**CHAPTER 1**

**INTRODUCTION**

The Propeller simulation using OpenGL is used to represent the working of a propeller i.e., the detailed view of its blades along different viewing angles. The different axis of rotation of propeller is displayed.

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.

**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 Mathematica), 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](http://en.wikipedia.org/wiki/2D_computer_graphics) 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.](http://en.wikipedia.org/wiki/Silicon_Graphics) (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.

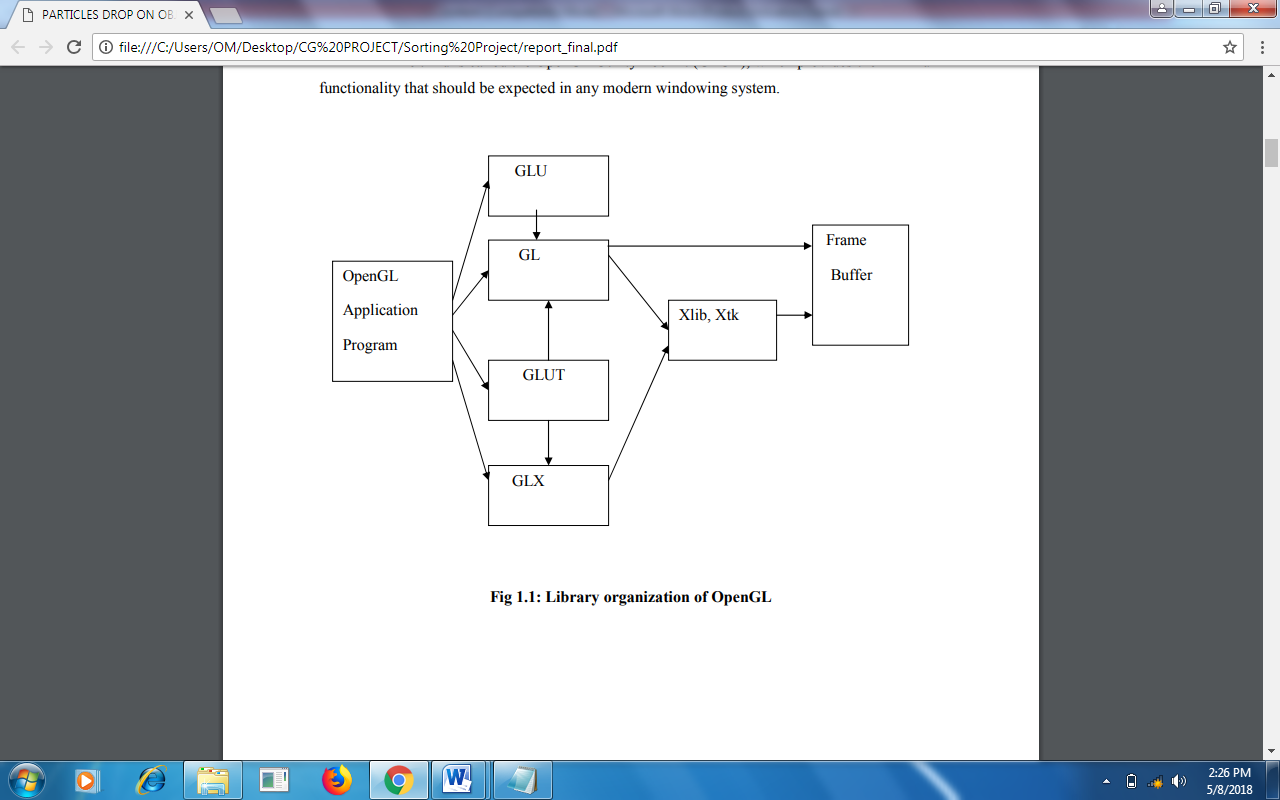


Figure 1.1: Library organization of OpenGL

* 1. **Overview of the project**

By implementing certain technical concepts like transformation, this project will demonstrate the use of three dimensional propellers using OpenGL. The project has detail information about simulation of the propeller in different forms as well as regaining its original shape

**The various concepts used in this application are as follows:**

**Transformations:**

A transformation is a function that takes a point or vector and maps that point or vector into another point or vector. In this application the transformation is used to make the plane to rotate in all 3 axis.

**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.

**The main features of the project are as follows**

* The application is user friendly, thereby it can be used by children and who do not know computer in depth too.
* As provision for future development.
* It has many options of keyboard functions.
* It has a good visual effect.
  1. **Aim of the project**

The aim of the project is to demonstrate the working of propeller using openGL functions.

**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**

In software engineering, a **functional requirement** defines a function of a software system or its component. A function is described as a set of inputs, the behavior, and outputs (see also software). Functional requirements may be calculations, technical details, data manipulation and processing and other specific functionality that define *what* a system is supposed to accomplish. Behavioral requirements describing all the cases where the system uses the functional requirements are captured in use cases.

**The various methods used in this project are as follows:-**

* **Display**

The module draws the output on the screen and the functions in it.

* **Keyboard**

The module specifies the action corresponding to the key board.

* **Mouse**

This module specifies the action corresponding to mouse click.

**2.3 Details of the software**

Here, the coding of our project is done in Microsoft Visual C++ which is a commercial integrated development environment (IDE) with OpenGL (Open Graphics Library) which is a standard specification to produce 2D and 3D computer graphics. We use, the OpenGL Utility Toolkit called GLUT which is a library of utilities for OpenGL programs.

**2.3.1 Microsoft Visual C++**

Microsoft Visual C++ is a commercial [integrated development environment](http://en.wikipedia.org/wiki/Integrated_development_environment) (IDE) product engineered by [Microsoft](http://en.wikipedia.org/wiki/Microsoft) for the [C](http://en.wikipedia.org/wiki/C_(programming_language)), [C++](http://en.wikipedia.org/wiki/C%2B%2B) and [C++/CLI](http://en.wikipedia.org/wiki/C%2B%2B/CLI) [programming languages](http://en.wikipedia.org/wiki/Programming_language). It has tools for [developing](http://en.wikipedia.org/wiki/Software_development_process) and [debugging](http://en.wikipedia.org/wiki/Debugging) C++ code, especially code written for the [Microsoft Windows](http://en.wikipedia.org/wiki/Microsoft_Windows) [API](http://en.wikipedia.org/wiki/Application_programming_interface), OpenGL API, the [Direct X](http://en.wikipedia.org/wiki/DirectX)[API](http://en.wikipedia.org/wiki/API) and the [Microsoft .NET](http://en.wikipedia.org/wiki/Microsoft_.NET) Framework.

