TRIBHUVAN UNIVERSITY
INSTITUTE OF ENGINEERING
CENTRAL CAMPUS, PULCHOWK

A
REPORT
ON
FOOTBALL STADIUM MODEL RENDERING

SUBMITTED BY:

Subash Tiwari (073BCT544)
Sabal K.C (073-BCT-537)
Rabindra Regmi (073-BCT-531)

SUBMITTED TO:

Department of Electronics and Computer Engineering

6th March, 2019

त्रिभुवन
विश्वविद्यालय
UNIVERSITY ORGANISED 1956 A.D
INCORPORATED 1959 A.D
TRIBHUVAN
KATHMANDU
NEPAL

TRIBHUVAN UNIVERSITY
INSTITUTE OF ENGINEERING
CENTRAL CAMPUS, PULCHOWK

A
REPORT
ON
FOOTBALL STADIUM MODEL RENDERING

SUBMITTED BY:
Subash Tiwari (073BCT544)
Sabal K.C (073-BCT-537)
Rabindra Regmi (073-BCT-531)

SUBMITTED TO:
Department of Electronics and Computer Engineering

6th March, 2019

ACKNOWLEDGEMENT

We heartily express our gratitude to our subject teacher Mr. Anil Verma for his valuable suggestions and guidelines for the project. We would like to express our sincere gratitude to the Department of Computer and Electronics Engineering for providing us such an opportunity and motivation for the project on Computer Graphics.

Also, we would like to extend our vote of thanks to our seniors, friends and to all of them for helping us directly or indirectly during the project development session and also to those who gave us valuable suggestions.

ABSTRACT

The report is written to be submitted to the Department of Computer and Electronics Engineering, Pulchowk Campus under the course ‘Computer Graphics’ in 5th semester of Bachelor of Computer Engineering. The target of the project is to implement the basic techniques and algorithms used in modeling and transformation of a 3D object. We have also implemented Visual Surface Detection. During rendering, we have used Phong’s illumination to model the lighting effects. Hence, the football stadium model rendering was completed.

TABLE OF CONTENTS

INTRODUCTION

The core idea of the project is to use various algorithms we have studied in class to render Football Stadium. For this project, we have selected Wembley football stadium as our object. The expected outcome will include a clear visualization of the football stadium. The simulation will be done in 3D that will give a clear picture to the user. The approaches we will be using in this project are visible surface detection, projection, rendering, 3D transformation, 3D object representation and many more.

The project football stadium visualization aims to give a comprehensive overview of the field of computer graphics. The overall focus will be on modeling and rendering. It will cover the basic concepts, mathematical foundations, fundamental theory and algorithms, software techniques, hardware and system issues, and application examples of computer graphics.

Brief Overview

Computer Graphics are the graphics created using computers and the representation of image data by a computer using graphic hardware and software. Graphics is a vital part of software era. Computer Graphics has remained one of the most exciting and rapidly growing area in the computer fields. Computer graphics is responsible for displaying art and image data effectively and meaningfully to the user. It is also used for processing image data received from the physical world. The interaction and understanding of computers and interpretation of data has been made easier because of computer graphics. Many tools have been developed to visualize data. Computer generated imagery can be categorized into several different types: two dimensional (2D), three dimensional (3D), and animated graphics. As technology has improved, 3D computer graphics have become more common, but 2D computer graphics are still widely used. Computer graphics has emerged as a sub-field of computer science which studies methods for digitally synthesizing and manipulating visual content. Over the past decade, other specialized fields have been developed like information visualization, and scientific visualization more concerned with "the visualization of three dimensional phenomena (architectural, meteorological, medical, biological, etc.), where the emphasis is on realistic renderings of volumes, surfaces, illumination sources, and so forth, perhaps with a dynamic (time) component".
Talking about the project its main focus is on the creation and transformation of 3D prototype of a stadium.

