|| JAI SRI GURUDEV ||

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama, Belgavi -590018, Karnataka State,India

A Mini Project Report
On

"POWER GENERATION USING WINDMILL"

Submitted in partial fulfillment of sixth semester

Computer Graphics and Visualization Laboratory (15CSL68)

BACHELOR OF ENGINEERING
IN
COMPUTER SCIENCE AND ENGINEERING

Submitted by:

ARPITHA S M
1SJ16CS012
VI Sem,CSE

CHAITHRA M N
1SJ16CS022
VI Sem,CSE

Under the guidance of:

APOORVA S
Assistant Professor
Department of CSE, SJCIT

S.J.C. INSTITUTE OF TECHNOLOGY
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
CHICKABALLAPUR-562101
2019

|| Jai Sri Gurudev ||

Sri Adichunchanagiri Shikshana Trust

S.J.C INSTITUTE OF TECHNOLOGY

Computer Science & Engineering Department

Chickballapur-562101

CERTIFICATE

This is to certify that the mini- project work entitled **“**Power generation using wind mill**”** is a bonafide work carried out at Computer Graphics and Visualization Laboratory by **Arpitha S M (1SJ16CS012) and Chaithra M N (1SJ16CS022)** in partial fulfilment for the award of **Bachelor of Engineering in Computer Science and Engineering** in **Sixth semester** of the **Visvesvaraya Technological University**,Belgaum during the year 2019. It is certified that all corrections/ suggestions indicated for Internal Assessment have been incorporated in the report deposited in the department library. The project report has been approved as it satisfies the academic requirements in respect to sixth semester mini project work.

....	....
Signature of the Guide	Signature of the HOD
Mr. Apoorva S	**Dr. Anitha T. N.**
Assistant Professor	**Professor &HOD**
Dept. of CSE	**Dept. of CSE**
S.J.C.I.T	**S.J.C.I.T**

ACKNOWLEDGEMENT

With Great pride we would like to convey our gratitude and appreciation to our alma-mater **“S.J.C Institute of Technology”** for giving us the required platform for the fulfillment of the project on Computer Graphics and Visualization as per the V.T.U requirements, for the sixth semester lab on the same subject.

We express our sincere thanks to **Dr. Ravikumar K M**, Principal of **S.J.C.I.T** for providing us with excellent infrastructure and facilities to complete the project.

We express whole hearted gratitude to **Dr. Anitha T. N.** Professor and HOD of **Computer Science and Engineering Department**. We wish to acknowledge her help in making our task easy by providing us with her valuable help and encouragement.

It is our pleasure to thank our guide **Mr. Apoorva S, Assistant Professor, Department of Computer Science and Engineering, SJCIT** for his guidance, encouragement and valuable suggestion from the beginning of the project work till the completion without which this project work would not have been accomplished. We greatly indebted to him.
And last but not the least, we would be very pleased to express our heart full thanks to the teaching and non-teaching staff of the Department of Computer Science and Engineering, SJCIT for their motivation and support.

We would also thank all those who extended their support and co-operation while bringing out this project.

Arpitha S M
(1SJ15CS012)

Chaithra M N
(1SJ15CS022)

ABSTRACT

Computer graphics is mainly concerned with all the aspects of producing pictures or images using computer. The field began humbly almost 50 years ago with display of few lines on CRT. Now we can create images by using graphics that are indistinguishable from photographic images of real objects.

Electricity generation is the process of generating electricity energy from other forms of energy.Wind power is the conversion of wind energy into useful form of energy ,such as using wind turbines to make electricity,windmills for mechanical power,wind pumps for water pumping or drainage.This project is to demostrate how the wind energy is converted into electrical energy.The total amount of economically extractable power available from the wind is considerably more than present human power use from all sources. To demonstrate this, we make use of some of the primitives available in Open Graphics Library(OpenGL).

LIST OF FIGURES

Chapter 1

INTRODUCTION

1.1 Introduction to mini project

Electricity generation is the process of generating electric energy from other forms of energy.Thefundamental principles of electricity generation were discovered during the 1820s and early 1830s by the British scientist Michael Faraday.