**2.3.2 OpenGL and GLUT**

OpenGL (Open Graphics Library) is a standard specification defining a cross-language, [cross-platform](http://en.wikipedia.org/wiki/Cross-platform) [API](http://en.wikipedia.org/wiki/Application_programming_interface) for writing applications that produce [2D](http://en.wikipedia.org/wiki/2D_computer_graphics) and [3D computer graphics](http://en.wikipedia.org/wiki/3D_computer_graphics), describing a set of functions and the precise behaviors that they must perform. From this specification, hardware vendors create implementations - libraries of functions created to match the functions stated in the OpenGL specification, making use of hardware acceleration where possible. Hardware vendors have to meet specific tests to be able to qualify their implementation as an OpenGL implementation.

GLUT is the OpenGL Utility Toolkit, a window system independent toolkit for writing OpenGL programs. It implements a simple windowing application programming interface (API)for OpenGL. GLUT makes it considerably easier to learn about and explore OpenGL programming. GLUT provides a portable API so you can write a single OpenGL program that works across all PC and workstation OS platforms.

**2.4 Software requirements**

* OPERATING SYSTEM **:** Windows 98, Windows XP,Windows Vista,

Windows 7

* FRONT END **:** Microsoft Visual Studio 2010
* CODING LANGUAGE **:** C++

**2.5 Hardware requirements**

* SYSTEM : Pentium IV 2.4 GHz or above
* HARD DISK : 40 GB, 80 GB, 160 GB or above
* MONITOR : 15 VGA colour
* RAM : 256 MB, 512 MB, 1 GB or above

**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.

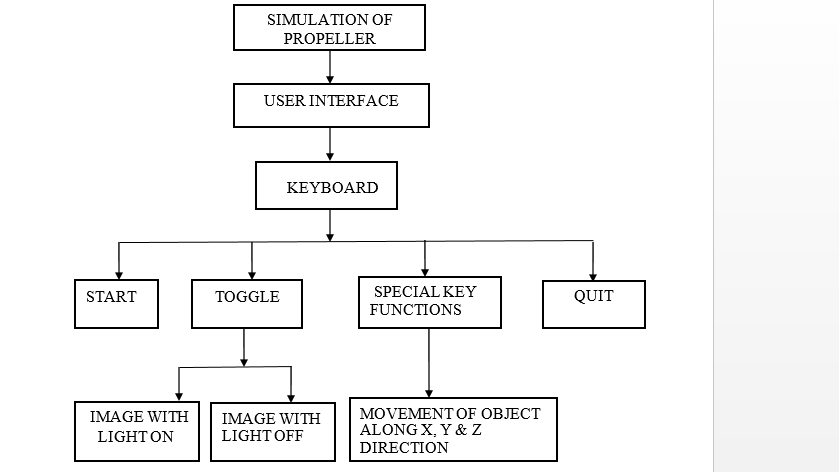


Figure 3.1:Data flow for Propeller Simulation.

The flow chart indicates the various key functions and the mouse functions used with in the system. Specific keys on keyboard have certain functions based on which the output changes. The mouse function is used to terminate the system.

**Chapter 4**

**Implementation**

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

* 1. **USER DEFINED FUNCTIONS**
* Void DrawTopTriangleSet() : Draw the top triangles of propeller.
* Void DrawTopTriangleSetNormalVector() : Draw the top triangles of propeller with normal vectors.
* Void DrawBottomTriangleSet() : Draw the bottom triangles of propeller.
* Void DrawBottomTriangleSetNormalVector() : Draw the bottom triangles of propeller with normal vectors.
* Void DrawBackTriangleSet() : Draw the back triangles of propeller.
* Void DrawBackTriangleSetNormalVector() : Draw the back triangles of propeller with normal vectors.
* Void DrawInsideTriangleSet() : Draw the inside triangles of propeller.
* Void writemessage() : Writes the message required to make the movements of propeller.
* Void reshape(int w, int h) : Reshapes the current window size.
* Void display() : The function is called by GLUT to display the text.
* Void keyboard(unsigned char key, int x, int y) : Alphabet specific keys for movement of propeller.
* Void SpecialKeys(unsigned char key, int x, int y) : Cursor key functions to move propeller.
* Void mouse(int btn, int state, int x, int y) : Mouse buttons used to quit the program.

**4.2 BUILT IN FUNCTIONS**

* **voidglClear (GLbitfield*mask*);**
* ***mask*** *–* Bitwise OR of masks that indicate the buffers to be cleared. The four masks are GL\_COLOR\_BUFFER\_BIT, GL\_DEPTH\_BUFFER\_BIT, GL\_ACCUM\_BUFFER\_BIT and GL\_STENCIL\_BUFFER\_BIT.
* It clears buffers to preset values.
* **voidglClearColor (GLclampf*red*, GLclampf*green*, GLclampf*blue,* GLclampf*alpha*);**
* *red, green, blue, alpha* – specify the red, green, blue and alpha values used when the color buffers are cleared. The initial values are all 0.
* It specifies clear values for the color buffers.
* **void glColor3f (GLfloat*red*, GLfloat*green,* GLfloat*blue*);**
* *red, green, blue –* specify new red, green, and blue values for the current color.
* It sets the current color.
* **glutCreateWindow (char \**name*);**
* *name –* ASCII character string for use as window name.
* It creates a top-level window.
* **voidglutDisplayFunc (void (*\*func*) (void));**
* *func* – the new display callback function.
* It sets the display callback for the current window.
* **voidglutInitWindowSize (int*width,* int*height*);**
* width– width in pixels.
* height – height in pixels.
* It is used to set the initial window size.
* **voidglutMainLoop (void);**
* It enters the GLUT event processing group. This routine should be called at most once in a GLUT program. Once called, this routine will never return. It will call as necessary any callbacks that have been registered.
* **voidglutPostRedisplay (void);**
* It marks the current window as needing to be redisplayed.
* **void display (void);**
* It contains the function definition for display callback.
* **voidglutIdleFunc (void (\**func*) (void));**
* It sets the global idle callback.

