtual Reality Environment**:

It is a recent application of computer graphics which is used to create virtual-reality environments in which a user can interact with the objects in a three-dimensional scene. Specialized hardware devices provide three-dimensional viewing effects and allow the user to “pick up” objects in a scene. Animations in virtual-reality environments are often used to train heavy equipment operators or to analyze the effectiveness of various cabin configurations and control placements. This allows the designer to explore various positions of the bucket or backhoe that might obstruct the operator’s view, which can then be taken into account in the overall tractor design.

**Data Visualization**:

Producing graphical representations for scientific, engineering, and medical data sets and processes is another fairly new application of computer graphics, which is generally referred to as scientific visualization. The term business visualization is used in connection with data sets related to commerce, industry, and other nonscientific areas. Numerical computer simulations, for example, frequently produce data files containing thousands and even millions of values. Similarly, satellite cameras and other recording sources are amassing large data files faster than they can be interpreted. Other visualization techniques include contour plots, renderings for constant-value surfaces or other spatial regions, and specially designed shapes that are used to represent different data types.

**Education and Training**:

Computer-generated models of physical, financial, political, social, economic, and other systems are often used as educational aids. Models of physical processes, physiological functions, population trends, or equipment, such as the color-coded diagram in for some training applications, special hardware systems are designed. Examples of such specialized systems are the simulators for practice sessions or training of ship captains, aircraft pilots, heavy-equipment operators, and air traffic-control personnel. Some simulators have no video screens; a flight simulator with only a control panel for instrument flying. But most simulators provide screens for visual displays of the external environment with multiple panels is mounted in front of the simulator.

**Entertainment:**

Television productions, motion pictures, and music videos routinely use computer-graphics methods. Sometimes graphics images are combined with live actors and scenes, and sometimes the films are completely generated using computer-rendering and animation techniques. Many TV series regularly employ computer-graphics methods to produce special effects, such as the scene in Figure from the television series Deep Space Nine. Some television programs also use animation techniques to combine computer-generated figures of people, animals, or cartoon characters with the live actors in a scene or to transform an actor’s face into another shape. And many programs employ computer graphics to generate buildings, terrain features, or other backgrounds for a scene.

**Computer Art**:

Both fine art and commercial art make use of computer-graphics methods. Artists now have available a variety of computer methods and tools, including specialized hardware, commercial software packages (such as Lumena), symbolic mathematics programs (such as Mathematical), CAD packages, desktop publishing software, and animation systems that provide facilities for designing object shapes and specifying object motions. Example: use of a paintbrush program that allows an artist to “paint” pictures on the screen of a video monitor. A paintbrush system, with a Wacom cordless, pressure-sensitive stylus, was used to produce the electronic painting. The stylus translates changing hand pressure into variable line widths, brush sizes, and color gradations.

**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, analyze images, or recognize visual patterns for robotics applications. However, image-processing methods are often used in computer graphics, and computer-graphics methods are frequently applied in image processing. Typically, a photograph or other picture is digitized into an image file before image-processing methods are employed. Then digital methods can be used to rearrange picture parts, to enhance color separations, or to improve the quality of shading OpenGL (Open Graphics Library) 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 [1].

**OpenGL**

OpenGL has become a widely accepted standard for developing graphics application. Most of our applications will be designed to access OpenGL directly through functions in three libraries. Functions in main GL library 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 begin with the letters glu.

The third is called the OpenGL Utility Toolkit (GLUT), which provides the minimum functionality that should be expected in any modern windowing system.

GLU

OpenGL

Application

Program

Frame

Buffer

GL

Xlib, Xtk

GLUT

GLX

**Fig 1.1: Library organization of OpenGL**

## **1.3 Overview of the project**

The project entitled **3D Bouncing Ball**, incorporates as many features of a simple graphics application as possible and adding a few of our own functions using the skills that have learnt in class. Here, a 3d spherical ball and various other objects are drawn using the knowledge of mathematics and OpenGL library. We can also make changes in the speed of bouncing of the ball and the angle of the camera around the surface in a 3d view.

**1.3.1 Various Concepts Used in the Application**

**glBegin:**

glBegin() and glEnd() delimit the vertices that define a primitive or a group of like primitives. glBegin accepts a single argument that specifies which of the ten ways the vertices are interpreted. Taking n as an integer count starting at one, and N as the total number of vertices specified, the interpretations are as follows:

* **GL\_POLYGON:** Draws a single, convex polygon. Vertices 1 through N define this polygon.

**Menus:**

OpenGL supports simple cascading pop-up menus. In this application simple pop-up menus along with the submenu are used to provide a simple user interface. The menu will pop up by clicking the right mouse button.

**Bitmap Character:**

OpenGL supports two kinds of texts: bitmap character and stroke character. Bitmap characters are basically 2D font without thickness. In this application bitmap character is used to display the texts.

**1.3.2 Main Features of the project**

* Visualized 3d bouncing ball on a plane surface.
* Menu to toggle the ball speed between slow, medium and fast.
* Menu to toggle between start and stop of bouncing the ball.
* Menu to toggle between different lighting options namely red, green and blue.
* Menu to toggle between start and stop of the camera spin.
* An option to change the angle of camera with the help of keyboard.
* Menu to toggle between the various different options stated above.

## **1.4 Aim of the project**

* The main objective behind this project is to design and visualize a 3d bouncing ball which serves the purpose to facilitate the bounce of a 3d ball.
* It can visualize a 3d ball which is spherical in nature.
* It helps to visualize the toggling of ball speed between slow, medium and fast.
* It helps to visualize the toggling of lighting between red, green and blue.
* The main purpose of this project is to graphically visualize the different nature of a 3d ball on a plane surface.

**CHAPTER 2**

# REQUIREMENT SPECIFICATION

A software requirement definition is an abstract description of the services which the system should provide, and the constraints under which the system must operate. It should only specify the external behavior of the system.

## **2.1 Functional Requirements**

Functional Requirements defines the internal working of the software, i.e., the calculations, technical details, data manipulation and processing and other specific functionality that show how the cases are to be satisfied and how they are supported by non-functional requirements, which impose constraints on the design or the implementation.

The following are the Functional requirements:

* The ability to perform correct operation when the corresponding keys are pressed.
* The ability to display the menu when the right mouse button is clicked.
* When the corresponding menu is selected, the corresponding option should be performed.
* Should be able to toggle between speed of bouncing of the ball.
* Should be able to toggle between different lighting options.
* Ability to switch between the camera angles.
* Menu to toggle between the various different options stated above.