In recent history, there has been a significant increase in the search for alternative sources of energy. Wind energy is one of the cleanest sources of energy today, however it is not the most efficient. Wind power now has the capacity to generate 430 TWh annually, which is about 2.5% of worldwide electricity usage. Wind turbines convert the kinetic energy in the wind into mechanical power.This mechanical power can be used for specific tasks (such as grinding grain or pumping water) or a generator can convert this mechanical power into electricity.A large wind farm may consist of several hundred individual wind turbines which are connected to the electric power transmission network.When combined with solar electricity, this energy source is great for developed and developing countries to provide a steady, reliable supply of electricity.

1.2 Introduction to Computer Graphics

Computer Graphics is concerned with all aspects of producing pictures or images using a computer. The field began humbly almost 50 years ago, with the display of a few lines on a cathode-ray tube (CRT); now, we can create images by computer that are indistinguishable from photographs of real objects. We routinely train pilots with simulated airplanes, generating graphical displays of a virtual environment in real time. Feature-length movies made entirely by computer have been successful, both critically and financially. Massive multiplayer games can involve tens of thousands of concurrent participants.

Perhaps the dominant characteristic of this new millennium is how computer and communication technologies have become dominant forces in our lives. Activities as wide – ranging as filmmaking, publishing, banking and education continue to undergo revolutionary changes as these technologies alter the ways in which we conduct our daily activities. The combination of computers, networks, and the complex human visual system, through computer graphics, has led to new ways of displaying information, seeing virtual worlds, and communicating with people and machines.

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 concerns with the pictorial synthesis of real or imaginary objects from their computer based models, where the related field of image processing treats the converse process, the analysis of scenes, or the reconstruction of models of 2D or 3D objects from their pictures. 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.

1.3 Introduction to OpenGL

OpenGL is a software interface to graphics hardware. This interface consists of about 150 distinct commands that you use to specify the objects and operations needed to produce interactive three-dimensional applications. OpenGL is easy to learn, and it possesses most of the characteristics of other popular computer graphics system.

OpenGL is designed as a streamlined, hardware-independent interface to be implemented on many different hardware platforms. To achieve these qualities, no commands for performing windowing tasks or obtaining user input are included in OpenGL; instead, you must work through whatever windowing system controls the particular hardware you're using. Similarly, OpenGL doesn't provide high-level commands for describing models of three-dimensional objects. Such commands might allow you to specify relatively complicated shapes such as automobiles, parts of the body, airplanes, or molecules. With OpenGL, you must build up your desired model from a small set of geometric primitives - points, lines, and polygons.

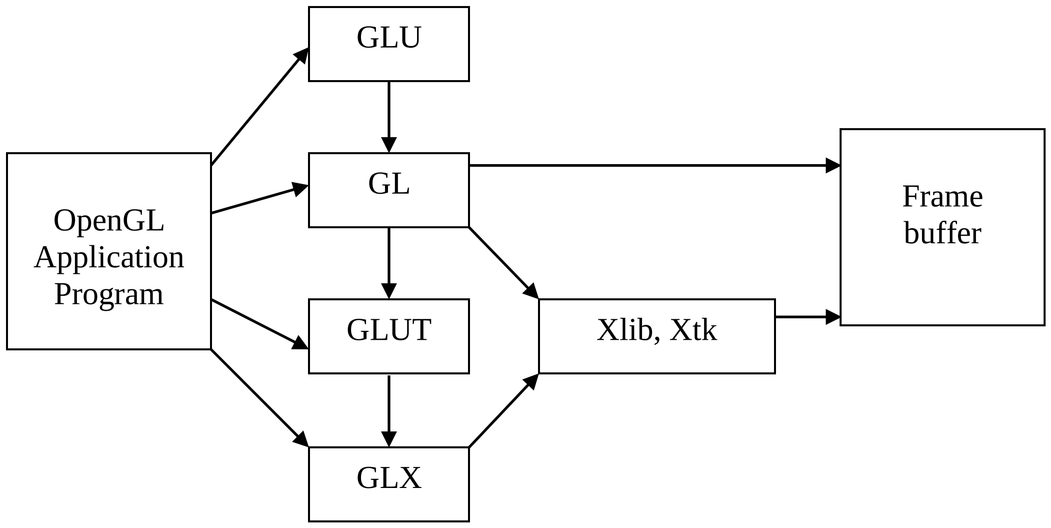


Figure 1.2.1 Library Organization

A sophisticated library that provides these features could certainly be built on top of OpenGL. The OpenGL Utility Library (GLU) provides many of the modeling features, such as quadric surfaces and NURBS curves and surfaces. GLU is a standard part of every OpenGL implementation. Also, there is a higher-level, object-oriented toolkit, Open Inventor, which is built atop OpenGL, and is available separately for many implementations of OpenGL.

