TRIBHUVAN UNIVERSITY

Institute Of Engineering

CENTRAL CAMPUS

PULCHOWK

COMPUTER GRAPHICS
PROJECT REPORT ON
3D VIEW OF SHANTI STUPA

Submitted To:

Dr. Basanta Joshi

Department of Electronics and Computer Engineering

Submitted by:

Prakash Dhakal (074BEX423)

Raj Adhikari (074BEX433)

Sunney Sharma (074BEX449)

Shashank Subedi (074BEX450)

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to our teacher Dr. Basanta Joshi for giving us this platform to convert our theoretical knowledge into practical form. This project on computer graphics which is taught to us by our Basanta sir to whom we are immensely grateful.

We would like to thank our lab teacher who assisted us to complete this project by providing us all the necessary ideas and help that we needed.
Last but not least, we would also like to acknowledge with much appreciation to the department of electronics and computer engineering for keeping computer graphics subject in our syllabus.

Finally, special thanks to our every friends and family members for their supports and assistance.

Yours' Sincerely

Prakash Dhakal (074BEX423)
Raj Adhikari (074BEX433)
Sunney Sharma (074BEX449)
Shashank Subedi (074BEX450)

Table of Contents

1. Introduction

Computer graphics are pictures and movies created using computers. Usually, the term refers to computer-generated image data created with the help from specialized graphical hardware and software. To display a picture of any size on a computer screen is a difficult process. Computer graphics are used to simplify this process. Various algorithms and techniques are used to generate graphics in computers. Although computer graphics is a vast field that encompasses almost any graphical aspect, we are mainly interested in the generation of images of 3-dimensional scenes. So, for the project we selected the 3D view of Shanti stupa of Pokhara.

So, designing the 3D model of one simple table and using the different ideas of computer graphics like rotation, scaling, surface rendering, and detecting the visible surfaces has covered most of the computer graphics concept. Some of the computer graphics concepts used in this project is described in following section.

1.1. Existing System

3D modeling is usually found in the architectural field. Many 3d rendering softwares like Autodesk Maya, Autodesk 3dsMax, Blender etc. are used for realistic modeling of the 3d structures. Among these softwares, Blender was be used for our project.

2. Objectives

The main objectives of the project are to fulfill the third year course of Computer Graphics and to implement our knowledge practically. Other objectives are:

- To design the 3D model of Shanti stupa of Pokhara.
- To implement the features of computer graphics learnt into the project.
- To implement the 3D rotational, scaling effects.
- To implement the color filling and lightning effects.

3. Methodology

- Visual Studio and C++ language (Opengl) was used.
- Perspective 2D projection from 3D model.
- Polyfill Scanline algorithm was used to render each quadrilateral.
- Z-buffer was attempted for VSD.
- Phong's Illumination model was used for lighting.

4. Viewing Pipeline

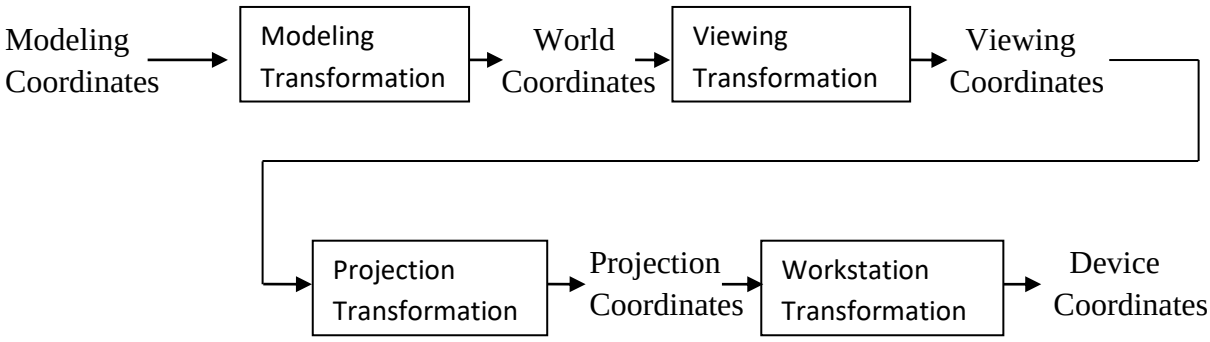


Fig : Three Dimensional transformation Pipeline

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 Shanti Stupa and converting the world-coordinates to device coordinates. First, the 3d object was modeled by generating coordinates manually. For modeling the structure of Shanti Stupa, blender was used. Since Shanti Stupa is symmetrical about quadrants, the coordinates for only 1st quadrant was modeled and coordinates for remaining quadrants were obtained via reflection about planes. 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 Phong's Shading Model.

