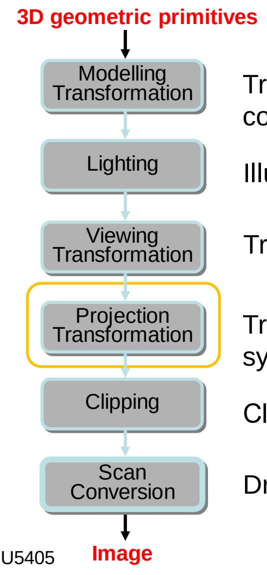
3D Graphics Programming Tools

Projection

The 3D rendering pipeline



3D geometric primitives

Modelling Transformation — Transform individual object coordinates into 3D world coordinate system

Lighting — Illuminate according to lighting and reflectance

Viewing Transformation — Transform 3D scene into 3D camera coordinate system

Projection Transformation — Transform 3D coordinates into 2D screen coordinate system

Clipping — Clip primitives outside camera’s view

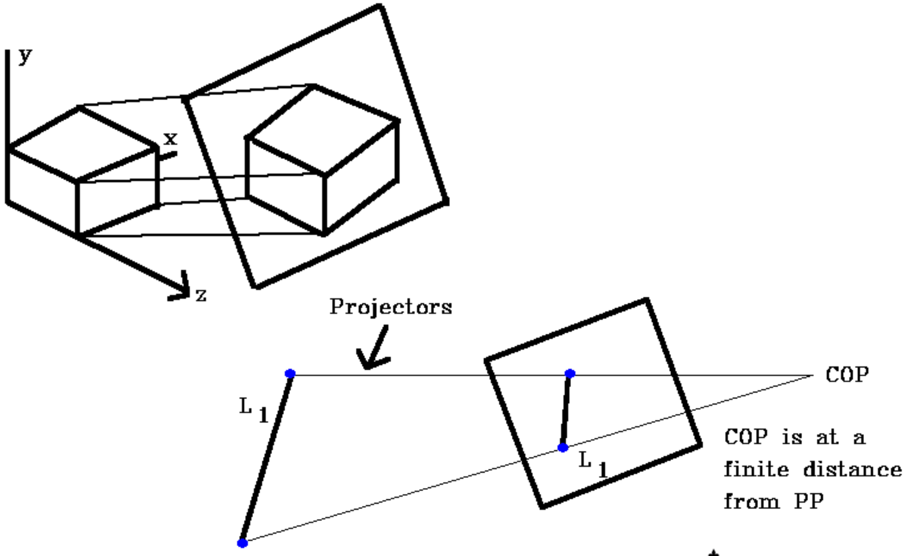
Scan Conversion — Draw pixels (includes texturing, hidden surface, ...)