Basically, our project is about designing of ship with its different views along with the movement which contains some of the objects like sky, mountain, hut, tree etc in the background. Here we are using various OpenGL geometric primitives such as GL_LINES, GL_POLYGON, GL_LINE_LOOP, and GL_POINTS. By using 3D transformations like Translation, Rotation, and Scaling different views of the ship is shown. We have used different interactions like Mouse, Keyboard and Menu. Here we are applying lighting effects using OpenGL functions for realistic effect.

1.4 SCOPE AND APPLICATIONS

The project helps to understand and modify the developed animation without much animation. Hence it can be used for both convenience and learning without bearing any effect on the system or the data contained in it. The project helps in understanding the graphics and its implementation and uses.

Chapter 2

LITERATURE SURVEY

2.1 Related work

Classification of overlaps between 3D graphics primitives and 2D regions of the screen is a fundamental problem in computer graphics, arising in hidden surface removal , collision detection , occlusion culling, and parallel rendering.

In particular, partitioning primitives based on tile overlaps is an important step in many high-performance polygon rendering systems utilizing image-space parallelism or bucket rendering. Different tile shapes have been used in these systems, including scan lines , horizontal strips, vertical strips , and rectangular areas. Tiles have been kept static or dynamically adjusted based on the distribution of graphics primitives on the screen. For each graphics primitive, the rendering system must determine which 2D screen-space tiles it overlaps so that it can invoke rendering operations on the appropriate processors.

Most current parallel polygon rendering systems are based on a sort-middle architecture. Some hardware implementations rely upon a fast, global interconnection to distribute primitives from geometry processors to rasterizers. For instance, SGI’s Infinite Reality Engine uses a fast Vertex Bus to broadcast screen space vertex information from semi-custom ASIC geometry processors to all ASIC rasterization processors. In this case, overlaps between primitives and tiles are not determined by the geometry processors, and every vertex is broadcast to every rasterization processor, requiring each rasterization processor to check all primitives to see if they overlap its screen region. Most other sort-middle systems have used the 2D bounding box of a primitive to test for overlaps with screen-space tile regions. For example, this method is used in UNC’s PixelPlanes 5 to sort primitives among tiles loaded onto a work queue to be processed by multiple rasterization processors . Other examples include Renderman, PixelFlow, and several commercial PC-based graphics accelerators .

2.2 Problem Statement

The purpose of this project is to basically demonstrate the various views and the outlooks of

primitives to the users with an interactive 3D animation. People who are less familiar with this 3D object after going through this can get to know the animations object how it is performed.
The existing system involves computer graphics. Computer Graphics started with the display of data on hard copy, plotters and cathode ray tube screens soon after the introduction of computer themselves. omputer graphics today is largely interactive – the user controls the contents , the structures and appearance of objects and their displayed images by using input devices such as keyboard, mouse or touch-sensitive Panel on the screen. Interactive computer graphics is the most important means of producing pictures since the invention of photography and television.

2.3 Proposed System

In the proposed system, the openGL, is a graphic software system designed as a streamlined , hardware - independent interface to be implemented on many different hardware. To achieve these qualities, no commands, for performing windowing tasks or obtaining user input are included in OpenGL. OpenGL doesn't provide high level commands specify relatively complicated shapes. With openGL, we must build up our desired model from a small set of geometric primitives – points , lines , and polygon.

.

2.4 Advantages

- Wind is free, meaning that wind farms need no fuel.
- It produce no waste or greenhouse gases, making them great for the environment.
- Wind is also a renewable resource; therefore it won't run out like coal or other fossil fuels.
- Wind power is also quite useful for supplying power to remote areas, meaning it can go where other power sources cannot.

2.5 Disadvantages

- They include the unpredictable behavior of wind.
- The unsightliness of the wind towers.
- Birds are sometimes killed in the blades.
- It can be somewhat noisy for an entire wind farm.

Chapter 3

SYSTEM REQUIREMENTS AND SPECIFICATION

The hardware and software requirements are very minimal and the software can run on most of the machines.

