Mathematics behind Engineering Drawing

COP290 Project

Arpan Mangal
Deepanshu Jindal

January 23, 2018

Contents

Chapter 1

Introduction

Engineering Drawing is a vital part of Mechanical Engineering. Mechanical students are given extensive training in this subject which consists of taking orthographic projections of a given 3 Dimensional object and retrieving the object from its 2 Dimensional orthographic projections.

The question is can this monotonous job be automated with the help of a computer. This report deals with developing a mathematical model to accomplish this task of automating the subject of Engineering Drawing.

1.1 Terminology and conventions used for describing 3D Objects

This section deals with the terminology used in this book for describing the object and its orthographic projections. We will work with the third angle of projection.

Consider a simple three dimensional object as in Fig. 1.1 below:

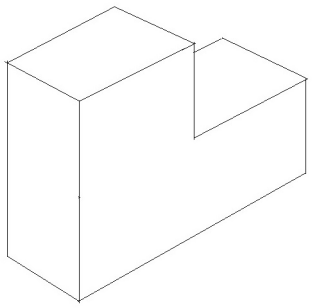


Figure 1.1: A 3-Dimensional object

We will see it in third angle of projection. For that we will assume

that orientation of coordinate axes as follows:

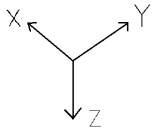


Figure 1.2: Orientation of axes

This is to say that the front view will lie on the Y-Z plane, top view on the X-Y plane and the side view on the Z-X plane.

The orthographic projections will then be plotted onto the three planes as below:

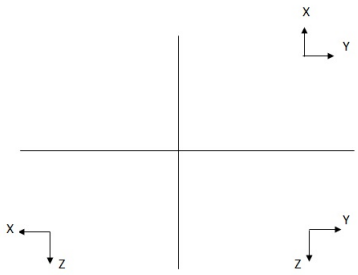


Figure 1.3: Terminology for specifying axes labels on the 3 planes

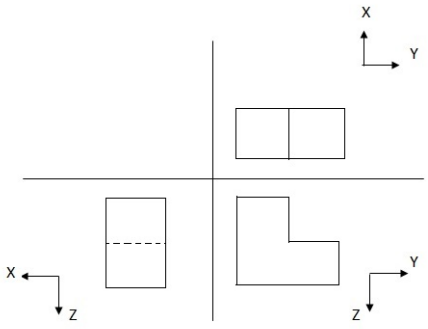


Figure 1.4: Actual orthographic projections of the object

1.2 Package Features

This software will consist of mainly two features:

1. Constructing orthographic projections from the given 3-Dimensional object.

2. Re-construction of the 3-Dimensional object from given orthographic projections and constructing isometric view of the 3-D reconstruction.

Each part has its own expected input and output as described below:

1.2.1 Orthographic Projections from 3-D

We will take a 3-dimensional graph, $G(V, E)$ as a set of 3 dimensional points $\boldsymbol{X_i}$'s as input along with the set of corresponding labelled edges $\boldsymbol{L_i}$'s between them.

The output shall consist of the three orthographic projections on the three orthographic planes as shown in Fig. 1.3. The user can also specify an arbitrary plane along which we can construct the auxiliary view of the object.

1.2.2 Re-construction of 3-D object from its orthographic projections

We will take the three orthographic projections from the user as shown in Fig. 1.3. Each projection will be set of points $\boldsymbol{X_i}$'s along with the connecting lines. Each point will consist of a label which will remain same across the three projections for the corresponding point.

Output shall be a 3-D plot and an isometric projection of the reconstructed object.

Chapter 2

Generating Projections of 3D Models

This chapter deals with projecting 3-dimensional models and figures to a 2-dimensional surface. This includes projection on the three Cartesian planes as well as any given cutting plane. We will take 3^{rd} angle of projection.

We will proceed by projecting points, lines and finally complete models which consists of lines and points.

2.1 Projection of a single point

This section describes projecting single point in 3D to a 2D surface. We will consider that we are given a point, $\boldsymbol{P}$ in the form of a vector:

$$P = \begin{pmatrix} X_i \\ Y_i \\ Z_i \end{pmatrix} \tag{2.1}$$

We will show that projection of the point on any plane can be obtained simply by multiplying suitably chosen matrix $\boldsymbol{M}$ with the vector $\boldsymbol{P}$.

