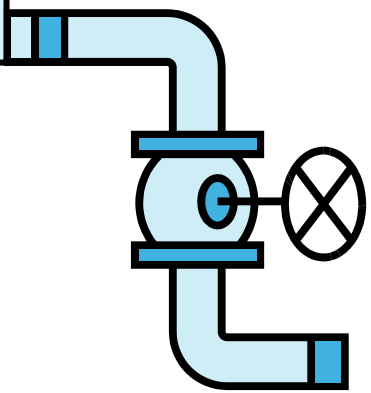
קורס גרפיקה ממוחשבת

2008 סמסטר ב'

ליאור שפירא

3D Polygon Rendering Pipeline



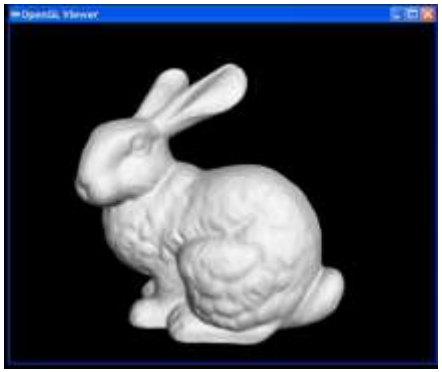
חלק מהשקפים מעובדים משקפים של פרדו דוראנד, טומס פנקהאוסר ודניאל כהן-אור

מה במצגת

- Overview
- The 3D rendering pipeline
- Transformations

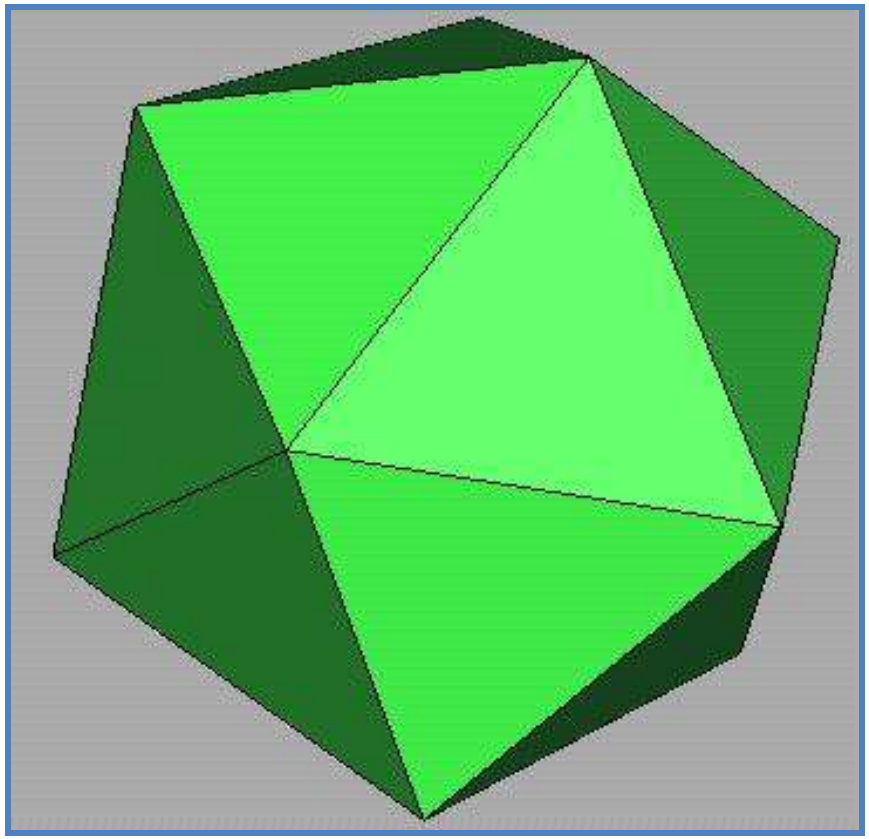
Rendering Scenarios

- אצווה (batch)
  - כל תמונה מיוצרת ברמת פירוט גבוהה ככל האפשר עבור סט ספציפי של פרמטרים
    - לוקח כמה זמן שצריך
    - שימושי לפוטוריאליזם, סרטים וכו'
- אינטראקטיבי
  - מייצרים תמונות בשבריר שנייה (לפחות 10 בשנייה) כאשר המשתמש שולט בפרמטרים של הרינדור
    - יש צורך להשיג את האיכות הגבוהה ביותר בהתחשב בזמן הנתון (הקצב הנדרש)
    - שימושי לויזואליזציות, משחקים וכו'