Features

1. Rotation of the model using 3d Transformations for different views
2. Zooming in/out moving camera and projection plane parallel
3. Rendering with color of calculated light intensity in each seen pixels used in building the model.
4. Front, top and opposite view of the stadium
5. Movement of light in the stadium
6. Day and night mode

OBJECTIVES

The main objective of this project is to become familiar with creating 3D and 2D graphics objects in computer. Creating graphics in computer require various graphics routines to be performed and our objective is to gain knowledge in using such routines and to list out, they are:

- To acquire knowledge in 2D and 3D Geometric Transformation including basic transformations such as translation, rotation, scaling etc.
- To know about 3 - dimensional object representation using polygon tables including vertex table, edge table and surface table.
- To implement the different colors in 3D objects using various fill algorithms.
- To apply algorithms for visible surface detection and hidden line detection (Z – buffer).
- To apply algorithms on Illumination Models and Surface Rendering Methods such as Ambient light, Diffuse Reflection, Specular Reflection and Polygon rendering methods as Gouraud Intensity Shading.

MOTIVATION

The motivation for doing this project was primarily an interest in undertaking a challenging project in an interesting area of computer graphics research. The opportunity to learn about the principals involved in computer graphics and its practical application in a system was appealing.

LITERATURE REVIEW

Stadium Visualization provides a way to visualize a stadium in a digital screen. Football Stadium Model has been chosen as our project for Computer graphics. The football stadium model will include in it creation of model, adding colors, lighting effects, shading, visible surface determination, transformation techniques that help in development of a vehicle along with the surrounding thus giving us a brief idea of implementation of various techniques used in graphics.

Computer graphics has been evolving and developing more and more over time. With increasing time the features and results of computer graphics have been more efficient than before. Since the very beginning, different projects of computer graphics are being developed which used different algorithms for their implementation.
3D model of a hotel interior is the latest computer graphics project where almost all the features are included. 3D uses in different areas of our life, in cinematography, in science, in architectural field, etc. When dealing with 3D graphics, the following five main steps which were used to obtain the final product are:

1. Modeling - creating objects that will be on stage
2. Texturing - determination of surface properties of objects to simulate the properties of real objects (color, texture, transparency, brightness, etc.)
3. Lighting - adding and placing light sources in the same way as it is done in the theater, for example; at this stage, also, setting shadows
4. The animation - changing the position of the object or its properties (color, transparency, etc.)
5. Visualization - creation of the final image or animation

METHODOLOGY

The steps that were taken to complete the project are listed below with their brief introductions.
Eclipse IDE and Java programming language was used.

- ➢ Consideration of triangle as the basic unit of 3D model and pixel as the basic unit of each quadrilateral.
- ➢ Orthogonal 2D projection from 3D model.
- ➢ Polyfill Scanline algorithm is used to render each quadrilateral.
- ➢ Z-buffer is attempted for VSD.
- ➢ Gouraud’s shading is used along with illumination model for lighting.

Eclipse IDE, Java and Swing and Blender

Eclipse is an integrated development environment used in computer programming, and is the most widely used Java IDE. It contains a base workspace and an extensible plug-in system for customizing the environment. Java is a general-purpose computer-programming language that is concurrent, class-based, object-oriented, and specifically designed to have as few implementation dependencies as possible. Swing is a GUI widget toolkit for Java. It is part of Oracle's Java Foundation Classes – an API for providing a graphical user interface for Java programs. Swing was developed to provide a more sophisticated set of GUI components than the earlier Abstract Window Toolkit.

We have used JFrame of Swing for displaying the window. Swing has also been used for the GUI elements of the project. Canvas has been used for drawing the elements.

Viewing Pipeline

Modeling World Viewing Coordinates Coordinates Coordinates
Viewing Transformation
Modeling Transformation
Device Projection
Projection Transformation
Workstation Transformation
Coordinates Coordinates