# CHAPTER 5

# TESTING

Testing has been conducted as tabulated below.

Table 5.1 Testing

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test case**  **Id** | **Test case description** | **Input** | **Actual output** | **Expected output** | **Remarks** |
| 1.1 | Initial view of propeller | Execute the program | Displays solid cone propeller and a light source, refer Figure 6.1 | Has to display solid cone propeller and a light source | Pass |
| 1.2 | Initial view of propeller | Execute the program | Code Crash | Has to display solid cone propeller and a light source | Fail |
| 2.1 | Propeller with more twist | Press Key 't' | Displays propeller with more twist, refer Figure 6.2 | Has to display propeller with more twist | Pass |
| 2.2 | Propeller with more twist | Press Key 't' | No display | Displays propeller with more twist, refer Figure 6.2 | Fail |
| 3.1 | Propeller with less twist | Press Key 'f' | Displays propeller with less twist, refer Figure 6.3 | Has to display propeller  with less twist | Pass |
| 3.2 | Propeller with less twist | Press Key 'f' | Displays propeller with no twist | Has to display propeller  with less twist | Fail |
| 4.1 | More number  of propellers | Press Key 'q' | Displays more number of propellers, refer Figure 6.4 | Has to display more number of propellers | Pass |
| 4.1 | More number  of propellers | Press Key 'q' | Propeller number remains same. | Has to display more number of propellers | Fail |
| 5.1 | Less number  of propellers | Press Key 'a' | Displays less number of propellers, refer Figure 6.5 | Has to display less number of propellers | Pass |
| 5.2 | Less number  of propellers | Press Key 'a' | Code crash | Has to display less number of propellers | Fail |

**CHAPTER 6**

**SNAPSHOTS**

### 

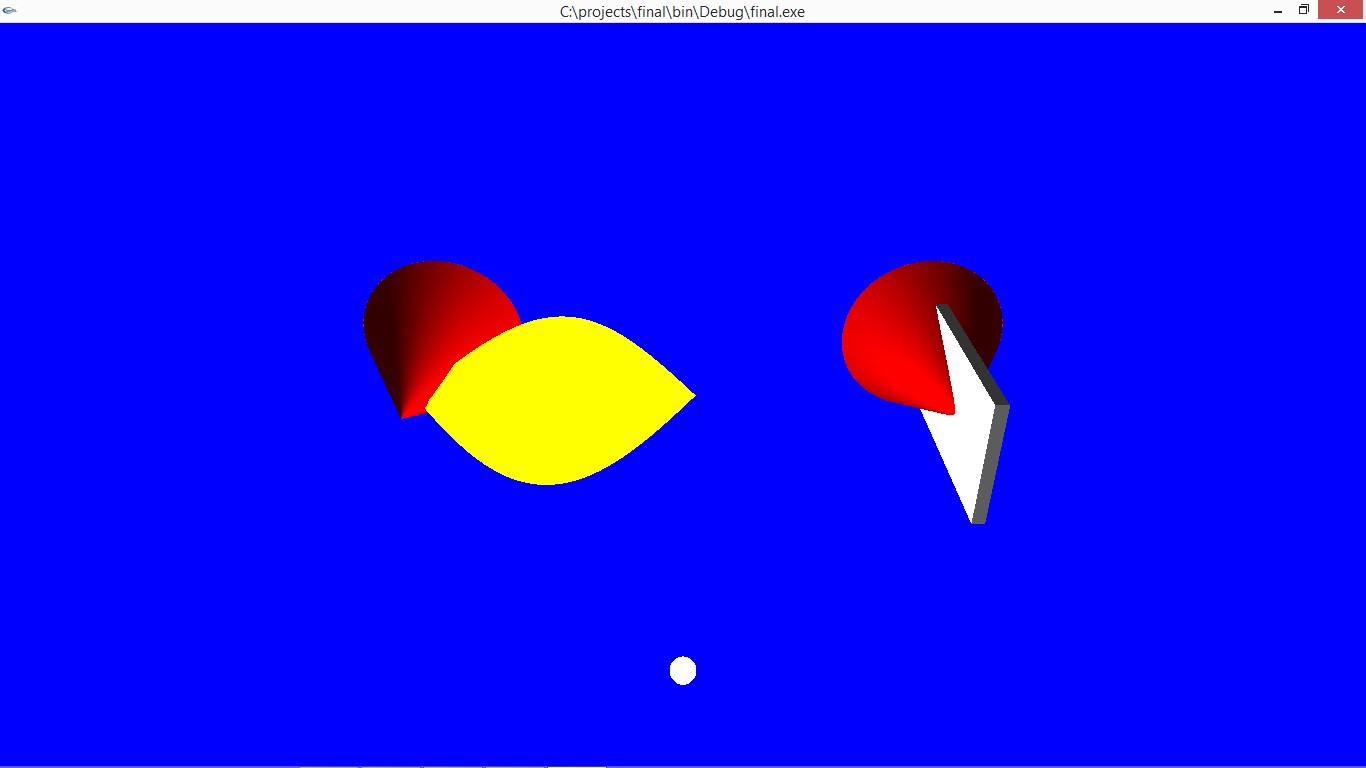


Figure 6.1:Initial view of propeller

Above Figure 6.1 illustrates the initial view of propeller containing solid cone, a propeller triangle and a light source.