5. Computer Graphics Algorithms Used:

5.1. Perspective Projection

In perspective projection, the distance from the center of projection to project plane is finite and the size of the object varies inversely with distance which looks more realistic.

The distance and angles are not preserved and parallel lines do not remain parallel. Instead, they all converge at a single point called center of projection or projection reference point. There are 3 types of perspective projections which are shown in the following chart.

One point perspective projection is simple to draw.

Two point perspective projection gives better impression of depth.

Three point perspective projection is most difficult to draw.

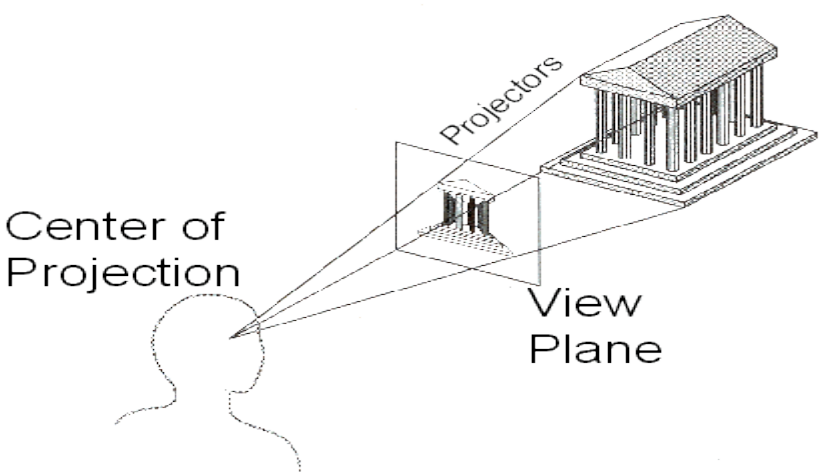


Fig: Perspective Projection

5.2. **Z BUFFER**

To accurately represent a complex model, it is imperative that those surfaces normally invisible from a certain point, be also invisible in the computer generated image. This problem normally called the visible surface problem or the hidden-surface problem and has been the fundamental research problem in computer graphics over the past 20 years. The effects of this algorithm are easily seen, even in relatively simple pictures. If we consider the following illustration, we can obtain much more information about the relative positions of the objects by using the right hand figure, rather than using the left hand one. Of all algorithms for visible surface determination, the depth-buffer is perhaps the simplest, and is the most widely used. For each pixel on the display, we keep a record of the depth of the object in the scene that is closest to the viewer, plus a record of the intensity that should be displayed to show the object. When a new polygon is to be processed, a z-value and intensity value are calculated for each pixel that lies within the boundary of the polygon. If the z-value at a pixel indicates that the polygon is closer to the viewer than the z-value in the z-buffer, the z-value and the intensity values recorded in the buffers are replaced by the polygon's values. After processing all polygons, the resulting intensity buffer can be displayed. The z-buffer, from which this algorithm derives its name, is an n × n array for which the (i, j)th element corresponds to the (i, j)th pixel. This array holds the image-space z value of the currently visible object at the pixel. There is also another n × n array whose elements correspond to the color that is to be assigned to the pixel.