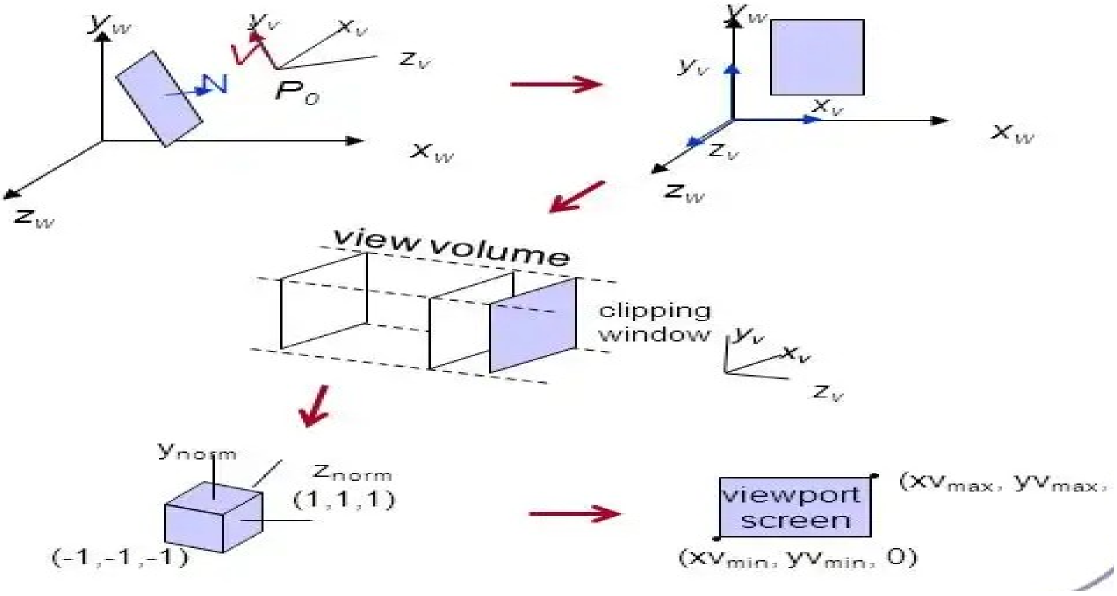


Fig 4.1. Three Dimensional transformation Pipeline

Generating a view of a three-dimensional scene is somewhat analogous to the processes involved in taking a photograph but we have more flexibility and many more options for generating views of a scene with a graphics package than with a camera. The figure above shows the general processing steps implemented for 3d modeling of the scene and converting the world-coordinates to device coordinates. First, the 3d object was modeled by generating coordinates manually. For modeling the structure, we made the design of the stadium using Blender and then exported in object format through which we got the coordinates to be manipulated by our code. And thus, world coordinates were obtained. Then, the world coordinates were converted to viewing coordinates. Next, perspective projection operations were performed to convert the viewing-coordinate to coordinate positions on the projection plane. Then visible surface were identified using depth buffer and were rendered using Goraud’s Shading Model.

3D Translation

A point is translated from position P=(x,y,z) to position P’ = (x’,y’,z’) with the following matrix:

$$\begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

or P’ = T.P

Parameters tx , ty ,tz specify translation distances for the coordinate directions x ,y and z. This matrix representation is equivalent to three equations:
x’ = x + tx y’ = y + ty z’ = z + tz

Perspective Projection

Perspective view is the real view as perceived by an eye. For real 3d modeling of any object, perspective projection algorithm is implemented to convert 3d world coordinates to 2d screen coordinates. For simplicity, camera/eye is aligned perpendicular to the XY-plane and the projection plane is parallel to XY-plane. Each vertex of the object in 3d is projected onto the projection plane and 2d projected coordinates is calculated. Then, necessary viewport transformations are applied in order to obtain the 2d screen coordinates. The viewing system used is **Right Hand Viewing System.**