## **2.2 Non-Functional Requirements**

Nonfunctional requirements are requirements which specify criteria that can be used to judge the operation of the system, rather than specific behaviors. This should be contrasted with functional requirements that specify specific behavior or functions. Typical nonfunctional requirements are reliability and scalability. Nonfunctional requirements are “constraints”, “quality attributes” and “quality of service requirements”.

**The following are the Non-Functional Requirements:**

* The application should provide a simple interface.

## **2.3 Software and Hardware Used**

**Software Used:**

* 1. OPERATING SYSTEM : Windows XP or Higher version
  2. FRONT END : Microsoft Visual Studio 2010.Net
  3. CODING LANGUAGE : C with OpenGL

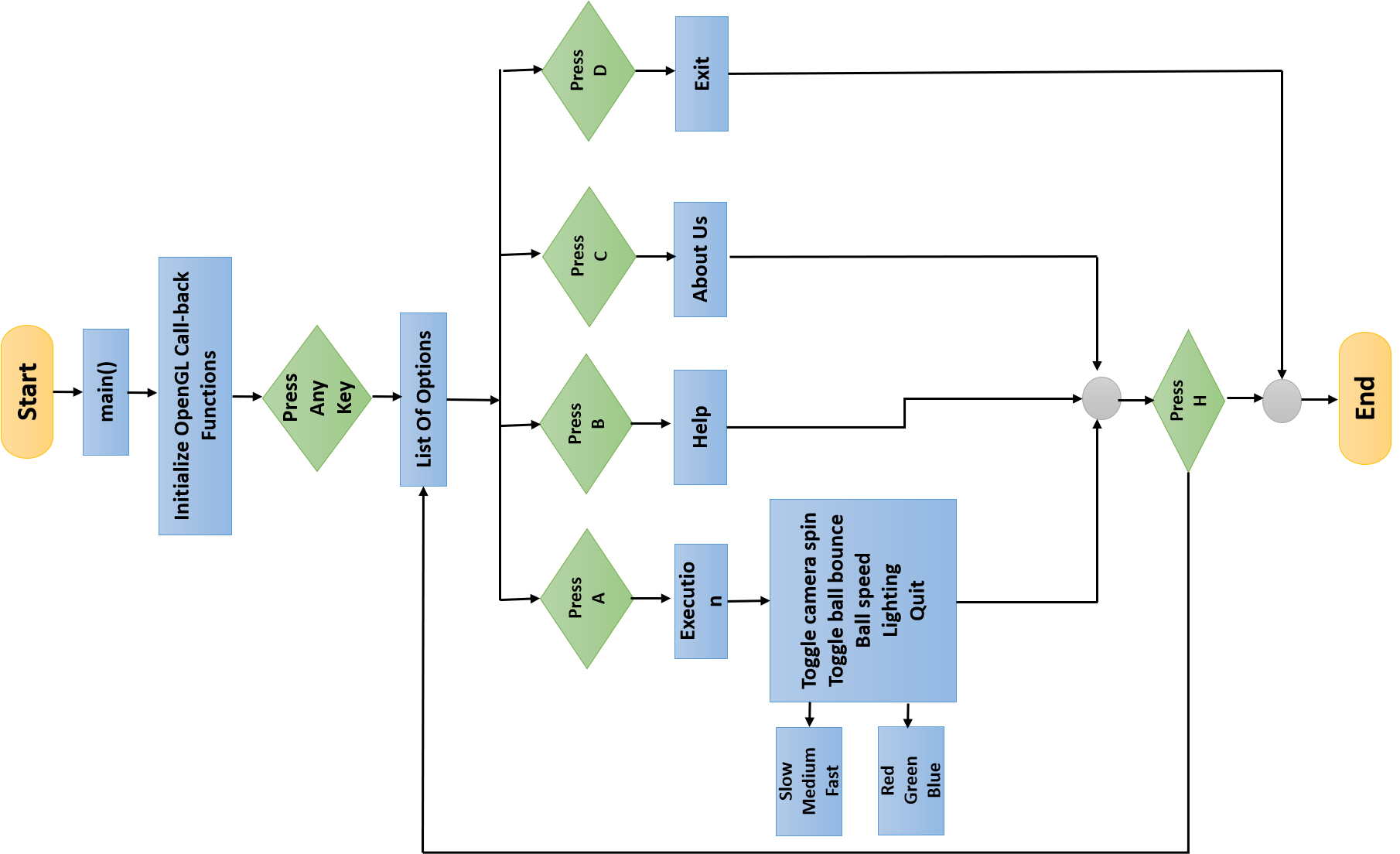
**Hardware Used:**

1. SYSTEM : Pentium IV 2.4 GHz
2. HARD DISK : 20 GB
3. MONITOR : 15 VGA color
4. RAM : 256 MB

**CHAPTER 3**

# DESIGN

Data flow design is as shown below - covering the flow of the data in the system. It describes the relation between user input and the system behavior.



**Fig 3.1: Flow chart for 3D Bouncing Ball.**

**CHAPTER 4**

# IMPLEMENTATION

To implement the Current system, we have used different functions of our project which are as follows:

**4.1 USER DEFINED FUNCTIONS:**

* **MakeGeometry ( ):** This functions generates various objects, 3d ball, platform, text on platform . It will create these objects by the given vertex position. The color of these objects are also provided here.
* **MakeLighting ( ):** This function is built using GL\_LIGHT\_MODEL\_AMBIENT, which is used to give the lightning of the window, where we used ambred, ambgreen and ambblue.
* **MakeCamera ( ):** This function is used to give projection, model view and also to spin the camera.
* **Toggleballkeys ( ):** This function is used to change the camera direction as well as turn camera angle up and down from the four key arrows of keyboard.
* **HandleMainMenu ( ):** This function takes input from menu of output window and is used to toggle camera spin and bounce of ball.
* **HandleSpeedMenu ( ):** This function takes input from menu of output window and is used to change the speed of bouncing ball.
* **HandleLightMenu ( ):** This function takes input from menu of output window and is used to change the lightening.
* **DrawTextXY ( ):** This function is used to draw the text below the platform.
* **Display ( ):** This function is used to coordinate the above functions and display the output.

