1 Chpater 3: Geometric Objects and Transformations

1.1 3.4 Frames in OpenGL

With OpenGL, we have six different “frames” in which we can think about the different objects in our scene:

- Object
- World
- Eye(camera)
- Clip coordinates
- Normalized device coordinates
- Window(or screen)

The **object frame** involves the way the object is oriented around the world. When we operate on this frame we would be doing things relative to the cube's size and position in the world. Next, the **world frame** refers to the frame around which everything else is set up, the *coordinate system* of the entire scene itself.

When direclty viewing a scene, we have to use the camera's coordinate system. This means that the center of the camera lens is also the origin of another frame, the *camera* frame. There are *affine* transformations which preserve the ratios and distances between the objects in the frame. With the application of these, we can, with the 4x4 *model-view matrix*, switch around from one to another.

After we are done placing the objects in the corresponding frame for the image, we then have to calculate whether or not it will be visible to the viewer. We have to see whether it fits into the *window* coordinates, which specify the size of the viewing window for the application.

A strategy that we can pursue is place the camera at the origin. Then, we can move the object in the z direction. This makes the illusion that the object is “moving to the front”. We replace the object's position by $(x, y, z - d)$, where d is the delta in the z direction.

1.2 3.7 Affine Transformations

A *transformation* is a just a remapping of one object's coordinates to another set of coordinates. However, the trannsformation function f has to be *linear*, that is

$$f(\alpha p + \beta q) = \alpha f(p) + \beta f(q)$$

. Essentially, we have to be able to split the overall transformation function into smaller pieces.

We can think of transformations as multiplying a 4x4 identity matrix C and a sort of displacement vector u.

1.3 3.8 Transformaiton, Rotation, and Scaling

1.3.1 Translation

With translation, we only need to specify the displacement vector by which we are to move. We don't really need a frame around which to translate, only the displacement δ.

1.3.2 Rotation

Rotation might be a little harder to describe than translation because we also have to specify an axis around which we rotate. We also have to describe the difference in angle between the positions we are rotating in. For example, if our point is at position (x, y), then (x', y') is :

$$x = \rho \cos(\phi)$$

$$y = \rho \sin(\phi)$$

$$x' = \rho\cos(\theta + \phi)$$
$$y' = \rho\sin(\theta + \phi)$$

Here, ϕ is is the angle between the axis and our original position, and θ is the angle between ϕ and the new position. Thus, to derive the new point we are just adding the difference in angle between the angles, and add that to our original calculation for the original point.