3D Polygon Rendering

- Many applications use rendering of 3D polygons with direct illumination

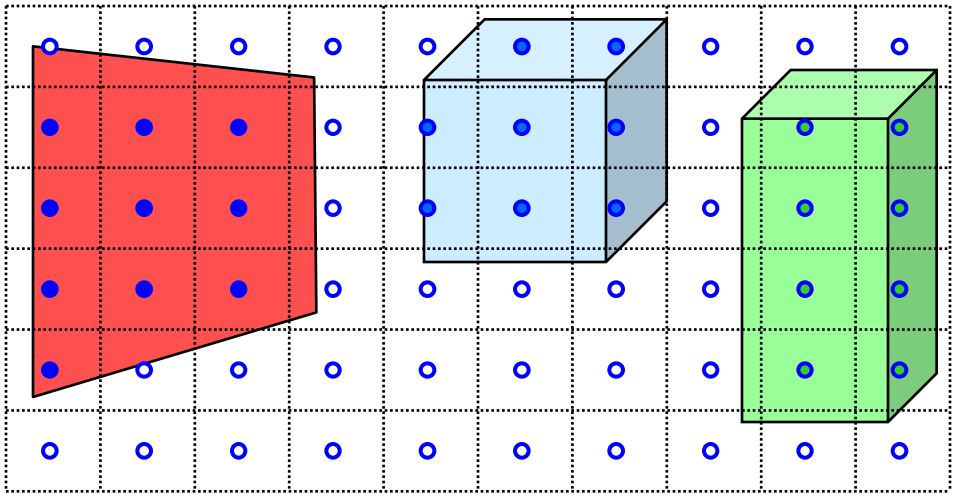


3D Polygon Rendering

- Many applications use rendering of 3D polygons with direct illumination

Ray Casting Revisited

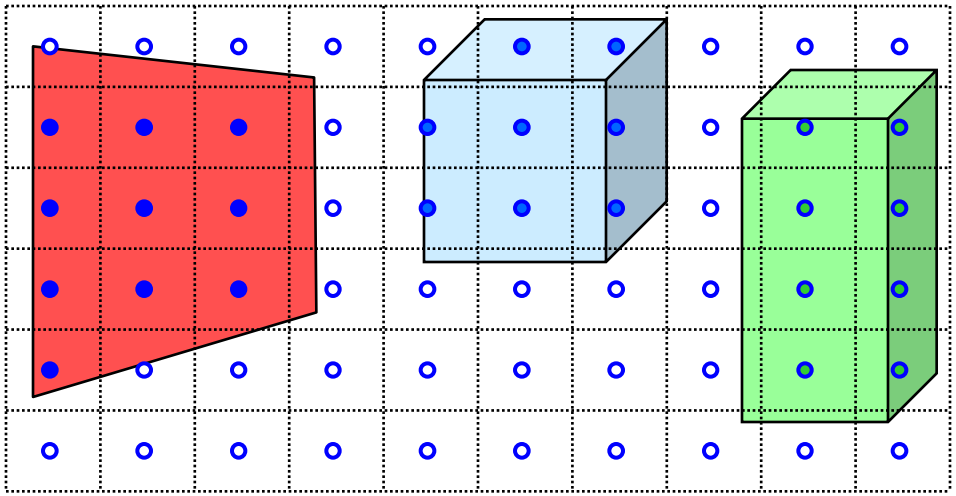
- For each sample …
  - Construct ray from eye position through view plane
  - Find first surface intersected by ray through pixel
  - Compute color of sample based on surface radiance



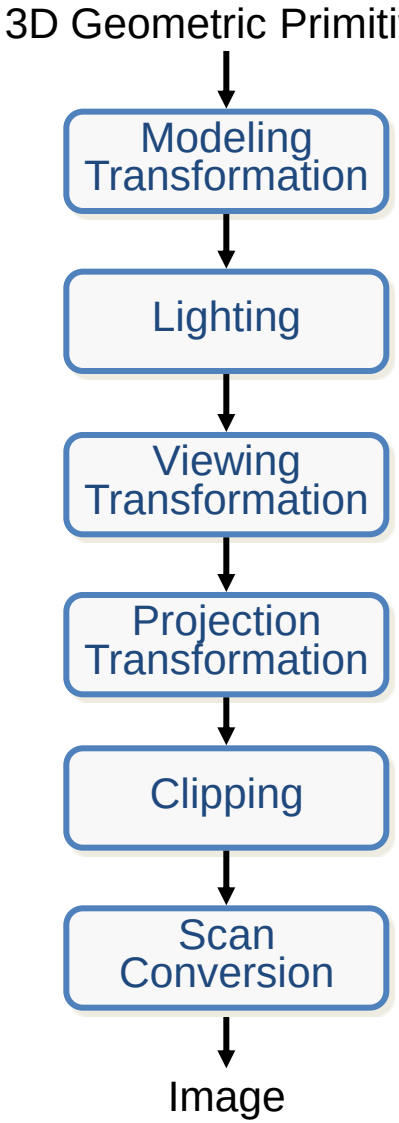
More efficient algorithms utilize spatial coherence!

3D Polygon Rendering

- What steps are necessary to utilize spatial coherence while drawing these polygons into a 2D image?

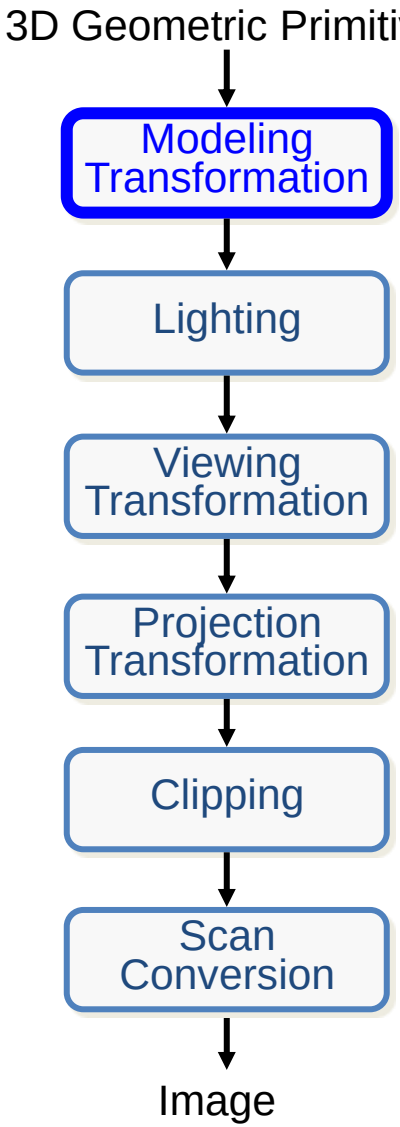


3D Rendering Pipeline (for direct illumination)



