ASSIGNMENT 1

MATHEMATICAL MODEL

DESIGN PRACTICES IN COMPUTER SCIENCE
COP290

Manish Tanwar - 2016CS10363

Abhinash Kumar - 2016CS10357

INTRODUCTION

The objective of this project is to design and implement a software package which reconstructs a 3-D object from its given orthographic views and from the given 3-D object gives its projections on a given cutting plane and its orthographic views(top, front, right side view). This document mainly talks about the mathematical model involved in this project and the algorithms to achieve the same. In this project we are considering only polyhedron 3-D objects which have flat polygon faces. For reconstruction of any polyhedron object, 3 orthographic views are sufficient and 2 orthographic views are necessary as some symmetric polyhedron objects can be uniquely reconstructed from 2 orthographic views.

ASSUMPTIONS

1. **3D to 2D Transformation -**
   (a) Our Software package targets Polyhedron 3D objects only.
   (b) Hidden lines also being shown as solid lines.
   (c) The information provided about the object forms a valid 3D object.
2. **2D to 3D Reconstruction -**
   (a) Provided projection forms a valid object.
   (b) The projected views provided in the input are top view, front view and right side view of the object.
   (c) Hidden lines are also treated as solid lines.

INPUT-OUTPUT FORMATS

1. **3D to 2D Transformation -**
   (a) Type-1 (Projection on the given plane)
      i. Input - The coordinates of vertices and edge set of the solid, and the plane on which the orthographic projection is to be taken.
      ii. Output - In the 2D plane the coordinates of the vertices and the edge set in the projected view.

(b) Type-2 (Orthographic views of the solid)

i. Input - The coordinates of vertices and the edge set of the solid.

ii. Output - In the 2D plane the coordinates of the vertices and the edge set in all the three projected views(Top view, Front view, Right side view).

2. **2D to 3D Reconstruction -**

(a) Input - For every vertex of the solid 2D coordinates in the top, front and right side, The edge set for each view and

(b) Output - The coordinates of vertices and edge set of the solid and further to be decided.

TRANSFORMATION OF A POINT

1. Rotations

(i) 2D Rotation

$$R(\theta) = \begin{bmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

(ii) 3D Rotation

Rotation about coordinate axes -

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\theta & -sin\theta \\ 0 & sin\theta & cos\theta \end{bmatrix} \qquad R_y(\theta) = \begin{bmatrix} cos\theta & 0 & sin\theta \\ 0 & 1 & 0 \\ -sin\theta & 0 & cos\theta \end{bmatrix}$$

$$R_z(\theta) = \begin{bmatrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Properties of Rotation Matrix -

• **Linearity -**

$$R(X+Y) = R(X) + R(Y)$$

• **Orthogonality -**

$$RR^\top = I$$

$$R^\top = R^{-1}$$

$$det(R) = 1$$

• **Normalized -** The squares of the elements in any row or column sum to 1.

• **Non-Commutative -** For two Rotation matrices R_1 and R_2 -

$$R_1R_2 \neq R_2R_1$$

2. Translation

Add a forth coordinate($w = 1$), so now Point $P(x, y, z)$ is represented as -

$$P(x,y,z) \Rightarrow \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Translation Matrix -

$$T = \begin{bmatrix} 1 & 0 & 0 & T_x \\ 0 & 1 & 0 & T_y \\ 0 & 0 & 1 & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} I_3 & T \\ 0 & 1 \end{bmatrix}$$

where I_3 is identity matrix and T is the amount of translation with respect to origin in all three dimensions.

$$T = \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix}$$

So translated point (say $P'(x',y',z')$) of Point $P(x,y,z)$ for translation amount T is -

$$P' = TP = \begin{bmatrix} 1 & 0 & 0 & T_x \\ 0 & 1 & 0 & T_y \\ 0 & 0 & 1 & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x+T_x \\ y+T_y \\ z+T_z \\ 1 \end{bmatrix} = T + P$$

3. Scaling

Scaling matrix for scaling with S_x and S_y for x and y dimension respectively -

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

Scaling matrix for scaling with S_x, S_y and S_z for x,y and z dimension respectively -

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

Scaling of point $P(x,y,z)$ by scaling factor S -

$$P' = SP = \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & S_z \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} S_x x \\ S_y y \\ S_z z \end{bmatrix}$$

Orthographic Projection

An orthographic projection is a parallel projection for which the direction of the projection is perpendicular to the view-plane.

