DESIGN AND IMPLEMENTATION OF A SOFTWARE PACKAGE FOR ENGINEERING DRAWING

COP 290 - Design Practices

Modelling and Analysis

Authors:
Sukriti Gupta
Vrittika Bagadia

Supervisor:
Dr. Subhashish Banerjee

January 2018

INDIAN INSTITUTE OF TECHNOLOGY DELHI

INTRODUCTION

This project aims at designing and implementing a software package for engineering drawing that can perform functions like generating projection of a solid figure (input in a suitable format) onto any cross section or cutting plane and reconstructing the solid figure from given orthographic views.

Note: It is assumed that the objects do not consist of curved edges/ surfaces.

1 2-D to 3-D

1.1 Input format:

We need to generate wireframe model from the given orthographic views. The input is given in the form of two lists, corresponding to two orthographic views. For edges we assume that all hidden and solid edges are given to us in the form of a list, with each edge represented as a tuple of the vertex numbers. The edges can be taken as an input using adjacency matrix as well.

1.2 Generating 3-D Model

Matching the corresponding points (we assume one to one correspondence) in these lists would give us the three coordinates of all the object points. Next, we need to identify the faces of the object from the edge list. For this, we use the following algorithm:

1. Choose a vertex such that there exists a pair of edges, "AB" and "BC" incident on that vertex that have not been considered together earlier. These edges should not be collinear as we want to define a plane using these edges only.

2. These two edges define the boundary of a face. Now, move the other endpoint of any one these edges, say to point A. Check for the edge that lies in plane of AB and BC. Add this edge to the boundary edges of the face, maintained as a list.

3. The above step is repeated till we circle back to vertex B. Now, all the edges that are adjacent to each other, that is share a vertex and lie on this face, have been considered together in pairs.

4. We repeat the above steps till all the adjacent edge pairs have been considered.

We are effectively finding out the shortest cycles in the graph as each simple cycle is actually a face. This algorithm would output all the faces/holes that the object has. To differentiate between the two, we consider the fact that only those "faces" would be holes which are completely contained in another face.

2 3-D to 2-D

Given an input 3 dimensional object, we wish to obtain its orthographic projections or its 2 dimensional projection onto an arbitrary plane.

2.1 Input format

Consists of the wireframe model with the faces enumerated as well. Each face has its associated list of edges that form its boundaries and a Boolean which states whether it is a hole or not.

2.2 Taking projection of object points

Coordinates of all points are stored in a 3*1 matrix M. To obtain transformations onto different planes, the matrix M of each point is multiplied by the transformation matrix T. Calculations for computing T for different planes of projection are shown below.

2.2.1 Projection onto x-y plane

The projection of any point $[x \quad y \quad z]^T$ is $[x \quad y \quad 0]^T$ as the z coordinate of any point on the x-y plane is zero. Hence the transformation matrix T can be represented as

$$T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

2.2.2 Projection onto y-z plane

$$T = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

2.2.3 Projection onto z-x plane

$$T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

2.2.4 Projection onto an arbitrary plane

Any plane onto which the projection has to be taken is represented by the unit vector perpendicular to the plane. This vector is represented as a 3*1 matrix of its direction cosines.

Our aim is to rotate this plane to become parallel to the x-y plane, so that the same transformation can be applied to the object and after that, the method of taking projection onto x-y plane gives us the required projection onto the given plane in the transformed coordinate system. This transformation can be

brought about using matrix multiplication. We rotate the unit normal vector of the plane to become parallel to the z-axis, $[001]^T$.

To get the answer in our original coordinate system, we multiply the obtained coordinates of the projection to the inverse of the transformation matrix. At this stage we are considering that the plane is not a cutting plane and it is just specifying the direction of viewing.

Finding out the transformation matrix: Let the direction cosines of the unit vector v perpendicular to the plane be represented by c_x, c_y, c_z. For this vector to coincide with the z axis, rotations are required about the other two axes. Let alpha and beta represent the angles of rotation about the x and y axes respectively. *Rotation by α brings our normal vector in the x-z plane and then rotation by β makes it coincide with the z axis. Once the rotations have been performed,the required projection is found out and the rotations are reversed to get the vector in its original configuration.*

Calculating α :

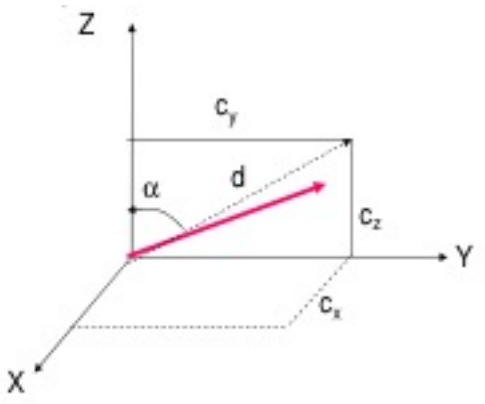

Figure 1: Calculating α

$$d = \sqrt{c_x^2 + c_z^2}$$

$$\cos\alpha = \frac{c_x}{d}$$

$$\sin\alpha = \frac{c_y}{d}$$

The x component is not affected by rotation about the x axis hence, for calculating alpha we only need to take into account the projection vector of v on the y-z plane. Alpha is thus the angle between the z axis and this projection vector.