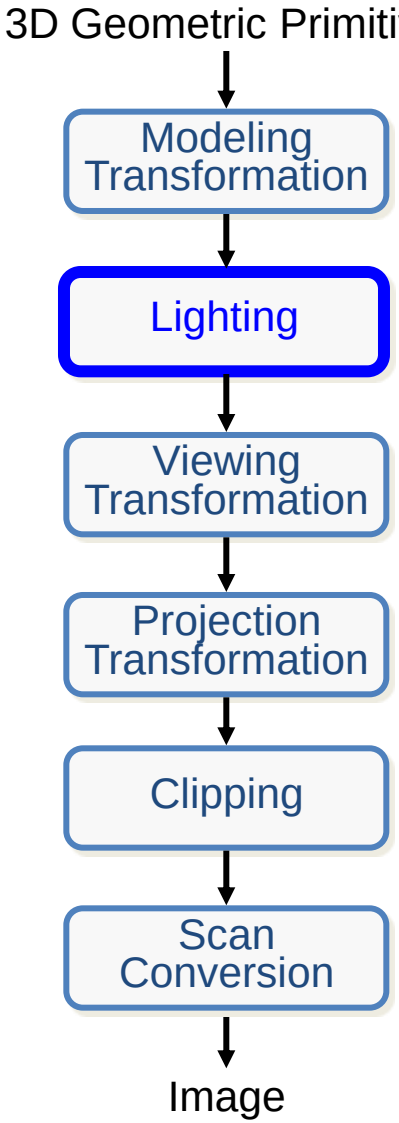
This is a pipelined sequence of operations to draw a 3D primitive into a 2D image

3D Rendering Pipeline (for direct illumination)



Transform into 3D world coordinate system

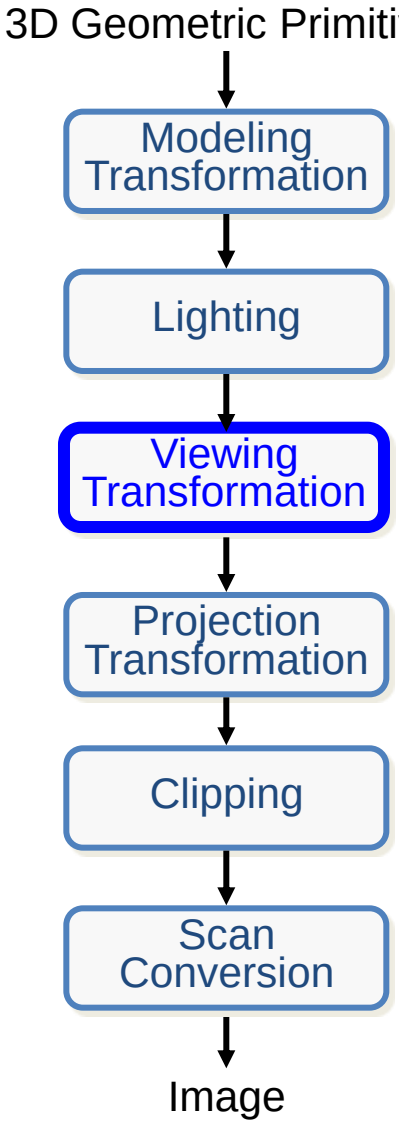
3D Rendering Pipeline (for direct illumination)



Transform into 3D world coordinate system

Illuminate according to lighting and reflectance

3D Rendering Pipeline (for direct illumination)

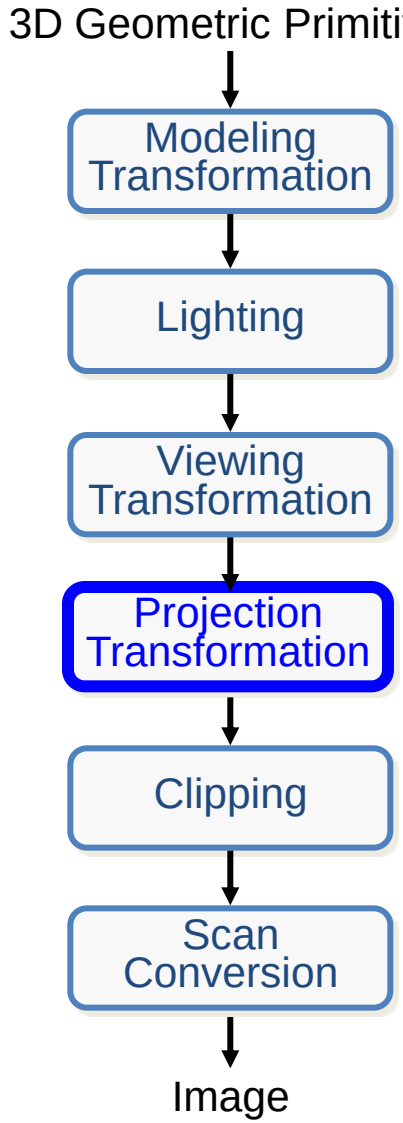


Transform into 3D world coordinate system

Illuminate according to lighting and reflectance

Transform into 3D camera coordinate system

3D Rendering Pipeline (for direct illumination)



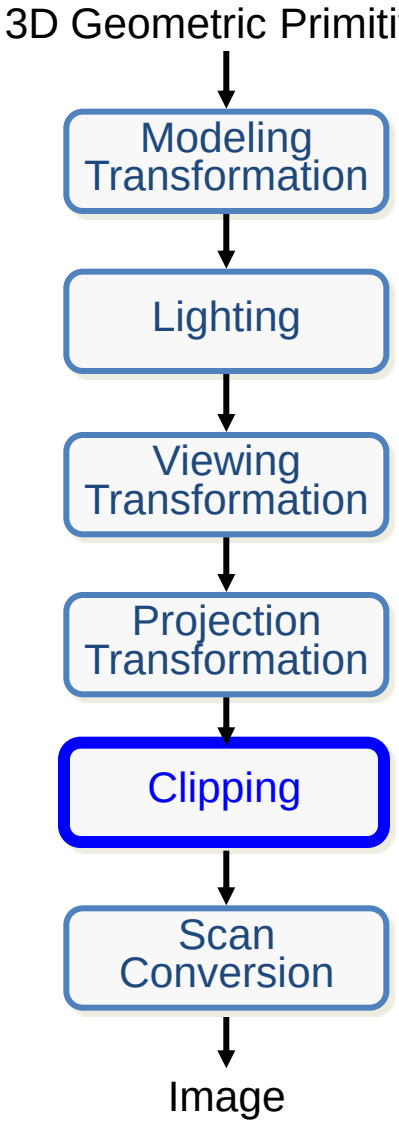
Transform into 3D world coordinate system