**4.2 BUILT IN FUNCTIONS:**

* **glTranslate ( )** : Alters the current matrix by a displacement of (x, y, z). TYPE is either GLfloat or GLdouble.
* **glutCreateMenu ( )** : Returns an identifier for a top level menu and register the callback function f that returns an integer value corresponding to the menu entry selected.
* **glutAddMenuEntry ( ) :** glutAddMenuEntry adds a menu entry to the bottom of the current menu. The string name will be displayed for the newly added menu entry. If the menu entry is selected by the user, the menu's callback will be called passing value as the callback's parameter.
* **glLoadIdentity ( )** : replaces the current matrix with the identity matrix. It is semantically equivalent to calling glLoadMatrix with the 4X4 identity matrix.
* **glClearColor ( )** : specifies the red, green, blue, and alpha values used by glClear to clear the color buffers. Values specified by glClearColor are clamped to the range [0 1].
* **glMatrixMode ( ) :** sets the current matrix mode. mode can assume one of four values:

GL\_MODELVIEW -Applies subsequent matrix operations to the modelview matrix stack.

GL\_PROJECTION -Applies subsequent matrix operations to the projection matrix stack.

* **glutKeyboardFunc ( ) :** This function sets the keyboard callback for the current window. When a user types into the window, each key press generating an ASCII character will generate a keyboard callback. The key callback parameter is the generated ASCII character.
* **glutMouseFunc ( ) :** This function sets the mouse callback for the current window*.* When a user presses and releases mouse buttons in the window, each press and each release generates a mouse callback. The button parameter is one of GLUT\_LEFT\_BUTTON, GLUT\_MIDDLE\_BUTTON, or GLUT\_RIGHT\_BUTTON.
* **glutDisplayFunc ( )** : Register the display function that is executed when the window needs to be redrawn.
* **glutPostRedisplay ( )** : Request that the display callback be executed after the current callback returns.
* **glutMainLoop ( )** : Cause the program to enter an event processing loop. It should be the last statement in main ().
* **glPushMatrix ( )** and **glPopMatrix ( )** : Pushes to and pops from the matrix stack corresponding to the current matrix mode.
* **glBegin ( )** : Initiates a new primitive of type mode and starts the collection of vertices. Values of mode include GL\_POINTS, GL\_LINES and GL\_POLYGON.
* **glEnd ( )** : Terminates a list of vertices.

**CHAPTER 5**

# TESTING

Testing has been conducted as tabulated below.

**Table no 5.1 Test Results**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl. No.** | **Functions with parameters under test** | **Expected result** | **Actual result** | **Comments** |
| 1 | All the output statement of the options page must be in their position mentioned | The initial project scenario must be displayed once any key is pressed. | The initial project scenario appeared after pressing any key | Pass |
| 2 | Execution Window | As soon as key ‘A’ or ‘a’ is pressed the execution window must appear. | When key ‘A’ or ‘a’ is pressed the execution window is opened. | Pass |
| 3 | Help Window | As soon as key ‘H’ or ‘h’ is pressed the help window must appear. | When key ‘H’ or ‘h’ is pressed the help window is opened. | Pass |
| 4 | Exit | As soon as key ‘E’ or ‘e’ is pressed the initial window must be closed. | When key ‘E’ or ‘e’ is pressed the initial window is closed. | Pass |
| 5 | Option Menu | When right button of mouse is clicked, an option menu must appear. | When right button of mouse is clicked, an option menu appeared. | Pass |

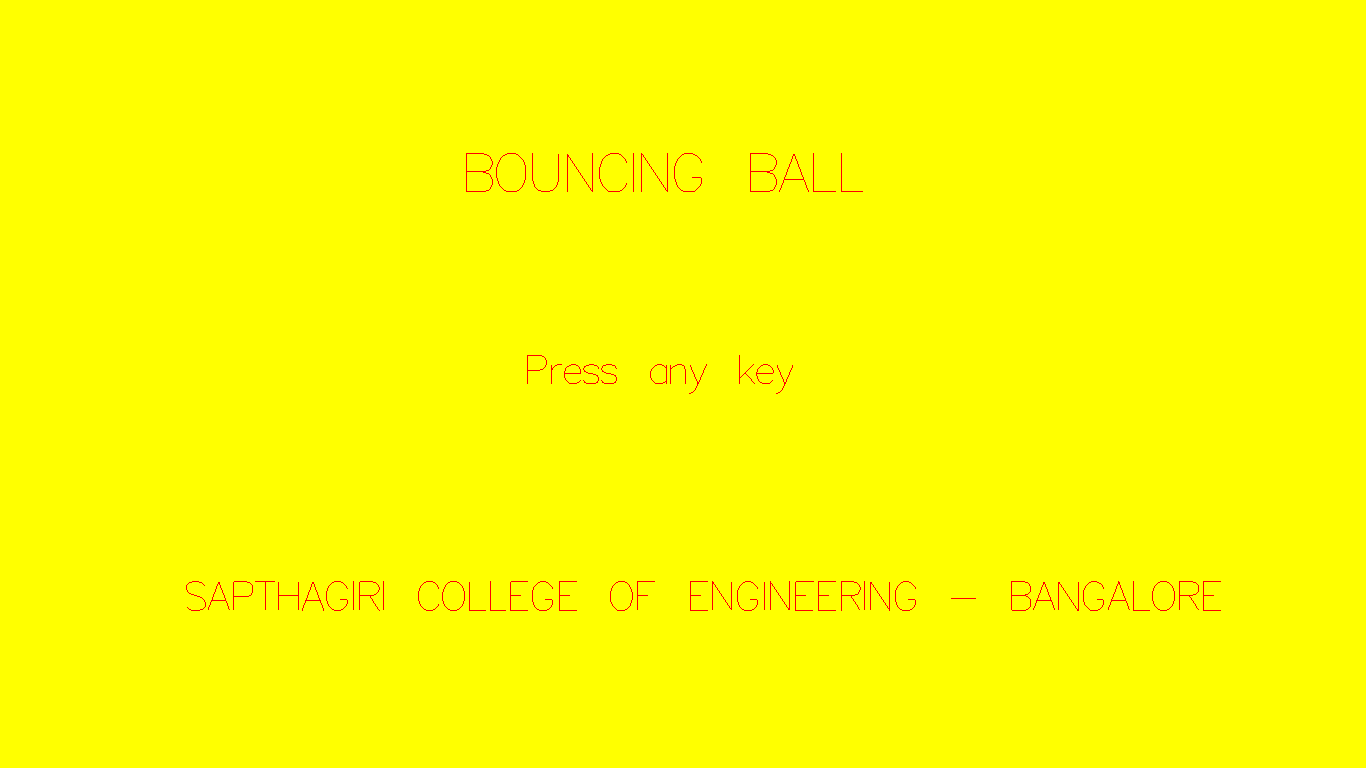
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 6 | Toggle Camera Spin | When ‘Toggle Camera Spin’ option is selected from option menu, camera spin must toggle between start and stop. | When Toggle Camera Spin’ option is selected from option menu, camera spin gets toggled between start and stop. | Pass |
| 7 | Toggle Ball Bounce | When ‘Toggle Ball Bounce’ option is selected from option menu, ball must toggle between start and stop bouncing. | When ‘Toggle Ball Bounce’ option is selected from option menu, ball gets toggled between start and stop bouncing. | Pass |
| 8 | Ball Speed | When ‘Ball Speed’ option is selected from option menu, another option menu must appear. | When ‘Ball Speed’ option is selected from option menu an option menu appeared. | Pass |
| 9 | Slow Ball Speed | When ‘Slow’ option is selected from option menu, ball bounce speed must get slow. | When ‘Slow’ option is selected from option menu, ball bounce becomes slowest. | Pass |
| 10 | Medium Ball Speed | When ‘Medium’ option is selected from option menu, ball bounce speed must get medium. | When ‘Medium’ option is selected from option menu, ball bounce speed becomes medium. | Pass |
| 11 | Fast Ball Speed | When ‘Fast’ option is selected from option menu, ball bounce speed must get fast. | When ‘Fast’ option is selected from option menu, ball bounce speed becomes fastest. | Pass |
| 12 | Lighting | When ‘Lighting’ option is selected from option menu, another option menu must appear. | When ‘Lighting’ option is selected from option menu an option menu appeared. | Pass |
| 13 | Red Lighting | When ‘Red’ option is selected from option menu, lighting must get changed to red color. | When ‘Red’ option is selected from option menu, lighting changes to red color. | Pass |
| 14 | Green Lighting | When ‘Green’ option is selected from option menu, lighting must get changed to green color. | When ‘Green’ option is selected from option menu, lighting changes to green color. | Pass |
| 15 | Blue Lighting | When ‘Blue’ option is selected from option menu, lighting must get changed to blue color. | When ‘Blue’ option is selected from option menu, lighting changes to blue color. | Pass |

