Assignment 2c - Due Tuesday February 28th

Add the following five RenderMan Interface Requests to your system and have them alter the graphics state as appropriate:

```
Projection
Clipping
Translate
Rotate
Scale
```

Then render persppoints.rib and rendertest.rib (both on my website). Hand in your final code and your renders.

Here is a summary of what you will need to do:

Projection

- Be sure to catch the optional "fov" parameter when the projection type is "perspective." As we derived in class, the focal distance is related to the field of view angle (theta) as follows:

$$f = \frac{1}{\tan(\theta / 2)}$$

- Add the perspective matrix to your matrix library. Recall that the matrix we derived in class for doing RenderMan-style perspective projection is:

$$M = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & f & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

- The Projection call is supposed to first multiply the current transformation by M and then set the screen transformation (Ms->r) to the current transformation. Lastly, it sets current back to identity.

Clipping

- Add the near and far clipping plane distances to the graphics state, and be sure that their defaults are the same as in the spec (RI_EPSILON and RI_INFINITY - see ri.h at the end of the spec for details).
- Wherever you do your drawing (WorldEnd, perhaps), be sure to test each point's camera space Z coordinate value against the clipping plane distances after the point has been transformed to camera space.

Translate, Rotate, Scale

- All three simply multiply the current transformation by the appropriate matrix. What could be simpler?
- Remember that Rotate takes four arguments: angle dx dy dz (where dx dy dz denote an arbitrary axis). Add this kind of rotation to your matrix library using the 5-rotation technique we derived in class:

$$\theta = \arctan 2(dy, dx)$$
$$R = \sqrt{dx^2 + dy^2 + dz^2}$$
$$\phi = \arccos(dz / R)$$

$$M = R_Z(-\theta) R_Y(-\phi) R_Z(angle) R_Y(\phi) R_Z(\theta)$$

- Based on the above, you will need Z and Y axis rotation functions in your matrix library as well.