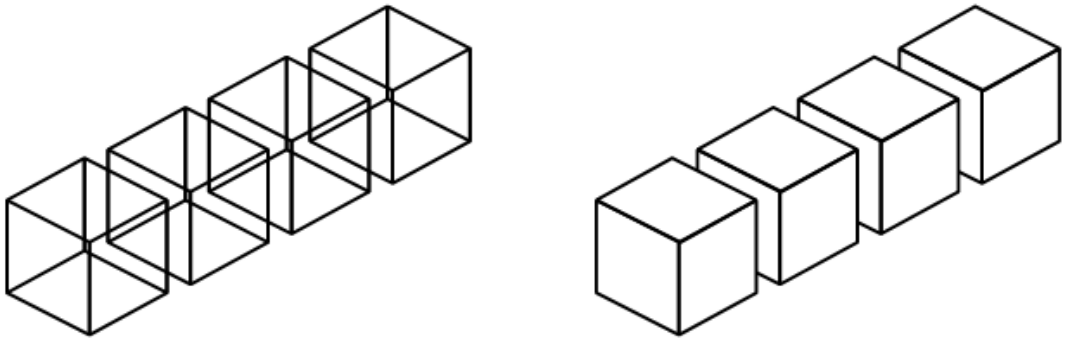


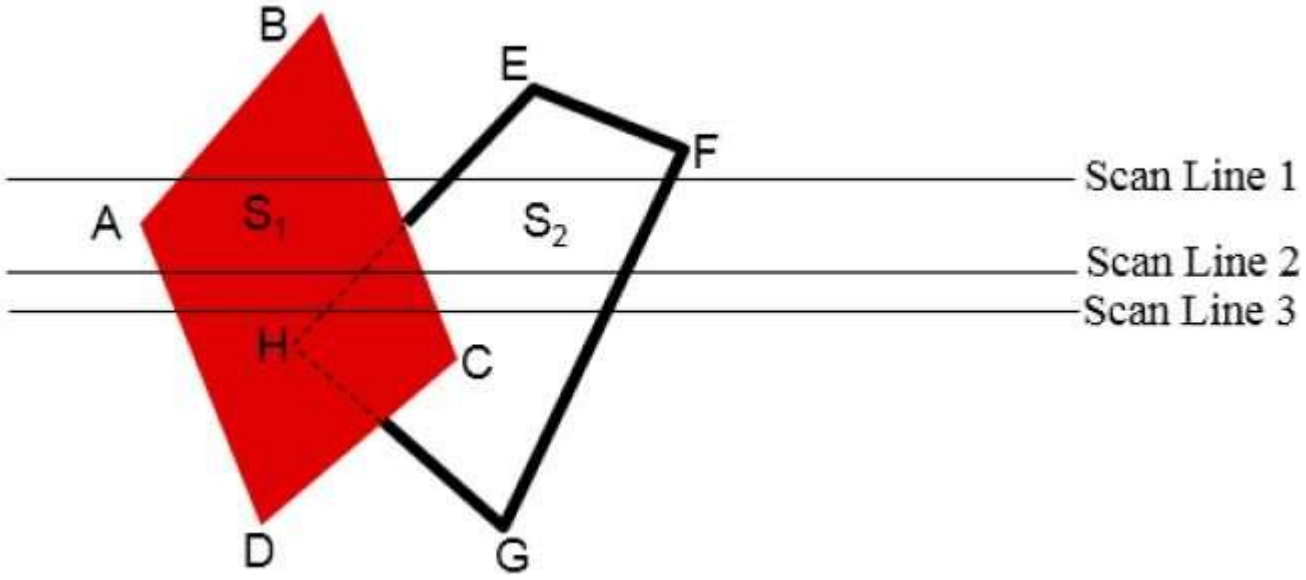
Fig. Z (Depth) Buffer

5.3. Scan-Line Method

It is an image-space method to identify visible surface. This method has a depth information for only single scan-line. In order to require one scan-line of depth values, we must group and process all polygons intersecting a given scan-line at the same time before processing the next scan-line. Two important tables,edge table and polygon table, are maintained for this.

The Edge Table − It contains coordinate endpoints of each line in the scene, the inverse slope of each line, and pointers into the polygon table to connect edges to surfaces.

The Polygon Table − It contains the plane coefficients, surface material properties, other surface data, and may be pointers to the edge table.



To facilitate the search for surfaces crossing a given scan-line, an active list of edges is formed. The active list stores only those edges that cross the scan-line in order of increasing x. Also a flag is set for each surface to indicate whether a position along a scan-line is either inside or outside the surface.

Pixel positions across each scan-line are processed from left to right. At the left intersection with a surface, the surface flag is turned on and at the right, the flag is turned off. You only need to perform depth calculations when multiple surfaces have their flags turned on at a certain scan-line position.