**CHAPTER 6**

# RESULTS AND SCREENSHOTS

Below are some screen shots of the project.

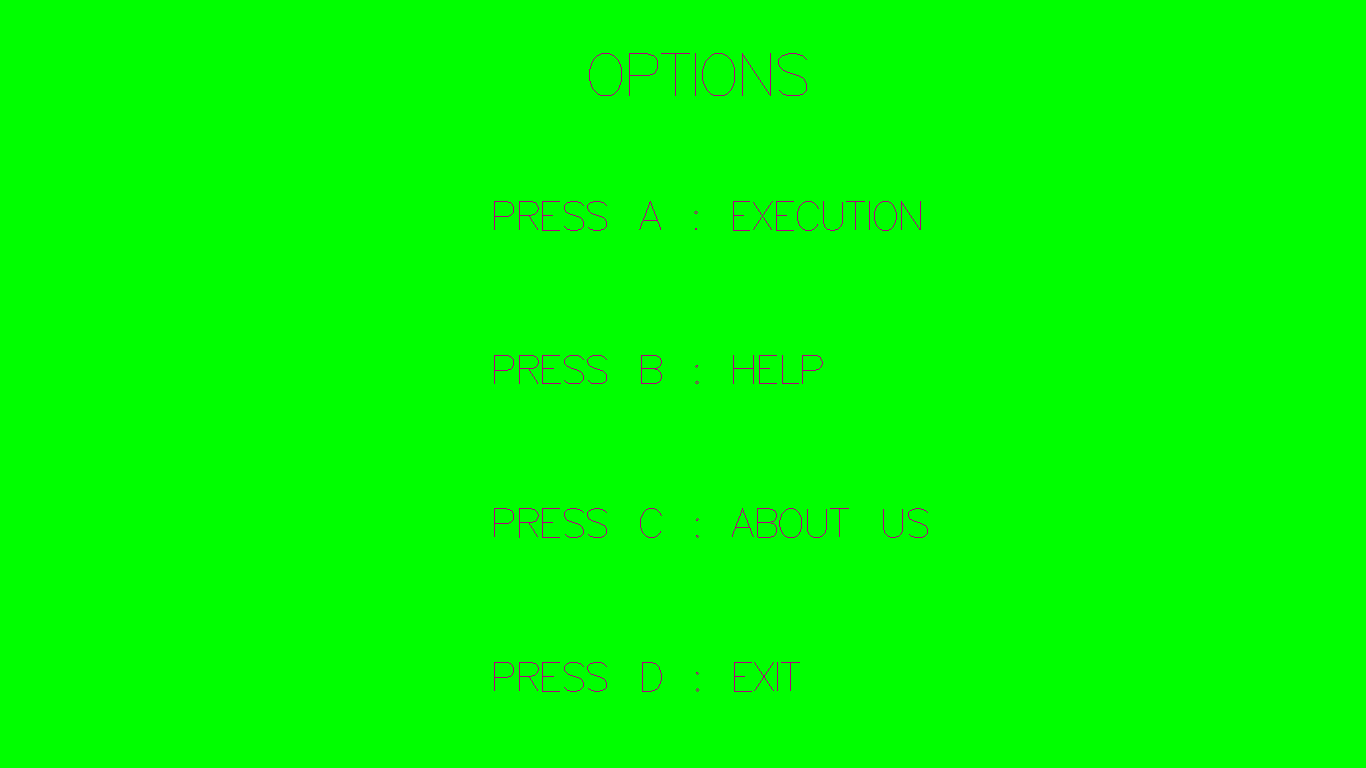
**1) Introduction Page**



**Fig 6.1: Introduction Page**

Fig 6.1Display the introduction page of the mini project

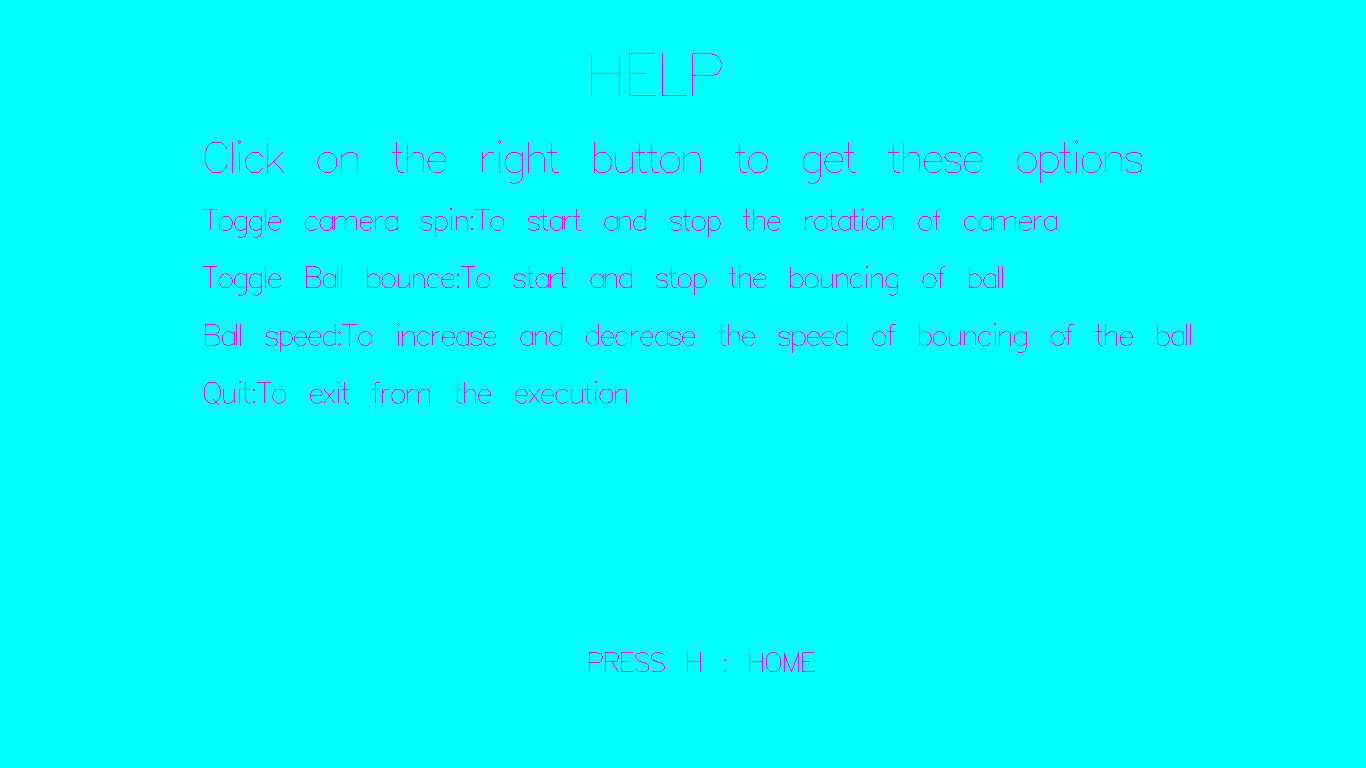
