Projections

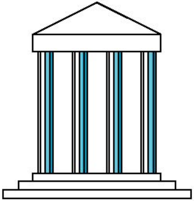
Computer Graphics and Visualization

Pedher Johansson
Department of Computer Science
Fall 2016

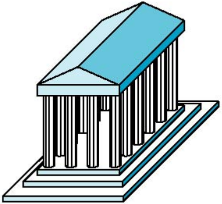
Planar Geometric Projections

- Standard projections project onto a plane
- Non-planar projections are needed for applications such as map construction

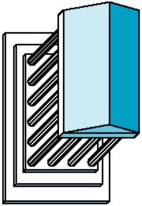
Classical Projections



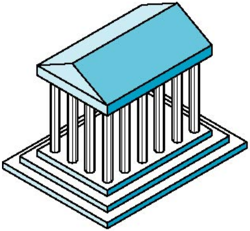
Front elevation



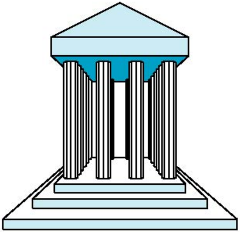
Elevation oblique



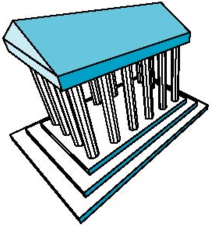
Plan oblique



Isometric



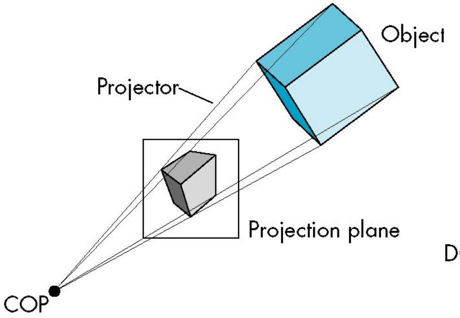
One-point perspective

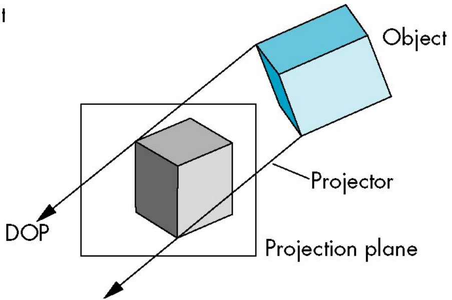


Three-point perspective

Perspective vs Parallel

- Projectors are lines that either
  - converge at a center of projection
  - are parallel





Perspective vs Parallel

- Classical viewing developed different techniques for drawing each type of projection
- Fundamental distinction is between parallel and perspective viewing even though mathematically parallel viewing is the limit of perspective viewing
- Computer graphics treats all projections the same and implements them with a single pipeline

Classes of Paralell Projections

Orthographic Projections Projection plane orthogonal to projectors. Usualy the projection plane is aligned orthogonal to the principal axes.

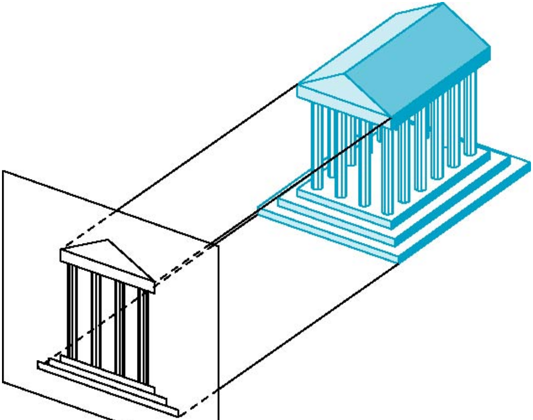
Axonometric Projections Principal axes are aligned to each other with some angle.

Oblique Projection Projectors hits projection plane at an angle $\neq 90^\circ$

Orthographic Projection

Projectors are orthogonal to projection plane

- Usualy the projection plane is aligned with the principal axes, giving a front, top or side view.