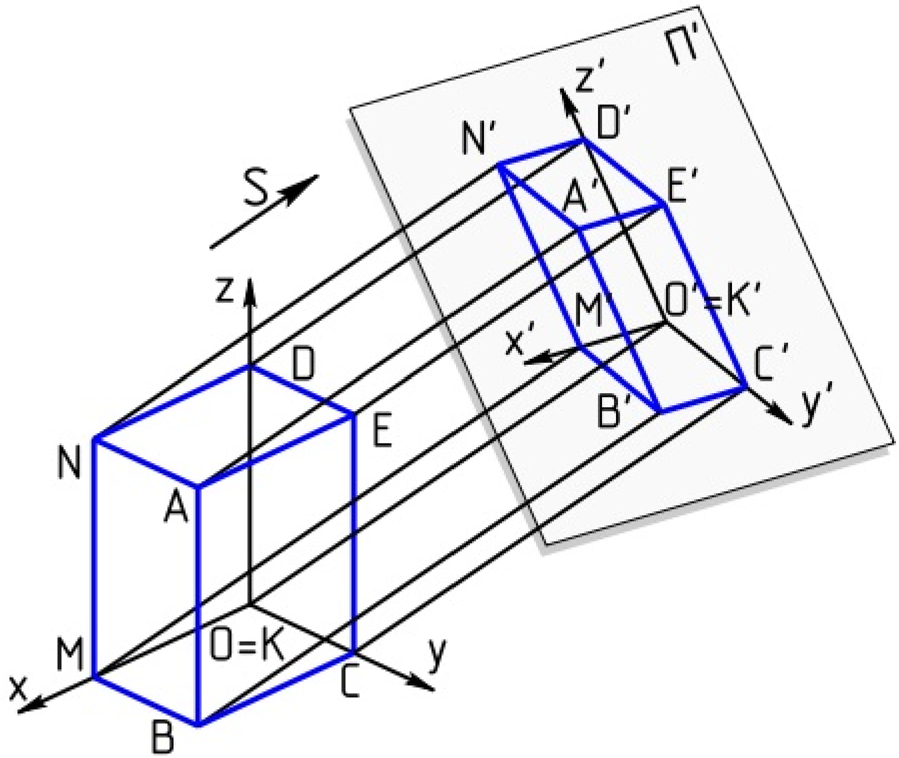

Figure 1: Orthographic View of a solid

1. Projection of a point on a Plane

To find the projection of a point on a plane

For a given plane $P: ax + by + cz + d = 0$, the projection of a point $Q(x_1, y_1, z_1)$ on P

Plane P can be represented as -

$$P: \begin{bmatrix} a & b & c & d \end{bmatrix} \begin{bmatrix} \vec{\Lambda} \\ 1 \end{bmatrix} = 0$$

where Λ is a variable point vector. A normal Vector of Plane P is given by -

$$\vec{n} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$$

P can also be written as -

$$P : \begin{bmatrix} \vec{n}^\top & d \end{bmatrix} \begin{bmatrix} \vec{\Lambda} \\ 1 \end{bmatrix} = 0$$

Any point on line passing through Q and normal to plane P is given by $\vec{q} + \lambda\vec{n}$ where $\lambda \in \mathbb{R}$ So to find the Projected point(say $\vec{R}$) substitute $\vec{R} = \vec{q} + \lambda\vec{n}$ in equation of plane P -

$$\begin{bmatrix} \vec{n}^\top & d \end{bmatrix} \begin{bmatrix} \vec{q} + \lambda\vec{n} \\ 1 \end{bmatrix} = 0$$

$$\Rightarrow \lambda = \frac{-d - det(\vec{q}\vec{n}^\top)}{det(\vec{n}\vec{n}^\top)}$$

therefore

$$\vec{R} = \frac{-d - det(\vec{q}\vec{n}^\top)}{det(\vec{n}\vec{n}^\top)}\vec{n} + \vec{q}$$

2. Projection of a line-segment on a Plane

We have two end points of the line-segment(say A and B), say projections of points A and B on the plane are A' and B'. So the projection the line segment AB is $A'B'$.