Image

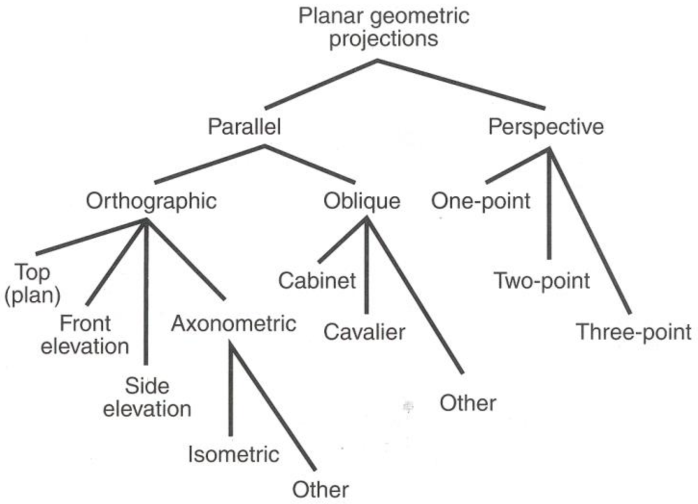
Today’s agenda

- Taxonomy of projections
- Parallel projection
- Perspective projection

Planar Geometric Projection

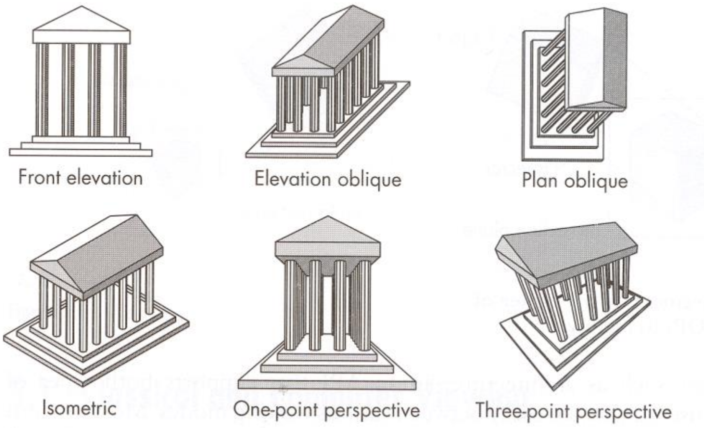


Taxonomy of projections



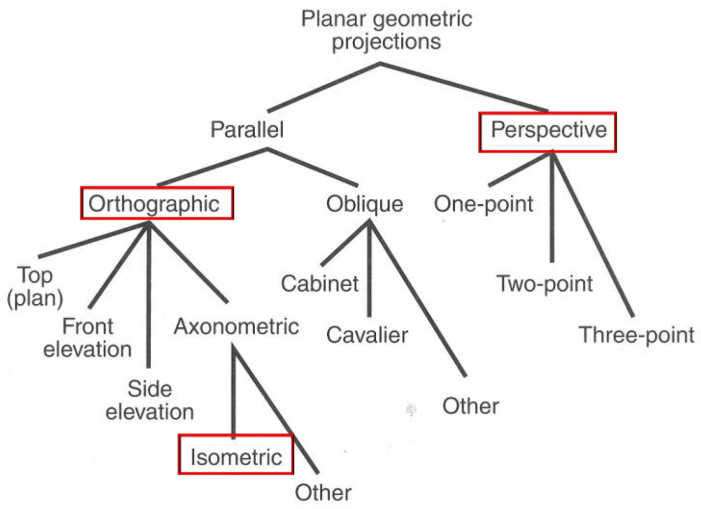
EBU5405

Classical projections



EBU5405

Taxonomy of projections



Planar geometric projections

Parallel

Perspective

Orthographic

Oblique

One-point

Top (plan)