Illuminate according to lighting and reflectance

Transform into 3D camera coordinate system

Transform into 2D camera coordinate system

3D Rendering Pipeline (for direct illumination)



Transform into 3D world coordinate system

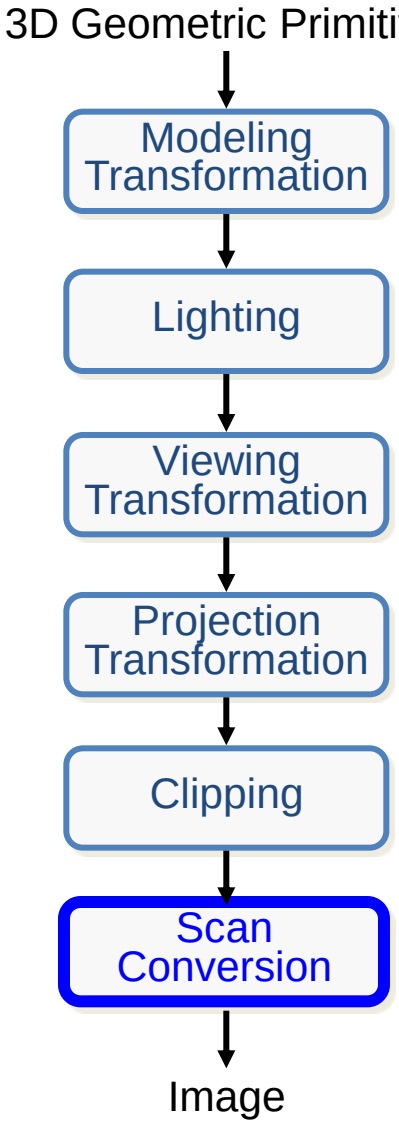
Illuminate according to lighting and reflectance

Transform into 3D camera coordinate system

Transform into 2D camera coordinate system

Clip primitives outside camera's view

3D Rendering Pipeline (for direct illumination)



Transform into 3D world coordinate system

Illuminate according to lighting and reflectance

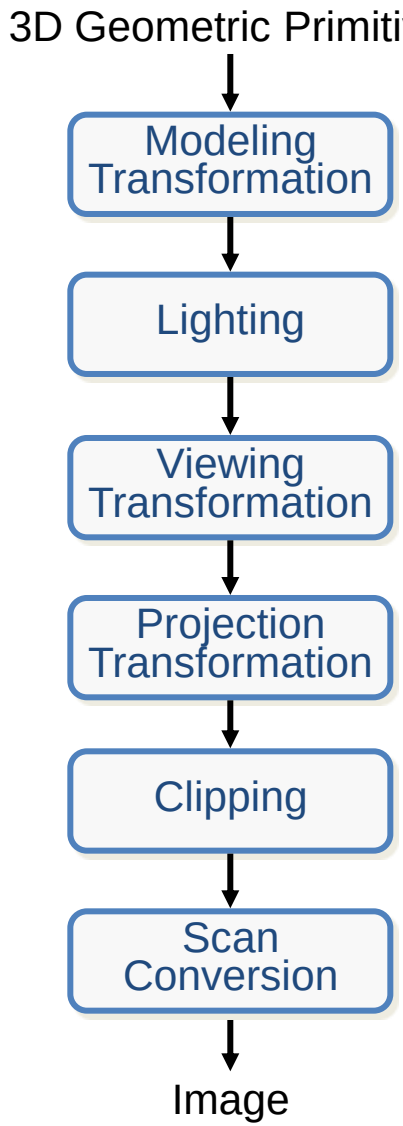
Transform into 3D camera coordinate system

Transform into 2D camera coordinate system

Clip primitives outside camera's view

Draw pixels (includes texturing, hidden surface, etc.)

Transformations



Transform into 3D world coordinate system

Illuminate according to lighting and reflectance

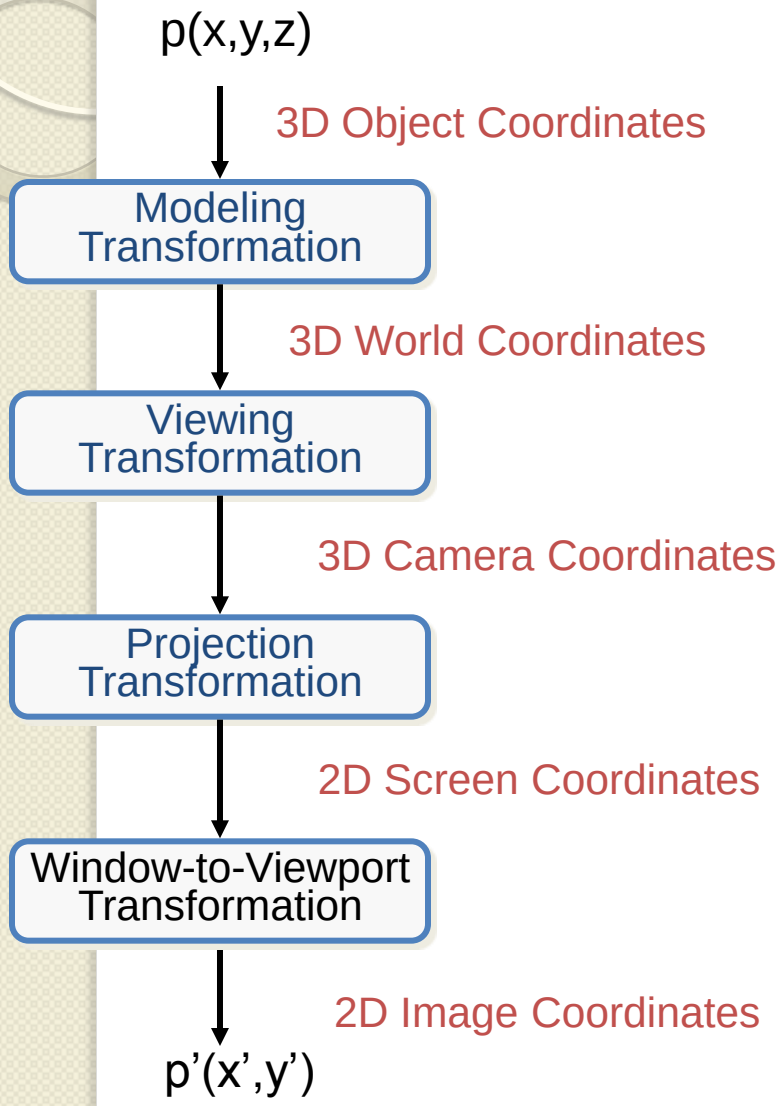
Transform into 3D camera coordinate system