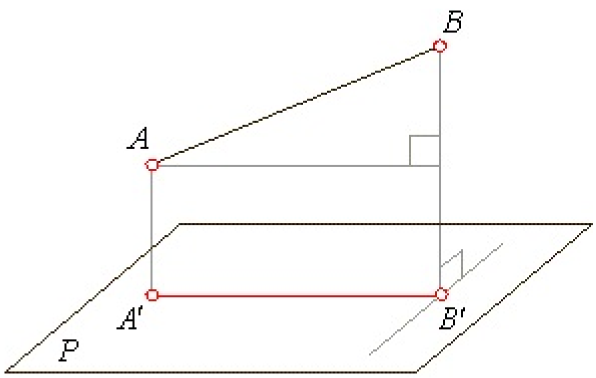

Figure 2: Projection of a line

3. PROJECTION OF SOLID ON THE GIVEN PLANE

We have all the vertices of the solid given to us. First, we will take the projection of all the points on the given plane by the method mentioned above and then we will join those projected points in projection which forms an edge in the solid. If any 2 points coincide and they form an edge in solid then we have to neglect that edge as projection of that edge is a point on the given plane.

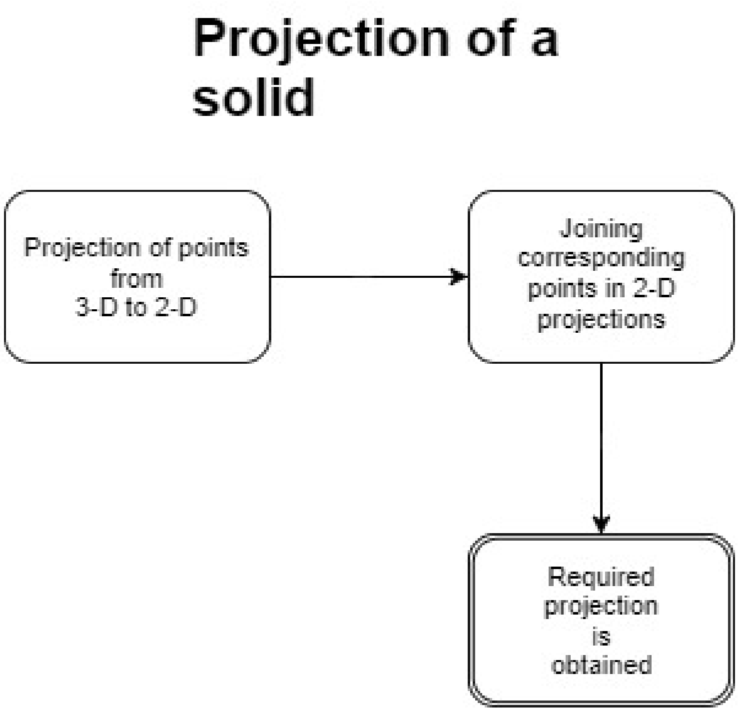

Figure 3: Block Diagram of taking Projection

ORTHOGRAPHIC VIEWS OF AN OBJECT

For Polyhedron Objects, 3 views are sufficient to accurately describe the object. In Engineering Drawing these three views are generally top view, front view and right side view. After finding the projection of all the vertices, in each view we have to join all those points which forms an edge in solid object. The points which coincide in projection plane are to be neglected. All the views are taken in accordance with the given coordinate axes as shown in Figure 6.