This projection vector (of length d) is rotated about the x axis by multiplying with rotation matrix $R_x(\alpha)$ and we obtain a vector v' of length d along the z axis.

Calculating β :

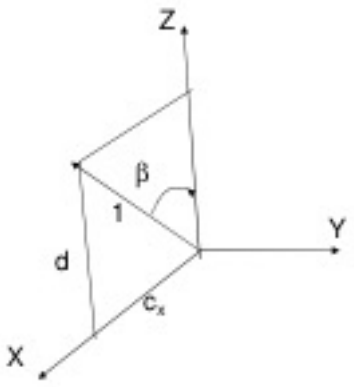

Figure 2: Calculating β

$$\cos\beta = d$$

$$\sin\beta = c_x$$

The resultant of v' and c_x (the x component of the vector v) lies in the x-z plane at an angle β from the z axis and is of unit length. It is rotated by angle β about the y axis by multiplying with matrix $R_y(\beta)$ and we obtain the transformed vector that is parallel to the z axis in the new coordinate system.

Finding the rotation matrices $R_x(\alpha)$ & $R_y(\beta)$:

We shall first consider rotation about the x axis. Point P $[x\ y\ z]^T$ is rotated by angle α to a new point P' $[x'\ y'\ z']^T$. Since the x coordinate remains unaffected by any rotation about the x axis, we have

$$x = x'$$

The problem is now reduced to a 2-D rotation about the origin. We can express P in polar coordinates as:

$$y = r\cos(\theta)$$

$$z = r\sin(\theta)$$

and the rotated point P' as:

$$y' = r\cos(\theta + \alpha)$$

$$z' = r\sin(\theta + \alpha)$$

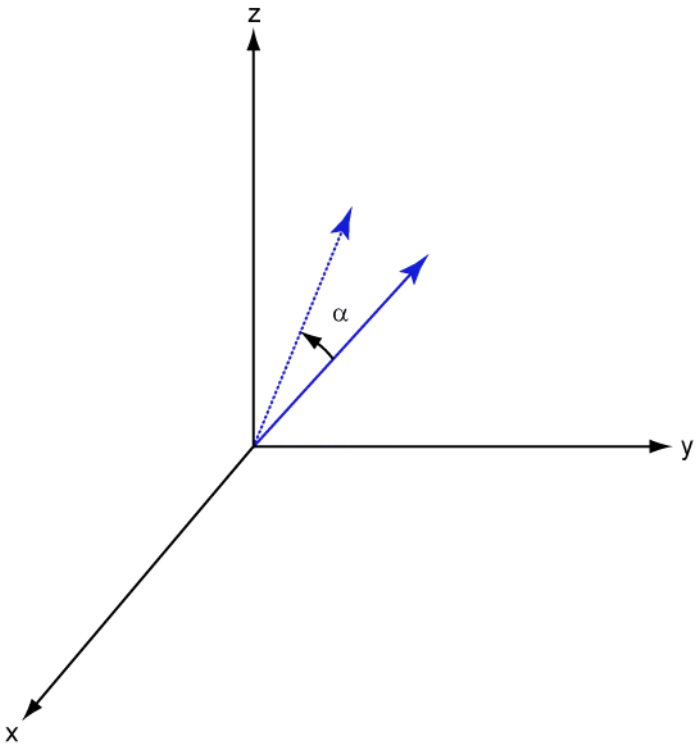

Figure 3: Calculating $R_x(\alpha)$

Simplifying these four equations, we obtain

$$y' = y\cos(\alpha) - z\sin(\alpha)$$

$$z' = y\cos(\alpha) + z\sin(\alpha)$$

Therefore the matrix for rotation about x axis can be represented as

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

Similarly the matrix for rotation about y axis can be derived and is calculated to be

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

Rotation of the coordinate axes with the vector fixed can be interpreted as rotation of the vector in the opposite direction. Thus the change in components

of a vector when the coordinate system rotates involves the transpose of the rotation matrix.

