**CHAPTER 1**

**INTRODUCTION**

**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**:

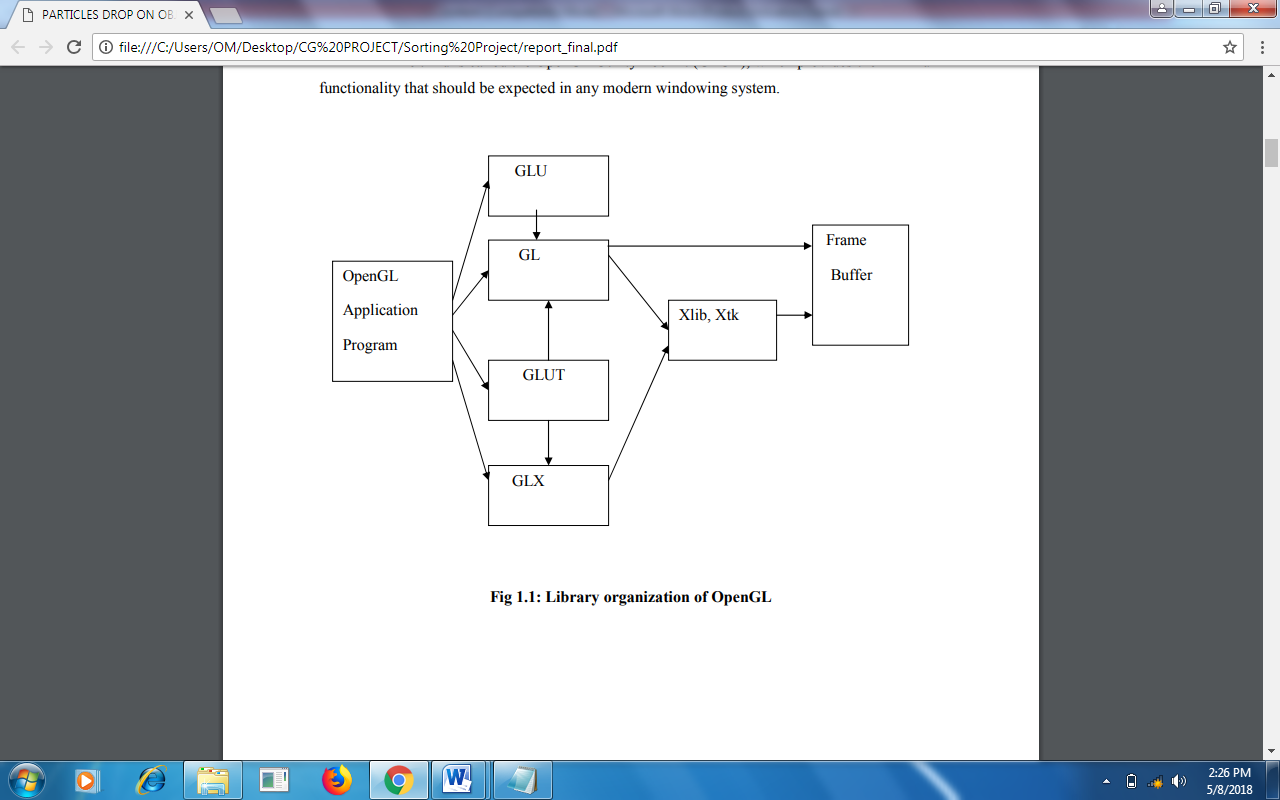
The modification or interpretation of existing pictures, such as photographs and TV scans, is called image processing. In computer graphics, a computer is used to create a picture. Image-processing techniques, on the other hand, are used to improve picture quality, analyze images, or recognize visual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



**Fig 1.1: Library organization of OpenGL**

* 1. **Overview of the project**

Implementing certain technical concepts like transformation. To show that implementation of bubble sort and selection sort graphically. Migration from text editor to OpenGL, consists of array of elements represented in the form of circles in random order. Sorting the randomized elements. Displaying the elements that are being swapped.