3.1 Hardware Requirements

- Processor : Pentium processor or AMD Athlon or higher CPU
- Ram : 2GB
- Hard disk space : 320GB
- Mouse and keyboard

3.2 Software Requirements

- Software : Microsoft Visual Studio 2010
- Operating system : windows XP or Windows Vista or Windows 7

3.3 Functional Requirements

The developed project displays the statements of services, the animation should provide, how the system should react to particular inputs and how the system should behave in particular situations.

3.4 Non Functional Requirements

The developed project has the constraints on services or functions offered by the system. They include constraints on development process, standard, etc.

Chapter 4

ANALYSIS AND DESIGN

4.1 High level and Low level design

Design is the creatiom of aplan or convention for the construction of an object or a system.Design has different connotations in different fields.In some cases the direct construction of an object is also considered to be design.Design involves problem-solving and creativity.A design may also be a more plan that does not include a production or engineering processes although a working knowledge of such processes is usually expected of designers.
High level design usually involves the usage of various methods and functions in the program.Low level design makes use of flowcharts,data-flow diagrams,use-case diagrams etc..for the simplified representation of the complex code used.

4.2 Design Principle

There are a lot of them out there in an amazing variety of designs and complexities. All of them had five things in common though:

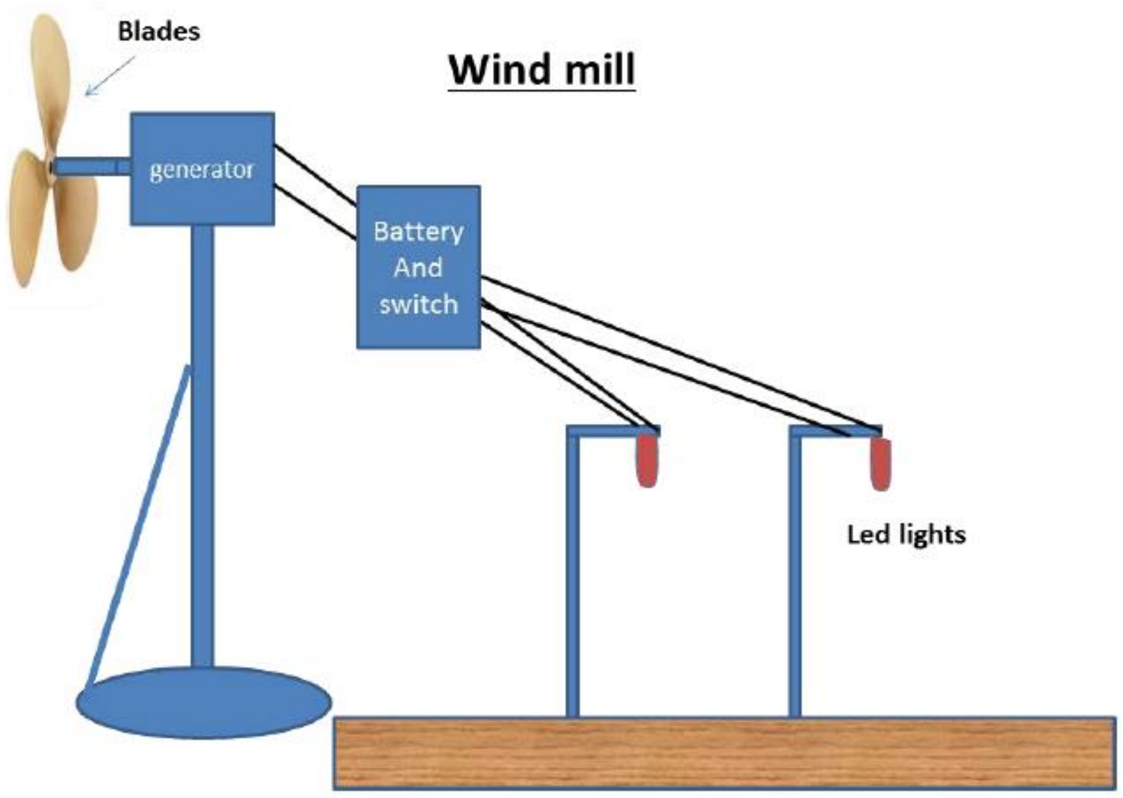
1. A generator
2. Blades
3. A tower to get it up into the wind

We reduced the project to just four little systems. If attacked one at a time, the project didn't seem too terribly difficult. We decided to start with the generator by using a permanent magnet dc motor. This looked like a simpler way to go. A Plastic blade attached with the motors sharp and fixed it on a wood stick. The o/p terminal of the motor is connected to a battery. When wind blows that turbine rotates and electricity generates and that stored in the battery. That battery o/p is connected to LED lights through a switch. This led works here as a street light.