Transform into 2D camera coordinate system

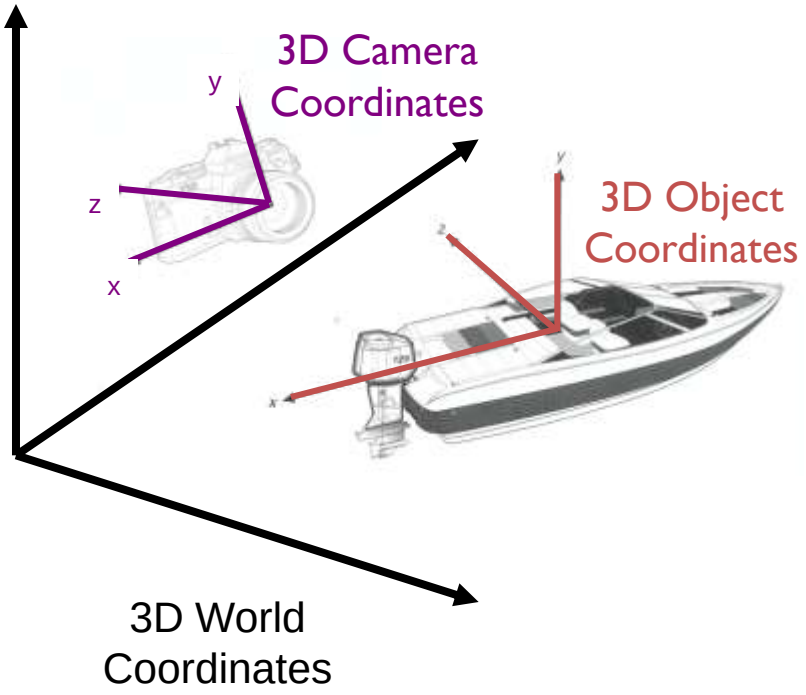
Clip primitives outside camera's view

Draw pixels (includes texturing, hidden surface, etc.)