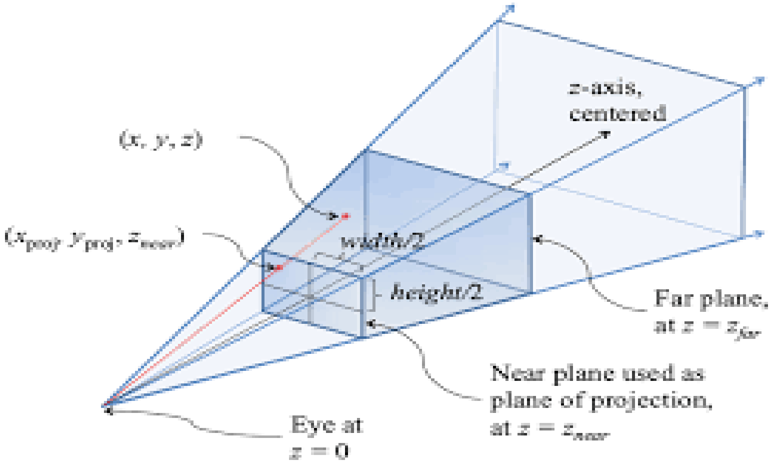


Fig 5.1. Perspective Projection

Depth Buffer

There are various methods for detecting the visible surfaces in a 3d scene. Some methods eliminate hidden surfaces and some identify the visible surfaces to be drawn. Visible Surface Detection methods can broadly be classified into two types namely: Object-Space Method and Image Space Method. Depth Buffer or Z-buffer is an Image Space Method. Depth buffer makes use of two buffer areas namely – Depth buffer and Refresh buffer. These buffers are used to store information of depth and color intensity of each pixel position (x,y) on the screen.

Algorithm

Step-1 − Set the buffer values −

Depthbuffer (x, y) = 0

Framebuffer (x, y) = background color

Step-2 − Process each polygon (One at a time)

For each projected (x, y) pixel position of a polygon, calculate depth z.

If Z > depthbuffer (x, y)

Compute surface color,

set depthbuffer (x, y) = z,

framebuffer (x, y) = surfacecolor (x, y)

For each polygon, the surface can be expressed in the form of

$Ax + By + Cz + D = 0$

The coefficients of equation of plane i.e. A, B, C, D is calculated using the following formula

$A = y1(z2 - z3) + y2(z3 - z1) + y3(z1 - z2)$

$B = z1(x2 - x3) + z2(x3 - x1) + z3(x1 - x2)$

$C = x1(y2 - y3) + x2(y3 - y1) + x3(y1 - y2)$

$D = -x1(y2z3 - y3z2) – x2(y3z1 - y1z3) – x3(y1z2 - y2z1)$

For each pixel on the polygon surface, respective 3d position of the pixel is calculated by projecting the point backwards to the 3d object. After the 3d point is calculated, the depth value of that point is calculated.

Goraud's Lighting Model

The second shading model, Gouraud shading, computes an intensity for each vertex and then interpolates the computed intensities across the polygons. Gouraud shading performs a bi-linear interpolation of the intensities down and then across scan lines. It thus eliminates the sharp changes at polygon boundaries.

The algorithm is as follows:

Compute a normal N for each vertex of the polygon.

From N compute an intensity I for each vertex of the polygon.

From bi-linear interpolation compute an intensity Ii for each pixel.

Paint pixel to shade corresponding to Ii.

3D Rotation

Rotation in 3d is performed about arbitrary axes parallel to coordinate axes passing through the centre of the structure. The whole structure is rotated in three directions by translating the arbitrary axes to the coordinate axes. The formulae used for rotation are:

Rotation about x axis:

y' = y*cos (xangle) - z*sin (xangle)

z' = y*sin (xangle) + z*cos (xangle)

Rotation about y axis:

x' = x*cos (yangle) + z*sin (yangle)