**2) Options Page**



**Fig 6.2: Options Page**

Fig 6.2 Display the various options to use after the introduction page.

**3) Help Page**



**Fig 6.3: Help Page**

Fig 6.3 Display various instructions that can be performed by a user.

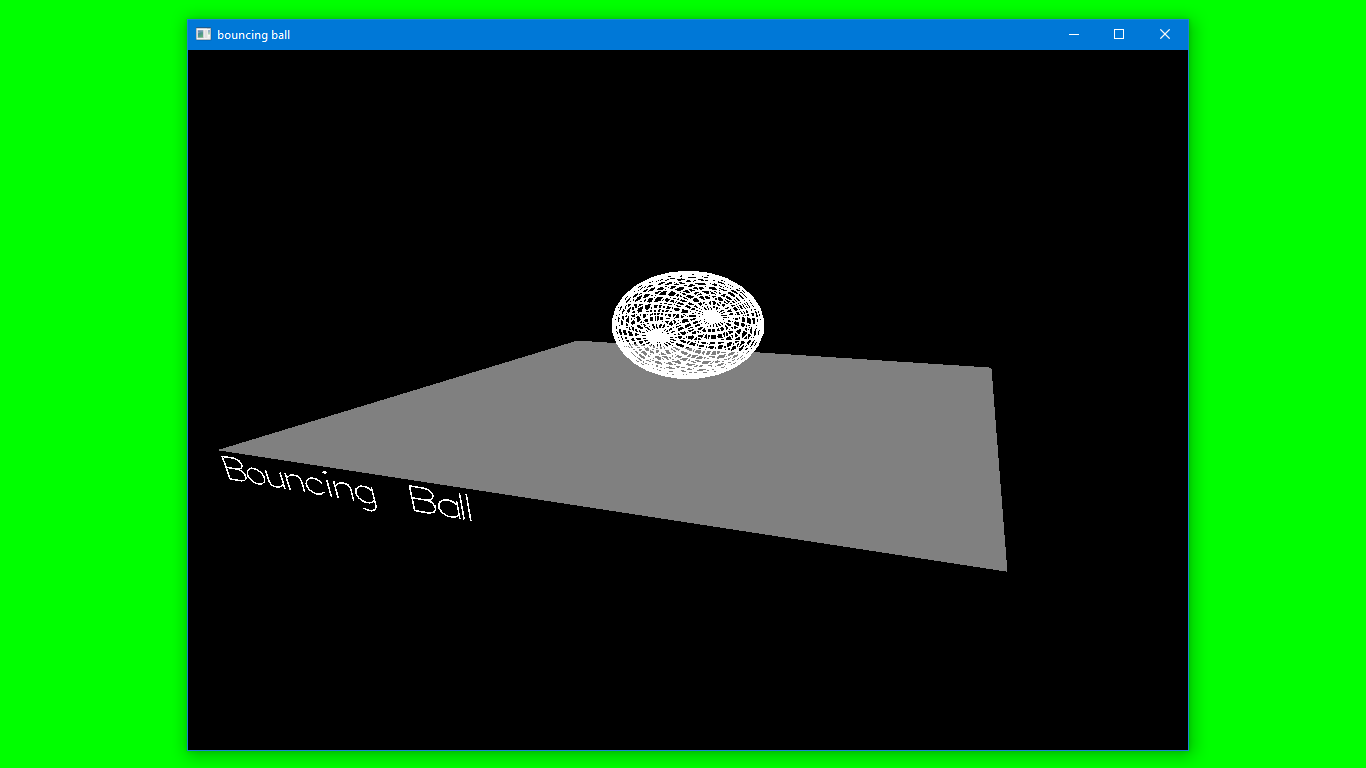
**4) About Us**



**Fig 6.4: About Us**

Fig 6.4 Display the information about the creator of project.