4.3 Design Specification

The design specification for a wind-turbine will contain a power curve and guaranteed availability. With the data from the wind resource assessment it is possible to calculate commercial viability. The typical operating temperature range is −20 to 40 °C (−4 to 104 °F).

In areas with extreme climate (like Inner Mongolia or Rajasthan) specific cold and hot weather versions are required.Wind turbines can be designed and validated according to IEC 61400 standards.



Chapter 5

IMPLEMENTATION

Computer graphics is concerned with all aspects of producing pictures or images using a computer. Here we use a particular graphics software system, OPEN GL, which has become a widely accepted standard for developing graphics application. This is a computer graphics case study for 3D model creation in OpenGL. It covers all features of graphics concepts likes (Transformation, Camera, Projection, and Animation).
The aim of our project is to study the effect of an

- Using a timer function.
- Uses display lists.
- Uses glut to manage windows and callbacks.

This project makes use of different OpenGL concepts such as Input and interaction, Menus and Display Lists, Transformations, Camera Movement, Coloring, Texturing, Lighting, Shading, Animation and Hidden Surface Removal.

5.1 OpenGL Functions

5.1.1 Specifying Simple Geometry

- **void glBegin (glEnum mode):**Initiates a new primitive of type mode and starts the collection of vertices.Values of mode include GL_POLYGON, GL_POINTS and GL_LINES.
- **void glEnd() :** Terminates a list of vertices.

5.1.2 Attributes

- **void glClearColor(GLclampf r, GLclampf g, GLclampf b, Glclampf a) :** Sets the present RGBA clear color used when clearing the color buffer. Variables of GL clampf floating–point numbers between 0.0 and 1.0.
- **void glColor3f(r,g,b) :** sets the RGB color to certain objects.

5.1.3 Working with the window

- **void glFlush() :** Forces any buffered OpenGL commands to execute.
- **void glutInit(int *argc,char **argv) :** Initializes GLUT. The arguments from main are passed in and can be used by the application.
- **int glutCreateWindow(char *title) :** Creates a window on the display.The string title can be used to label the window.
- **void glutInitDisplayMode(unsigned int mode) :** Requests a display with properties in mode.The value of mode is determined by logical OR of options including the color model (GLUT_RGB,GLUT_INDEX) and buffering (GLUT_SINGLE,GLUT_DOUBLE)
- **glutInitWindowSize(int width,int height) :** Specifies the initial height and width of the window in pixel.
- **void glutInitWindowPosition(int x,int y) :** Specifies the initial position of the top-left corner of the window in pixel.
- **void glutMainLoop() :** Cause the program to enter an event-processing loop.It should be the last statement in main.
- **void glutDisplayFunc(void (*func)(void)) :** Registers the display function func that is executed after the current callback returns.
- **void glutPostRedisplay() :** Requests that the display callback be executed after the current callback returns.

5.1.4 Transformations

- **void glMatrixMode(GLenum mode) :** Specifies which matrix will be affected by subsequent transformations. Mode can be GL_MODELVIEW, GL_PROJECTION or GL_TEXTURE.
- **void glLoadIdentity() :** Sets the current transformation matrix to an identity matrix.
- **void glPushMatrix() & void glPopMatrix() :** Pushes to and pops from the matrix stack corresponding to the current matrix mode.
- **void glRotate[fd](TYPE angle, TYPE dx, TYPE dy, TYPE dz) :** Alters the current matrix by a rotation of angle degrees about the axis(dx, dy, dz).
- **void glTranslate[fd](TYPE x, TYPE y, TYPE z) :** Alters the current matrix by a displacement of (x, y, z).
- **void glScale[fd](TYPE sx, TYPE sy, TYPE sz) :** Alters the current matrix by a scaling

of (sx, sy, sz).