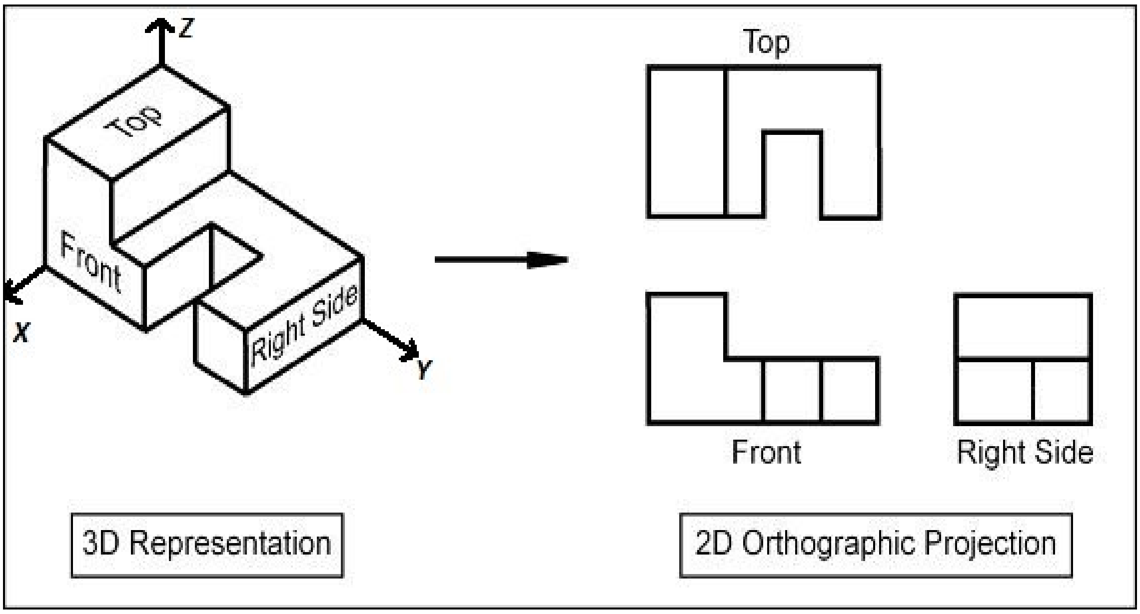

Figure 4: Orthographic views of a solid

Top View

The coordinates of a vertex on the object $A(x, y, z)$ in the top view is given by (say $A_T(x', y')$) -

$$A_T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} A$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix}$$

Front View

Rotation Matrix about Y axis is given by -

$$R_y(\theta) = \begin{bmatrix} cos\theta & 0 & sin\theta \\ 0 & 1 & 0 \\ -sin\theta & 0 & cos\theta \end{bmatrix}$$

The coordinates of a vertex on the object $A(x, y, z)$ in the front view is given by (say $A_F(x', y')$) -

$$A_F = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} R_y(-90°) A$$

$$\Rightarrow \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} -z \\ y \\ x \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} -z \\ y \end{bmatrix}$$

Right Side View

Rotation Matrix about X axis is given by -

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\theta & -sin\theta \\ 0 & sin\theta & cos\theta \end{bmatrix}$$

The coordinates of a vertex on the object $A(x, y, z)$ in the right side view is given by (say $A_S(x', y')$) -

$$A_S = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} R_y(90°) A$$

$$\Rightarrow \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ -z \\ y \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ -z \end{bmatrix}$$

RECONSTRUCTION OF SOLID

To obtain solid from given orthographic views there are mainly three process involved -

1. to obtain the vertices of the solid from the given views and thus obtain a wireframe.
2. to aggregate these vertices to form the planes of the object.
3. Finally obtaining the polyhedron with the help of the obtained planes.

There can be largely many wireframes corresponding to a given orthographic views.So the main problem is to remove the redundant edges which are formed in wireframe. After these redundant edges are removed then we head on to obtain planes of the polyhedron. Once we have got all the vertices, edges and planes of the solid then our solid is fully defined and our solid is finally obtained.