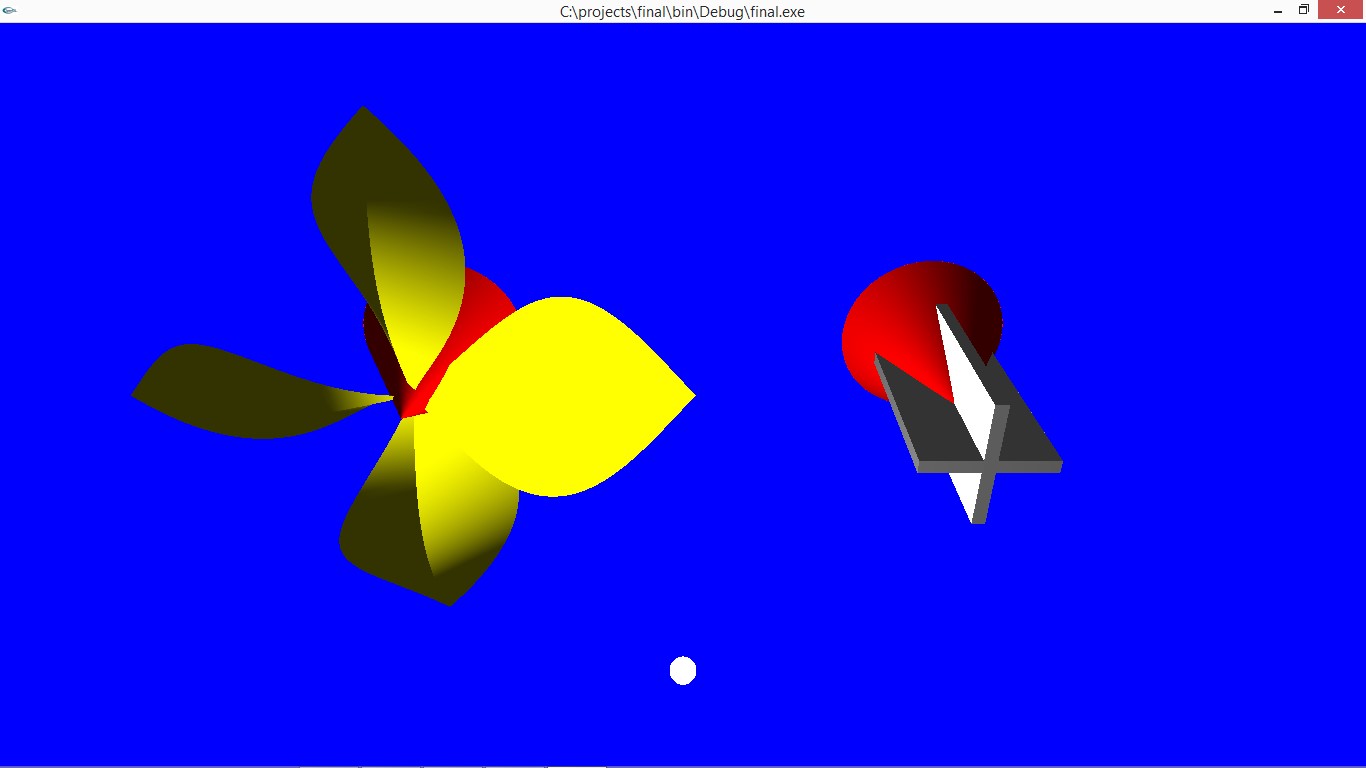


Figure 6.2: Propeller with more twist

Above Figure 6.2 illustrates the more twist of propeller that is obtained by pressing key 't' in keyboard.

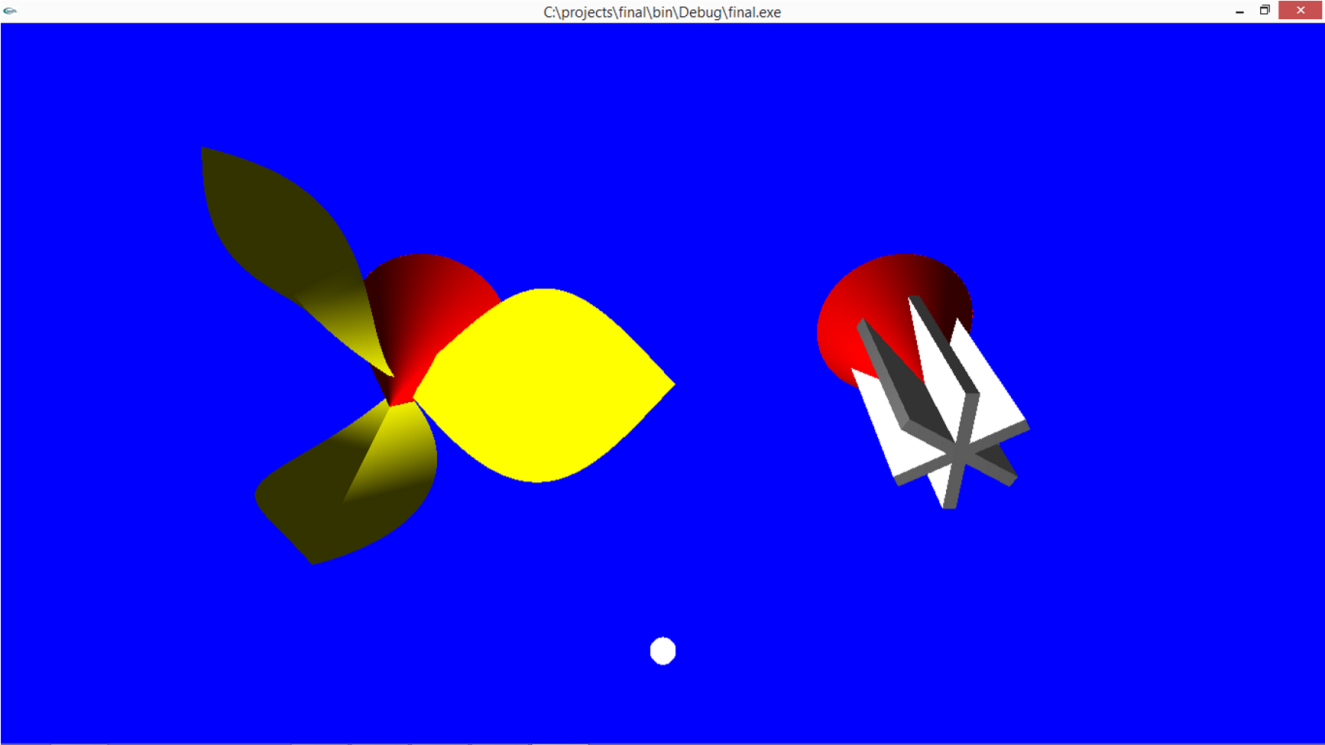


Figure 6.3: Propeller with less twist

Above Figure 6.3 illustrates the less twist of propeller that is obtained by pressing key 'f' in keyboard.

