Camera Models

Spring 2021

Daniel G. Aliaga
Department of Computer Science
Purdue University
Thin Lens System

Object point

Optical axis

Lens System

Focal point

Retina A
Thin Lens Equation

\[ \frac{1}{d_0} + \frac{1}{d_i} = \frac{1}{f} \]
Pinhole Camera Model

- **Pinhole**
- **Object**
- **Image Plane**

Variables:
- $f$
- $d$
“Pinhole” Camera Image
Digression:
Non-Pinhole Camera Models...

• Why restrict the camera to a pinhole camera model?
Multiperspectve Imaging

Hand-crafted

semi-automated...

to produce this...

[Roman-Vis04]
Multiperspective Imaging

[Seitz-CGA03]
Multiple COP Images

[Rademacher-SIG98]
Multiple COP Images

Figure 5 Castle model. The red curve is the path the camera was swept on, and the arrows indicate the direction of motion. The blue triangles are the thin frusta of each camera. Every $64^{th}$ camera is shown.

Figure 6 The resulting 1000×500 MCOP image. The first fourth of the image, on the left side, is from the camera sweeping over the roof. Note how the courtyard was sampled more finely, for added resolution.

Figure 7 The projection surface (image plane) of the camera curve.

Figure 8 Three views of the castle, reconstructed solely from the single MCOP image above. This dataset captures the complete exterior of the castle.

[Rademacher-SIG98]
Multiperspective Imaging for Cel Animation

Figure 1 A multiperspective panorama from Disney’s 1940 film *Pinocchio*. (Used with permission.)
Multiperspective Imaging for Cel Animation

[Wood-SIG97]
Multiperspective Imaging for Cel Animation

[Wood-SIG97]
Multiperspective Imaging for Cel Animation

[Wood-SIG97]
Occlusion-Resistant Cameras

Input images

Output images

[Aliaga-CGA07]
Occlusion Cameras

---

**Simulator Images**

- Figure 1: Depth image (DI).
- Figure 2: Images rendered from DI and OCRI, viewpoint 4" left of reference viewpoint.
- Figure 3: OCRI.
- Figure 4: Images rendered from DI and OCRI. Wireframe shows spherical display volume.

**Photographs of 3D Display**

- Figure 5: 3D images rendered from DI (left), OCRI (middle), and original geometric model (right), all photographed from reference view.
- Figure 6: DI and OCRI 3D images from viewpoint translated 4" left.
- Figure 7: DI and OCRI 3D images from side view.

[Popescu-JDT06]
Graph Cameras

Portal-based graph camera image (top left and fragment right) and PPC image for comparison (bottom left)

Occluder-based graph camera image (top left), PPC image for comparison (bottom left), and ray visualizations (right)

Enhanced street-level navigation

[Popescu-SIGA09]
Let’s get back on track: Pinhole Camera Model...

- **f**: Image plane
- **d**: Pinhole
- **object**: Object

Diagram showing the relationship between the object, pinhole, and image plane.
Photographic Camera History 101

• First photos:
  – 1826:
    • https://www.youtube.com/watch?v=sOkt8ObhN_M

• First videos:
  – 1874
    • https://www.youtube.com/watch?v=VC-yTYyE2w0

• Femto camera (10^{12} frames per second)
  – 2013
    • https://www.youtube.com/watch?v=EtsXgODHMWk
Camera Resolution

• First digital camera (~1985): 0.01 MP (100x100 pixels)
• Today: consumer ware up to 50 MP (~7000x7000 pixels)
• Gigapixel camera (assembled pics):
  – 320 GP of London (565,000 x 565,000 pixels)
    • https://www.dpreview.com/articles/7683341128/320-gigapixel-photo-of-london-is-the-worlds-largest-panoramic-photo
What is perspective projection?

- **Object**
- **Image Plane**
- **Point** $(x, y)$
- **Point** $(X, Y, Z)$

Equations:

- $x = f \frac{X}{Z}$
- $y = f \frac{Y}{Z}$
“Computer Graphics” Pinhole Camera Model

- f
- d
- eye
- image plane
- object
What is perspective projection?

\[ y = \frac{Y}{\frac{1}{f} + \frac{Z}{Z}} \]

\[ y = f \times \frac{Y}{Z} \quad \text{and} \quad x = f \times \frac{X}{Z} \]

- \( f \) is the focal length.
- \( (X, Y, Z) \) is the 3D world coordinate.
- \((x, y)\) is the 2D pixel coordinate on the image plane.
- The eye or viewpoint is at the origin.
- The optical axis is the line from the eye to the viewpoint.
- The image plane is the projection plane parallel to the screen.
Perspective Camera Parameters

- **Intrinsic/Internal**
  - Focal length $f$
  - Principal point (center) $p_x, p_y$
  - Pixel size $s_x, s_y$
  - (Distortion coefficients) $k_1,...$

- **Extrinsic/External**
  - Rotation $\phi, \varphi, \psi$
  - Translation $t_x, t_y, t_z$
Perspective Camera Parameters

• Intrinsic/Internal
  – Focal length $f$
  – Principal point (center) (=middle of image)
  – Pixel size (=1, irrelevant)
  – (Distortion coefficients) (=0, assuming no bugs 😊)

• Extrinsic/External
  – Rotation $\phi, \varphi, \psi$
  – Translation $t_x, t_y, t_z$
Focal Length

Assume \( c = 0 \):

\[
\begin{bmatrix}
  x \\
  y \\
  fY/Z
\end{bmatrix} = \begin{bmatrix}
  fX/Z \\
  fY/Z \\
  fY/Z
\end{bmatrix} = \begin{bmatrix}
  fX \\
  fY \\
  Z
\end{bmatrix} = \begin{bmatrix}
  X \\
  Y \\
  Z
\end{bmatrix}
\]
Focal Length

