Project 1 Review

Ming Jin
10/03/2011
Suggested Workflow

• Start with the simplest: Cube and CameraView sliders.
• Work on the projections
• Get WorldView working (add ViewVolume)
Model-View Transformations
Model-View Transformations

• Model
  \[ M = T(t_x^M, t_y^M, t_z^M)R_z(\theta_z^M)R_y(\theta_y^M)R_x(\theta_x^M)S(\text{scale}^M) \]

  Make sure you have the orders right!

• View
  \[ V = T(t_x^V, t_y^V, t_z^V)R_z(\text{roll}^V)R_y(\text{yaw}^V)R_x(\text{pitch}^V) \]

  \( V \) looks backward... It specifies how to move the world so it’s “in front” of the canonical OpenGL camera
Projection Transformations

WorldView.projection()
CameraView.projection()

gl.glFrustum(left, right, bottom, top, near, far)
gl.glOrtho(left, right, bottom, top, near, far)

Think about how to preserve the aspect ratio!
Combining the two views

• Draw the ViewVolume in WorldView:
  – Think about the diagram again!
  – How do you relate the two views?
Clipping

• Six clipping planes for the view volume:
  
  GL.GL_CLIP_PLANE0,
  GL.GL_CLIP_PLANE1,
  ...
  GL.GL_CLIP_PLANE5

• Each plane has a formula:
  \[ Ax + By + Cz + D = 0 \]

  \( \mathbf{n} = (A, B, C) \) is the unit-length normal vector of the plane

  Given any point \( p \) on the plane: \( D = -p \cdot \mathbf{n} \)
Clipping

• Specify the clipping planes:

```c
void glClipPlane(int plane,
                double[] equation,
                int equation_offset)
```

*plane* – plane identifier
*equation* – array of four values specifying the plane
*equation_offset* – starting offset within the equation array
Clipping

• Placing the clipping planes:
  – Clipping planes are transformed just like other geometry – they are placed using the current matrix transformation!

• Applying the clipping:
  – Turn clipping planes on by calling glEnable()
Questions?