5.4. TRANSLATION

Translation can best be described as linear change in position. This change can be represented by a delta vector [Tx, Ty, Tz], where Tx represents the change in the object's x position, Ty represents the change in its y position, and Tz represents its change in z position.

Translation is defined by a delta vector v = (Tx, Ty, Tz) and by a transformation matrix

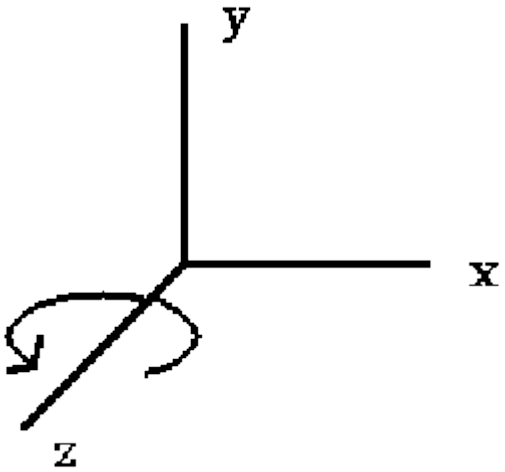
$$Ap = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ Tx & Ty & Tz & 1 \end{vmatrix}$$

5.5. 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:

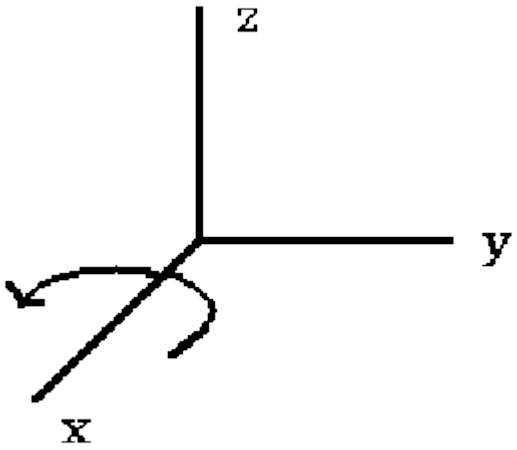
5.5.1. Z-Axis Rotation

Z-axis rotation is identical to the 2D case:



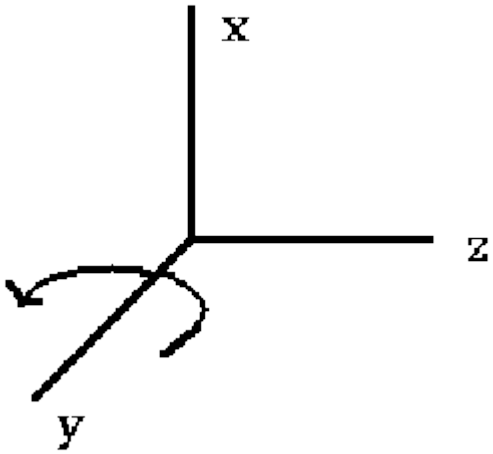
$$x' = x*\cos q - y*\sin q$$
$$y' = x*\sin q + y*\cos q$$
$$z' = z$$

5.5.2. **X-Axis Rotation**



y' = y*cos q - z*sin q
z' = y*sin q + z*cos q
x' = x

5.5.3. Y-Axis Rotation



z' = z*cos q - x*sin q
x' = z*sin q + x*cos q
y' = y

5.6. SCALING

We can change the size of an object using scaling transformation. In the scaling process, we either expand or compress the dimensions of the object. Scaling can be achieved by multiplying the original coordinates of the object with the scaling factor to get the desired result. In 3D scaling operation, three coordinates are used. Let us assume that the original coordinates are (X, Y, Z), scaling factors are (SX, SY, SZ) respectively, and the produced coordinates are (X', Y', Z').

