


Transformations in OpenGL

- OpenGL supports built-in transformations
 - `glLoadIdentity`
 - Clear previous transforms
 - `glTranslate*(dx,dy,dz)`
 - `glRotate*(angle,nx,ny,nz)`
 - `glScale*(sx,sy,sz)`
 - Can also reflect with $s < 0$

1




Other Transformations

- What about shears, etc.?
 - User specified matrices are possible
 - `glLoadMatrix*(const type* array)`
 - `glMultMatrix*(const type* array)`

$$M = \begin{bmatrix} m_0 & m_4 & m_8 & m_{12} \\ m_1 & m_5 & m_9 & m_{13} \\ m_2 & m_6 & m_{10} & m_{14} \\ m_3 & m_7 & m_{11} & m_{15} \end{bmatrix}$$

2



Transformations and State

- A "global" transformation is in the state
- Transform commands modify the state
 - Loads replace the state
 - Other transforms compose with the state
 - Ex. `glTranslate`

$$M_{\text{state}} = M_{\text{state}}^T$$
 - The right multiply is important!

3

OpenGL Transformation Pipeline

- vertex
- Modelview Transform
 - eye coordinates
- Projection Transform
 - clip coordinates
- Perspective division
 - normalized device coordinates
- Viewport Transformation
 - screen coordinates

4

OpenGL Transform States

- OpenGL maintains 3 transforms in its state
 - Modelview
 - Projection
 - Viewport
- Modelview and Projection are routinely manipulated
- Viewport is set by window size, position, and clip specifications

5

Specifying Which Transform

- `glMatrixMode(mode)`
 - `GL_MODELVIEW`
 - Adjusted routinely during drawing
 - `GL_PROJECTION`
 - Set during initialization
 - Adjusted during reshape events

6

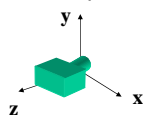
Why Combine Model and View?

- Roughly similar goals
 - Place objects in position about the view point
- Programmer must isolate the two
- Typically we code the view first then model
 - Sounds backward
 - Recall transforms right multiply
 - Last transform is applied first to the vertex
 - OpenGL models coordinate system changes

7

OpenGL Default View

- By default OpenGL sets the following view
 - Camera at the origin
 - Facing toward negative z-axis
 - View up is the y-axis

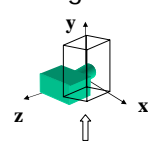
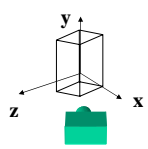


```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
```

8

Viewing Example (1)

- Viewing an edge from a distance

- Object perspective (fixed camera)
 - Rotate object -45° about y-axis
 - Move object back from camera $0,0,-5$

9

Viewing Example (2)

- Camera/coordinate system perspective (fixed object)
 - Move origin away from camera 0,0,-5
 - Rotate coordinate system -45° about y-axis
- OpenGL code
 - `glTranslatef(0.0, 0.0, -5.0);`
 - `glRotatef(-45.0, 0.0, 1.0, 0.0);`

10

Simplifying Viewing

- `gluLookAt`
 - Camera/eye position
 - Point looked at
 - View-up vector
 - May not be parallel to view direction

11

Projection Transformations (1)

- Always with respect to the view point
- Orthographic


```
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
glOrtho(left,right,bottom,top,near,far);
```

12

Projection Transformations (2)

- Perspective (1)


```
glFrustum(left, right, bottom, top, near, far);
```

13

Projection Transformations (3)

- Perspective (2)



```
gluPerspective(fovy, aspect, near, far);
```

14

Non-Standard Projections

- Examples
 - Axonometric
 - Non-right frustum
- Use `gl{Load,Mult}Matrix*`
 - Refer to projection matrix forms from last time


15



Matrix Stacks (1)

- Allow preservation and recovery of transformation state
- Transformation chosen by `glMatrixMode`
- `glPushMatrix`
 - Remember current coordinate system
- `glPopMatrix`
 - Recall last saved coordinate system


16



Matrix Stacks (2)

- Stack limits
 - `GL_MODELVIEW` – 32
 - `GL_PROJECTION` – 2
- Typical use
 - Draw in model coordinates
 - Perhaps a set drawing function
 - Use transforms to “place” models
 - Prior to drawing


17



Clipping

- Most clipping set by the projection
- Additional clip planes can be added
 - `glClipPlane`
 - Specify plane # with `GL_CLIP_PLANE i` , $i = 1..6$
 - Specify plane with array (A,B,C,D)
 - Must be enabled via `glEnable`
 - `glEnable(GL_CLIP_PLANE i)`


18



Viewport Transformation

- Clip extents are
 - Scaled to window size
 - Translated to window position
- Pay attention to the aspect ratio
 - Window size is passed to the reshape handler
 - `QGLWidget::resizeGL(int w, int h)`
 - A sub-window can be selected with `glViewport`

19



Black Screen Effect

- Why is nothing visible?
 1. Drawing in background color
 2. Near and far clip planes
 - ◆ Remember you backed off the camera
 3. In perspective transforms don't put 'near' too close to the camera. (Roundoff)
 4. Do you have the right transforms?
 5. Are you looking in the right direction?

20
