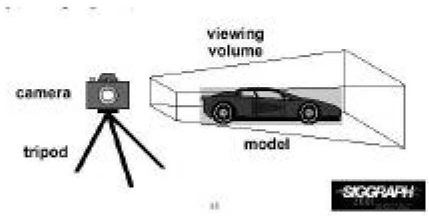
CS 4731: Computer Graphics
Lecture 11: 3D Viewing

Emmanuel Agu

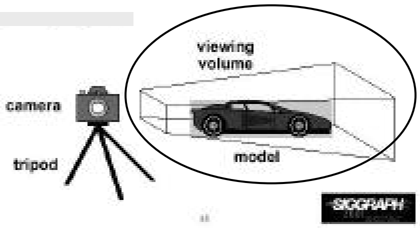
3D Viewing

- Similar to taking a photograph



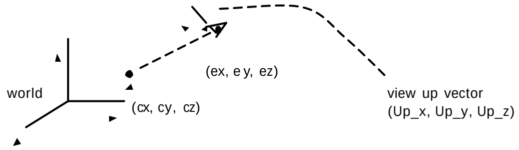
Viewing Transformation

- Control the “lens” of the camera
- Project the object from 3D world to 2D screen



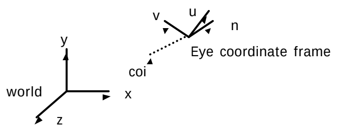
Viewing Transformation

- Control the “lens” of the camera
- Important camera parameters to specify
  - Camera (eye) position (Ex, Ey ,Ez) in world coordinate system
  - Center of interest (coi) (cx, cy, cz) or lookAt point
  - Orientation (which way is up?): Up vector (Up_x, Up_y, Up_z)



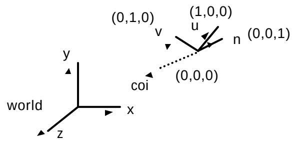
Viewing Transformation

- Transformation?
  - Form a camera (eye) coordinate frame
  - Transform objects from world to eye space



Viewing Transformation

- Eye space?
  - Transform to eye space can simplify many downstream operations (such as projection) in the pipeline



Viewing Transformation

- OpenGL way:
  - gluLookAt (Ex, Ey, Ez, cx, cy, cz, Up_x, Up_y, Up_z)
  - The view up vector is usually (0,1,0)
  - Remember to set the OpenGL matrix mode to GL_MODELVIEW first
- Recall: OpenGL uses 3 matrices:
  - Modelview matrix:
  - Projection matrix:
  - Viewport matrix:
- Modelview matrix:
  - combination of modeling matrix *M* and Camera transforms *V*

Viewing Transformation

- OpenGL Code:

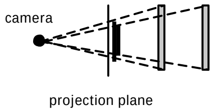
```
void display()
{
     glClear(GL_COLOR_BUFFER_BIT);
     glMatrixMode( GL_MODELVIEW);
     glLoadIdentity();
     gluLookAt(0,0,1,0,0,0,0,1,0);
     display_all();     // your display routine
}
```

Projection Transformation

- Different types of projection: parallel, perspective, orthographic, etc
- Important to control
  - Projection type: perspective or orthographic, etc.
  - Field of view and image aspect ratio
  - Near and far clipping planes

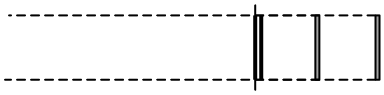
Perspective Projection

- Similar to real world
- Characterized by object foreshortening
- Objects appear larger if they are closer to camera
- Need:
  - Projection center
  - Projection plane
- Projection: Connecting the object to the projection center



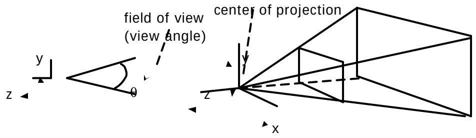
Orthographic Projection

- No foreshortening effect – distance from camera does not matter
- The projection center is at infinite
- Projection calculation – just drop z coordinates



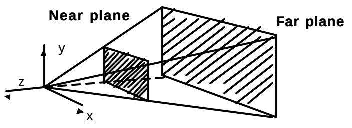
Field of View

- Determine how much of the world is taken into the picture
- Larger field of view = smaller object projection size



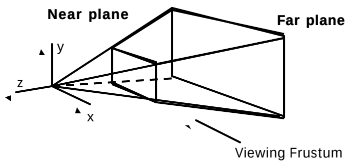
Near and Far Clipping Planes

- Only objects between near and far planes are drawn
- Near plane + far plane + field of view = Viewing Frustum