Hence **to find the transformation matrix T,** the complete sequence of operations can be summarized as:

$$T = [R_x(\alpha)]^T [R_y(\beta)]^T [R]([R_y(\beta)]^T)^{-1}([R_x(\alpha)]^T)^{-1}$$

Since $R^(\alpha)$ and $R^(\beta)$ are square matrices with real entries whose columns and rows are orthogonal unit vectors, therefore the inverse is equal to the transpose. Hence the expression reduces to

$$T = [R_x(\alpha)]^T [R_y(\beta)]^T [R][R_y(\beta)][R_x(\alpha)]$$

where R is the matrix used to take projection onto the x-y plane.

This expression can be simplified using the standard libraries.

2.3 Dealing with hidden lines

1. We project each point onto the xy plane, in order of increasing z coordinate.
2. For each edge whose both endpoints have been projected, we mark it with a solid line.
3. For each face, as soon as all its edges are marked, we consider the area covered by it. All the edges lying in this area are now marked in dashed lines.
4. This procedure is repeated till all the points are projected.

2.4 Steps in case only orthographic views are given

Now we consider the case in which only those edges are given which are visible in the orthographic views. Since in case of two or less views, it may or may not be possible to get a unique 3D model, we consider the case of 3 orthographic projections being given, which are sufficient to uniquely determine a 3D model, if it exists. We also assume that the corresponding points are given as otherwise, the problem becomes NP hard. For each edge visible in the views, we can determine the exact endpoints of those edges, considering the fact whether or not they will be visible in other views, and if they are, whether they are solid or hidden and the points themselves are hidden or visible. The only thing we need to do is to find out the edges which are not visible in any of the views. For each point not connected to atleast 3 other points, we consider all the points it is not connected to. For each of these, we see whether or not an edge can exist in between these two points according to the views given. For example, if we consider a cube (Figure 4), the orthographic views would be labelled squares (1234, 1485, 4378). From the front view, the vertex 5 is known to be

connected with vertices 1 and 8 as the edges 5-2 and 5-7 would have manifested as hidden lines in the side view and top view respectively. The only possible vertex connected to 5 (other than 1 and 8) can be 6 as the edge 5-3 would have shown in the front view and side view as a hidden line while 5-4 would have manifested as a solid line in the front view.

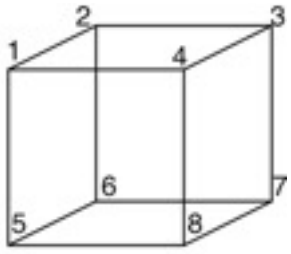

Figure 4

2.5 Cutting planes

If the viewing plane specified is a cutting plane, we first consider the object to have no holes. Now, we determine all the points of intersection of the cutting plane and our object. We do this by considering each plane of the object at a time (equation can be derived as we know atleast three non-collinear points lying on the plane) and simultaneously solving it with the cutting plane to determine whether or not there is any line of intersection. If there is a line of intersection, we determine the endpoints of this line on our object. We remove all the vertices and edges that would be hashed, that is parts of the object to one side of the cutting plane by determining each element's distance to the cutting plane. We now take the projection of the modified object as we normally do. After taking the projection, we hash all the simple cycles formed by the edges that were formed due to intersection of the cutting plane. Now we consider any hole that may be present. If there is any intersection between the hole and the hashed planes (determined by simultaneously solving the two planes), we un-hash the intersecting part in the projection.

3 3-D to Isometric

The isometric drawing can be expressed as a pair of coordinates along the two perpendicular axes, U and V in the 2D plane. In isometric drawings, all the axes x, y and z are placed at 120° to each other and thus, to get the position of any point in the isometric drawing, each component in 3D can be viewed as the addition of certain components along the perpendicular axes of the isometric drawing. The coordinates along these axes are given as:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \cos(\alpha) & cos(\alpha + 120°) & cos(\alpha - 120°) \\ \sin(\alpha) & sin(\alpha + 120°) & sin(\alpha - 120°) \end{bmatrix} \times \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

where α is the angle made by the U axis with the x axis in the clockwise direction. V is at an angle $(90° + \alpha)$ in anti-clockwise direction.

3.1 Dealing with hidden lines

We can deal with it in the same way as we would have dealt with the projection on a plane whose normal vector is given as $(\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}})$. Any point would be hidden only if there exists another point (belonging to a face or otherwise) that lies on the line that passes through the first point and has direction cosines given by $(\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}})$. This is effectively the exact same condition that determines hidden elements in the orthographic projections.

There exists a one to one mapping between the isometric view and the projection on the specified plane. In both the cases, we take the component along the line passing through the origin (direction cosine $(\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}})$) and perpendicular to it. The first component is made zero, while the other component stays the same for orthographic projections and goes through certain angle change for isometric view. Something hidden in the orthographic view implies that the perpendicular vector for two objects was the same. This implies that they will be mapped to the same point in the isometric view as well, thereby hiding the same point in this view as well.

REFERENCES

1. 3-D Rotations using matrices
http://web.iitd.ac.in/ hegde/cad/notes.html