Front elevation

Side elevation

Axonometric

Isometric

Other

Cabinet

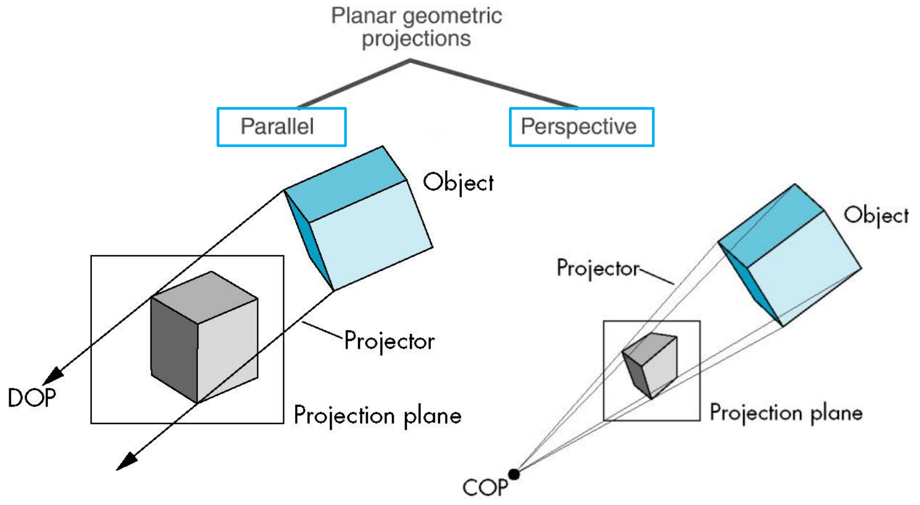
Cavalier

Other

Two-point

Three-point

Planar geometric projections



Planar geometric projections

Parallel

Perspective

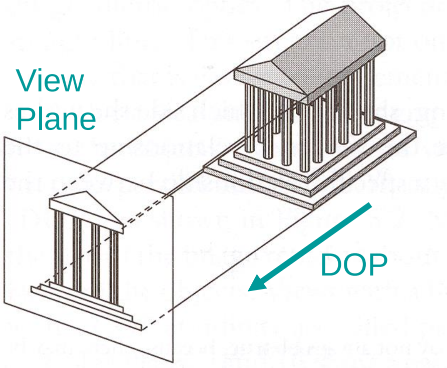
Today’s agenda

- Taxonomy of projections
- Parallel projection
- Perspective projection

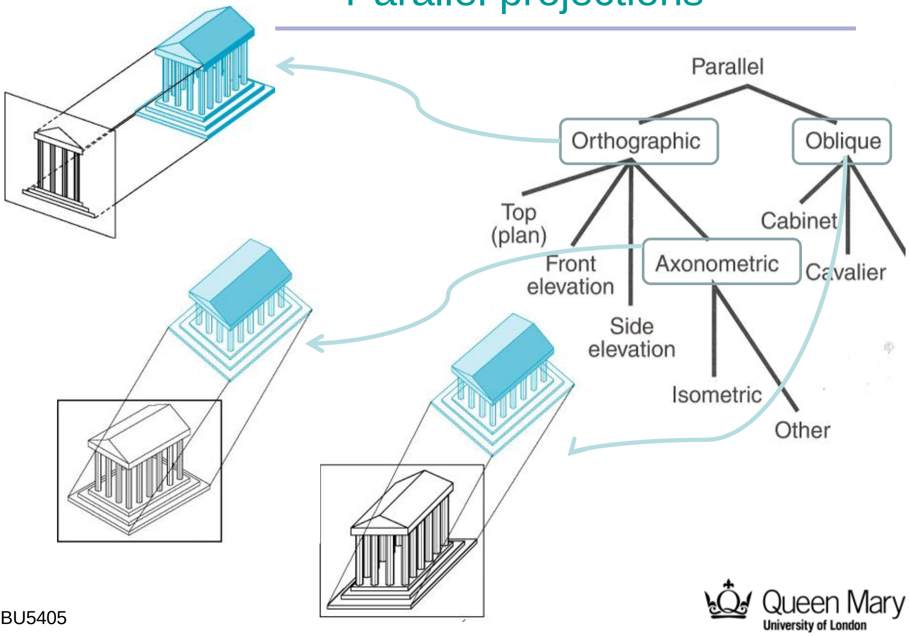
Parallel projection

Center of projection is at infinity

- Direction of projection (DOP) is the same for all points



Parallel projections

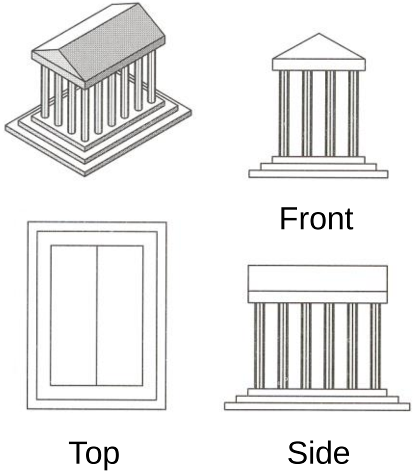


Orthographic projections

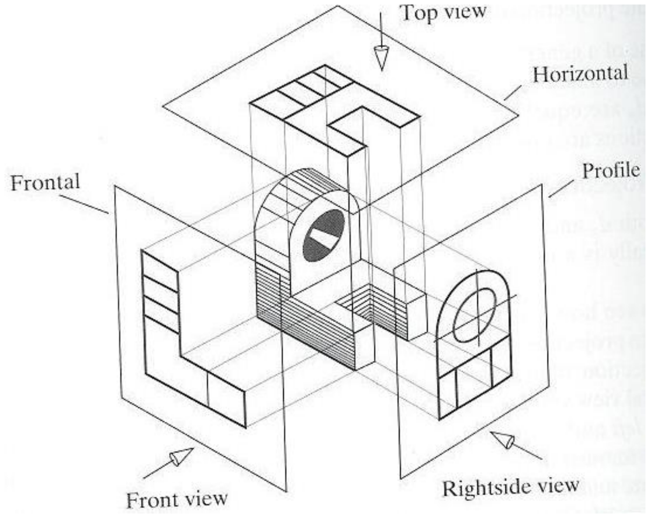
DOP is perpendicular to the view plane

- Advantage: you can make accurate measurements of image features in the two dimensions that remain.