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

**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.

**Blending:**

The project uses blending to make objects appear transparent.

**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 axes.

**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 project has been implemented to show the working of Windmill Simulation in such a way that how the propellers of the windmill can rotate, and the speed tracking can be done. Also, it is more advantageous for future modifications as when modifications are to be made instead of searching the whole program module search can be useful as it decreases the time consumption.

**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.

* **Menu**

This module specifies the action corresponding to menu entry.

* **Keyboard**

The module specifies the action corresponding to the key board.

* **Idle**

This module is used to display the object more times using some delay t.

Reading 10 random numbers between 1 and 20

Perform selection sort in descending order.

**2.2 Non-functional requirements:**

These are constraints on the services or functions offered by the system. They include timing constraints, constraints on the development process and standards. Non-functional requirements often apply to the system.

**Non-Functional Requirements are as follows: -**

**2.2.1 Dependability:**

The dependability of a computer system is a property of the system that equates to its trustworthiness. Trustworthiness essentially means the degree of user confidence that the system will operate as they expect, and that the system will not ‘fail’ in normal use.

**2.2.2 Availability:**

The ability of the system to deliver services when requested. There is no error in the program while executing the program.

**2.2.3 Reliability:**

The ability of the system to deliver services as specified. The program is compatible with all types of operating system without any failure.

**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 [DirectX](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.

**2.4Softwarerequirements**

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

Windows 7, Windows 10

* 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**

Flow chart 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.

MAIN

INITIALIZE CALLBACK FUNCTIONS

MAIN SCREEN DISPLAYED

MOUSE BUTTON

RIGHT BUTTON

MIDDLE BUTTON

LEFT BUTTON

SIDE VIEW 2

BACK VIEW

FRONT VIEW

CUSTOM VIEW

SIDE VIEW 1

EVENTS

**Fig 3.1: Flowchart for Simulation of Windmill**

**EXPLANATION oF FLOWCHART:**

The flow of control in the above flow chart is respected to the Texture Package. For any of the program flow chart is compulsory to understand the program. We consider the flow chart for the texture project in which the flow starts from start and proceeds to the main function after which it comes to the initialization of call back functions and further it proceeds to mouse and keyboard functions after all the function, the flow comes to quit which is the end of the flow chart.

**Chapter 4**

**Implementation**

To implement the Current system, the program uses different functions which are as follows:

* 1. **USER DEFINED FUNCTIONS**
* **bitmap\_output(int x, int y, char \*string, void \*font)**

Specify the raster position for pixel operations and render the bitmap character at required position

* **circle\_draw(circle c)**

To draw circle by drawing consecutive triangle fans and to display the value of circle below

* **swap\_circles(circle \*cc1,circle \*cc2)**

Swaps circles from one circle to another circle by changing their centers.

* **bubb\_sort()**

The bubble sort makes multiple passes through a list. It compares adjacent items and exchanges those that are out of orders. Each pass through the list places the next largest values in its proper place.

* **sel\_sort()**

The selection sort algorithm sorts an array by repeatedly finding the maximum element (considering descending order) from unsorted part and putting it at the beginning.

**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 colour 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: Test case for menu option**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sl no. | Test case description | Expected Result | Actual Result | Remarks |
| 1. | Click the MIDDLE mouse button on the display screen | Menu with   * Side View 1 * Side View 2 * Back View * Front View * Custom View   Should be displayed | Menu with   * Side View 1 * Side View 2 * Back View * Front View * Custom View   Is displayed  If not displayed | Pass  Fail |
| 2. | Click on Side View 1 option | A Side View of the Windmill structure (Right Side)  Should be displayed | A Side View of the Windmill structure  Is displayed | Pass |
| 3. | Click on Side View 2 option | A Side View of the Windmill structure (Left Side)  Should be displayed | A Side View of the Windmill structure  Is displayed  A side view of the windmill is not displayed | Pass  Fail |
| 4. | Click on Back View option | A Back-Side View of the Windmill structure  Should be displayed | A Back-Side View of the Windmill structure  Is displayed | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 5. | Click on Front View option | A Front Side View of the Windmill structure  Should be displayed | A Front Side View of the Windmill structure  Is displayed | Pass |
| 6 | Click on Custom View option | Option to enter any custom angle.  Should be displayed  The Computer asks for any custom angle, on entering the values, it rotates the Windmill Structure to that angle. | Option to enter any custom angle.  Is displayed  The Computer asks for any custom angle, on entering the value, it rotates the Windmill Structure to that angle. | Pass |