5.15 Viewing

- **void glOrtho(GLdouble left, GLdouble right, GLdouble bottom, GLdouble top, GLdouble near, GLdouble far) :** Defines an orthographic viewing volume with all parameters measured from the centre of projection plane.

5.16 Other functions

- **glShadeModel(mode) :** To enable the flat shading we must set the shade as follows glShadeModel(GL_FLAT);
- **glutReshapeFunc(void (*func)(int width, int height))** : registers the callback function that GLUT calls on window reshape events.
- **glutKeyboardFunc(void (*func)(unsigned char key, int x, int y)** : registers the callback function that GLUT calls on a keyboard event. The parameter key is the ASCII value of the key pressed. Parameters x and y indicate the mouse position at the time of the key press.
- **glRasterPos{234}{sifd}{v}(…)** : sets the current raster position (for what comes next) in either 2, 3, or 4-D coordinates (see glVertex( )).
- **glutBitmapCharacter(void* font, char c)** : draws one bitmapped character using the indicated font.
- **glVertex{234}{sifd}{v}(…)** : specify a vertex in either 2, 3, or 4-D coordinates, using either 16 or 32 bit integers or floats or doubles, and either listing all coordinates or using an array parameter.
- **glLineStipple(int factor, ushort pattern)** : uses 16-by pattern to draw dashed or dotted lines at factor stretching.
- **glutSolidSphere(GLdouble radius,GLint slices, GLint stacks)** : render a solid sphere
- **glutSolidCube(GLdouble size) :** render a solid cube. The cube is centered at the modeling coordinates origin with sides of length size.
- **glutAddMenuEntry(char *name, int value)** - adds a menu entry to the bottom of the *current menu.*
- **glutAttachMenu(int button)** : attaches a mouse button for the *current window* to the identifier of the *current menu.*

Chapter 6

DISCUSSION and SNAPSHOTS

6.1 Discussion

A power generation project is a 2D animation of generating a power.In our project,there is no electricity production in the village prior to the execution.Once the blades of the wind turbine is rotated due to the wind,it send the energy to the generator to convert it into electrical energy. A blades can move in any direction like clockwise or anti-clockwise.A simple DC motor has a coil of wire that can rotate in a magnetic field.The current in the coil is supplied via two brushes that make moving contact with a split ring. The coil lies in a steady magnetic field. The forces exerted on the current-carrying wires create a torque on the coil.Here we are using it as a Dynamo (Generator). A dynamo is a device can convert a mechanical rotation into an electrical current. This process is done with a magnetic field and a coil. The more turns in the coil the higher the dynamo voltage. The o/p of the motor is connected tonumber of leds through a battery.If more wind is encountered ,the street light is on else power is provided only to the houses.So that it is possible to provide the power to the houses and to the roads.