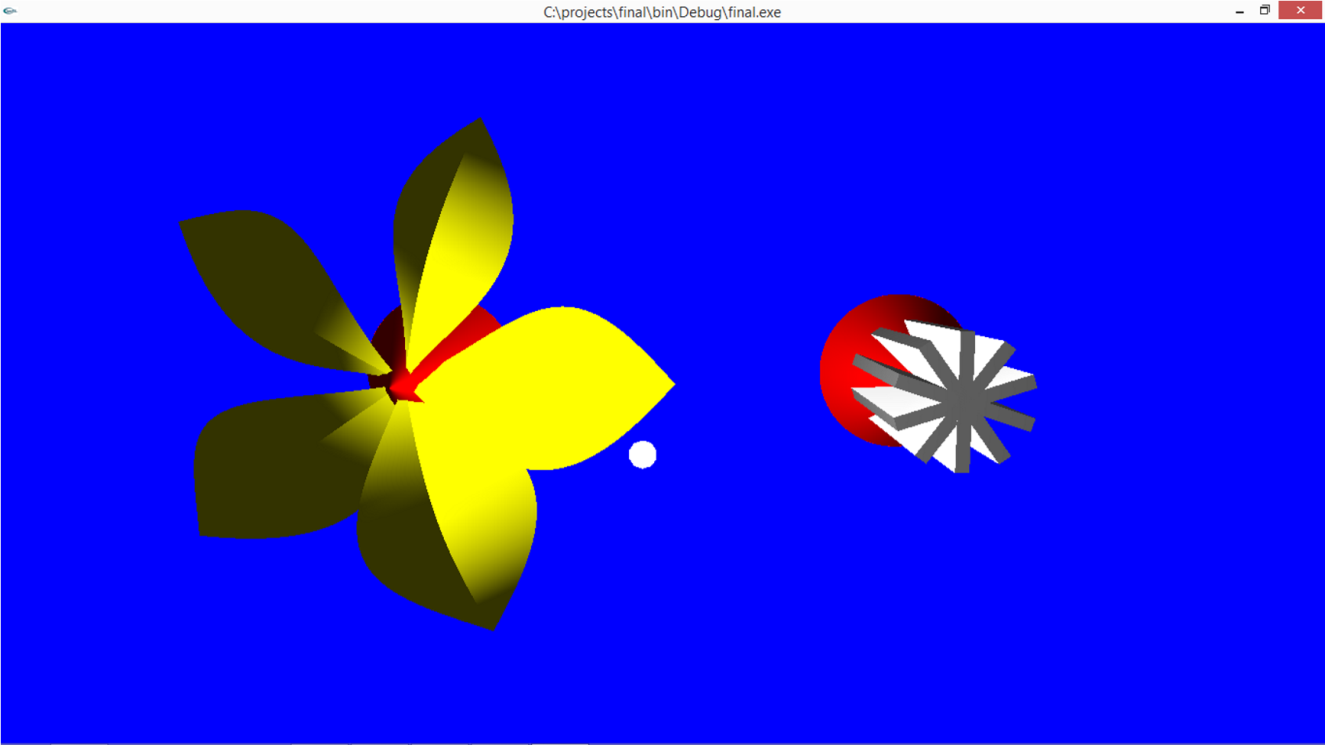


Figure 6.4: More number of propeller

Above Figure 6.4 illustrates the more number of propellers that is obtained by pressing key 'q' in keyboard.

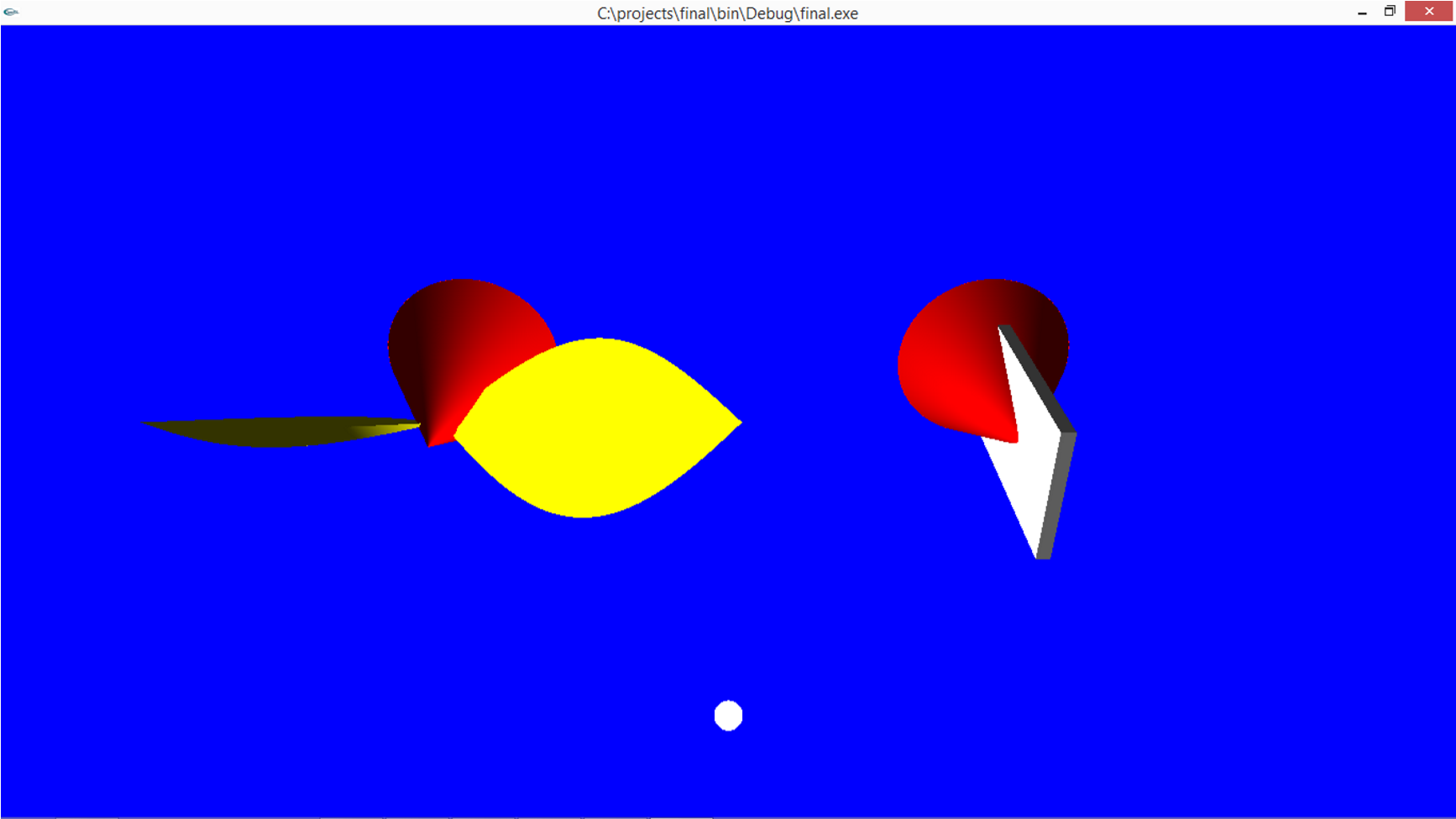


Figure 6.5:Less number of propeller

Above Figure 6.5 illustrates the less number of propellers that is obtained by pressing key 'a' in

keyboard.