- Disadvantage: images don't appear natural (i.e. they lack perspective foreshortening).

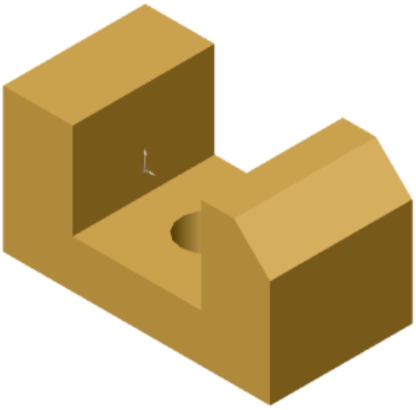


Orthographic projection



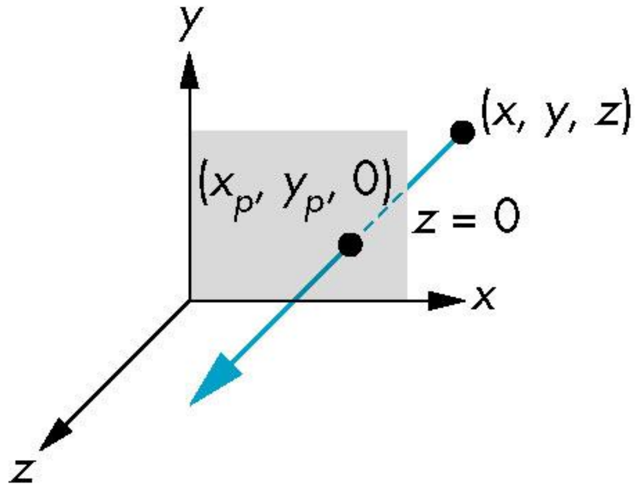
Exercise

Draw the top, front and right side views



Orthographic projection

$x_p = x$
$y_p = y$
$z_p = 0$



Orthographic projection

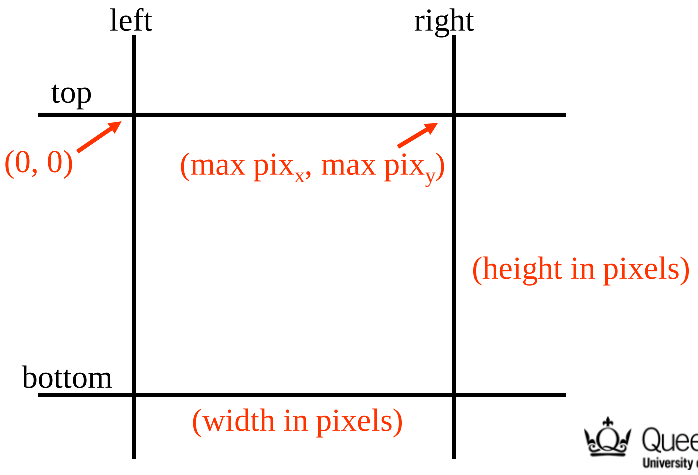
- Simple orthographic transformation

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \quad & \quad & \quad & \quad \\ & & & \\ & & & \\ & & & \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

- Notice that the parallel lines of the tiled floor remain parallel after orthographic projection.

Orthographic: screen space transformation

- glOrtho (left, right, bottom, top, near, far)



left

right

top

bottom

(0, 0)

(max pix$_x$, max pix$_y$)

(height in pixels)

(width in pixels)

