Automatic Texture Generation Notes

Texture Generation:

- The function to know is: `void glTexGen*( coord, param_name, param );`
  - *coord* should be either GL_S, GL_T, GL_R, or GL_Q, depending on which coordinate you want to effect.
  - *param_name* should be either GL_TEX_GEN_MODE, GL_OBJECT_PLANE, or GL_EYE_PLANE.
  - *param* should be a vector specifying the object or eye plane if using *param_name* of GL_OBJECT_PLANE or GL_EYE_PLANE.
  - *param* should be (for this class) either GL_OBJECT_LINEAR or GL_EYE_LINEAR if *param_name* is GL_TEX_GEN_MODE.
- To setup texture generation for this class, you'll need to make 3 OpenGL calls *for each* texture coordinate (s, t, r, or q) you want to automatically generate.
  - `glTexGeni( GL_S, GL_TEX_GEN_MODE, GL_EYE_LINEAR );`
  - `glTexGenfv( GL_S, GL_EYE_PLANE, plane );`
  - `glEnable( GL_TEXTURE_GEN_S );`
  - Where plane is defined as `GLfloat plane[4] = {A, B, C, D};` for floating point values A, B, C, and D.
  - To enable texture generation for coordinates other than *s*, replace GL_S with GL_T, GL_R, or GL_Q, and replace GL_TEXTURE_GEN_S with GL_TEXTURE_GEN_T, GL_TEXTURE_GEN_R, or GL_TEXTURE_GEN_Q.
  - To use an object plane instead of an eye plane, replace GL_EYE_LINEAR by GL_OBJECT_LINEAR and GL_EYE_PLANE with GL_OBJECT_PLANE.

How Does Linear Texture Generation Work?

- The basic idea is that for each vertex V that you create OpenGL computes the *s* coordinate as the distance between vertex V and the plane $Ax + By + Cz + Dw = 0$ specified by the function call `glTexGenfv( GL_S, GL_???_PLANE, plane );`
- How does this happen?
  - If V = (x_o, y_o, z_o, w_o), then this distance is $Ax_o + By_o + Cz_o + Dw_o$ in object coordinates.
  - Computing this distance in eye coordinates is a little tricky, since the plane and vertex must *first* be transformed to eye coordinates. How is this done? The modelview matrix is applied, of course! But remember transformations affect points and vectors differently. How do we get the correct plane? To find out, read page 699. What happens is that if $\mathbf{M}$ is the modelview matrix *at the time glTexEnvfv(GL_?, GL_EYE_PLANE, plane); is called*, then the transformed vertex V' is V' = $\mathbf{M}$V, let's call V' = (x_e, y_e, z_e, w_e). The plane must also be transformed to get a new plane $A'x + B'y + C'z + D'w = 0$ We get $(A', B', C', D') = (A, B, C, D)\mathbf{M^{-1}}$. The distance is then computed as $A'x_e + B'y_e + C'z_e + D'w_e$.
- Why is this the distance?
  - Think about the equation for a plane $Ax+By+Cz+Dw = 0$. (A,B,C) is the plane's normal. How do you compute D? You plug in a point P you know to be on the plane. If you use a homogenous point, w can be assumed to be 1 (do the homogeneous division). This means $-D = Ap_x + Bp_y + Cp_z$ for this plane. This can be written $-D = N \cdot \vec{P}$, where $\vec{P}$ is the direction from the origin to point P. Thus, -D is the distance from the origin to the plane (times the length of the normal $N = (A, B, C)$, if N is non-unit length).
  - If you think about it, the distance to the plane is $(V - P) \cdot N$. Using associativity, $(V - P) \cdot N = V \cdot N - P \cdot N$. Note from above that $-P \cdot N = D$. So, the distance from V to the plane is $V \cdot N + D = AV_x + BV_y + CV_z + D$.