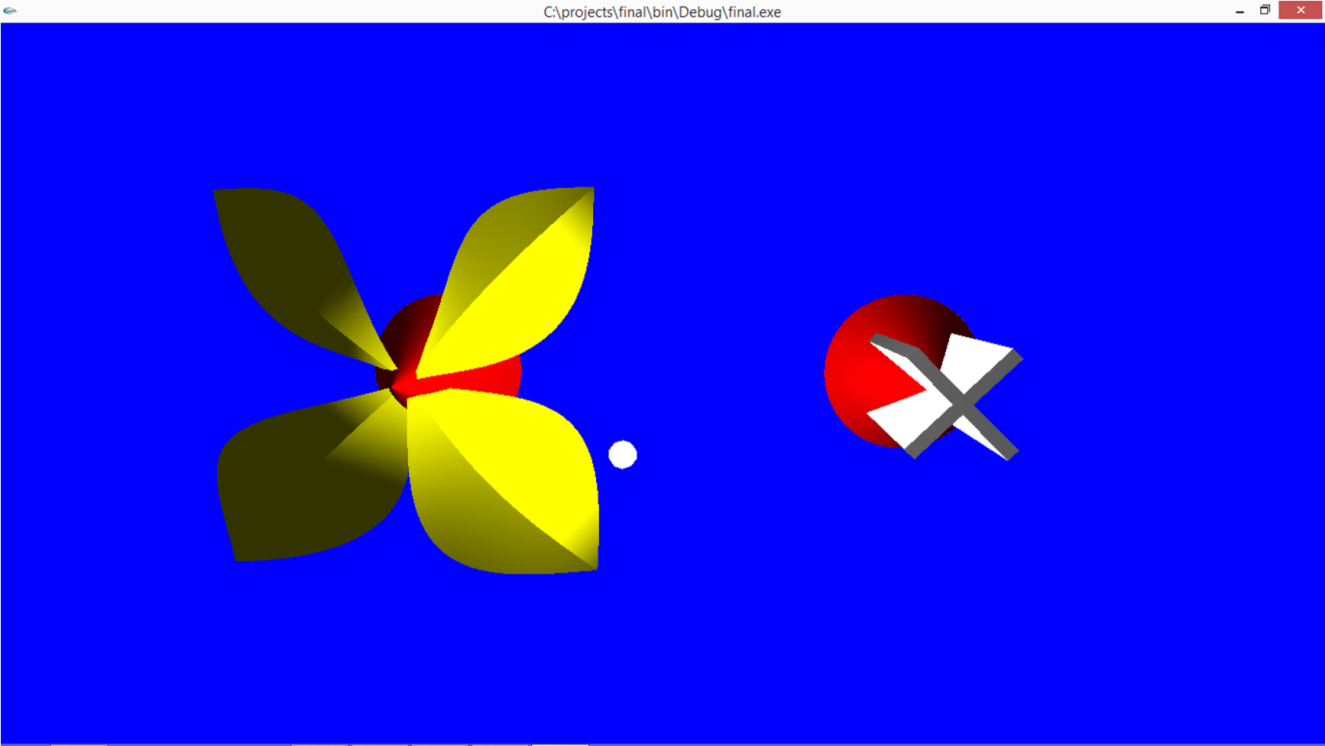


Figure 6.6:Rotation of propeller

Above Figure 6.6 illustrates the rotation of pro peller that is obtained by pressing key 'r' in keyboard.