Screen space transformation

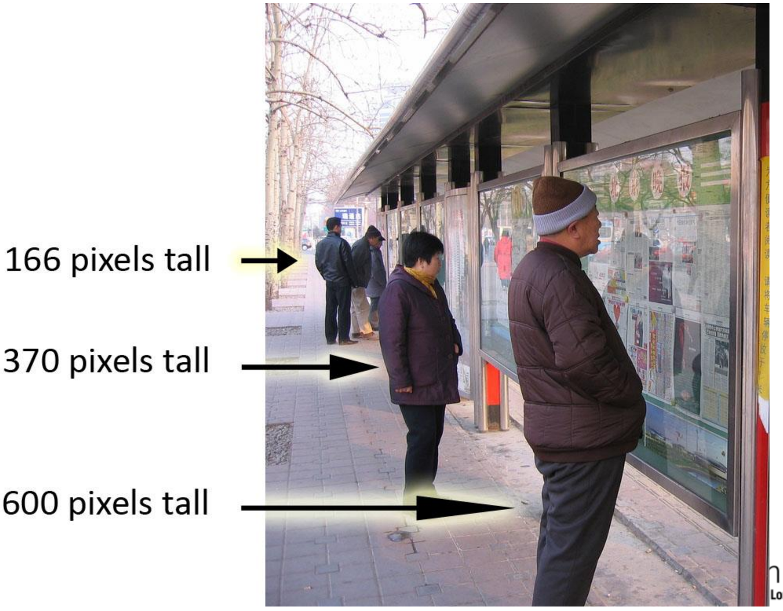
$$\begin{bmatrix}x'\\y'\\z'\\1\end{bmatrix} = \begin{bmatrix} \frac{width}{right-left} & 0 & 0 & \frac{-left\times width}{right-left}\\ 0 & \frac{height}{bottom-top} & 0 & \frac{-top\times height}{bottom-top}\\ 0 & 0 & \frac{z_{max}}{far-near} & \frac{-near\times z_{max}}{far-near}\\ 0 & 0 & 0 & 1\end{bmatrix}\begin{bmatrix}x\\y\\z\\1\end{bmatrix}$$

- This matrix scales and translates to accomplish the transition in units
  - Left, right, top, bottom refer to the viewing frustum (view volume) in modelling coordinates
  - width and height are in pixel units (viewport)

Today’s agenda

- Taxonomy of projections
- Parallel projection
- Perspective projection

Perspective



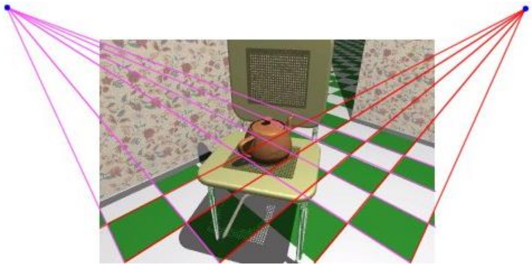
166 pixels tall

370 pixels tall

600 pixels tall

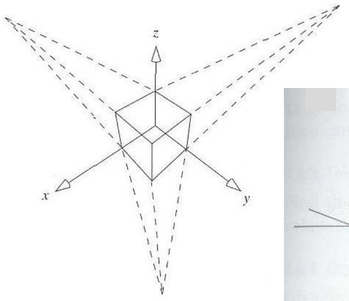
Perspective projection

- In the real world, objects exhibit perspective foreshortening
  - distant objects appear smaller
  - objects closer to viewer look larger
- Parallel lines appear to converge to single point (vanishing point)
- First discovered by Donatello, Brunelleschi, and Da Vinci during Renaissance

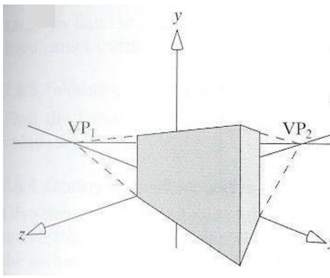


Perspective projection

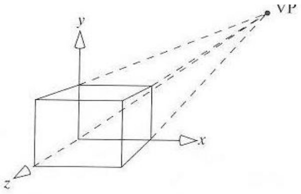
How many vanishing points?