**Table 5.2: Test case for mouse**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sl no. | Test case Description | Expected result | Actual Result | Remarks |
| 1. | Click on left button on display screen. | The Speed of the Windmill wheel should increase. | Increases the speed of the Windmill wheel. | Pass |
| 2. | Click on right button on display screen. | The Speed of the Windmill wheel should decrease. | Decreases the speed of the Windmill wheel. | Pass |

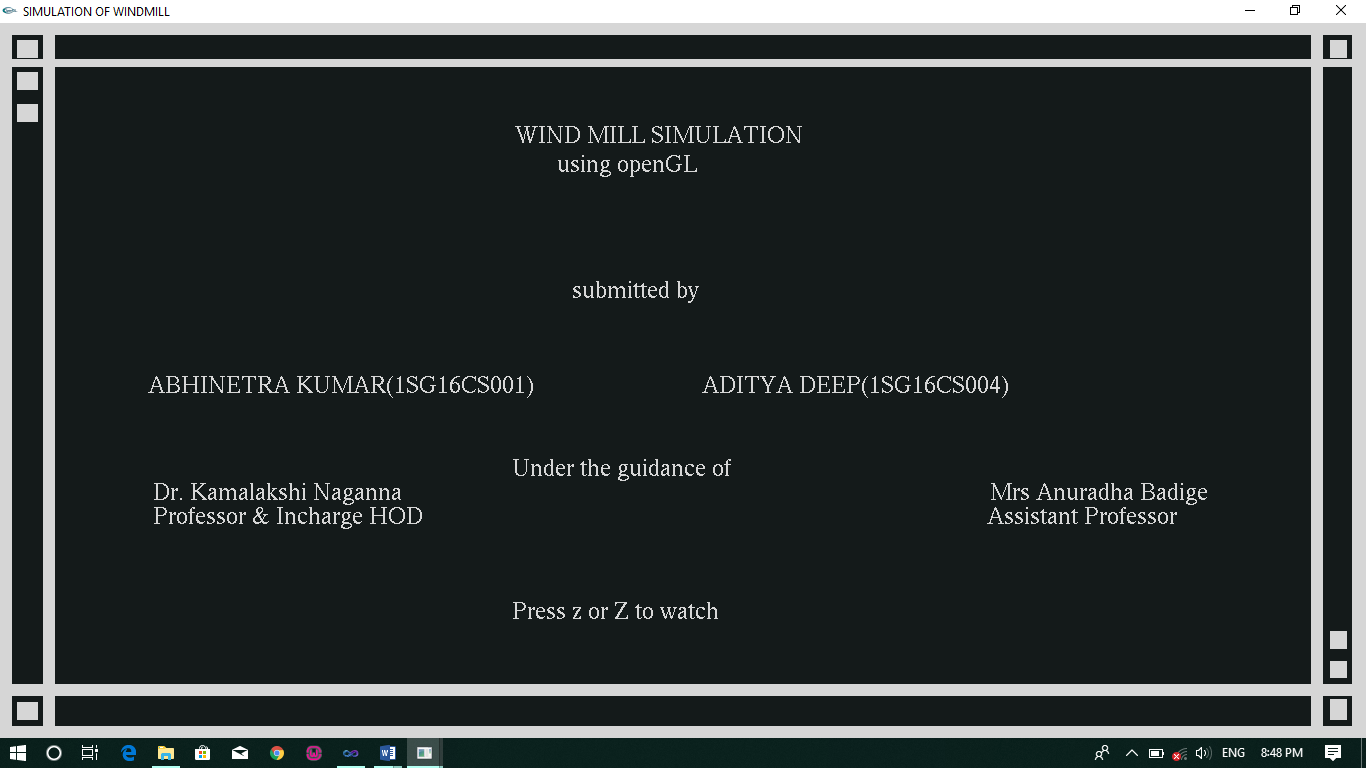
**Table 5.3: Test case for keyboard**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sl no. | Test case Description | Expected result | Actual result | Remarks |
| 1. | Press Left Arrow key on the display screen. | Free anti-clockwise movement the Windmill Structure | Able to move the windmill structure anti-clockwise freely | Pass |
| 2. | Press Right Arrow key on the display screen. | Free clockwise movement the Windmill Structure | Able to move the windmill structure clockwise freely | Pass |