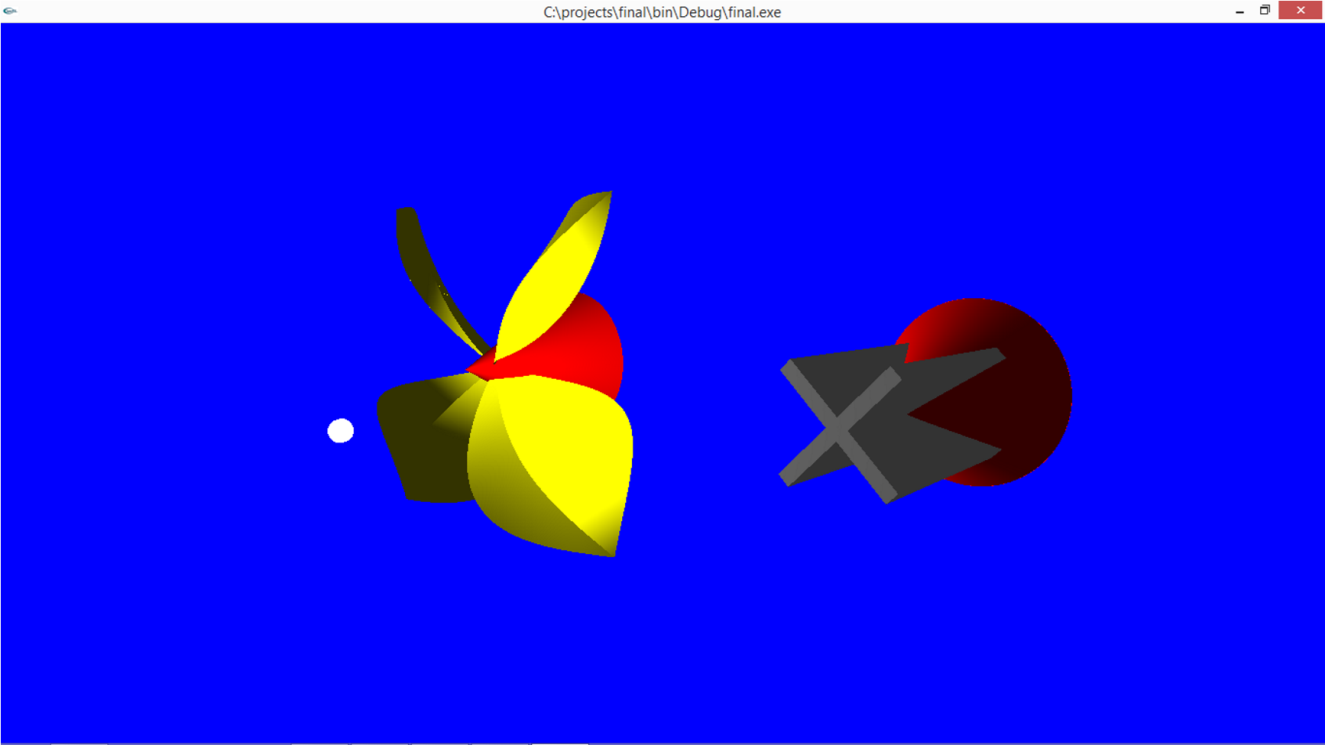


Figure 6.7:Left view of the propeller

Above Figure 6.7 illustrates the left view of the propeller that is obtained by moving the light source towards left using left arrow key in keyboard.

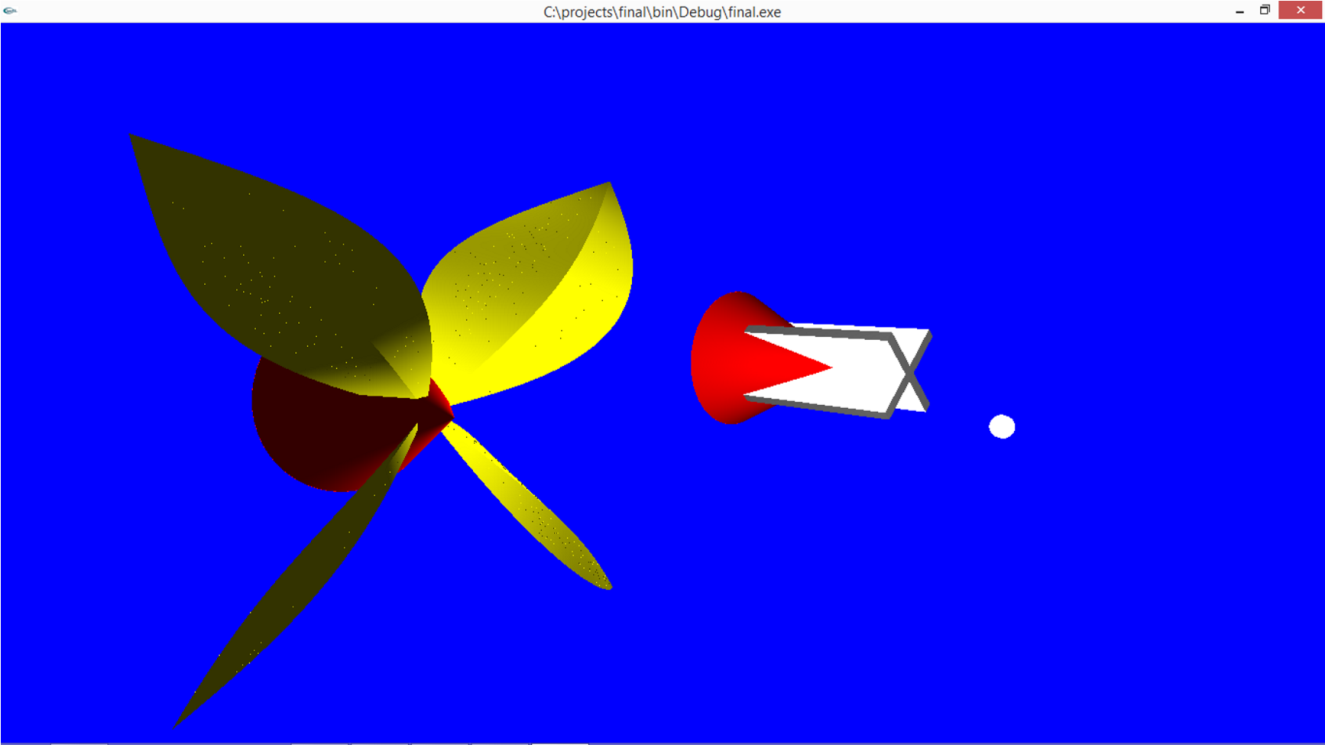


Figure 6.8:Right view of the propeller

Above Figure 6.8 illustrates the right view of the propeller that is obtained by moving the light

source towards right using right arrow key in keyboard.