3-point perspective

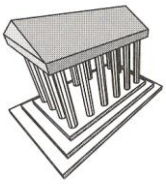


2-point perspective

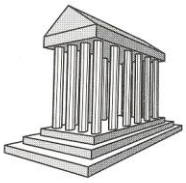


1-point perspective

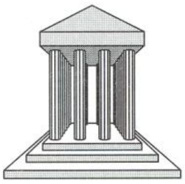
Perspective projection



3-point perspective



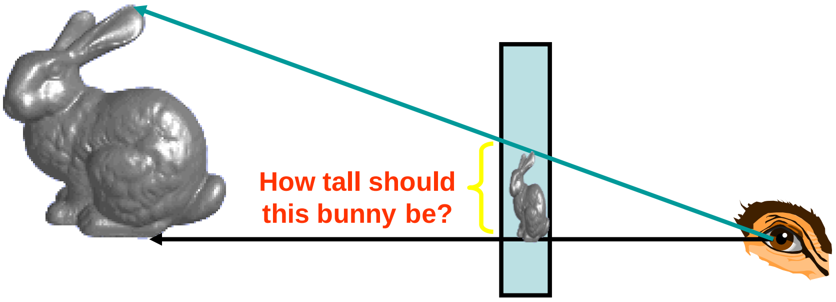
2-point perspective



1-point perspective

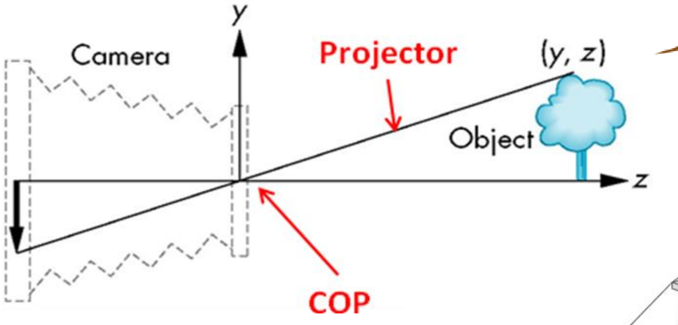
Perspective projection

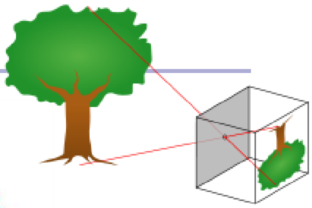
- 3-D graphics → think of the screen as a 2-D window onto the 3-D world

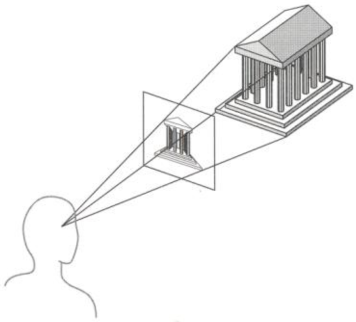


How tall should this bunny be?

Pinhole Camera



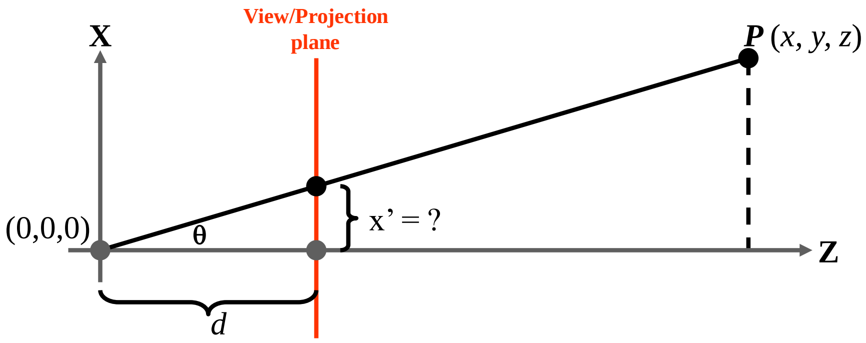




Synthetic Camera

- The geometry of the situation is that of similar triangles.

View from above



$$\frac{x'}{d} = \frac{x}{z}$$

What is x' ?

Perspective projection

- Desired result for a point $[x, y, z, 1]^T$ projected onto the view plane:

$$\frac{x'}{d} = \frac{x}{z}, \quad \frac{y'}{d} = \frac{y}{z}$$

$$x' = \frac{d \cdot x}{z} = \frac{x}{z/d}, \quad y' = \frac{d \cdot y}{z} = \frac{y}{z/d}, \quad z' = d$$

What could a matrix look like to do this?

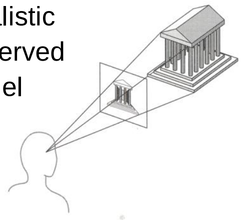
Perspective projection matrix

$$x' = \frac{x}{z/d}, \quad y' = \frac{y}{z/d}, \quad z' = d$$

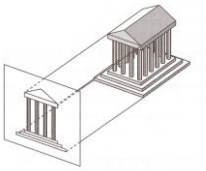
$$M_{perspective} = \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix}$$

Perspective vs. Parallel

- Perspective projection
  - + Size varies inversely with distance - looks realistic
  - – Distance and angles are not (in general) preserved
  - – Parallel lines do not (in general) remain parallel