Advantages and Disadvantages

- Preserves both distances and angles
  - Shapes preserved
  - Can be used for measurements
    - Building plans
    - Manuals
- Cannot see what object really looks like because many surfaces hidden from view
  - Often we add the isometric

Axonometric Projections

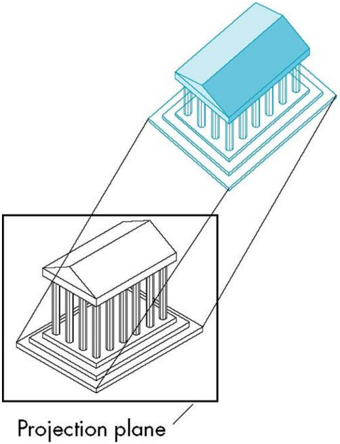
- Allow projection plane to move relative to object
- Still Orthographic!

Classify by how many angles of a corner of a projected cube are the same

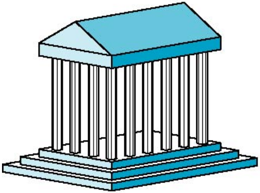
none: trimetric

two: dimetric

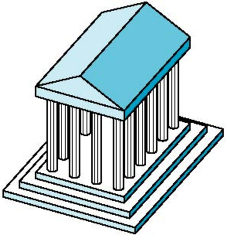
three: isometric



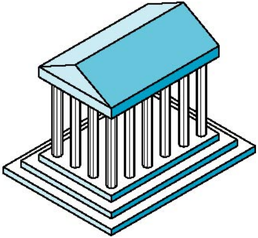
Types of Axonometric Projections



Dimetric



Trimetric



Isometric

Advantages and Disadvantages

- Lines are scaled (foreshortened) but can find scaling factors
- Lines preserved but angles are not
  - Projection of a circle in a plane not parallel to the projection plane is
- Can see three principal faces of a box-like object
- Some optical illusions possible
  - Parallel lines appear to diverge
- Does not look real because far objects are scaled the same as near objects
- Used in CAD applications

Orthographic Projection Matrix

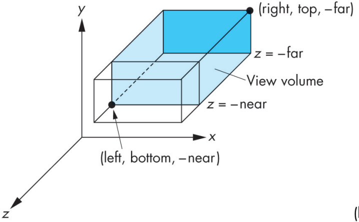
- Orthographic projection
- Never done in the vertex shader (depth needed in the pipeline)

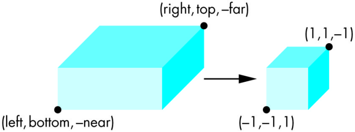
$$q_p = \mathbf{M}_{\mathrm{orth}} q$$

$$q = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \qquad \mathbf{M}_{\mathrm{orth}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Orthographic Normalization

We want to define a volume (frustum) that should be projected to our screen. Not just whats inside the NDC cube.





- Translate and scale

Orthographic Normalized Projection

$$\mathbf{T} = \mathbf{T}(-(right + left)/2, -(top + bottom)/2, +(far + near)/2)$$

$$\mathbf{S} = \mathbf{S}(2/(right - left), 2/(top - bottom), 2/(near - far))$$

$$\mathbf{ST} = \begin{bmatrix} \frac{2}{right-left} & & & -\frac{left+right}{right-left} \\ & \frac{2}{top-bottom} & & -\frac{top+bottom}{top-bottom} \\ & & -\frac{2}{far-near} & -\frac{far+near}{far-near} \\ & & & 1 \end{bmatrix}$$

$$\mathbf{ST} = \mathbf{P}_{\mathrm{orth}}$$

Aspect Ratio

- The ratio between *top* − *bottom* and *right* − *left*, should be the same as the aspect ratio of the viewport.
- The frustum first gets disproptional in NDC, but then scaled out again when mapped on the viewport.
- the aspect ratio is usually defined by the ration between width and height (e.g. 4:3 or 16:9)
- *left*, *right*, *top* and *bottom* can also be calculated from near and the field of view angle

