Today

- Windowing and Viewing Transformations
  - Windows and viewports
  - Orthographic projection
  - Perspective projection
Screen Space

- Monitor has some number of pixels
  - e.g. 1024 x 768
- Some sub-region used for given program
  - You call it a window
  - Let's call it a viewport instead

![Diagram of viewport](image)

Screen Space

- May not really be a “screen”
  - Image file
  - Printer
  - Other
  - Little pixel details
  - Sometimes odd
    - Upside down
    - Hexagonal

From Shirley textbook.
Screen Space

- Viewport is somewhere on screen
  - You probably don’t care where
  - Window System likely manages this detail
  - Sometimes you care exactly where
- Viewport has a size in pixels
  - Sometimes you care (images, text, etc.)
  - Sometimes you don’t (using high-level library)

Canonical View Space

- Canonical view region
  - 2D: [-1,-1] to [+1,+1]
**Canonical View Space**

- Canonical view region
  - 2D: [-1,-1] to [+1,+1]

\[
\begin{bmatrix}
  x' \\
  y' \\
  1
\end{bmatrix} = \begin{bmatrix}
  \frac{n_x}{2} & 0 & \frac{n_x-1}{2} \\
  0 & -\frac{n_y}{2} & \frac{n_y-1}{2} \\
  0 & 0 & 1
\end{bmatrix} \begin{bmatrix}
  x \\
  y \\
  1
\end{bmatrix}
\]

**Canonical View Space**

- Canonical view region
  - 2D: [-1,-1] to [+1,+1]
  - Define arbitrary window and define objects
  - Transform window to canonical region
  - Do other things (we’ll see clipping latter)
  - Transform canonical to screen space
  - Draw it.
**Canonical View Space**

World Coordinates (Meters)  Canonical  Screen Space (Pixels)

Note distortion issues...

---

**Projection**

- Process of going from 3D to 2D
- Studies throughout history (e.g. painters)
- Different types of projection
  - Linear
    - Orthographic
    - Perspective
  - Nonlinear
Projection

- Process of going from 3D to 2D
- Studies throughout history (e.g. painters)
- Different types of projection
  - Linear
    - Orthographic
    - Perspective
  - Nonlinear

Many special cases in books just one of these two...

Orthographic is special case of perspective...
Perspective Projections

Linear Projection

- Projection onto a planar surface
- Projection directions either
  - Converge to a point
  - Are parallel (converge at infinity)
Linear Projection

- A 2D view

Perspective

Orthographic

Linear Projection

Orthographic

Perspective
Linear Projection

Orthographic

Perspective

Note how different things can be seen. Parallel lines “meet” at infinity.

A 2D view
Orthographic Projection

- No foreshortening
- Parallel lines stay parallel
- Poor depth cues

Canonical View Space

- Canonical view region
  - 3D: [-1,-1,-1] to [+1,+1,+1]
- Assume looking down -Z axis
  - Recall that “Z is in your face”
Orthographic Projection

- Convert arbitrary view volume to canonical
Orthographic Projection

- Step 1: translate center to origin
- Step 2: rotate view to -Z and up to +Y
Orthographic Projection

- Step 1: translate center to origin
- Step 2: rotate view to -Z and up to +Y
- Step 3: center view volume

Orthographic Projection

- Step 1: translate center to origin
- Step 2: rotate view to -Z and up to +Y
- Step 3: center view volume
- Step 4: scale to canonical size
Orthographic Projection

- Step 1: translate center to origin
- Step 2: rotate view to -$Z$ and up to $+Y$
- Step 3: center view volume
- Step 4: scale to canonical size

\[ M = S \cdot T_2 \cdot R \cdot T_1 \]
Perspective Projection

- Foreshortening: further objects appear smaller
- Some parallel line stay parallel, most don’t
- Lines still look like lines
- $Z$ ordering preserved (where we care)

Pinhole a.k.a center of projection

Image from D. Forsyth
Perspective Projection

Foreshortening: distant objects appear smaller

Perspective Projection

- Vanishing points
  - Depend on the scene
  - Not intrinsic to camera

“One point perspective”
Perspective Projection

- Vanishing points
  - Depend on the scene
  - Not intrinsic to camera

“Two point perspective”

Perspective Projection

- Vanishing points
  - Depend on the scene
  - Not intrinsic to camera

“Three point perspective”
Perspective Projection

![Diagram of Perspective Projection]