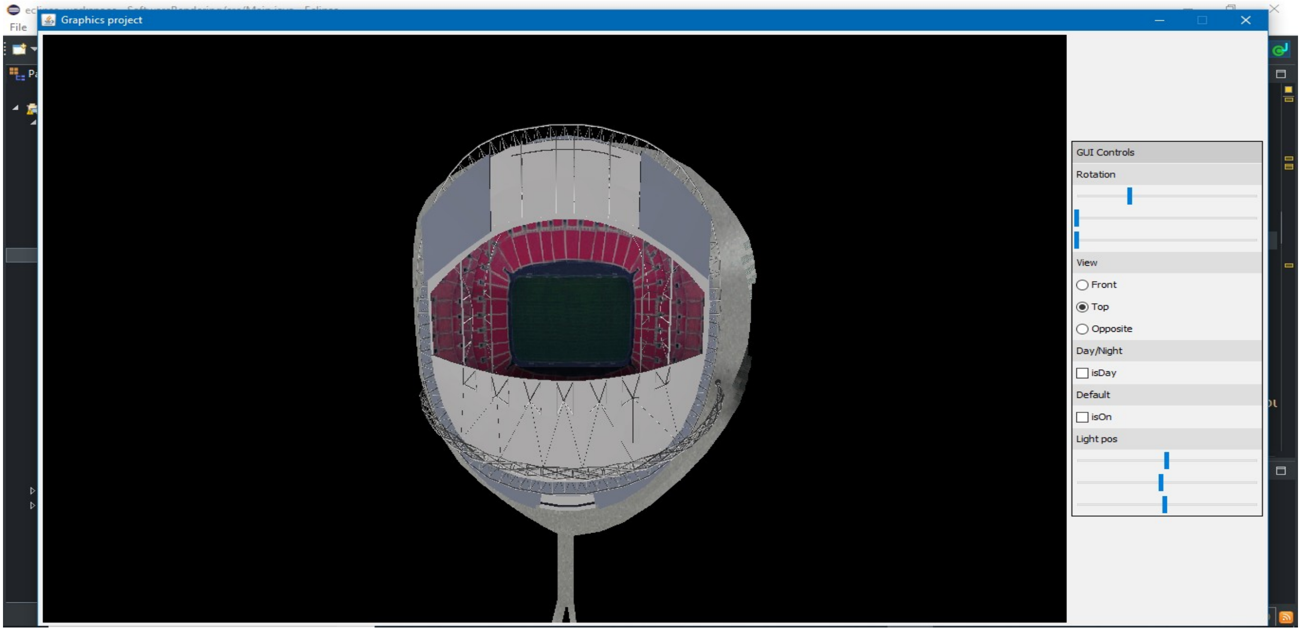
z' = -y*sin (yangle) + z*cos (yangle)

Rotation about z axis:

x' = x*cos (zangle) + y*sin (zangle)

y' = x*sin (zangle) - y*cos (zangle)