2.1.1 Projecting on one of the Cartesian planes

We have three Cartesian planes: X-Y, Y-Z and Z-X. We will project the point in each of the three planes.

Consider the projection of $\boldsymbol{P}$ on X-Y plane. It would simply be equivalent to dropping the z-coordinate and keeping the x and y coordinates intact, i.e. the projection $\boldsymbol{p_{xy}}$ of $\boldsymbol{P}$ would be:

$$p_{xy} = \begin{pmatrix} X_i \\ Y_i \end{pmatrix} \tag{2.2}$$

This could be written as:

$$p_{xy} = \begin{pmatrix} x_i \\ y_i \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} X_i \\ Y_i \\ Z_i \end{pmatrix} \tag{2.3}$$

or,

$$p_{xy} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} P \tag{2.4}$$

Similarly,

$$p_{yz} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} P \tag{2.5}$$

and,

$$p_{zx} = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} P \tag{2.6}$$

Equations 2.4 through 2.6 describe the projection of point $\boldsymbol{P}$ on the three Cartesian planes.

2.1.2 Projection on an arbitrary plane surface

Any arbitrary plane surface, $\boldsymbol{\Pi}$, can be represented by the position of its origin, $\boldsymbol{O'}$ and the direction of its axes $\boldsymbol{X'}$, $\boldsymbol{Y'}$, and $\boldsymbol{Z'}$ w.r.t. the original axes. (Here $\boldsymbol{Z'}$ is the normal to the plane $\boldsymbol{\Pi}$, and $\boldsymbol{X'}$ & $\boldsymbol{Y'}$ form the plane surface)

First, lets consider translation only i.e. shifting of origin with fixed direction of coordinate axes. Suppose, the shifted origin is at point $(\boldsymbol{X_o}, \boldsymbol{Y_o}, \boldsymbol{Z_o})$. Then the new coordinated for point $\boldsymbol{P}$ shall be

$$P' = \begin{pmatrix} X_i \\ Y_i \\ Z_i \end{pmatrix} - \begin{pmatrix} X_o \\ Y_o \\ Z_o \end{pmatrix} = \begin{pmatrix} X_i - X_o \\ Y_i - Y_o \\ Z_i - Z_o \end{pmatrix} = P - O' \tag{2.7}$$

Now, consider that the coordinate axes are rotated without translation.
Let the rotated coordinated axes be $\boldsymbol{X'}$, $\boldsymbol{Y'}$, and $\boldsymbol{Z'}$
Let the unit vectors corresponding to $\boldsymbol{X'}$, $\boldsymbol{Y'}$, and $\boldsymbol{Z'}$ (w.r.t. old Cartesian coordinates $\boldsymbol{X}$, $\boldsymbol{Y}$, and $\boldsymbol{Z}$)respectively be

$$\begin{pmatrix} \alpha_1 \\ \beta_1 \\ \gamma_1 \end{pmatrix}, \begin{pmatrix} \alpha_2 \\ \beta_2 \\ \gamma_2 \end{pmatrix}, \begin{pmatrix} \alpha_3 \\ \beta_3 \\ \gamma_3 \end{pmatrix} \tag{2.8}$$

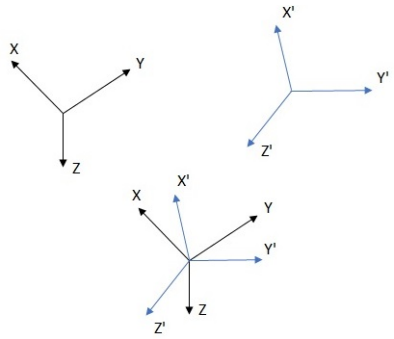


Figure 2.1: Rotation of Coordinate Axes

The new coordinates of $\boldsymbol{P}$ can be obtained by multiplying with transformation matrix[1] as

$$P' = \begin{pmatrix} \alpha_1 & \beta_1 & \gamma_1 \\ \alpha_2 & \beta_2 & \gamma_2 \\ \alpha_3 & \beta_3 & \gamma_3 \end{pmatrix} \begin{pmatrix} X_i \\ Y_i \\ Z_i \end{pmatrix} \tag{2.9}$$

or,

$$P' = RP \tag{2.10}$$

Now, to consider any arbitrary plane we need to take into account both translation and rotation. Thus,

$$P' = R(P - O') \tag{2.11}$$

Now the projection of the point $\boldsymbol{P}$ on this plane will be

$$p' = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} R(P - O') \tag{2.12}$$