View Frustum

Distance to image plane

Top $i$

Bottom $b$

Near $n$

Far $f$

Center

Up

Y

-Z

i
Perspective Projection

- Step 1: Translate center to origin
- Step 2: Rotate view to -Z, up to +Y
Perspective Projection

- Step 1: Translate center to origin
- Step 2: Rotate view to -Z, up to +Y
- Step 3: Shear center-line to -Z axis
- Step 4: Perspective
Perspective Projection

- Step 1: Translate center to origin
- Step 2: Rotate view to \(-Z\), up to \(+Y\)
- Step 3: Shear center-line to \(-Z\) axis
- Step 4: Perspective

\[
\begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & \frac{i + f}{i} & f \\
0 & 0 & -\frac{1}{i} & 0
\end{bmatrix}
\]

Perspective Projection

- Step 4: Perspective
  - Points at \(z = -i\) stay at \(z = -i\)
  - Points at \(z = -f\) stay at \(z = -f\)
  - Points at \(z = 0\) goto \(z = \pm \infty\)
  - Points at \(z = -\infty\) goto \(z = -(i+f)\)

  - \(x\) and \(y\) values divided by \(-z/i\)

  - Straight lines stay straight
  - Depth ordering preserved in \([-i,-f]\)
  - Movement along lines distorted
Perspective Projection

- Step 4: Perspective
  - Points at $z=-i$ stay at $z=-i$
  - Points at $z=-f$ stay at $z=-f$
  - Points at $z=0$ go to $z=\pm\infty$
  - Points at $z=-\infty$ go to $z=-(i+f)$
  - $x$ and $y$ values divided by $-z/i$
  - Straight lines stay straight
  - Depth ordering preserved in $[-i,-f]$
  - Movement along lines distorted

$$\begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & i+f & 0 \\
0 & 0 & \frac{-1}{i} & 0
\end{bmatrix}$$

From Shirley textbook.

Perspective Projection

From Shirley textbook.
Perspective Projection

![Diagram of perspective projection]

Some horizontal lines

View vector

Near

Far

Top

"Eye" plane

39

Perspective Projection

39

[1 0 0 0]
0 1 0 0
0 0 i+f f
0 0 -1 i
0 0 0

40

Perspective Projection

40
Perspective Projection

Visualizing division of $x$ and $y$ but not $z$

Motion in $x,y$
Perspective Projection

Note that points on near plane fixed

Recall that points on far plane will stay there...
Perspective Projection

When we also divide $z$ points must remain on straight lines

Lines extend outside view volume
Perspective Projection

Motion in $z$

Perspective Projection

Motion in $z$
Perspective Projection

Motion in $z$

Total motion
Perspective Projection

- Step 1: Translate center to orange
- Step 2: Rotate view to -Z, up to +Y
- Step 3: Shear center-line to -Z axis
- Step 4: Perspective
- Step 5: center view volume
- Step 6: scale to canonical size

\[ M = M_o \cdot M_p \cdot M_v \]
There are other ways to set up the projection matrix

- View plane at $z=0$ zero
- Looking down another axis
- etc...

Functionally equivalent

Vanishing Points

Consider a ray:

$$r(t) = p + t\ d$$
Vanishing Points

- Ignore Z part of matrix
- X and Y will give location in image plane
- Assume image plane at z=-i

\[
\begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
\text{whatever} & & & \\
0 & 0 & -1 & 0
\end{bmatrix}
\rightarrow
\begin{bmatrix}
I_x \\
I_y \\
I_w
\end{bmatrix}
= \begin{bmatrix}
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & -1
\end{bmatrix}
\begin{bmatrix}
x \\
y \\
z
\end{bmatrix}
\]

Vanishing Points

\[
\begin{bmatrix}
I_x \\
I_y \\
I_w
\end{bmatrix}
= \begin{bmatrix}
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & -1
\end{bmatrix}
\begin{bmatrix}
x \\
y \\
z
\end{bmatrix}
= \begin{bmatrix}
x \\
y \\
-z
\end{bmatrix}
\]

\[
\begin{bmatrix}
I_x / I_w \\
I_y / I_w
\end{bmatrix}
= \begin{bmatrix}
-x / z \\
-y / z
\end{bmatrix}
\]
Vanishing Points

- Assume $d_z = -1$

$$\begin{bmatrix} I_x / I_w \\ I_y / I_w \end{bmatrix} = \begin{bmatrix} -x / z \\ -y / z \end{bmatrix} = \begin{bmatrix} \frac{p_x + td_x}{-p_z + t} \\ \frac{p_y + td_y}{-p_z + t} \end{bmatrix}$$

$$\lim_{t \to \pm \infty} = \begin{bmatrix} d_x \\ d_y \end{bmatrix}$$

Vanishing Points

- All lines in direction $d$ converge to same point in the image plane -- the vanishing point
- Every point in plane is a v.p. for some set of lines
- Lines parallel to image plane ($d_z = 0$) vanish at infinity