Transformations

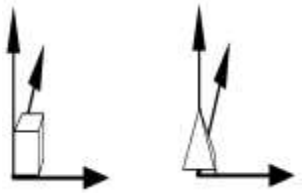


Transformations map points from one coordinate system to another

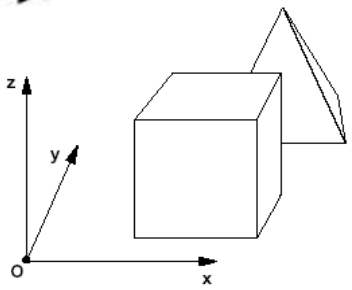


Coordinate systems

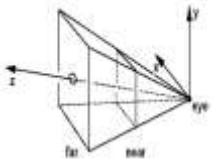
- Object space
  - local to each object



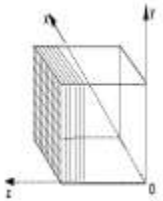
- World space
  - common to all objects



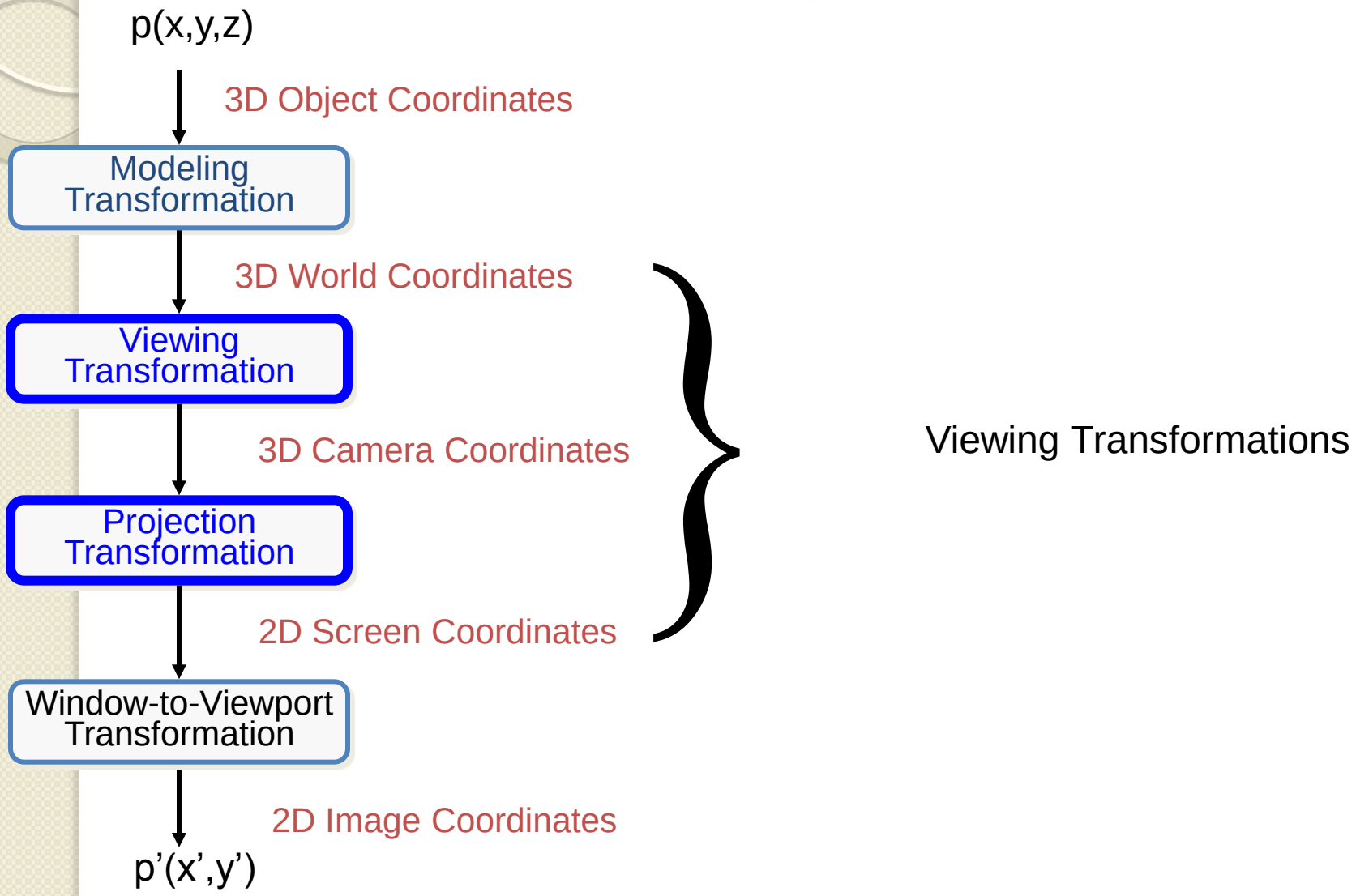
- Eye space / Camera space
  - derived from view frustum