2.2 Projection of a Line

A line can be uniquely determined using two distinct points lying on it (say $\boldsymbol{P_1}$ and $\boldsymbol{P_2}$).
We will represent the line as a 3 x 2 matrix represented as ($\boldsymbol{P_1}$, $\boldsymbol{P_2}$)

$$L = \begin{pmatrix} X_1 & X_2 \\ Y_1 & Y_2 \\ Z_1 & Z_2 \end{pmatrix} \tag{2.13}$$

[1]Details about finding rotational transformation matrix in the Appendix

As seen in section 2.1, projection of a single point can be obtained by multiplication of point vector by suitable matrix. Now, the projection of line on the plane will be same as a line drawn on the plane joining the projections of the points.

2.2.1 Projecting on one of the Cartesian planes

$$L_{xy} = \begin{pmatrix} x_1 & x_2 \\ y_1 & y_2 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} X_1 & X_2 \\ Y_1 & Y_2 \\ Z_1 & Z_2 \end{pmatrix} \tag{2.14}$$

or,

$$L_{xy} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} L \tag{2.15}$$

Similarly,

$$L_{yz} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} L \tag{2.16}$$

and,

$$L_{zx} = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} L \tag{2.17}$$

2.2.2 Projection on an arbitrary plane surface

Just as we derived for the case of projection of point on arbitrary plane $\mathbf{\Pi}$, similarly the projection of the line, $\boldsymbol{L}$ on the $\boldsymbol{X' - Y'}$ plane will be given by:

$$L'_p = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} R(L - O') \tag{2.18}$$

(Here $\boldsymbol{O'} \equiv (\boldsymbol{O'}, \boldsymbol{O'})$)

2.3 Projection of 3D Models

For representational purposes a 3D model can be thought of as a set of lines making up the edges and planes making up the faces. By projecting all the edges and all the faces we would be able to obtain the projection of the entire object.

A set of bounding edges would define a plane. Thus if we are able to project the edges, then their projections shall give us the projection of plane bounded by these edges.

2.3.1 Spatial arrangement of planes and hidden lines

With the naive implementation of the procedure stated above we can develop a projection drawing by considering the 3D object as a wireframe, however with that we would lose out on the information about spatial orientation of plane and hidden lines in the drawing.

To know which plane lies above others we need to follow a procedure in selecting the order in which points are projected.

Firstly, we arrange the points in order of decreasing distance from the plane of projection. We start projecting points onto the plane in this order.

As soon as both of the points defining an edge are projected on the plane we join the on the projection making up the projection of edge. We continue doing this process. As soon as all the lines bounding a plane are projected, we know that the topmost plane has been projected. Now any edge projected thereafter which pass through this plane will have to be a hidden edge.[2]

We keep adding points, edges and planes in this order and we can be sure that all the visible parts are projected on the plane first and any edges added thereafter which pass through a plane has to be a hidden edge.

[2]Details about finding that a point lies within a polygon in appendix

Chapter 3

3D Reconstruction from projections

This chapter deals with creating the 3-dimensional reconstruction of an object given its three (or atleast two) orthographic projection. We will also see about the existence and uniqueness of the solution.

We will proceed as before, first reconstructing points, then lines and finally 3D objects.

3.1 Reconstruction of a single point from given orthographic projection

Let us take a single point $\boldsymbol{P_i}$ shown in each of the three projections.