What’s a horizon?
Perspective Tricks

Right Looks Wrong (Sometimes)

From Correction of Geometric Perceptual Distortions in Pictures, Zorin and Barr SIGGRAPH 1995
The Ambassadors by Hans Holbein the Younger

Ray Picking

- Pick object by picking point on screen
- Compute ray from pixel coordinates.
Ray Picking

- Transform from World to Screen is:
  \[
  \begin{bmatrix}
  I_x \\
  I_y \\
  I_z \\
  I_w
  \end{bmatrix}
  = \mathbf{M}
  \begin{bmatrix}
  W_x \\
  W_y \\
  W_z \\
  W_w
  \end{bmatrix}
  \]

- Inverse:
  \[
  \begin{bmatrix}
  W_x \\
  W_y \\
  W_z \\
  W_w
  \end{bmatrix}
  = \mathbf{M}^{-1}
  \begin{bmatrix}
  I_x \\
  I_y \\
  I_z \\
  I_w
  \end{bmatrix}
  \]

- What Z value?

---

Ray Picking

- Recall that:
  - Points at \( z=-i \) stay at \( z=-i \)
  - Points at \( z=-f \) stay at \( z=-f \)

\[
\mathbf{r}(t) = \mathbf{p} + t \mathbf{d}
\]

\[
\mathbf{r}(t) = \mathbf{a}_w + t(\mathbf{b}_w - \mathbf{a}_w)
\]

\[
\mathbf{a}_s = [s_x, s_y, -i]
\]

\[
\mathbf{b}_s = [s_x, s_y, -f]
\]

depends on screen details, YMMV
General idea should translate...
Depth Distortion

- Recall depth distortion from perspective
  - Interpolating in screen space different than in world
  - Ok, for shading (mostly)
  - Bad for texture

Ok, for shading (mostly)
Bad for texture
Screen
World
Half way in world space
Half way in screen space

Depth Distortion

\[ S_1 = \frac{P_1}{h_1} \]
\[ S_2 = \frac{P_2}{h_2} \]
\[ S_3 = \frac{P_3}{h_3} \]
\[ S_4 = \frac{P_4}{h_4} \]

\[ P_1 \]
\[ P_2 \]
\[ P_3 \]
\[ P_4 \]
Depth Distortion

We know the $S_i$, $P_i$, and $b_i$, but not the $a_i$. 

\[ X = \sum_i S_i b_i \]
\[ Q = \sum_i P_i a_i \]

\[ X = Q/h = \left( \sum_i P_i a_i \right) / \left( \sum_j h_j a_j \right) \]
\[
S_1 = \frac{P_1}{h_1} \\
S_2 = \frac{P_2}{h_2} \\
S_3 = \frac{P_3}{h_3} \\
S_4 = \frac{P_4}{h_4} \\

X = \sum_i S_ib_i \\

\sum_i S_ib_i = \left( \sum_i P_ia_i \right) / \left( \sum_j h_ja_j \right) \\

\sum_i P_ib_i/h_i = \left( \sum_i P_ia_i \right) / \left( \sum_j h_ja_j \right) \\

\text{Depth Distortion}
\]
Depth Distortion

Independent of given vertex locations.

\[ \sum_i P_i b_i / h_i = \left( \sum_i P_i a_i \right) / \left( \sum_j h_j a_j \right) \]

\[ b_i / h_i = a_i / \left( \sum_j h_j a_j \right) \quad \forall i \]

Linear equations in the \( a_i \).

\[ \left( \sum_j h_j a_j \right) b_i / h_i - a_i = 0 \quad \forall i \]
Depth Distortion

Linear equations in the $a_i$.

$$\left( \sum_j h_j a_j \right) b_i/h_i - a_i = 0 \quad \forall i$$

Not invertible so add some extra constraints.

$$\sum_i a_i = \sum_i b_i = 1$$

For a line:

$$a_1 = h_2 b_1 / (b_1 h_2 + h_1 b_2)$$

For a triangle:

$$a_1 = h_2 h_3 b_1 / (h_2 h_3 b_1 + h_1 h_3 b_2 + h_1 h_2 b_3)$$

Obvious Permutations for other coefficients.