Assume \( c \neq 0 \):

\[
\begin{pmatrix}
    x \\
    y
\end{pmatrix}
= \begin{pmatrix}
    fX' / Z' \\
    fY' / Z'
\end{pmatrix}
= \begin{pmatrix}
    fX' \\
    fY' \\
    Z'
\end{pmatrix}
= \begin{pmatrix}
    X \\
    Y \\
    Z \\
    1
\end{pmatrix}
\]
Focal Length

In general, where $M$ is some 4x4 matrix:

$$
\begin{pmatrix}
    x \\
    y \\
    fX' / Z' \\
    fY' / Z'
\end{pmatrix}
= \begin{pmatrix}
    fX' \\
    fY' \\
    fY' / Z'
\end{pmatrix} = \begin{bmatrix}
    f & 0 & 0 & 0 \\
    0 & f & 0 & 0 \\
    0 & 0 & 1 & 0
\end{bmatrix} M_{4x4}
\begin{pmatrix}
    X \\
    Y \\
    Z \\
    1
\end{pmatrix}
$$
World to Camera Matrix $M$

$$
\begin{align*}
\tilde{x}_c &= R(\tilde{X} - C) \\
\tilde{x}_c &= R\tilde{X} - RC \\
\end{align*}
$$

$$
\tilde{x}_c = 
\begin{bmatrix}
R & t \\
0 & 1
\end{bmatrix}
\begin{bmatrix}
X \\
Y \\
Z
\end{bmatrix}
$$

$$
R = R_x R_y R_z
$$

3x3 rotation matrices

$$
t = 
\begin{bmatrix}
t_x \\
t_y \\
t_z
\end{bmatrix}^T
$$

translation vector
Perspective Projection Process

• Thus, given \( \tilde{X} = \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \)

...the perspective projection is

\( \tilde{x}_p = PM \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \)

\[
\begin{bmatrix}
\tilde{x}_p_x / \tilde{x}_p_z \\
\tilde{x}_p_y / \tilde{x}_p_z
\end{bmatrix}
\]
OpenGL Equivalent

...  
 glMatrixMode(GL_PROJECTION);  
 ...
 gluPerspective(60, 1.0, 0.1, 1000.0);  
 ...
 glMatrixMode(GL_MODELVIEW);  
 ...
 glTranslatef(tx,ty,tz);  
 glRotatef(rx,1,0,0);  
 glRotatef(ry,0,1,0);  
 glRotatef(rz,0,0,1);

/* or glLoadMatrixf(mat); */
...

Projection Transformations

void gluPerspective(GLdouble fovy, GLdouble aspect, GLdouble near, GLdouble far);
void glFrustum(GLdouble left, GLdouble right, GLdouble bottom, GLdouble top, GLdouble near, GLdouble far);
void glOrtho(GLdouble left, GLdouble right, GLdouble bottom,
            GLdouble top, GLdouble near, GLdouble far);

void gluOrtho2D(GLdouble left, GLdouble right,
                GLdouble bottom, GLdouble top);
Example OpenGL Rendering

- Perspective is “mathematically” accurate...
(3D) Perception

• There is lot more to it...
Perception

• Size-distance relationship
  – Same size object at two distances...
Perception

• Importance of ground plane, shadows...
Perception

• Alto Relief

• Sunken Relief
• We will see this type of things later...
Perception: Bas-Relief
Perception: Bas-Relief
Perception: Bas Relief Ambiguity

\[ NL^T = C \]

or

\[ NRR^T L^T = C \]
\[ NGG^{-1} L^T = C \]

??
Perception: Inconsistencies

• Escher
Perception: Nekker Cube
Perception: Nekker Cube
Perception: Muller-Lyer Illusion
Perception: Muller-Lyer Illusion
Perception: Muller-Lyer Illusion
Back to Projection Matrices...

• Basic Perspective Projection:

\[
\begin{bmatrix}
  f & 0 & 0 \\
  0 & f & 0 \\
  0 & 0 & 1 \\
\end{bmatrix}
\]

• Basic Orthographic:

\[
\begin{bmatrix}
  1 & 0 & 0 \\
  0 & 1 & 0 \\
  0 & 0 & 1 \\
\end{bmatrix}
\]
• Problem (maybe?):
  – The formulation give does not pass z, or depth, though the transformation pipeline
  – (see white/blackboard)
  – What can we do?

IDEAS????
Options

• Trivial changes?
• Funky matrices...
• Re-mapping options...
Projection Matrices

- Passthrough Z Perspective Projection:

\[
\begin{bmatrix}
    n & 0 & 0 & 0 \\
    0 & n & 0 & 0 \\
    0 & 0 & n+f & -nf \\
    0 & 0 & 1 & 0
\end{bmatrix}
\]

If \((Z = n)\), then \(z = \left( n + f - \frac{nf}{n} \right) = n\)

If \((Z = f)\), then \(z = \left( n + f - \frac{nf}{n} \right) = f\)
Projection Matrices

• To map [-1,+1] to viewing frustum \((l,r,b,t,n,f)\):

\[
\begin{bmatrix}
\frac{2n}{(r-l)} & 0 & 0 & -\frac{(r+l)}{(r-l)} \\
0 & \frac{2n}{(t-b)} & 0 & -\frac{(t+b)}{(t-b)} \\
0 & 0 & \frac{2nf}{(n-f)} & -\frac{(n+f)}{(n+f)} \\
0 & 0 & 0 & 1 \\
\end{bmatrix}
\]