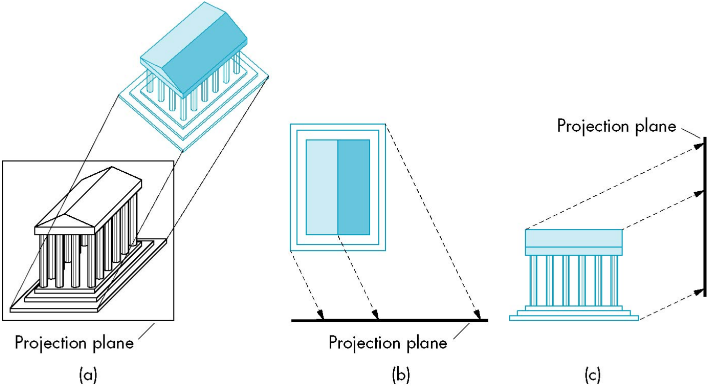
Transformations of Objects

1. Transformation of objects, $\mathbf{T}_{object}$
2. Align the set to the camera, $\mathbf{V}$
3. Projection and normalization to NDC, $\mathbf{P}$
4. Orthogonal projection of the NDC cube, $\mathbf{M}_{\mathrm{orth}}$
5. Scaling to view frame (screen coordinates)

- 1 done on CPU or in shader on GPU
- 2-3 done in shader on GPU
- 4-5 done by OpenGL on GPU

Oblique Projection

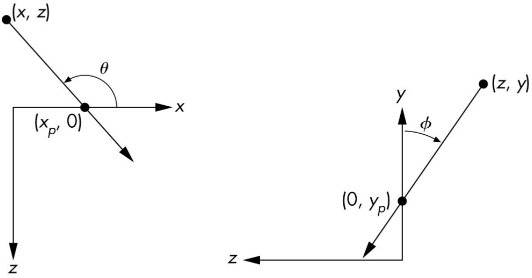
Arbitrary relationship between projectors and projection plane



Advantages and Disadvantages

- Can pick the angles to emphasize a particular face
- Lengths and angles in faces parallel to projection plane are preserved
- Lengths in faces orthogonal to the projection plane are scaled

Oblique Projection



(a) (b)

$$\tan\,\theta = \frac{z}{x_p - x}$$

$$x_p = x + z\cot\,\theta$$

$$y_p = y + z\cot\,\phi$$

Oblique Projection

$$x_p = x + z\cot\theta$$

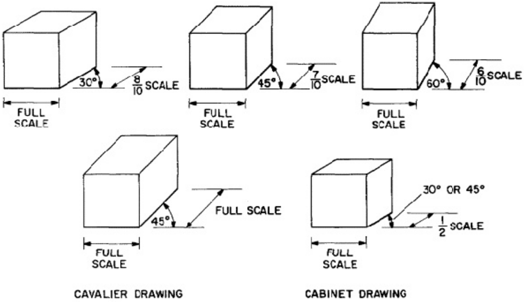
$$y_p = y + z\cot\phi$$

$$\mathbf{H}(\theta,\phi) = \begin{bmatrix} 1 & & \cot\theta & \\ & 1 & \cot\phi & \\ & & 1 & \\ & & & 1 \end{bmatrix}$$

$$\mathbf{STH}(\theta,\phi) = \mathbf{P}_{\text{oblique}}$$

Oblique Projection

- Projector angles are not intuitive
- Instead scaling and angle of the z-dimension is ofthe used



Oblique Projection

$$x_p = x + d \cdot z\cos\,\alpha$$

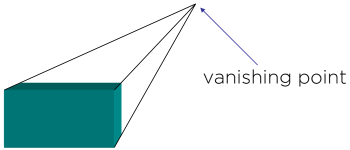
$$y_p = y + d \cdot z\sin\,\alpha$$

$$\mathbf{H}(\theta,\phi) = \begin{bmatrix} 1 & & d\cdot\cos\,\alpha & \\ & 1 & d\cdot\sin\,\alpha & \\ & & 1 & \\ & & & 1 \end{bmatrix}$$

- Cabinet: $d = 0.5$, Cavalier: $d = 1$

Perspective Projections

- Parallel lines (not parallel to the projection plan) on the object converge at a single point in the projection (the vanishing point)
- Drawing simple perspectives by hand uses these vanishing point(s)



Classes of Perspective Projections

- Classical they are defined by how the projecion plane is aligned with the
- Makes no differnce in math or implementaion.

