Orthographic Projection

- How? Draw parallel lines from each object vertex
- The projection center is at infinite
- In short, use (x,y) coordinates, just drop z coordinates
Perspective Projection

- After setting view volume, then projection transformation
- Projection?
  - **Classic**: Converts 3D object to corresponding 2D on screen
  - How? Draw line from object to projection center
  - Calculate where each intersects projection plane
The Problem with Classic Projection

- Keeps \((x,y)\) coordinates for drawing, drops \(z\)
- We may need \(z\). Why?

\[
\begin{align*}
x_p &= x \\
y_p &= y \\
z_p &= 0
\end{align*}
\]

Classic Projection Loses \(z\) value
Normalization: Keeps z Value

- Most graphics systems use *view normalization*
- **Normalization**: convert all other projection types to orthogonal projections with the *default view volume*
Parallel Projection

- normalization $\Rightarrow$ find 4x4 matrix to transform user-specified view volume to canonical view volume (cube)

\[ \text{glOrtho}(\text{left}, \text{right}, \text{bottom}, \text{top}, \text{near}, \text{far}) \]
Parallel Projection: Ortho

- Parallel projection: 2 parts
  1. **Translation**: centers view volume at origin
Parallel Projection: Ortho

2. **Scaling:** reduces user-selected cuboid to canonical cube (dimension 2, centered at origin)
Parallel Projection: Ortho

- Translation lines up midpoints: E.g. midpoint of $x = (\text{right} + \text{left})/2$
- Thus translation factors:
  $-(\text{right} + \text{left})/2, -(\text{top} + \text{bottom})/2, -(\text{far} + \text{near})/2$
- Translation matrix:

$$
\begin{pmatrix}
1 & 0 & 0 & -(right + left)/2 \\
0 & 1 & 0 & -(top + bottom)/2 \\
0 & 0 & 1 & -(far + near)/2 \\
0 & 0 & 0 & 1
\end{pmatrix}
$$
Parallel Projection: Ortho

- Scaling factor: ratio of ortho view volume to cube dimensions
- Scaling factors: \( \frac{2}{(\text{right} - \text{left})} \), \( \frac{2}{(\text{top} - \text{bottom})} \), \( \frac{2}{(\text{far} - \text{near})} \)
- Scaling Matrix M2:

\[
\begin{pmatrix}
\frac{2}{\text{right} - \text{left}} & 0 & 0 & 0 \\
0 & \frac{2}{\text{top} - \text{bottom}} & 0 & 0 \\
0 & 0 & \frac{2}{\text{far} - \text{near}} & 0 \\
0 & 0 & 0 & 1
\end{pmatrix}
\]
Parallel Projection: Ortho

Concatenating \textbf{Translation} \times \textbf{Scaling}, we get Ortho Projection matrix

\[
P = ST = \begin{bmatrix}
\frac{2}{\text{right} - \text{left}} & 0 & 0 & \frac{-\text{right} - \text{left}}{	ext{right} - \text{left}} \\
0 & \frac{2}{\text{top} - \text{bottom}} & 0 & \frac{-\text{top} + \text{bottom}}{\text{top} - \text{bottom}} \\
0 & 0 & \frac{2}{\text{far} - \text{near}} & \frac{-\text{far} + \text{near}}{\text{far} - \text{near}} \\
0 & 0 & 0 & 1
\end{bmatrix}
\]
Final Ortho Projection

- Set $z = 0$
- Equivalent to the homogeneous coordinate transformation

$$M_{\text{orth}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Hence, general orthogonal projection in 4D is

$$P = M_{\text{orth}}ST$$
References

- Interactive Computer Graphics (6\textsuperscript{th} edition), Angel and Shreiner
- Computer Graphics using OpenGL (3\textsuperscript{rd} edition), Hill and Kelley