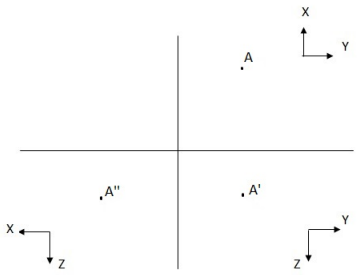


Figure 3.1: Projections of single point

The actual point in 3D will be of the form:

$$P_i = \begin{pmatrix} X_i \\ Y_i \\ Z_i \end{pmatrix} \tag{3.1}$$

and its corresponding projections in the three planes would be:

XY Plane:

$$p = \begin{pmatrix} x_i \\ y_i \end{pmatrix} \tag{3.2}$$

YZ Plane:

$$p' = \begin{pmatrix} y'_i \\ z'_i \end{pmatrix} \tag{3.3}$$

ZX Plane:

$$p'' = \begin{pmatrix} z''_i \\ x''_i \end{pmatrix} \tag{3.4}$$

Using the method of projecting 3D point in 2D as in 2.1,

$$p = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} P \tag{3.5}$$

$$p' = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} P \tag{3.6}$$

$$p'' = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} P \tag{3.7}$$

resulting in the equation,

$$\begin{pmatrix} x_i \\ y_i \\ y'_i \\ z'_i \\ z''_i \\ x''_i \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} P \tag{3.8}$$

Now, for existence of a solution
$x_i = x''_i, y_i = y'_i, z'_i = z'_i$,[1].

This gives,

$$\begin{pmatrix} x_i \\ y_i \\ y_i \\ z_i \\ z_i \\ x_i \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} P \tag{3.9}$$

Removing redundant rows,

$$\begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} P \tag{3.10}$$

[1] If not then the given orthographic projections are incompatible and hence a reconstruction is not possible.

This has unique solution namely,

$$\begin{pmatrix} X_i \\ Y_i \\ Z_i \end{pmatrix} = \begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix} \tag{3.11}$$

which is the required point in 3D.

Clearly, for projection of a single point we need only the three coordinates for which two orthographic projections are both necessary and sufficient.

3.2 Reconstruction of a line

Let us take a single line $\boldsymbol{L_i}$ joined by points $\boldsymbol{A_i}$ and $\boldsymbol{B_i}$.
In the three projections it will correspond to three lines

$$l_i \equiv (a_i, b_i), l_i' \equiv (a_i', b_i'), l_i'' \equiv (a_i'', b_i'') \tag{3.12}$$

If the line is normal to a Cartesian plane then, in the corresponding orthographic projection, the points $\boldsymbol{A_i}$ and $\boldsymbol{B_i}$ shall be coinciding.

We will use the same method as defined above in section 3.1 to derive the points

$$A_i = \begin{pmatrix} A_{i\_x} \\ A_{i\_y} \\ A_{i\_z} \end{pmatrix} \tag{3.13}$$

and,

$$B_i = \begin{pmatrix} B_{i\_x} \\ B_{i\_y} \\ B_{i\_z} \end{pmatrix} \tag{3.14}$$

.

Our line, $\boldsymbol{L_i}$ will then be $\boldsymbol{L_i \equiv (A_i, B_i)}$.

3.3 Reconstructing a wire frame of a 3D object

Let us consider a 3-Dimensional object as shown in Fig. 3.2 . It will consist of lines $\boldsymbol{L_i}$'s and points $\boldsymbol{P_i}$'s.

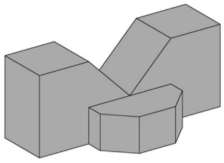


Figure 3.2: A 3-D Object

We will be given its three orthographic projections as shown in Fig. 3.3 and we will assume that we are given point correspondence in each view i.e., for a point $\boldsymbol{p_i}$ in a view, corresponding points $\boldsymbol{p_i'}$ and $\boldsymbol{p_i''}$ are shown in the two other views.

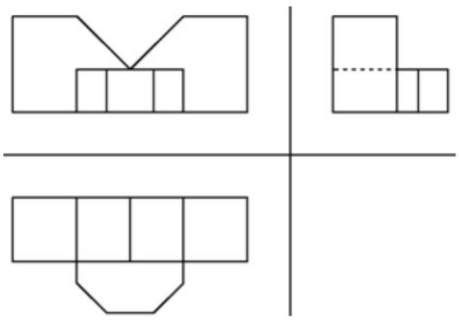


Figure 3.3: Orthographic Projections of the object (1^{st} angle)

Further we are given various $\boldsymbol{l_i}$'s in the three orthographic projections. The lines given to us may not be exhaustive and could pose several problems. This is discussed in the following section.