One-Point Perspective

- One principal face parallel to projection plane
- One vanishing point for cube

One-Point Perspective

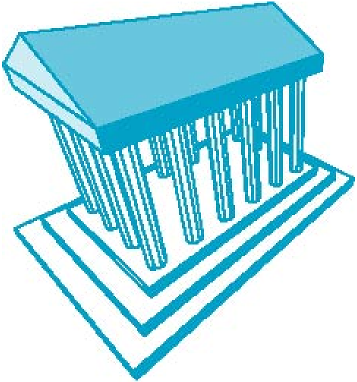
Two-Point Perspective

- On principal direction parallel to projection plane
- Two vanishing points for cube

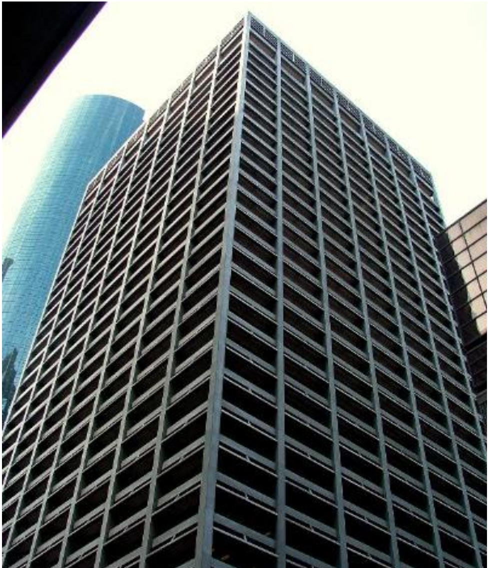
Two-Point Perspective

Three-Point Perspective

- No principal face parallel to projection plane
- Three vanishing points for cube



Three-Point Perspective

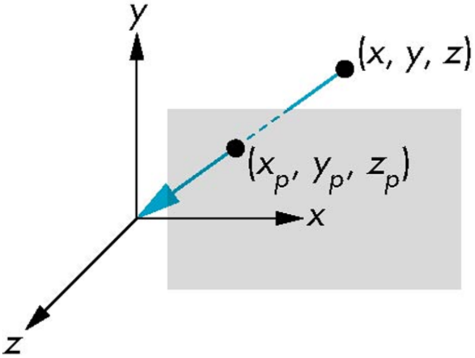


Advantages and Disadvantages

- Objects further from viewer are projected smaller than the same sized objects closer to the viewer (diminution)
  - Looks realistic
- Equal distances along a line are not projected into equal distances (nonuniform foreshortening)
- Angles preserved only in planes parallel to the projection plane
- More difficult to construct by hand than parallel projections (but not more difficult by computer)

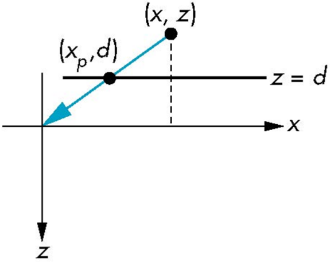
Simple Perspective

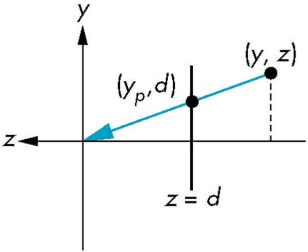
- Center of projection at the origin
- Projection plane $z = d$, $d < 0$



Perspective Equations

Consider top and side views





$$x_p = \frac{x}{z/d} \qquad y_p = \frac{y}{z/d} \qquad z_p = d$$

Homogeneous Coordinate Form

consider $q_p = \mathbf{M}q$ where $\mathbf{M} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/d & 0 \end{bmatrix}$

$$q = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \quad \Rightarrow q_p = \begin{bmatrix} x \\ y \\ z \\ z/d \end{bmatrix}$$

