#### **Executar viewer:**

~/assig/grau-g/Viewer/GLarenaSL (dins del directori on volem que se'ns guardin els shaders que creem)

### **Viewer Plugins:**

/dades/nacho.llado.cortes/G/Viewer/GLarenaPL

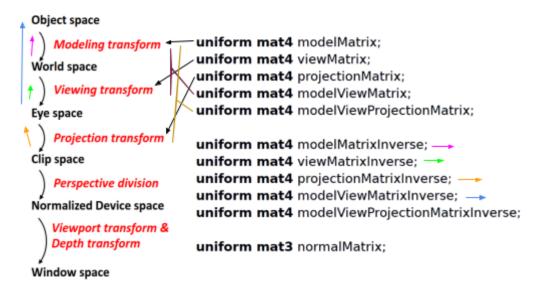
### **Activar Snippets:**

~/assig/grau-g/gedit-config (activar plugin snippets i escriure defs[tab] per a que surti tot)

### Web Assignatura:

https://www.cs.upc.edu/~virtual/G/

# Sistemes de coordenades i matrius



#### Model de Phong:

```
N = vector normal unitari = normalize(normalMatrix * normal);
L = vector unitari cap a la font de llum = lightPosition.xyz - P;
P = vector unitari de la càmera cap al vèrtex = (modelViewMatrix * vec4(vertex.xyz,1)).xyz;
V = vector unitari del vèrtex cap a la càmera = -P
R = reflexió del vector L respecte N = 2 * (N * L) * N - L
```

https://learnopengl.com/Getting-started/Coordinate-Systems

La normal a eye space multiplicar per la normalMatrix. La resta entre el lightPosition i el punt V per obtenir el vector L es fa en eye space!

Per treballar les coordenades de textura quan hem d'agafar fragments, reescalar-les al domini que ens és fàcil treballar  $[0, 1] \rightarrow {}^*6 \rightarrow [0, 6]$ 

# Transformacions bàsiques

# **Object space**

Modeling transform

World space

) Viewing transform

Eye space

Projection transform

Clip space

Perspective division

**Normalized Device space** 

Viewport transform & Depth transform

Window space

### **Modeling transforms**

translate(
$$t_x, t_y, t_z$$
) 
$$T = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

scale(
$$s_x, s_y, s_z$$
) 
$$T = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

rotate(a,x,y,z) 
$$T = \begin{bmatrix} x^2d + c & xyd - zs & xzd + ys & 0 \\ yxd + zs & y^2d + c & yzd - xs & 0 \\ xzd - ys & yzd + xs & z^2d + c & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$
 c=cos(a), s=sin(a), d=1-cos(a)

# Transformacions bàsiques

### **Object space**

**Modeling transform** 

**World space** 

Viewing transform

Eve space

Projection transform

Clip space

Perspective division

Normalized Device space

Viewport transform & Depth transform

Window space

## **Modeling transforms**

$$\mathbf{glRotate*}(a, 1, 0, 0) : \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos a & -\sin a & 0 \\ 0 & \sin a & \cos a & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

**glRotate\***
$$(a, 0, 1, 0)$$
: 
$$\begin{bmatrix} \cos a & 0 & \sin a & 0 \\ 0 & 1 & 0 & 0 \\ -\sin a & 0 & \cos a & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{glRotate*}(a, 0, 0, 1) \colon \begin{bmatrix} \cos a - \sin a & 0 & 0 \\ \sin a & \cos a & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