3.3.1 Intermediate vertices and collinear edges

Consider a orthographic projection as below:

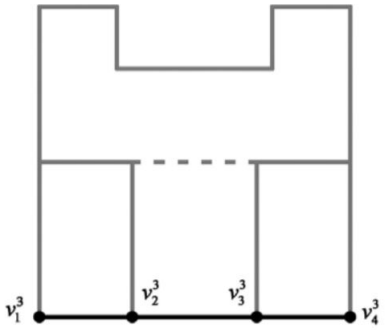


Figure 3.4: A projected view

Consider the lowest line. This line consist of many edges, for ex. v_1v_2, v_2v_4, v_1v_4 etc. Now we do not expect the user to enter each of the possible permutations. The user will input either v_1v_4 or combinations like v_1v_2 and v_2v_4.

First we will follow a method called segmentation. In this for each edge we will iteratively check for all the possible existence of intermediate vertices.

If no intermediate vertex is found, the procedure stops. Otherwise the found intermediate vertex causes the creation of two new edges (unless one of them already exists). We will perform this for each set of edges belonging to a projection, thus adding new edges to our edge set.

This is depicted in Fig. 3.5.

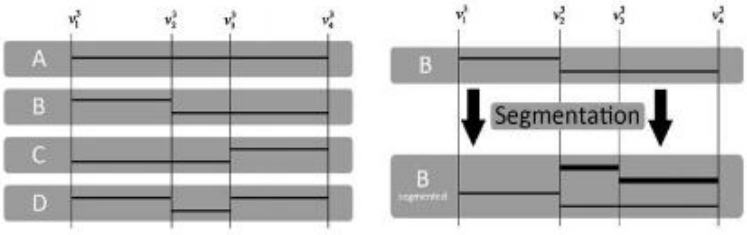


Figure 3.5: Segmentation procedure for creating additional edges

This segmentation task will create additional short edges out of larger edges like v_1v_4 but still it may not be exhaustive. For ex. say it may divide v_1v_4 into v_1v_2 and v_2v_4, the latter being further subdivided. It will leave out permutations such as v_1v_3.

So after the segmentation task, a check of collinearity of edges is performed. This is fundamental when two non contiguous vertices are linked by two or more edge all collinear one with each other. This procedure will add new edges like v_1v_3 to our set of edges.

This is depicted in Fig. 3.6 .

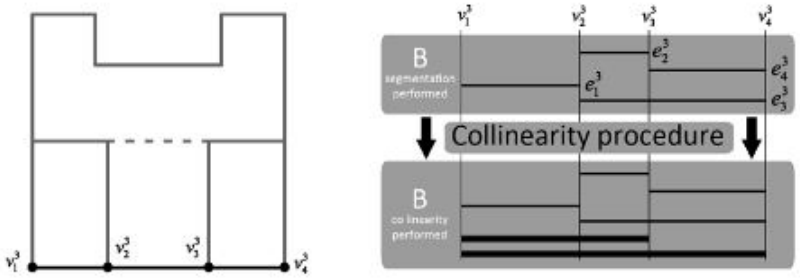


Figure 3.6: Collinearity procedure for creating additional edges

After performing these two procedures we will get a set of edges, $\boldsymbol{E_i}$ for each of the three projections (i is the label for the three planes). Each of the set will contain edges/lines $\boldsymbol{l_j}$ joined by some points $\boldsymbol{a_i}$ and $\boldsymbol{b_i}$. From the three sets so obtained we will remove out all the edges which conflict in the orthographic projections. (A non conflicting relationship exists if there exist a line between the two points in all the three projections or the points are coincident in one of the projection.)

On the remaining edges we will follow the same procedure for lines as detailed in section 3.2 . This would give us the corresponding points $\boldsymbol{A}$ and $\boldsymbol{B}$ in 3-D which join to form our line $\boldsymbol{L}$.

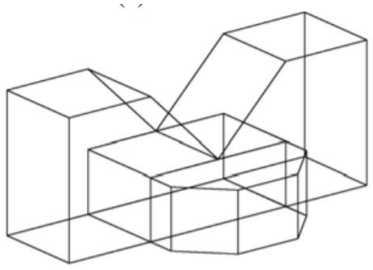


Figure 3.7: Pseudo-Wire frame of re-constructed object

The set of points $\boldsymbol{P}$ and edges $\boldsymbol{E}$ in 3-D constructed in this manner will give us our reconstruction of the object in the form of a wireframe model. It will be a wireframe model of the true 3-D object similar to that shown in Fig. 3.7. However this wireframe may contain some extra connecting lines between two points which should not be there. We will call this a "pseudo wireframe".