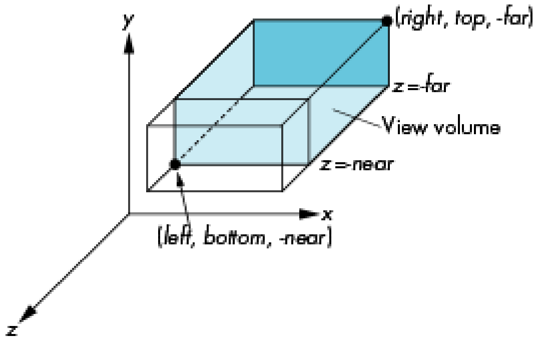


- Parallel projection
  - + Good for exact measurements
  - + Parallel lines remain parallel
  - – Angles are not (in general) preserved
  - – Less realistic looking



OpenGL Orthogonal Viewing

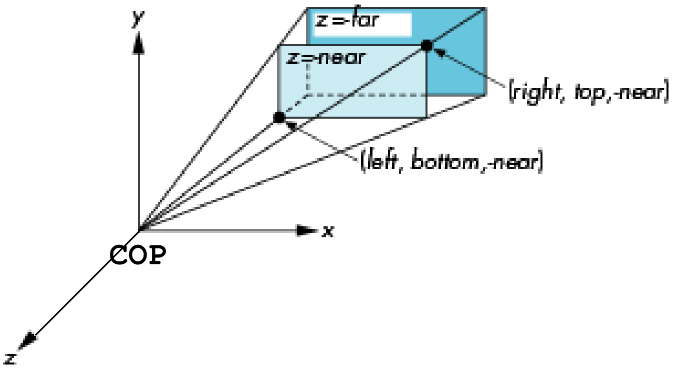
`glOrtho(left,right,bottom,top,near,far)`



`near` and **`far`** measured <u>from</u> camera

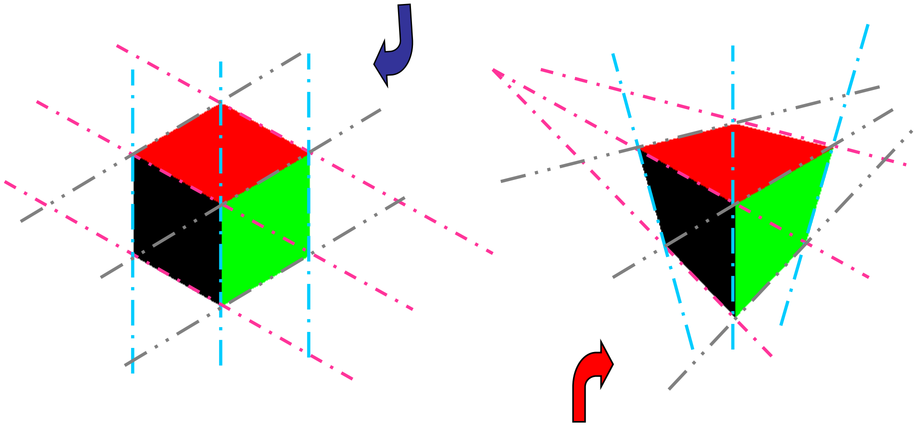
OpenGL Perspective

`glFrustum(left,right,bottom,top,near,far)`



“Isometric” view

`glOrtho(-2.0,2.0,-2.0,2.0,-2.0,2.0);`

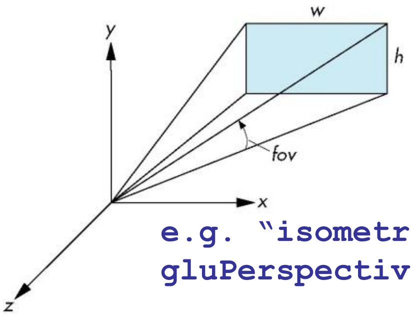


`glFrustum(-0.25,0.25,-0.25,0.25,0.2,2.0);`

Using Field of View

- With **`glFrustum`** it is often difficult to get the desired view
- **`gluPerpective(fov, aspect, near, far)`**

often provides a better interface



near plane

`aspect = w/h`

`e.g. “isometric” view of the cube:`
`gluPerspective(100,1.0,0.2,2.0);`

What did we learn?

- Taxonomy of projections
- Parallel projection
- Perspective projection