Mathematically, $X' = X. SX$; $Y'=Y. SY$; $Z'=Z. SZ$;

$$S = \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

5.7. PHONG SHADING

In Phong's model, the light reflected off an object is the sum of three components, ambient, diffuse and specular, based on the reflectance properties of its surface. A point light source is considered and light intensity of RGB is calculated for each pixels. Ambient part of the light is scattered equally in all directions from the surface of object. In addition to ambient part of light source, there is presumed to be a global ambient light as well. Diffuse part of the light travels as coherent beam from source towards object and is scattered equally in all directions. The direction of the light source does matter in case of diffuse reflectance. The specular part of the light also travels in a coherent beam from source to object and then again is reflected in a coherent beam. So, both the direction of the light source and viewer matter in case of specular reflectance. Diffuse models soft light with little focus while specular models hard light with a focus.

3*3 light property matrix is:

$$\begin{bmatrix} L_{amb,\,R} & L_{amb,\,G} & L_{amb,\,B} \\ L_{dif,\,R} & L_{dif,\,G} & L_{dif,\,B} \\ L_{spec,\,R} & L_{spec,\,G} & L_{spec,\,B} \end{bmatrix}$$

The global ambient vector:

$[globAmb_{\,R}\ globAmb_{\,G}\ \ globAmb_{\,B}]$

3*3 material properties matrix is:

$$\begin{bmatrix} V_{amb,\,R} & V_{amb,\,G} & V_{amb,\,B} \\ V_{dif,\,R} & V_{dif,\,G} & V_{dif,\,B} \\ V_{spec,\,R} & V_{spec,\,G} & V_{spec,\,B} \end{bmatrix}$$

Lighting equation that gives the color intensity V_X at each pixels, where X may be any of RGB is:

$$V_X = globAmb_{\,X} * V_{amb,\,X} + L_{amb,\,X} * V_{amb,\,X} + |L \,.\, N| * L_{dif,\,X} * V_{dif,\,X}$$

$$+ |S \,.\, N| * L_{spec,\,X} * V_{spec,\,X}$$

Here, L is the light to pixel vector, N is the outward normal and S is the halfway vector. The dot product of the two vectors gives cosine of angle between them.