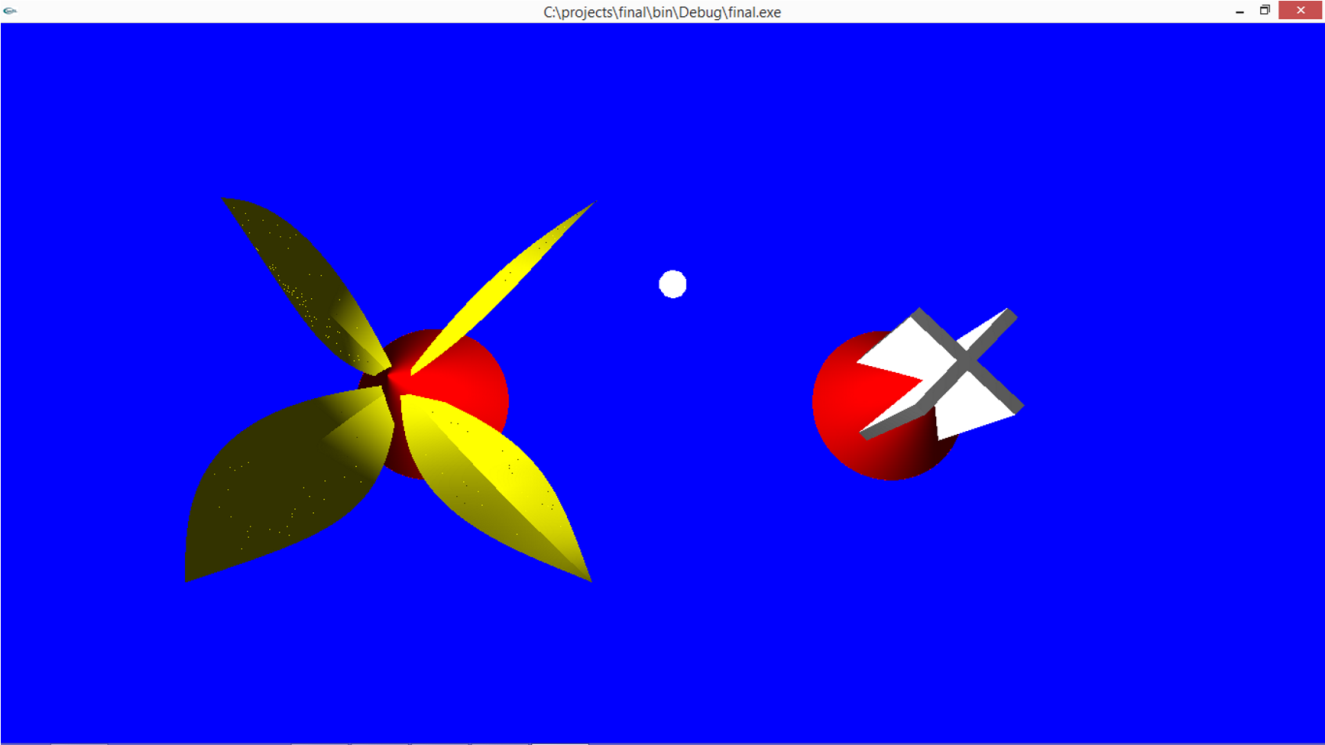


Figure 6.9:Top view of the propeller

Above Figure 6.9 illustrates the top view of the propeller that is obtained by moving the light source towards up using up arrow key in keyboard.

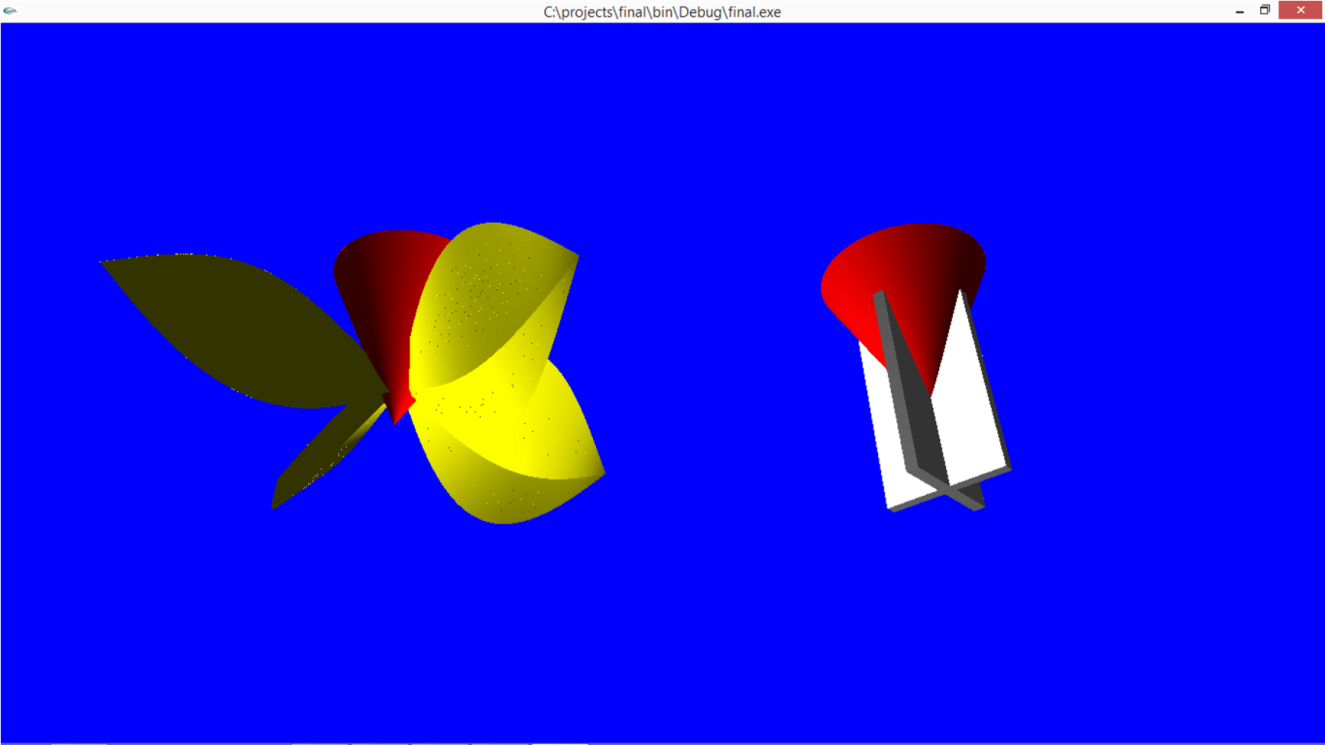


Figure 6.10:Bottom view of the propeller

Above Figure 6.10 illustrates the bottom view of the propeller that is obtained by moving the light source towards down using down arrow in keyboard.

**CHAPTER 7**

**CONCLUSION**

The project has involved the designing of propeller. It is implemented using somein-built functions which are provided in the standard graphics package. We have exhibited the view of propeller and its movements. This project draws a propeller and its movements. The project consist of propeller’s different models like propeller with normal vectors, propeller with more and less triangles. The propeller rotate about its axis, moves down, right and left. We have light source which can move in all x, y, z direction, which illuminates the specific part of propeller where light falls. We can zoom in and out the scene. We can also switch on and off the light source. We can start and stop animation by left and right mouse key button respectively. The movements are done with the help of keyboard and mouse functions which have been successfully implemented.

**Bibilography**

**Reference Books**

[1] Donald Hearn & Pauline Baker: Computer Graphics with OpenGL Version,3rd / 4th

Edition, Pearson Education,2011

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

5th edition. Pearson Education, 2008

[3] James D Foley, Andries Van Dam, Steven K Feiner, John F Huges Computer graphics

with OpenGL: pearson education