We need to remove these extra edges to obtain the real object's wireframe as shown in Fig. 3.8.

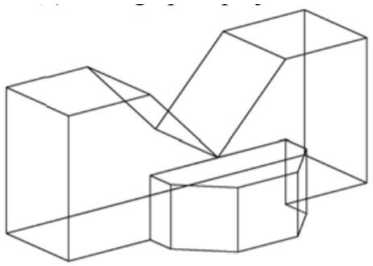


Figure 3.8: Wire frame of re-constructed object

It is for sure that no actual line will be left out because for every line $\boldsymbol{L_i}$ there will correspond a line $\boldsymbol{l_i}$ in each of the three views whether solid or hidden, from which our algorithm will infer an actual line $\boldsymbol{L_i}$ in 3-D.

3.3.2 Removing extra edges from Pseudo-wireframe

Many of the false edges and features derived from the drawing can be removed by following what is knows as **Pruning Strategy** derived by *Markovsky et al.*

According to this strategy keep iterating over the following four steps until no feature is removed in a loop:

1. A vertex lying on just one or no edges is removed, together with that edge.
2. A vertex lying on just two collinear edges is removed and the two edges are replaced by a single edge.
3. Three or more edges meeting at a vertex where all of the edges are coplanar, are removed unless they have a collinear extension.
4. A vertex lying on just two non-collinear edges is removed together with those edges.

This **Pruning Strategy** shall remove the false features from the wire-frame giving us the desired wireframe reconstruction of the 3D object.

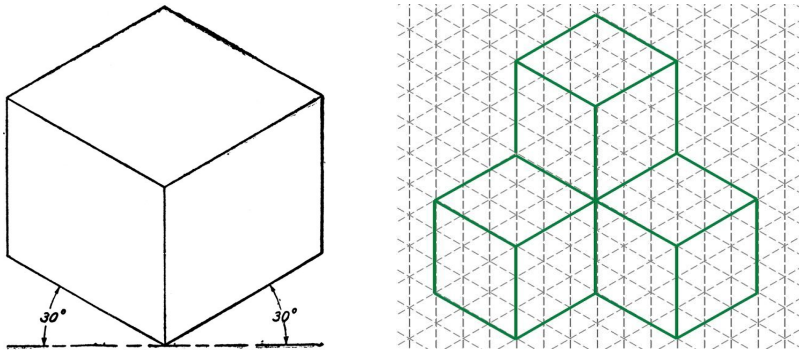
3.3.3 Rotation of the Wireframe

Rotation of the wireframe obtained for observation by the user can be carried out by multiplying with rotation transformation matrix as illustrated in sections 2.1.2 and 2.2.2.

Chapter 4

Representing 3D Models in Isometric View

Isometric projection is a method for visually representing three-dimensional objects in two dimensions.



(a) Isometric drawing for a cube (b) Isometric graph

Figure 4.1: Isometric representation

Isometric view can be obtained for a 3D object by suitable rotations followed by dimension reduction.The object is first rotated around the vertical axis by 45° followed by a rotation about horizontal axis by 35.264°.

The isometric transform from a point A in 3D space to a point b in 2D space looking into the first octant can be obtained as:

$$\begin{pmatrix} b_x \\ b_y \\ 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & cos\alpha & sin\alpha \\ 0 & -sin\alpha & cos\alpha \end{pmatrix} \begin{pmatrix} cos\beta & 0 & -sin\beta \\ 0 & 1 & 0 \\ sin\beta & 0 & cos\beta \end{pmatrix} \begin{pmatrix} A_x \\ A_y \\ A_z \end{pmatrix} \tag{4.1}$$

where $\alpha = 35.264°$ and $\beta = 45°$

or,

$$b = \frac{1}{\sqrt{6}} \begin{pmatrix} \sqrt{3}A_x - \sqrt{3}A_z \\ A_x + 2A_y + A_z \end{pmatrix} \tag{4.2}$$

To find the isometric projections for the edges, we just have to project the corresponding endpoints of an edge, and by drawing a line between them we will obtain the projection of the edge.