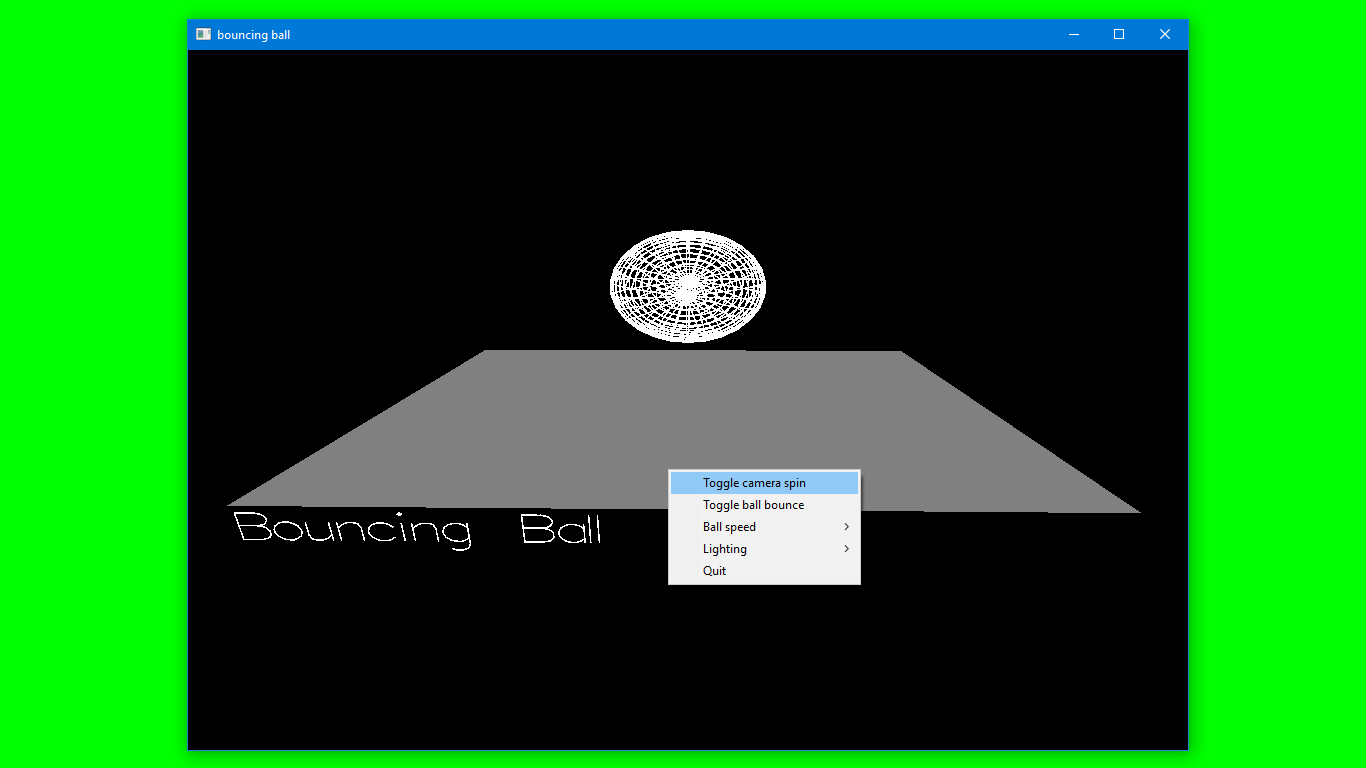
**5) Default Page**



**Fig 6.5: Default Page**

Fig 6.5 Display the default output window.

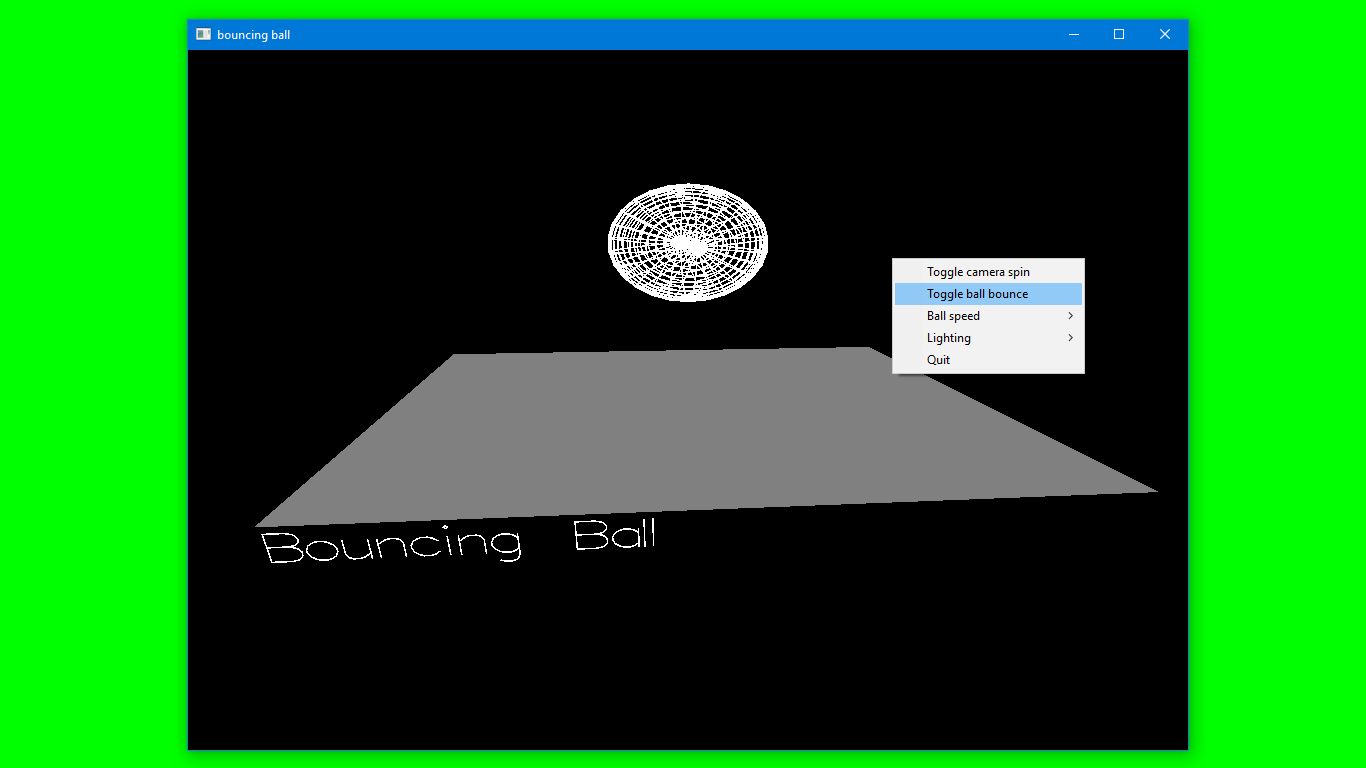
**6) Toggle Camera Spin**



**Fig 6.6: Toggle Camera Spin**

Fig 6.6 Camera movement can be turned on or off.