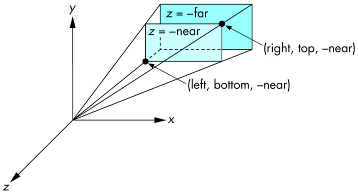
Perspective Division

- However $w \neq 1$, so we must divide by w to return from homogeneous coordinates
- This perspective division yields

$$x_p = \frac{x}{z/d} \qquad y_p = \frac{y}{z/d} \qquad z_p = d$$

the desired perspective equations

Perspective Matrix



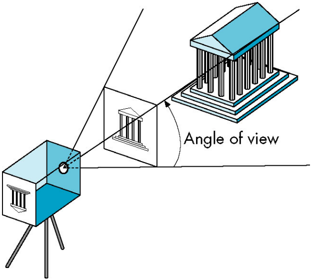
- How to fit that *frustum* into the NDC cube?

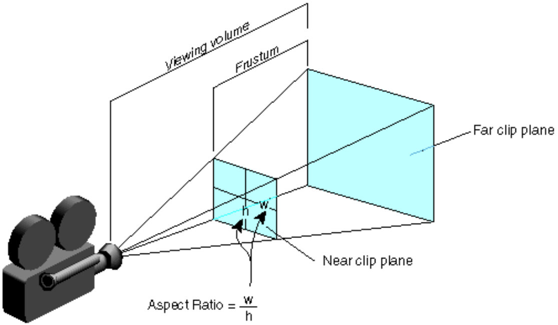
Perspective Matrix

- Make the frustum symetric (scew), $\mathbf{H}$
- Scale the back planes to the $z = \pm 1$ planes, $\mathbf{S}_{\mathrm{z}}$
- Scale the sides to the $x = \pm 1$ and $y = \pm 1$ planes, $\mathbf{S}_{\mathrm{xy}}$

$$\mathbf{P}_{\mathrm{persp}} = \mathbf{S}_{\mathrm{xy}}\mathbf{S}_{\mathrm{z}}\mathbf{H} = \begin{bmatrix} \frac{2 \cdot near}{right - left} & & \frac{right + left}{right - left} & \\ & \frac{2 \cdot near}{top - bottom} & \frac{top + bottom}{top - bottom} & \\ & & -\frac{far + near}{far - near} & \frac{-2 \cdot far \cdot near}{far - near} \\ & & -1 & \end{bmatrix}$$

Perspective Matrix





Perspective Matrix

- To simplify, we use

$$
\begin{aligned}
\mathit{left} &= -\mathit{right} \\
\mathit{bottom} &= -\mathit{top} \\
\mathit{top} &= \mathit{near} \cdot \tan\theta \\
\mathit{right} &= -\mathit{top} \cdot \mathit{aspect}
\end{aligned}
$$

$$
\mathbf{P}_{\mathrm{persp}} = \begin{bmatrix} \frac{\mathit{near}}{\mathit{right}} & & & \\ & \frac{\mathit{near}}{\mathit{top}} & & \\ & & -\frac{\mathit{far}+\mathit{near}}{\mathit{far}-\mathit{near}} & \frac{-2\cdot\mathit{far}\cdot\mathit{near}}{\mathit{far}-\mathit{near}} \\ & & -1 & \end{bmatrix}
$$

Transformations of Objects

1. Transformation of objects, $\mathbf{T}_{object}$
2. Align the set to the camera, $\mathbf{V}$
3. Projection and normalization to NDC, $\mathbf{P}$
4. Orthogonal projection of the NDC cube, $\mathbf{M}_{\mathbf{orth}}$
5. Scaling to view frame (screen coordinates)

- 1 done on CPU or in shader on GPU
- 2-3 done in shader on GPU
- 4-5 done by OpenGL on GPU

Taxonomy of Planar Geometric Projections

Parallel Projection

- Orthographic
  - Top
  - Front
  - Side
  - Axonometric
    - Isometric
- Oblique
  - Cabinet
  - Cavalier

Perspective Projections

- One point
- Two point
- Three point
- Camera model