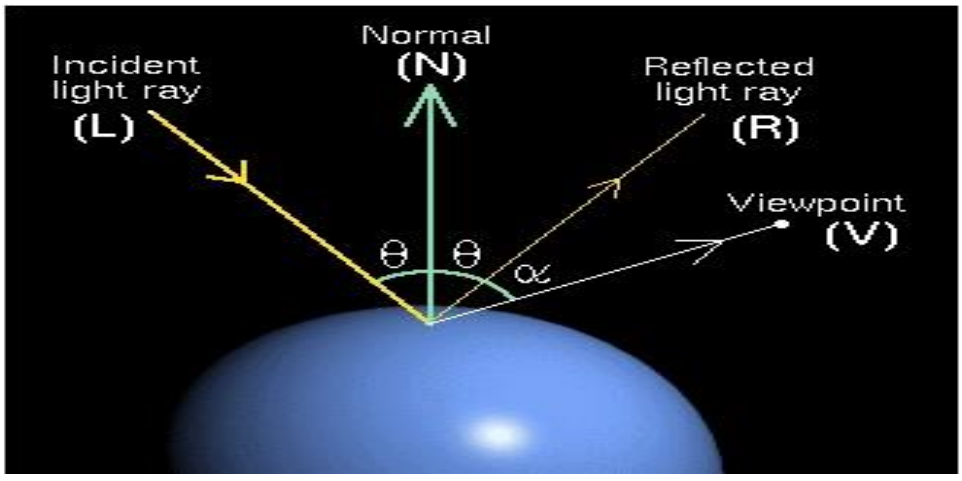
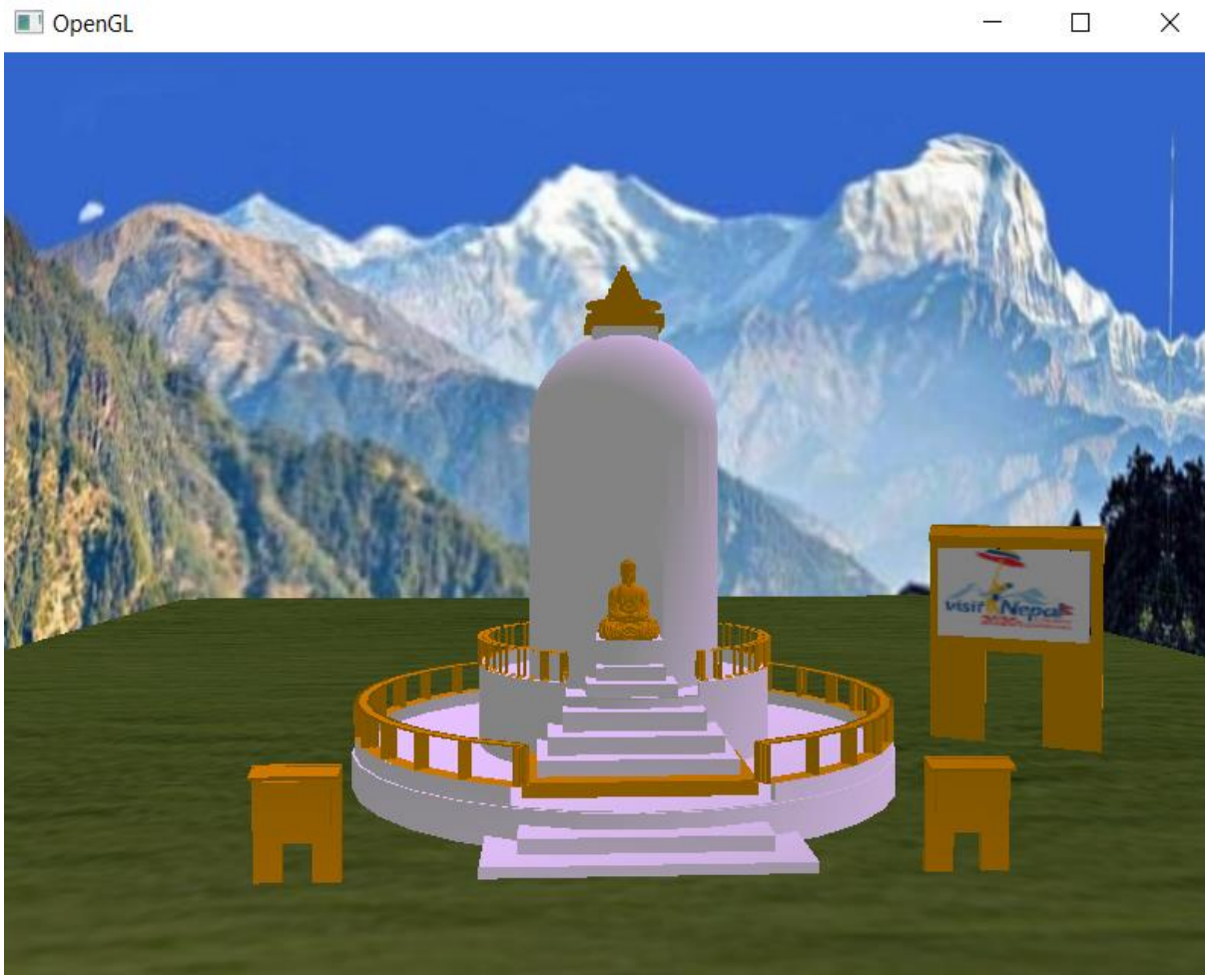


Fig: Light reflection

6. Output:



7. Conclusion

After completing this project we now became familiar with some basic concepts use in computer graphics. This project is the 3D view of Shanti Stupa. In this project we have used the basic ideas of computer graphics like, rotation, scaling, surface rendering, z buffer method to find the depth of the surfaces of the object. We used rotation about x-axis, y-axis and z-axis, so that our object can be rotate about all three major axis. For zooming in and zooming out, we used scaling concepts. And to hide one surface with another surface we used Z buffer technique. By comparing the value of the z coordinate of different surfaces with same value of the x and y, we can show one surface by hiding the other surfaces behind it. For the purpose of surface rendering we used the Phong shading method. Since it applies illuminates a surface using the unit normal vector of a point it is better than other method of illumination technique. Hence by applying these basic concepts of computer graphics we completed a small project which was very useful.

8. References:

- Hearn, Donald **"Computer Graphics C Version"**, 2nd Edition, Singapore: Pearson Education Pte. Ltd., 2004
- Gotsman, Elber **"Polygon Filling Drawing Geometry on Raster Displays"**, 9-12