Thus given coordinates for vertices and edges for any 3D solid, we can obtain its isometric projection using the above given formula. For obtaining isometric projection from different octants we just have to multiply the point coordinates with a suitably chosen rotational transformation matrix.

Appendix A

Rotational Transformation Matrix

We look into the problem of finding a rotational transformation matrix when an arbitrary rotated plane is specified

A.0.1 When plane is specified as a point lying on it and normal vector

We consider any arbitrary line passing through not parallel to normal. Taking cross product of this line with normal(say $\vec{n}$) will give us a vector which lies perpendicular to the normal vector and hence lies on the plane. We normalise this obtained vector to get a unit vector(say $\vec{p}$) lying in the plane. Now taking a cross product of $\vec{n}$ and $\vec{p}$ will give us another unit vector $\vec{q}$ lying in plane orthogonal to $\vec{p}$.

Now together $\vec{p}$,$\vec{q}$ and $\vec{n}$ for a new orthogonal system for coordinates. Let these vectors with respect to the Cartesian coordinate system be represented as :

$$\begin{pmatrix} p_1 \\ p_2 \\ p_3 \end{pmatrix}, \begin{pmatrix} q_1 \\ q_2 \\ q_3 \end{pmatrix} and \begin{pmatrix} n_1 \\ n_2 \\ n_3 \end{pmatrix} \tag{A.1}$$

The rotation transformation matrix for transforming coordinates

$$\begin{pmatrix} p_1 & p_2 & p_3 \\ q_1 & q_2 & q_3 \\ n_1 & n_3 & n_3 \end{pmatrix} \tag{A.2}$$

A.0.2 When plane is specified as a set of point lying on it

For unique determination of a plane we need atleast three no collinear points. Let P_1, P_2 and P_3 be three such points on plane. Then, we consider two vectors namely $\boldsymbol{P_1P_2}$ and $\boldsymbol{P_1P_3}$. Now taking a cross product of the two gives us the normal vector $\vec{n}$ of the plane.
Also normalising $\boldsymbol{P_1P_2}$ gives us a vector lying on plane $\vec{p}$. Now taking its crossproduct with $\vec{n}$ and continuing as above gives us the required rotation tranformation matrix.

Appendix B

Point inside a polygon

This section addresses the problem of determining whether a point lies inside the polygon or not.

Firstly, if the coordinate of point lie beyond the extreme coordinates of the polygons, then we can be sure that the point is outside the polygon. If this is not the case, then we must do something else. We will use an algorithm known as **ray casting**. The idea of the algorithm is pretty simple: Draw a virtual ray from anywhere outside the polygon to the point under consideration and count how often it hits a side of the polygon. If the number of hits is even, it's outside of the polygon, if it's odd, it's inside.

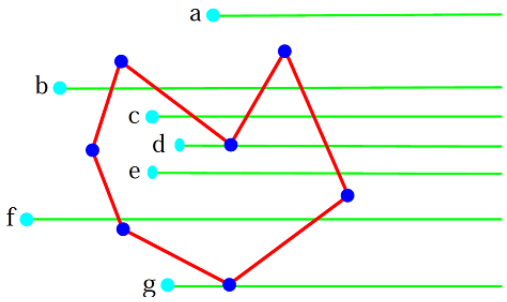


Figure B.1: Ray Casting

The special cases such as that for the point **g** in the above figure, or the case when an edge is coincident with the casted ray, we need to handle things differently.

If a point lies on the casted ray then we can either add both the edges sharing that point as a hit or ignore them altogether, since the parity remains same in both the case.

If an edge is coincident with the casted ray then we must ignore that edge. The edges adjacent to this edge can either both be counted or left out, again due to indifference to the parity caused by this.

Bibliography

[1] From 2D Orthographic views to 3D Pseudo-Wireframe: An Automatic Procedure http://www.ijcaonline.org/volume5/number6/pxc3871296.pdf

[2] Graphics Recognition: Algorithms and Systems *Karl Tombre*

[3] https://en.wikipedia.org/wiki/Isometric_projection#Mathematics

[4] https://stackoverflow.com/questions/217578/how-can-i-determine-whether-a-2d-point-is-within-a-polygon