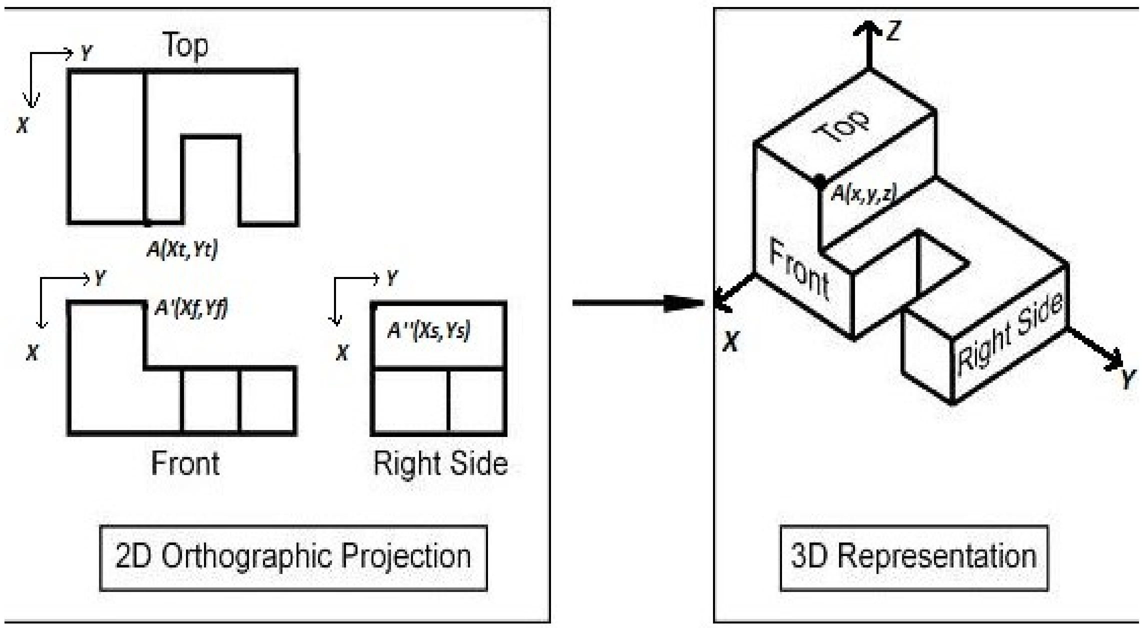

Figure 5: Reconstruction of solid from its orthographic views

Notations Used-

1. **Vertices** -
   Representation of a point A in all three view -
   (a) Top View => $A(x_T, y_T)$
   (b) Front View => $A'(x_F, y_F)$
   (c) Right Side View => $A''(x_S, y_S)$
2. **Vertex Sets** -
   (a) Top View Vertex Set => V_T
   (b) Front View Vertex Set => V_F
   (c) Right Side View Vertex Set => V_S
3. **Edge Sets**
   (a) Top View Edge Set => E_T
   (b) Front View Edge Set => E_F
   (c) Right Side View Edge Set => E_S
4. **Others** -
   (a) Possible Vertex Set in 3D Object => V_P
   (b) Possible Edge Set in 3D Object => E_P

1. Generation of 3D vertices -

For a point $A(x_T, y_T), (x_F, y_F), (x_S, y_S)$ the corresponding 3D coordinates(say $A(x, y, z)$) are given by these equations (from the above section)-

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x_T \\ y_T \end{bmatrix}$$

$$\begin{bmatrix} z \\ y \end{bmatrix} = \begin{bmatrix} -x_F \\ y_F \end{bmatrix}$$

$$\begin{bmatrix} x \\ z \end{bmatrix} = \begin{bmatrix} x_S \\ -y_S \end{bmatrix}$$

2. Generation of 3D Edges -

Possible edges are very useful in obtaining the correct wireframe of the object. After obtaining the vertex set of the solid object, we obtain the corresponding edges by checking all the views in the folowing manner -
First we select a vertex from V_T. Let say select $v_i \in V_T$ then check for all $v_j \in V_T$ such that $\exists \quad e = \quad (v_i, v_j) \in E_T$, then it is a potential edge in solid if $(e \in E_F \mid e = (v'_i, v'_j))$ and $(e \in E_F \mid e = (v''_i, v''_j))$.
Note: Here edge $\in$ means it is either an edge or end points of e overlaps in that view.
Here there is a correspondence between the points in all the three views like here v_i ,v'_i, v''_i are all the same vertex of the solid respectively in top,front and side view.
All the edges obtained in this process are potential edges in 3D object and all this edges are stored in E_p