- Screen space
  - indexed according to hardware attributes

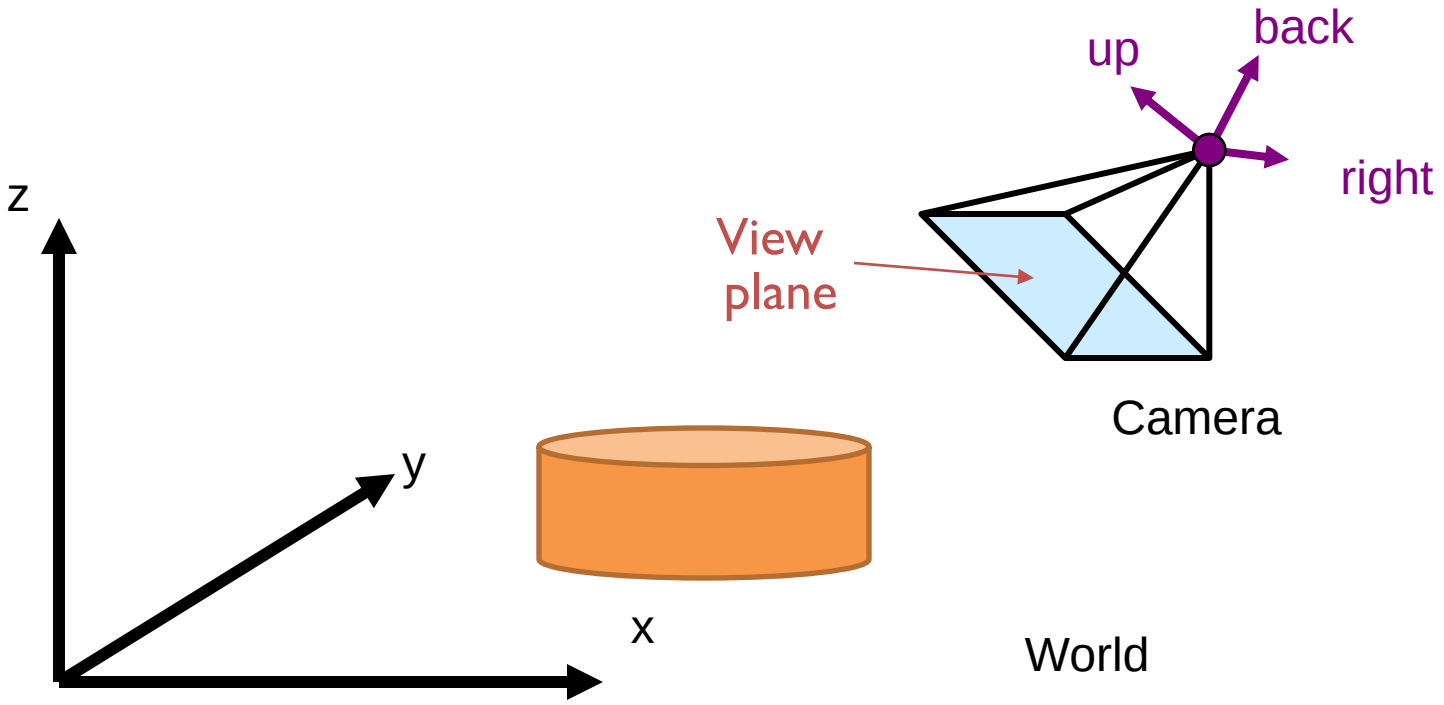


Viewing Transformations



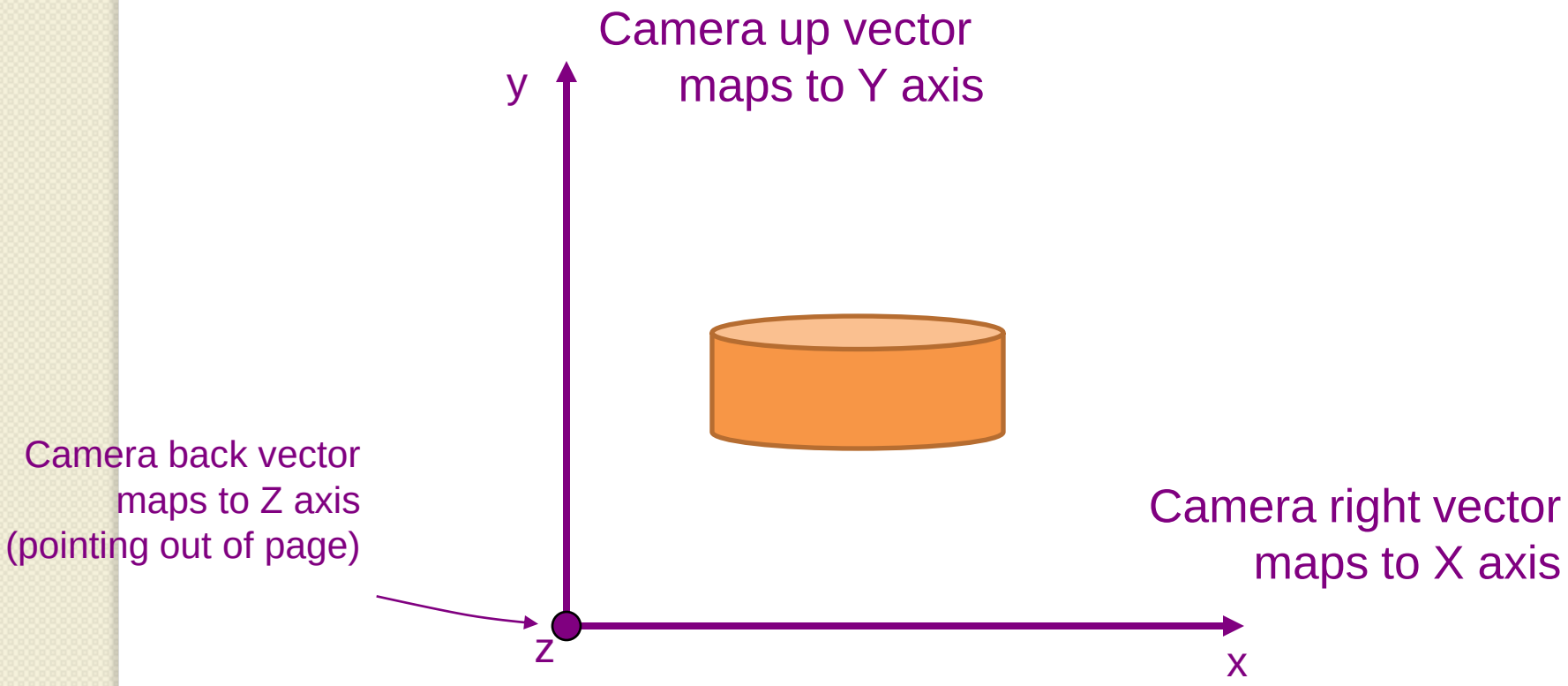
Viewing Transformation

- Mapping from world to camera (viewing) coordinates
  - Eye position maps to origin
  - Right vector maps to X axis
  - Up vector maps to Y axis
  - Back vector maps to Z axis



Camera Coordinates

- Canonical coordinate system
  - Convention is right-handed (looking down -z axis)
  - Convenient for projection, clipping, etc.



Finding viewing transformation

- We have the camera (world coordinates)
- We want a transformation T taking objects from world to camera

$$p^c = Tp^w$$

- Trick: find T^{-1} taking objects in camera to world

$$p^w = T^{-1}p^c$$

$$\begin{bmatrix} x' \\ y' \\ z' \\ w' \end{bmatrix} = \begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ m & n & o & p \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} \quad ?$$

Finding viewing transformation

- Trick: map from camera coordinates to world
  - Origin → eye position
  - Z axis → back vector
  - Y axis → up vector
  - X axis → right vector

Invert to get T. Easy!

$$\begin{bmatrix} x' \\ y' \\ z' \\ w' \end{bmatrix} = \begin{bmatrix} R_x & U_x & B_x & E_x \\ R_y & U_x & B_y & E_y \\ R_z & U_z & B_z & E_z \\ R_w & U_w & B_w & E_w \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix}$$