6.2 Snapshots

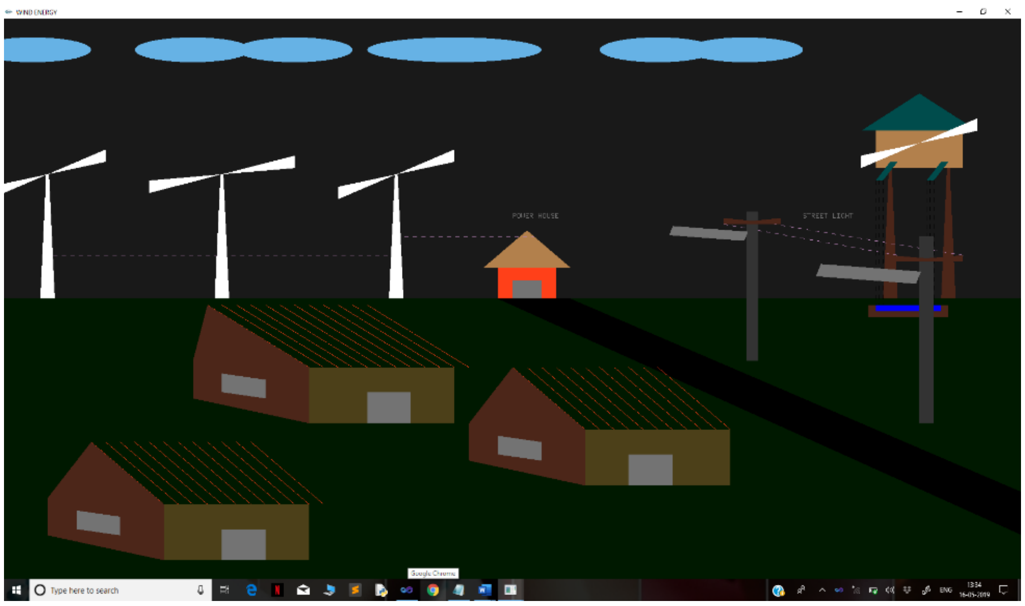


Figure 1 : A scenic view of wind turbine

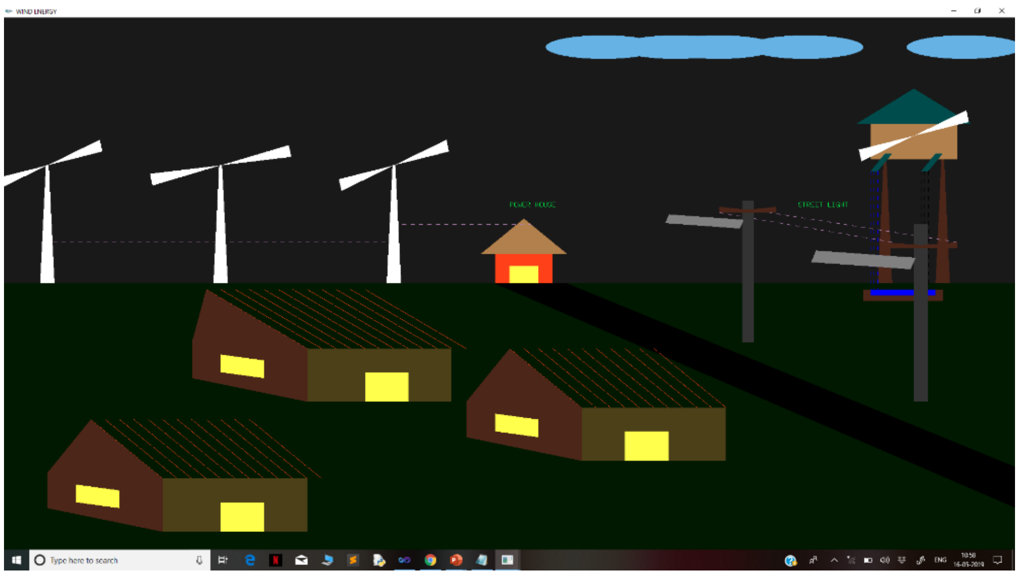


Figure 2 : Power provided to houses due to less wind

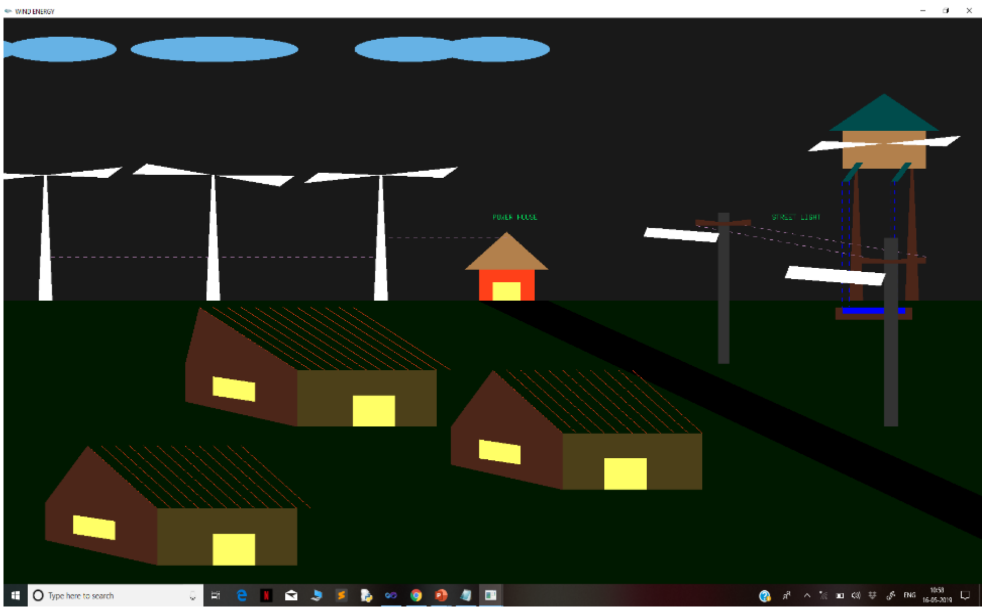


Figure 3 : Street light is on due to more wind

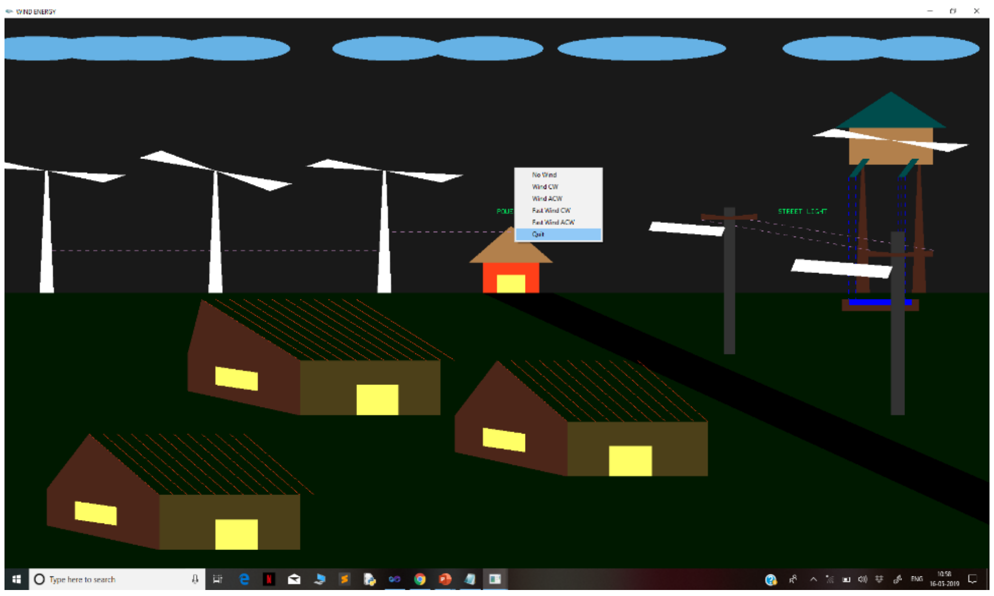


Figure 4 : Number of options for rotation of blades

Chapter 7

CONCLUSION AND FUTURE ENHANCEMENT

7.1 Conclusion

The aim of this project is study of implementation of 3D objects. It works well with the window platform. The idea behind this project can be used for matching purpose and fun purpose. In this mini-project, we have attempted to implement low level design of space shuttle. We have seen the functions performed by and code for the same. In the OpenGL, C/C++ supports enormous flexibility in the design and the use of C/C++ graphics programs.

The presence of many in-built OpenGL functions and libraries to take care of many of the functionalities reduces the burden of coding and makes the implementation simpler. The mini-project has made use of quite a few opengl functions for design of patterns, for updating of colours, position, etc.

We do hereby conclude that the natural wind power can be maneuverly used for the production of electricity generation. These are the renewable resources that are needed to be widely used for the power production which can play essential roles in the development of cities. The sea beach and the coastal zones can affably accommodate the benefits of the renewable resource cause they have the abundance of free wind power in a continuous manner .The widened use the windmills can provide free electrical power to be used in the public areas, such as street lights , parks etc.

7.2 Future Enhancement

Future work includes efficient network communication, load balancing, and image composition. In particular, networking bandwidth limitations present very significant challenges.

The project can be designed using better languages and better graphical interfaces making use of inputs from the mouse and keyboard.

Chapter 8

REFERENCES

1. Interactive Computer Graphics A Top-Down Approach with OpenGL **-**Edward Angel, 5th Edition, Addison-Wesley, 2008.

2. James D Foley, Andries Van Dam, Steven K Feiner, John F Hughes, **Computer Graphics**, Pearson Education 1997.

3. OpenGL Super Bible: Comprehensive Tutorial and Reference (5th Edition) By Richard S. Wright, Nicholas-Haemel , Graham Sellers, Benjamin Lipchak.

4. F.S. Hill Jr**.: Computer Graphics using OpenGL**, 3rd Edition, PHI, 2009.