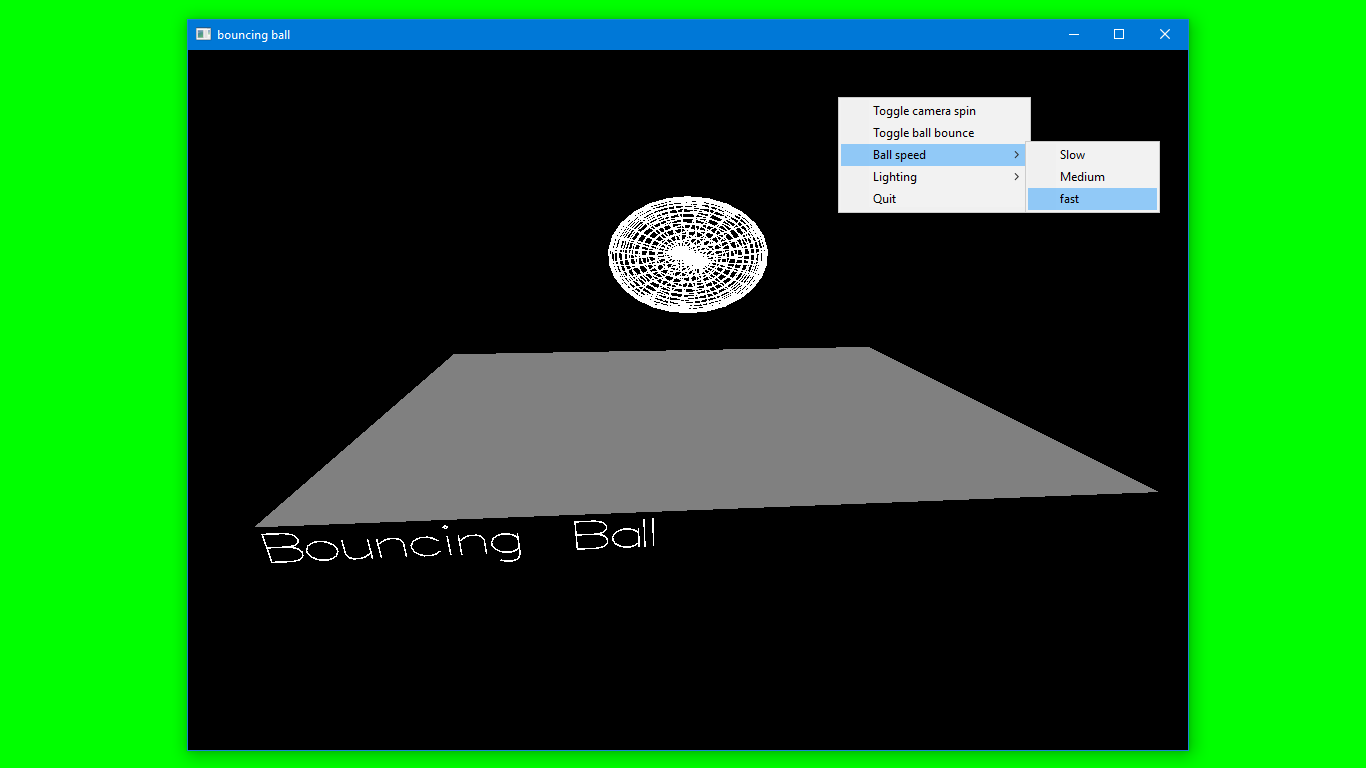
**7) Toggle Ball Bounce**



**Fig 6.7: Toggle Ball Bounce**

Fig 6.7 Ball bouncing can be turned on or off.