3. Construction of Wireframe -

Some of the generated potential edges(E_P) may partially overlap and they are redundant so Procedure for removing redundant edges is as follows -

1. Initially mark all edges in E_P as non-visited.
2. If all the edges in E_P are visited then the procedure ends here. Select an non-visited edge $e_i \in E_P$ and make set E_{Temp} as e_i.
3. $\forall e_j \in E_P,\ e_j \neq e_i$, if e_j and e_i are collinear and e_j overlaps with one edge in E_{Temp}, then set $E_{Temp} \leftarrow E_{Temp} \cup e_j$
4. If $\left|E_{Temp}\right| = 1$ then mark this edge of E_{Temp} visited and go to step (b), otherwise, continue.
5. If $\left|E_{Temp}\right| > 1$ then make $E_P \leftarrow E_P - E_{Temp}$ and sort all the vertices of the edges in E_{Temp} with respect to their coordinate values. Let say $v_1, v_2, v_3 ... v_m$ is the sorted sequence of vertices of the edges in E_{Temp}. Now add all the edges $(v_k, v_{k+1}) \forall k \in 1,2,3..,m-1$ into E_P and mark them visited. Go to step (b).

By proceeding with the above steps all the edges in E_P are unique and non-overlapping.

4. Generation of Planar Graphs

Planes are generated with the help of obtained wireframe , V_p and E_p . Following steps should be followed to obtain planes -

1. First adjacent edges are stored of each vertex in V_p .
2. Any plane can be obtained from any two non-collinear vector in space so we select two edges $e1 = (v_i, v_j)$ and $e2 = (v_i, v_k) \in E_p$ with v_i as the common vertex.
3. Eliminate duplicate planes by checking if there exists two planes which have the same direction and same distance from the origin.
4. Eliminate illegal planes by checking if the boundary is closed or not.

5. Generation of Face Loops

1. First we consider a plane and select that edge (lets say e1) which lie on this plane.
2. After this we will consider the adjacent edge e2 of e1 which lie on this plane.
3. We repeat the above procedure for e2 until we reach e1 again.
4. This gives us a basic loop.
5. Face loops are formed by taking union of the basic loops such that for a face loop F_k, $\exists \; L \in F_k$ such that L_i lies in $L \quad \forall \; L_i \in F_k - \{L\}$

6. Generation of Body Loops

1. First of all we assign directions(+ve side and -ve side) to normals of every face loop.
2. Now we consider a face loop with a certain direction and find out it's adjacent face loops' direction such that they both direct to interior or exterior of the body loop.
3. Repeat the above step till a group of face loops form a closed body which forms a body loop.
4. Repeat step 2 and 3 till all the face loops are visited from both directions.
5. Removing illegal body loops -

(a) It should not have dangling face loop.

(b) Every edge in a body loop should be shared by exactly two face loops.

(c) Body loops can be described in two ways -

   i. Inner body loop => Valid

   ii. Outer body loop => Invalid

   To differentiate inner and outer body loops - We consider one of the face loops and extend it's normal (say l) if the number of intersecting points made by l and all the face loops of the body loop(except the face with normal l) is one or more then the body loop is classified as inner body loop, otherwise outer body loop.

7. Construction of the Solid

There are a number of body loops formed but not all of them contribute in the construction of the solid. Let's say we have n body loops, then the number of objects that can be formed is $2^n - 1$. So we have to check the legality of the possible objects. When two body loops are combined then there is a possibility that

1. A face loop becomes redundant.

2. Some edges become redundant.

To find the correct object cross-check the orthographic views of object with the given input.

REFERENCES

1. Efficient algorithm for the reconstruction of 3D objects from orthographic projections, Qing-Wen Yan, C L Philip Chen and Zesheng Tangt

2. Wikipedia : `https://en.wikipedia.org/wiki/Rotation_matrix`