Viewing Frustrum

- 3D counterpart of 2D world clip window
- Objects outside the frustum are clipped

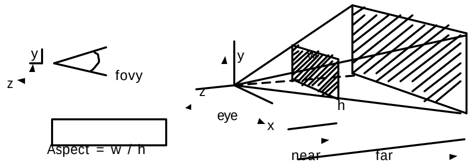


Projection Transformation

- In OpenGL:
  - Set the matrix mode to GL_PROJECTION
  - Perspective projection: use
    - gluPerspective (fovy, aspect, near, far) **or**
    - glFrustum (left, right, bottom, top, near, far)
  - Orthographic:
    - glOrtho(left, right, bottom, top, near, far)

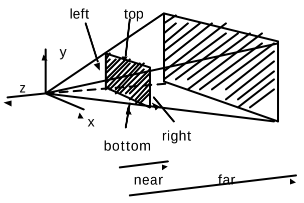
gluPerspective(fovy, aspect, near, far)

- Aspect ratio is used to calculate the window width



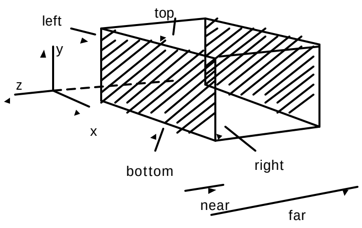
glFrustum (left, right, bottom, top, near, far)

- Can use this function in place of gluPerspective()



glOrtho(left, right, bottom, top, near, far)

- For orthographic projection

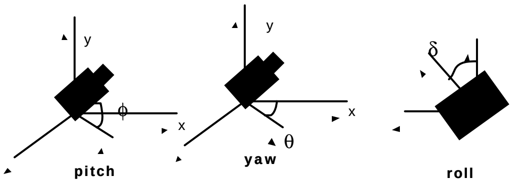


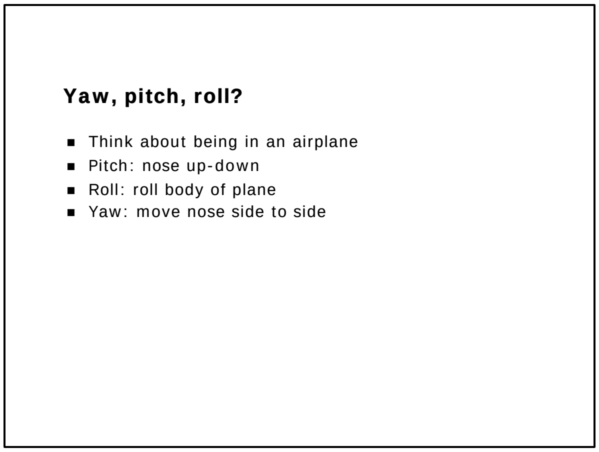
Example: Projection Transformation

```
void display()
{
      glClear(GL_COLOR_BUFFER_BIT);
      glMatrixMode(GL_PROJETION);
      glLoadIdentity();
      gluPerspective(fove, aspect, near, far);
      glMatrixMode(GL_MODELVIEW);
      glLoadIdentity();
      gluLookAt(0,0,1,0,0,0,0,1,0);
      display_all();     // your display routine
}
```

Flexible Camera Control

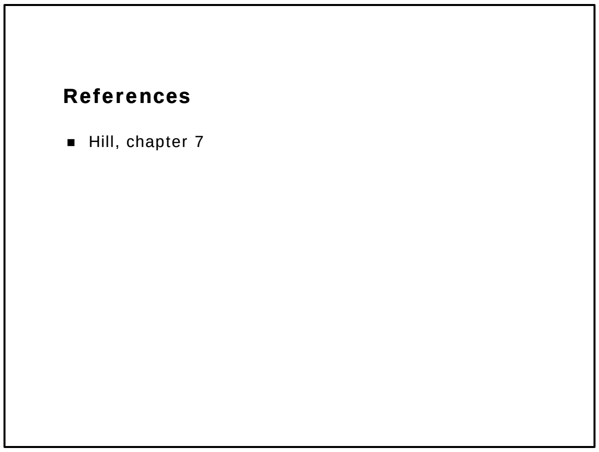
- Sometimes, we want camera to move
- Just like control a airplane's orientation
- Use aviation terms for this





Yaw, pitch, roll?

- Think about being in an airplane
- Pitch: nose up-down
- Roll: roll body of plane
- Yaw: move nose side to side



References

- Hill, chapter 7