OUTPUTS



GUI Controls

Rotation

View

○ Front

● Top

○ Opposite

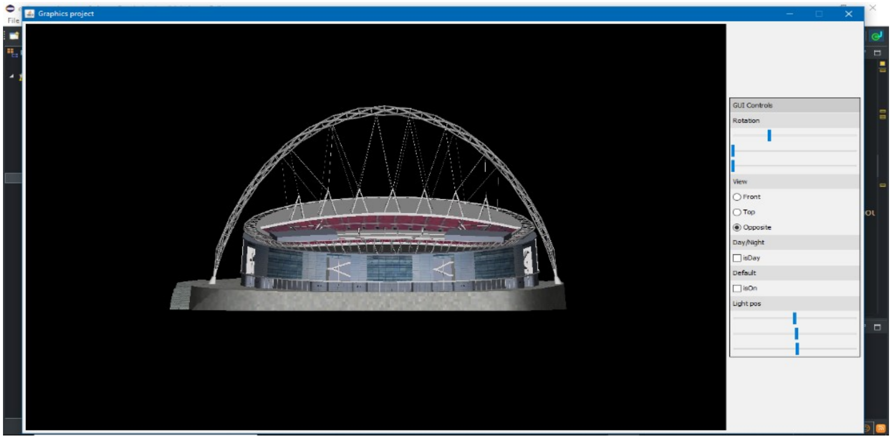
Day/Night

□ isDay

Default

□ isOn

Light pos



GUI Controls

Rotation

View

○ Front

○ Top

● Opposite

Day/Night

□ isDay

Default

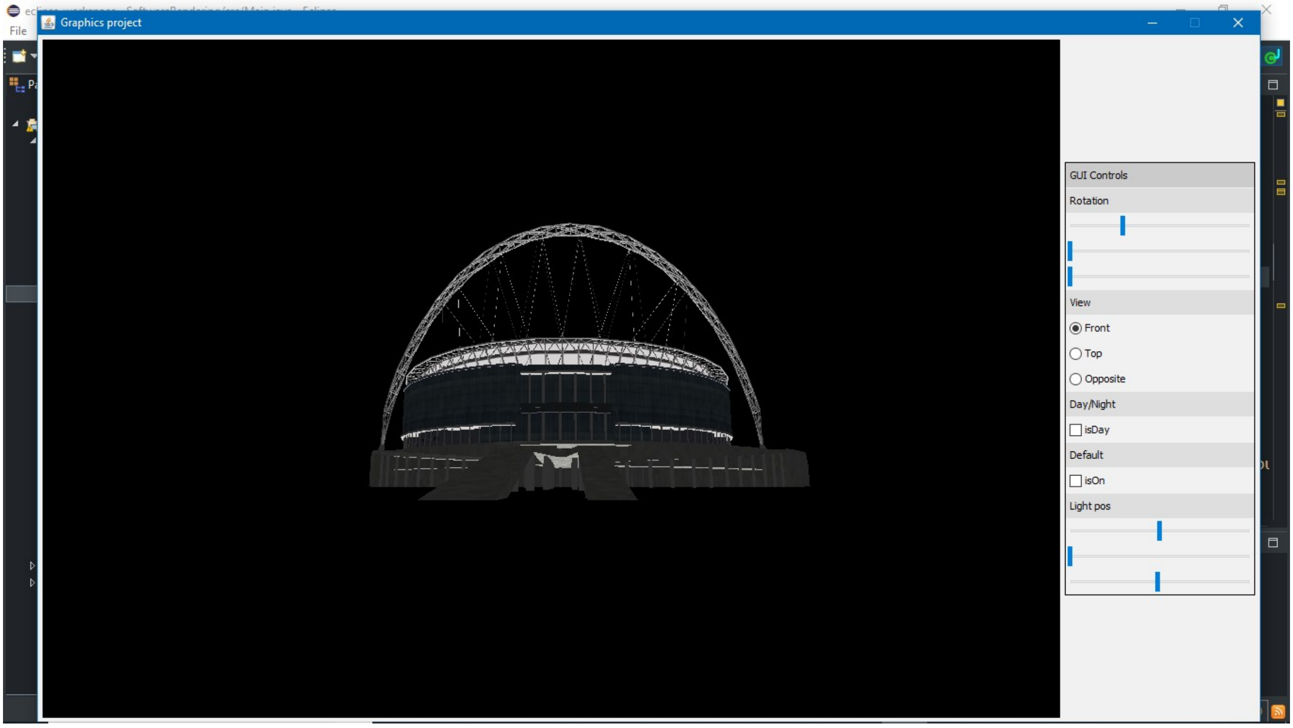
□ isOn

Light pos

LIMITATIONS

The project seems to be quite complete but there are many things that can to be improved. Following are some of the limitations of the project:

- Optimizations can be made to the rendering process. Phong shading computations were very slow and gouraud shading had to be used.
- The coloring of the model is same for every face.
- The system is key controlled and the User Interface(UI) is very minimalistic.
- Only opaque objects are modeled.



View

◉ Front

○ Top

○ Opposite

Day/Night

□ isDay

Default

□ isOn

FUTURE ENHANCEMENTS

- The rendering process can be enhanced by utilizing external GPUs, multiple threads. Thus, can support complex models that depict real visualization.
- Multiple models can be rendered in parallel.
- Multiple light source can be added along with various other properties such as textures, bump maps,etc
- The coloring property and shadows can be added to make the model more real and sophisticated.
- The User Interface can be improved with mouse facilities, buttons and drag and drop facilities.
- Various illumination techniques and backface detection algorithms can be implemented to consider transparent, translucent surfaces.
- The system can be improved to support Web Rendering, mobile rendering and other devices.

CONCLUSION

The project helped us to implement the theoretical algorithms of the course in a real rendering application. In this project we gained sufficient amount of knowledge related to Computer Graphics that can also be applied to other graphics related projects. Also the application of computer graphics principle to develop a system useful in real life was made. We also learned to work as a team and learned the optimization of system, as the field of computer graphics requires extensive processing.

We also learned to know our limitations which can be very useful for future projects and systems. The model of visualization thus developed can be used by traffic departments to visualize traffic at various areas if the project can be improved to some extent.

REFERENCES

1. https://docs.blender.org/manual/en/dev/

2. https://docs.oracle.com/javase/tutorial/uiswing/components/index.html

3. https://help.eclipse.org/2018-12/index.jsp

4. Real-Time Rendering, Third Edition 3rd Edition by Tomas Akenine-Moller, Eric Haines, Naty Hoffman

5. Computer Graphics, Donald Hearn, M.Pauline Baker
6. Youtube, thebennybox: software rendering