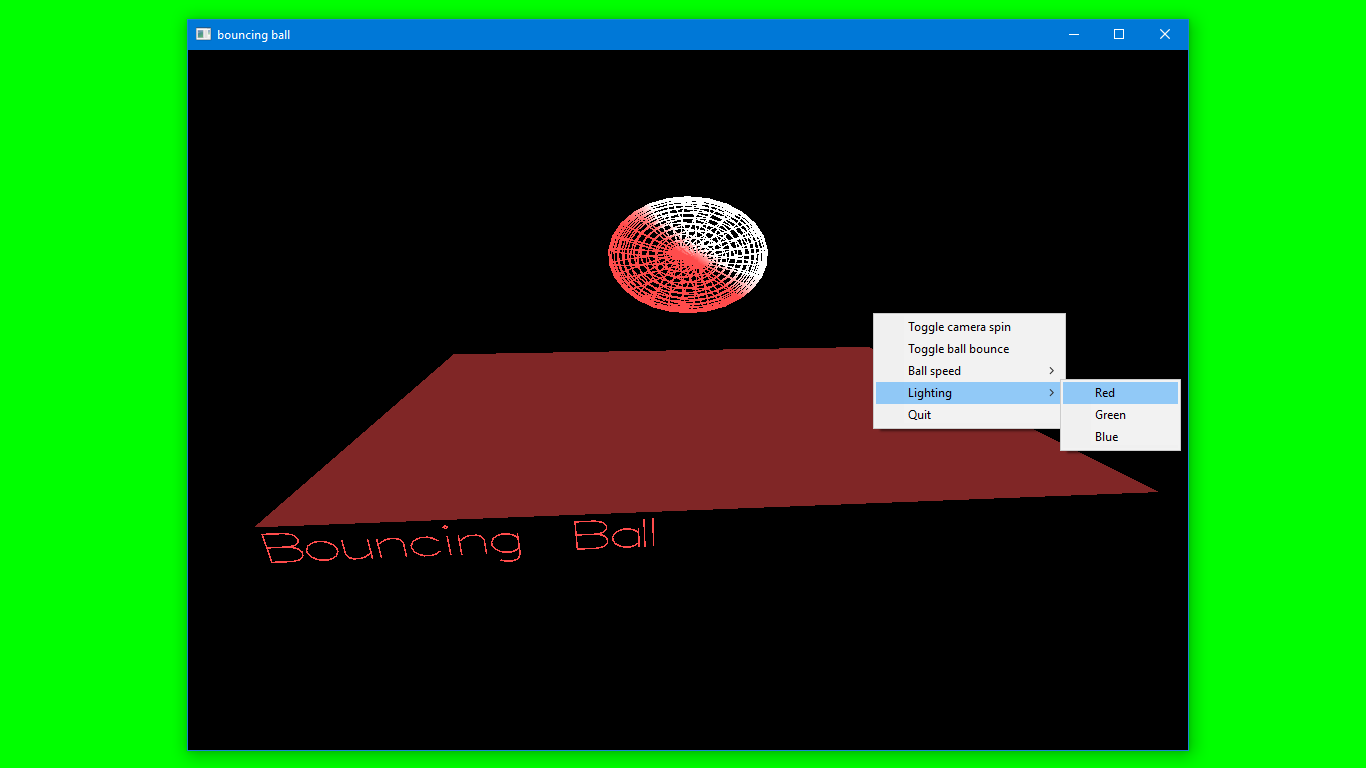
**8) Ball Speed**



**Fig 6.8: Ball Speed**

Fig 6.8Ball speed can be toggled between slow, medium and fast.

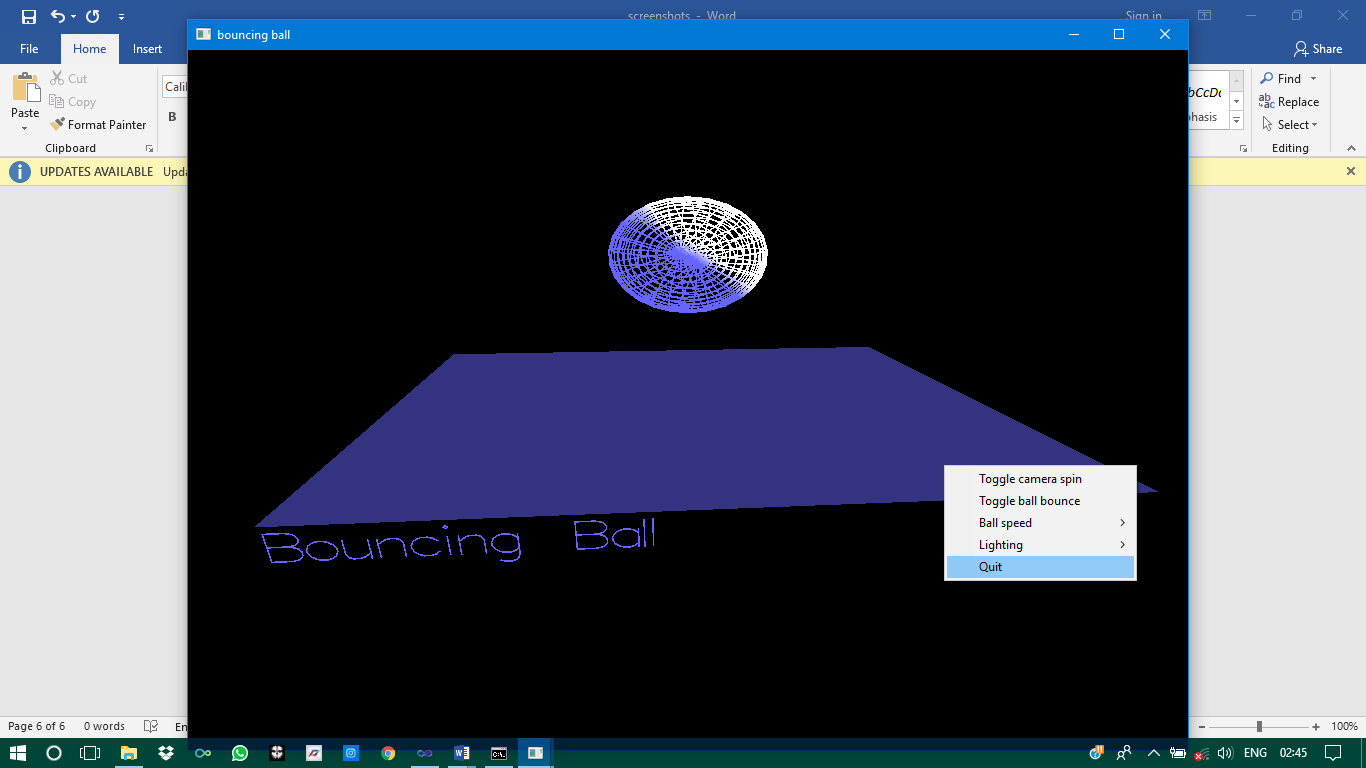
**9) Lighting**



**Fig 6.9: Lighting**

Fig 6.9 Lighting can be toggled between red, green and blue.

**10) Quit**



**Fig 6.10: Quit**

Fig 6.10 Clicking upon it will destroy the output window.

**CHAPTER 7**

# CONCLUSION

# 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 application **3d Bouncing Ball** would be helpful in visualizing the bouncing of a 3d ball which is spherical in nature on a plane surface.

The project has been implemented with a simple interface so that the user can use it without the thorough knowledge of OpenGL.

This project has been demonstrated to fulfil the requirements. The functionality of all the modules and the module level integration is found to be satisfactory.

The development of the mini project has given us a good exposure to OpenGL by which we have learnt some of the technique which helps 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.

**CHAPTER 8**

# FUTURE ENHANCEMENTS

The mini project developed has a scope for future enhancement too as follows:

1. Different textures can be added to the ball.
2. Perspective viewing to objects can be added.
3. The 3d ball can be also implemented in other programs.
4. Also, there is scope to add various colors of the ball.

# 

# BIBLIOGRAPHY

**BOOKS**

[1] Donald Hearn & Pauline Baker: Computer Graphics with OpenGL Version, 4th

Edition, Pearson Education, 2011.

[2] Edward Angel: Interactive Computer Graphics - A Top Down approach with OpenGL,

5th edition. Pearson Education, 2008.

**WEBSITES**

[1] www.opengl.org [Courtesy: http://www.opengl.org/documentation/specs/glut]

[2] www.glprogramming.com [Courtesy: <http://glprogramming.com/blue/ch01.html>]