

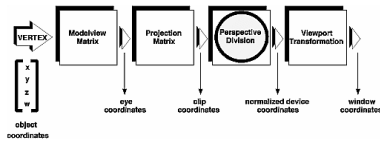
Transformaciones en OpenGL



Contenido

- Transformaciones y sistemas de coordenadas
- Ejemplo de transformación
- Dualidad de la transformación
- Funciones de matrices
- Transformaciones de modelizado y visualizado
- La pila de matrices

Transformaciones y sistemas de coordenadas



Ejemplo de transformación

```
void reshape(int width, int height) {
    glViewport(0, 0, width, height);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluPerspective(60.0,
        (GLfloat)height / (GLfloat)width, 1.0, 128.0);
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
    gluLookAt(0.0, 1.0, 3.0,
        0.0, 0.0, 0.0,
        0.0, 1.0, 0.0);
}
```

Dualidad de la transformación

- Las siguientes sentencias producen el mismo efecto:

```
gluLookAt(0.0, 1.0, 3.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);
```

```
glTranslatef(0.0, -1.0, -3.0);
```

- Es equivalente mover la cámara en una dirección a mover el objeto en la dirección contraria

Modos de la matriz

```
glMatrixMode(GLenum mode)
GL_MODELVIEW
GL_PROJECTION
GL_TEXTURE
```

Funciones de matrices

- `glLoadIdentity(void)`
- `glLoadMatrix(fd)` (cont TYPE *m)
- `glMultMatrix(fd)` (cont TYPE *m)

$$M = \begin{bmatrix} m_1 & m_5 & m_9 & m_{13} \\ m_2 & m_6 & m_{10} & m_{14} \\ m_3 & m_7 & m_{11} & m_{15} \\ m_4 & m_8 & m_{12} & m_{16} \end{bmatrix}$$

Transformaciones de modelizado y visualizado

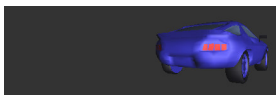
- Se combinan en una misma matriz
`glMatrixMode(GL_MODELVIEW)`
- Transforman de las coordenadas del objeto a las coordenadas de la cámara (eye coordinates)
- Funciones de modelizado:
 - `glTranslate(fd)` (TYPE x, TYPE y, TYPE z)
 - `glRotate(fd)` (TYPE angle, TYPE x, TYPE y, TYPE z)
 - `glScale(fd)` (TYPE x, TYPE y, TYPE z)
- Funciones de visualizado:
 - `gluLookAt()`

Orden de las tranformaciones

- `traslacion(1,0,0)`
- `rotacionY(135°)`



```
glRotatef(135.0, 0.0, 1.0, 0.0)
glTranslatef(1.0, 0.0, 0.0)
```



```
glTranslatef(1.0, 0.0, 0.0)
glRotatef(135.0, 0.0, 1.0, 0.0)
```

Orden de las transformaciones en OpenGL

```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
glMultMatrixf(M1);
glMultMatrixf(M2);
glMultMatrixf(M3);
```

- Se aplica al vector v:

$$M1 \cdot M2 \cdot M3 \cdot v = [M1 \cdot [M2 \cdot [M3 \cdot v]]]$$

Nota: en realidad se aplica: $[M1 \cdot M2 \cdot M3] \cdot v$

Sistema de referencia

- En un sistema global, las transformaciones se tienen que definir de la última a la primera
- En un sistema local, las transformaciones se definen en el orden que se aplican
- Caso particular: tortuga de logo

Pila de matrices

- `glPushMatrix(void)`
- `glPopMatrix(void)`

