Lecture 7: Computer Projection 1
Review

• In the last lecture
  – Definition: view point, view plane, projectors

  – Types of projection
    • Parallel (orthographic, oblique)
    • Perspective
Review

- Vanishing points in perspective
  - Lines parallel in 3D intersect at a vanishing point
In this lecture (and next)

- How to perform projection in the computer? Or, given a point in 3D, where do I draw it on the 2D computer screen?

World Coordinate: \((X_w, Y_w, Z_w)\)

? \( \downarrow \)

Screen Coordinate: \((X_s, Y_s)\)
Virtual Camera

• Parameters
  – Position of camera
  – Orientation
  – Field of view (wide angle, telephoto)
  – Clipping plane (near distance, far distance)
  – Perspective or parallel projection?
  – Focal distance
Position

- From where the camera is
  - Like a photographer choosing the vantage point to shoot a photo
- 3D point \( P = (P_x, P_y, P_z) \)
Orientation – Look Vector

- Where the camera is looking
- 3D vector \( L = (L_x, L_y, L_z) \)
  - Not necessarily a unit vector
- Look vector alone is not sufficient to describe orientation…
Orientation – Up Vector

• How the camera is rotated around the look vector
  – If you are holding the camera horizontally or vertically, or in between.

• 3D vector $\mathbf{U} = (U_x, U_y, U_z)$
  – Not necessarily orthogonal to look vector $\mathbf{L}$
  – Actual “Up-right” direction, $\mathbf{U'}$, is:

  \[ \mathbf{U}' = ? \]

  (projecting $\mathbf{U}$ onto the plane orthogonal to $\mathbf{L}$)
Orientation – Up Vector

• How the camera is rotated around the look vector
  – If you are holding the camera horizontally or vertically, or in between.

• 3D vector \( U = (U_x, U_y, U_z) \)
  – Not necessarily orthogonal to look vector \( L \)
  – Actual “Up-right” direction, \( U' \), is:

\[
U' = U - \left( \frac{L}{|L|} U \right) \frac{L}{|L|}
\]

(projecting \( U \) onto the plane orthogonal to \( L \))
Default Position, Orientation

- Camera at origin, looking down –Z axis, and in upright pose
  - E.g., in OpenGL

\[ P=(0, 0, 0) \]
\[ L=(0,0,-1) \]
\[ U=(0,1,0) \]
Viewing Angle

- Describes the field of view
  - Like choosing a specific type of lens, e.g., a wide-angle lens or telephoto lens
- Width and height angles $q_w, q_h$
  - Assuming the view region is a rectangle
Viewing Angle

- Determines amount of perspective distortion
  - Small angles result in near-parallel projectors, hence little distortion
  - Large angles result in widely varying projectors with large distortion
Viewing Angle

- When keeping the size of the main object in view, longer distances gives narrower view angle.

Close-up (wide angle)  Far away (narrow angle)
Viewing Angle

Close-up (wide angle)  Far away (narrow angle)
Viewing Angle

Close-up (wide angle)  Far away (narrow angle)
Viewing Angle

- Fun example: dolly-zoom effect (or “Hitchcock zoom”)
  - Moves away the camera and shrinks the viewing angle at the same time, so that the main subjects stays the same size on screen
  - The background gets “closer”, and perspective distortion lessens
Aspect Ratio

- **Aspect Ratio** \( a \)
  - Ratio of width over height of the screen
    - 1:1 (square)
    - 4:3 (NTSC)
    - 16:9 (HDTV)
    - 2.35:1 (Widescreen Films)

- Width angle as aspect ratio and height angle
  - Compute width angle:
    ?
**Aspect Ratio**

- **Aspect Ratio** \( a \)
  - Ratio of width over height of the screen
    - 1:1 (square)
    - 4:3 (NTSC)
    - 16:9 (HDTV)
    - 2.35:1 (Widescreen Films)

- **Width angle as aspect ratio and height angle**
  - Compute width angle:
    \[
    q_w = 2 \arctan(\tan(q_h/2) \times a)
    \]
Clipping Planes

- Restricts visible volume between *near* and *far* clipping planes
  - Objects closer than the near plane or further than the far plane are not drawn
  - Objects intersecting the two planes are clipped
- Defined as distances $d_n$, $d_f$ from camera along look vector
Clipping Planes

• Why do we need near plane
  – Avoid things too close to camera
    • They will appear with large distortion, and may block view
  – Avoid things behind the camera
    • They will appear upside-down and inside-out

• Why do we need far plane
  – Avoid drawing things too far away
    • They will complicate the scene
    • They appear small on the screen anyway
    • Saving rendering time
Perspective Camera Model

- Position
- Up vector
- Look vector
- Front clipping plane
- Back clipping plane
- Width angle
- Height angle
Perspective Camera Model

- View frustum: a truncated pyramid region that the camera can “see”
Orthographic Camera Model

- Width $w$ and height $h$ replace viewing angles
  - Both width angle and height angle are effectively zero
Orthographic Camera Model

- Width $w$ and height $h$ replace viewing angles
  - Both width angle and height angle are effectively zero

\[ w = h \cdot a \]
Film Plane and Viewport

- **Film Plane**
  - *Any* plane parallel to the near/far clipping planes.
- **Viewport**
  - A rectangular region on the screen displaying what’s projected on the film plane (may have different aspect ratio as the film plane)
Film Plane and Viewport

- Move the film plane further from the camera.
- What would happen to Viewport?
Other Camera Models

- **Focal length**
  - Approximates behavior of real camera lens
  - Objects at distance of focal length are in focus; other objects get blurred

![Diagram showing focal length and focus points](image)
Other Camera Models

- Focal length

Rendering with focal blur
Depth of field

Portraits (family photos)

Computer vision experiments
More light

LIGHT ON SENSOR

Less light

Large Aperture

Medium Aperture

Small Aperture

f/2.0  f/4.0  f/5.6  f/11  f/22

DOF - DEPTH OF FIELD

Little DOF (blurred back)

Much DOF (whole picture sharp)
Pupils
Pupils
Other Camera Models

• **Focal length**
  
  - Focal blur can serve as a cue for depth and (even) size

Held et al., "Making Big Things Look Small: Blur combined with other depth cues affects perceived size and distance", 2008
Other Camera Models

- **Oblique projection**
  - Look vector *not* perpendicular to film plane

Nikon PC-E Nikkor 24mm Tilt/Shift lens
Rolling shutter cameras
What next?

- **Three steps**
  - **Clipping**: removes geometry outside the frustum
  - **Projecting**: transforms 3D coords. to 2D coords. on the film plane
  - **Viewport transformation**: gets pixel coordinate in the viewport