CS 351: Perspective Viewing

Instructor: Joel Castellanos
e-mail: joel@unm.edu
Web: http://cs.unm.edu/~joel/

2/16/2017

Perspective Projection

Diagram showing perspective projection with labels for back clipping plane, view plane, image plane, and front clipping plane. Diagram includes points for camera, VRCS, and frustum limits.
Frustum

In computer graphics, the viewing frustum is the three-dimensional region which is visible on the screen which is formed by a clipped pyramid.

Properties of Perspective Projections

Property 1: The perspective projection of an object becomes smaller as the object gets farther away from the center of projection.

What are some non-obvious, and interesting effects of this?
Properties of Perspective Projections

Property 2: As an object is rotated, its projected width becomes smaller. This is known as **foreshortening**.

Property 3: Perspective projections preserve straight lines.
Properties of Perspective Projections

Property 4: Sets of parallel lines that are parallel to the view plane remain parallel when projected onto the view plane.

Property 5: Sets of parallel lines that are not parallel to the view plane converge to a vanishing point on the view plane.

Equation of a Checkerboard

// Checkerboard in x-z plane with y = 0.

public static Color getCheckerboardGroundColor(
    double x, double y, double z)
{
    if (Math.abs((Math.floor(x))) % 2 == Math.abs((Math.floor(z))) % 2)
        return Color.BLACK;
    return Color.WHITE;
}
Axis-Aligned Perspective Projection
Straight down Y-Axis

- Eye: (0, 10, 0)
- One Point on View Plane: \((x_{vp}, 5, z_{vp})\)

\[
\text{ray} = \text{eye} + d(\text{ViewPlanePt} - \text{eye})
\]

\[
\begin{bmatrix}
 x_{hit} \\
 z_{hit}
\end{bmatrix} = \begin{bmatrix}
 0 \\
 0
\end{bmatrix} + d \begin{bmatrix}
 x_{vp} - 0 \\
 z_{vp} - 0
\end{bmatrix}
\]

\[
0 = 10 + d (5 - 10) \\
0 = y_{eye} + d (y_{vp} - y_{eye}) \\
d = \frac{y_{eye}}{(y_{eye} - y_{vp})}
\]

- \(x_{hit} = x_{eye} + d (x_{vp} - x_{eye})\)
- \(z_{hit} = z_{eye} + d (z_{vp} - z_{eye})\)

A Practical Viewing System

- The **virtual pinhole camera** implements perspective viewing with the following features:

  - An arbitrary eye point.
  - An arbitrary view direction (The view plane is defined as being perpendicular to the view direction and centered on the ray from the eye point).
  - An arbitrary orientation about the view direction.
  - An arbitrary distance between the eye point and the view plane.
Physical Pinhole Camera

- Clear inverted image with small pinhole
- Fuzzy out-of-focus image with larger hole

Lens Aperture

- Aperture ≠ Shutter
- Why color reflections?
Large Glass is Expensive

- Canon, Prime 50mm f/1.8 USM Lens: $125.00
- Canon, Prime 50 mm f/1.4 USM Lens: $399.00
- Canon, Prime 50 mm f/1.2 USM Lens: $1,549.00

Depth of Field

- f/2.8
- f/5.6
- f/11
- f/32
Circle of Confusion

Real lenses do not focus all rays perfectly.

Thus, at best focus, a point is imaged as a spot rather than a point.

The smallest such spot that a lens can produce is often referred to as the circle of least confusion.

Perspective Views of Boxes

- How many vanishing points are there in each image?
- In each image, does the view direction point up or down or is it horizontal? How can you tell when it's horizontal?
Quiz

In the equation: \( \vec{a} = \vec{b} \times \vec{c} / \|\vec{b} \times \vec{c}\| \)

\( \vec{a}, \vec{b} \) and \( \vec{c} \) are vectors. This means they have both magnitude and direction. What can be said about the magnitude and direction of \( \vec{a} \)?

Virtual Pinhole-Camera Viewing System

User Input:
- The eye point, \( e \).
- The look-at point, \( l \).
- The up vector, \( \text{up} \).
- The view-plane distance \( d \).
Primary-Ray Calculation

The \((x_v, y_v)\) coordinates of a sample point \(p\) on the pixel in row \(r\) and column \(c\) are:

\[
x_v = s \left( c - \frac{h_{res}}{2} + p_x \right)
\]

\[
y_v = s \left( r - \frac{v_{res}}{2} + p_y \right)
\]

The primary-ray direction \(d\) is:

\[
\vec{d} = x_v \vec{u} + y_v \vec{v} - d \vec{w}
\]

3D Translation

A translation is a geometric transformation that moves every point of a figure or a space by the same amount in a given direction.

A translation transformation does not change the object's shape or size.

\[
\begin{bmatrix}
  d_x \\
  d_y \\
  d_z
\end{bmatrix} +
\begin{bmatrix}
  x \\
  y \\
  z
\end{bmatrix} =
\begin{bmatrix}
  d_x + x \\
  d_y + y \\
  d_z + z
\end{bmatrix}
\]
3D Scaling

Uniform scaling is a linear transformation that enlarges (increases) or shrinks (diminishes) objects by a scale factor that is the same in all directions.

The result of uniform scaling is similar (same shape different size) to the original.

\[
\begin{bmatrix}
  r & 0 & 0 \\
  0 & r & 0 \\
  0 & 0 & r
\end{bmatrix}
\begin{bmatrix}
  x \\
  y \\
  z
\end{bmatrix}
= 
\begin{bmatrix}
  r \cdot x \\
  r \cdot y \\
  r \cdot z
\end{bmatrix}
\]

Uniform scaling

\[
\begin{bmatrix}
  r_x & 0 & 0 \\
  0 & r_y & 0 \\
  0 & 0 & r_z
\end{bmatrix}
\begin{bmatrix}
  x \\
  y \\
  z
\end{bmatrix}
= 
\begin{bmatrix}
  r_x \cdot x \\
  r_y \cdot y \\
  r_z \cdot z
\end{bmatrix}
\]

Non-uniform scaling

3D Rotation about the x-axis

\[
R_x(\alpha) =
\begin{bmatrix}
  1 & 0 & 0 \\
  0 & \cos \alpha & -\sin \alpha \\
  0 & \sin \alpha & \cos \alpha
\end{bmatrix}
\begin{bmatrix}
  x \\
  y \\
  z
\end{bmatrix}
= 
\begin{bmatrix}
  1 \cdot x + 0 \cdot y + 0 \cdot z \\
  0 \cdot x + (\cos \alpha) \cdot y - (\sin \alpha) \cdot z \\
  0 \cdot x + (\sin \alpha) \cdot y + (\cos \alpha) \cdot z
\end{bmatrix}
= 
\begin{bmatrix}
  x \\
  (\cos \alpha) \cdot y - (\sin \alpha) \cdot z \\
  (\sin \alpha) \cdot y + (\cos \alpha) \cdot z
\end{bmatrix}
\]
3D Rotation about the $y$-axis and $z$-axis

\[
R_y(\beta) = \begin{bmatrix}
\cos \beta & 0 & \sin \beta \\
0 & 1 & 0 \\
-\sin \beta & 0 & \cos \beta
\end{bmatrix}
\begin{bmatrix}
x \\
y \\
z
\end{bmatrix}
\]

\[
R_z(\gamma) = \begin{bmatrix}
\cos \gamma & -\sin \gamma & 0 \\
\sin \gamma & \cos \gamma & 0 \\
0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
x \\
y \\
z
\end{bmatrix}
\]