**CHAPTER 6**

**SNAPSHOTS**

* 1. **Main page**



**Figure 6.1: Main Page**

The figure 6.1 shows the main page of the project

* 1. **Windmill working**
     1. **Initial Screen**

****

**Figure 6.2.1: Initial state of windmill**

Figure 6.2.1 shows the Initial state of the Windmill

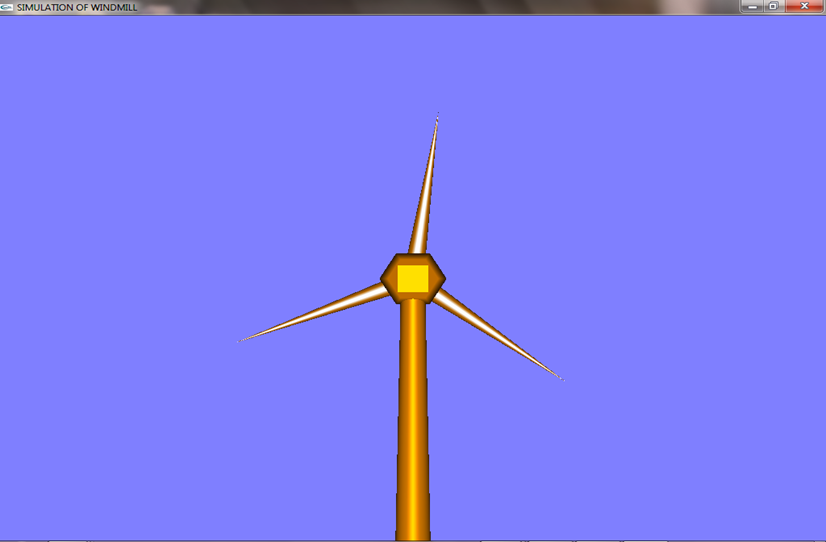
* + 1. **Screen displaying menus**



**Figure 6.2.2: Screen displaying menus**

Figure 6.2.2 shows the menus which is used to toggle the windmill

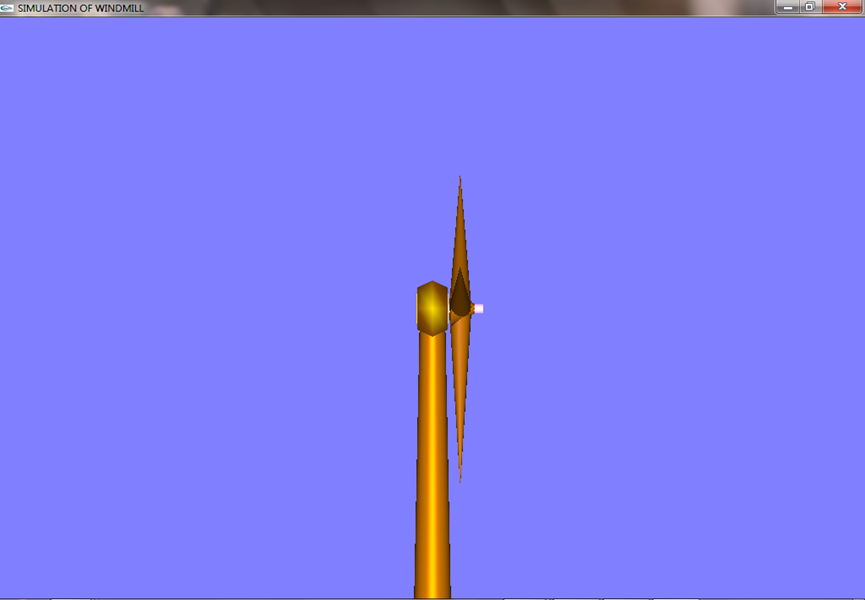
* 1. **Back view**



**Figure 6.3: Back view of windmill**

Figure 6.3 shows the back view of the windmill

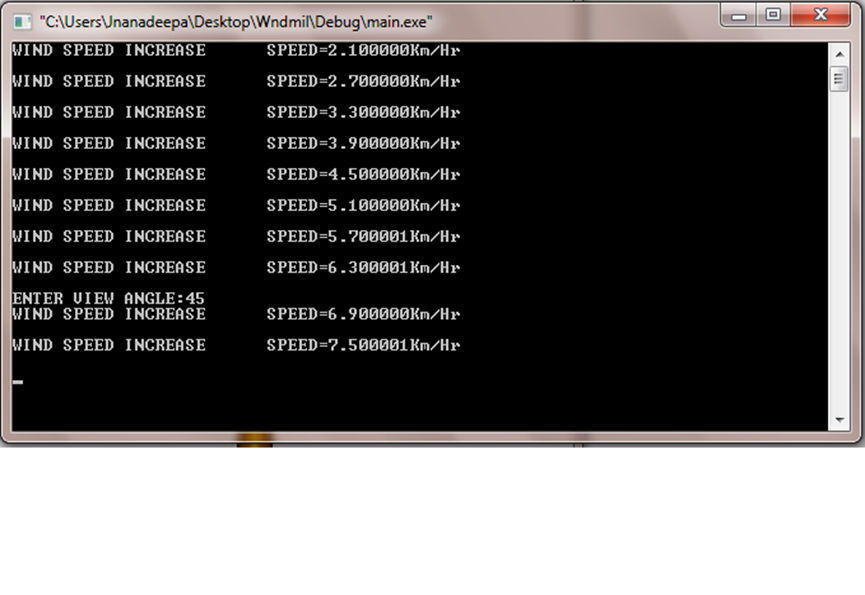
* 1. **Side view**



**Figure 6.4: Side view of windmill**

Figure 6.4 shows the right-side view of the windmill

* 1. **Speed Tracking on mouse clicks**



**Figure 6.5: Speed tracking**

Figure 6.5 shows how speed can be tracked on the screen and even angular position.

**CHAPTER 7**

**CONCLUSION**

**Simulation of Windmill** is a designed and implemented using a graphics software system called **OpenGL** which has become a widely accepted standard for developing graphic application. Using openGL functions user can create geometrical objects and can use **translation, rotation**, scaling with respect to the co-ordinate system.

The project Visual Transformation Techniques using openGL is based on Rotation and Translation processes using shading effects and is demonstrated using Visual C++.

The development of the **Simulation of Windmill** project has given us a good exposure to OpenGL by which we have learnt some of the technique which help in development of animated pictures, gaming. Hence it is helpful for us even to take up this field as our career too and develop some other features in OpenGL and provide as a token of contribution to the graphics world.

**Simulation of Windmill** consist of many users defined function such as increasing windmill fan speed, decreasing windmill fan speed, side views, front and back views, custom angle of rotation of entire windmill structure. All these function makes this project an example of animation in **OpenGL.**

**BibLIography**

**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

**Websites**

* www.opengl.org
* www.google.co.in(OpenGLprojects)
* www.wikipedia.com