Finding viewing transformation

- Lets give it a try…

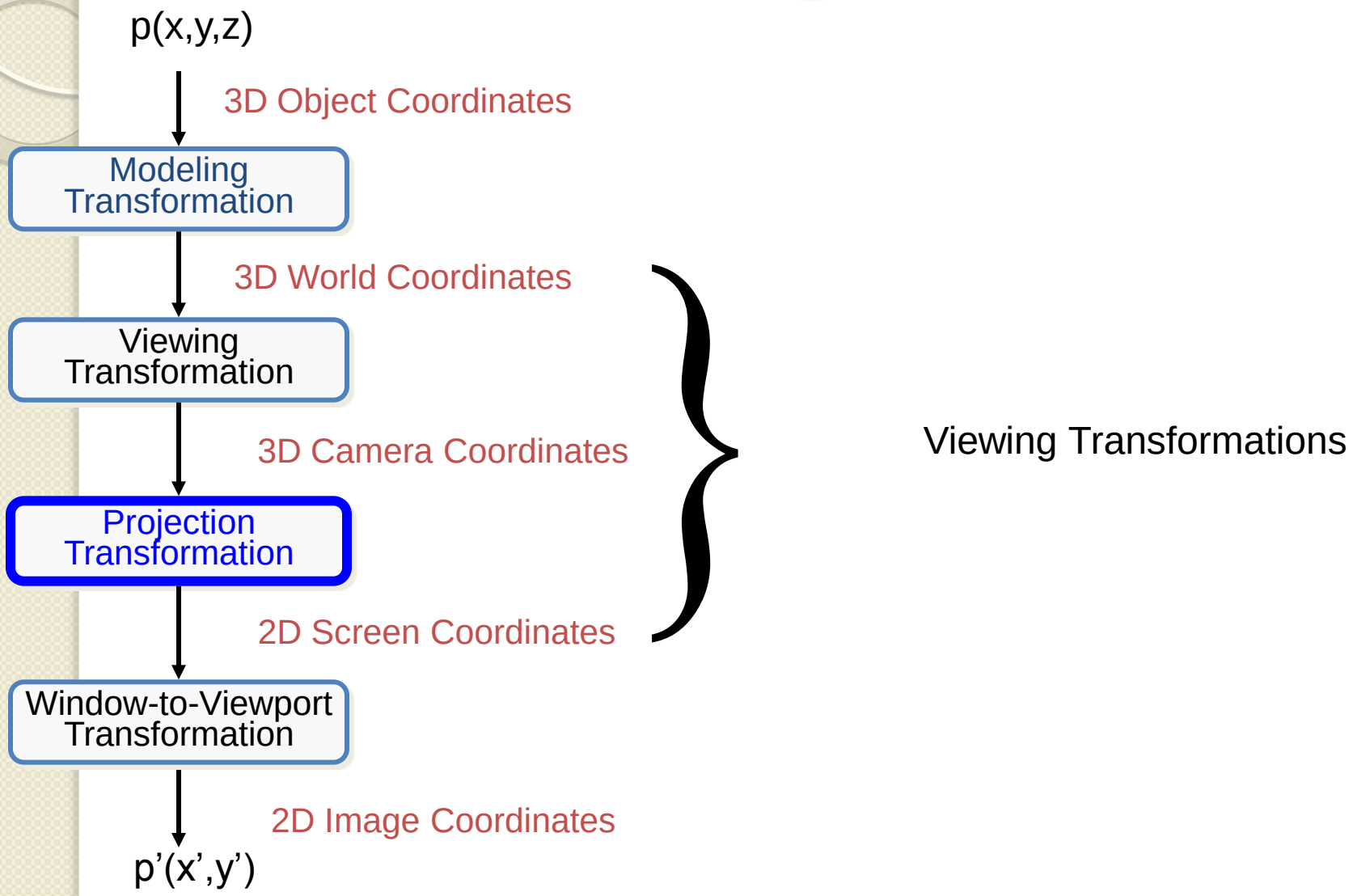
$$\begin{bmatrix} x' \\ y' \\ z' \\ w' \end{bmatrix} = \begin{bmatrix} R_x & U_x & B_x & E_x \\ R_y & U_x & B_y & E_y \\ R_z & U_z & B_z & E_z \\ R_w & U_w & B_w & E_w \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix}$$

$$\begin{bmatrix} E_x \\ E_y \\ E_z \\ E_w \end{bmatrix} = \begin{bmatrix} R_x & U_x & B_x & E_x \\ R_y & U_x & B_y & E_y \\ R_z & U_z & B_z & E_z \\ R_w & U_w & B_w & E_w \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} R_x + E_x \\ R_y + E_y \\ R_z + E_z \\ R_w + E_w \end{bmatrix} = \begin{bmatrix} R_x & U_x & B_x & E_x \\ R_y & U_x & B_y & E_y \\ R_z & U_z & B_z & E_z \\ R_w & U_w & B_w & E_w \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

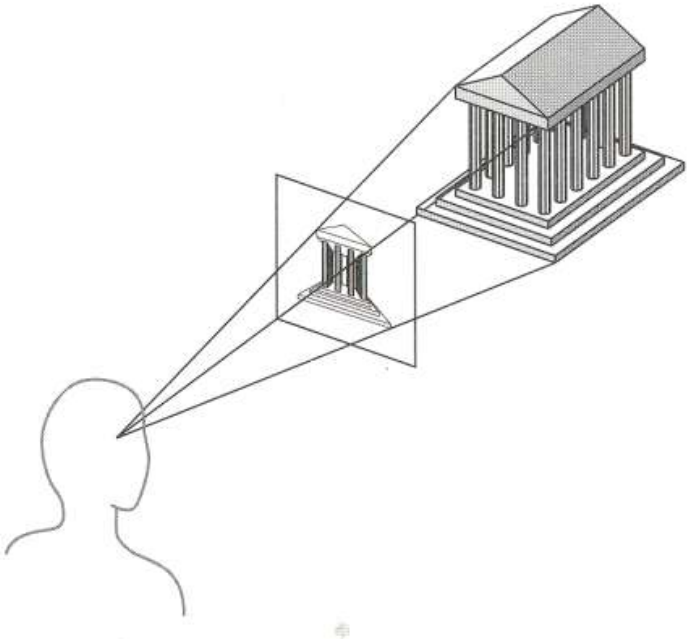
(0,0,0) עובר לנק' העין, הצירים עוברים לוקטורים המגדירים את מערכת הצירים של המצלמה

Viewing Transformations

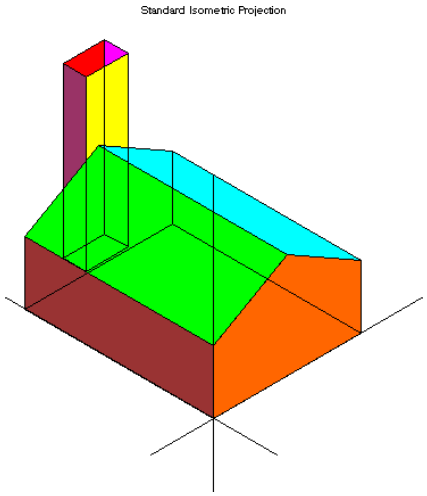


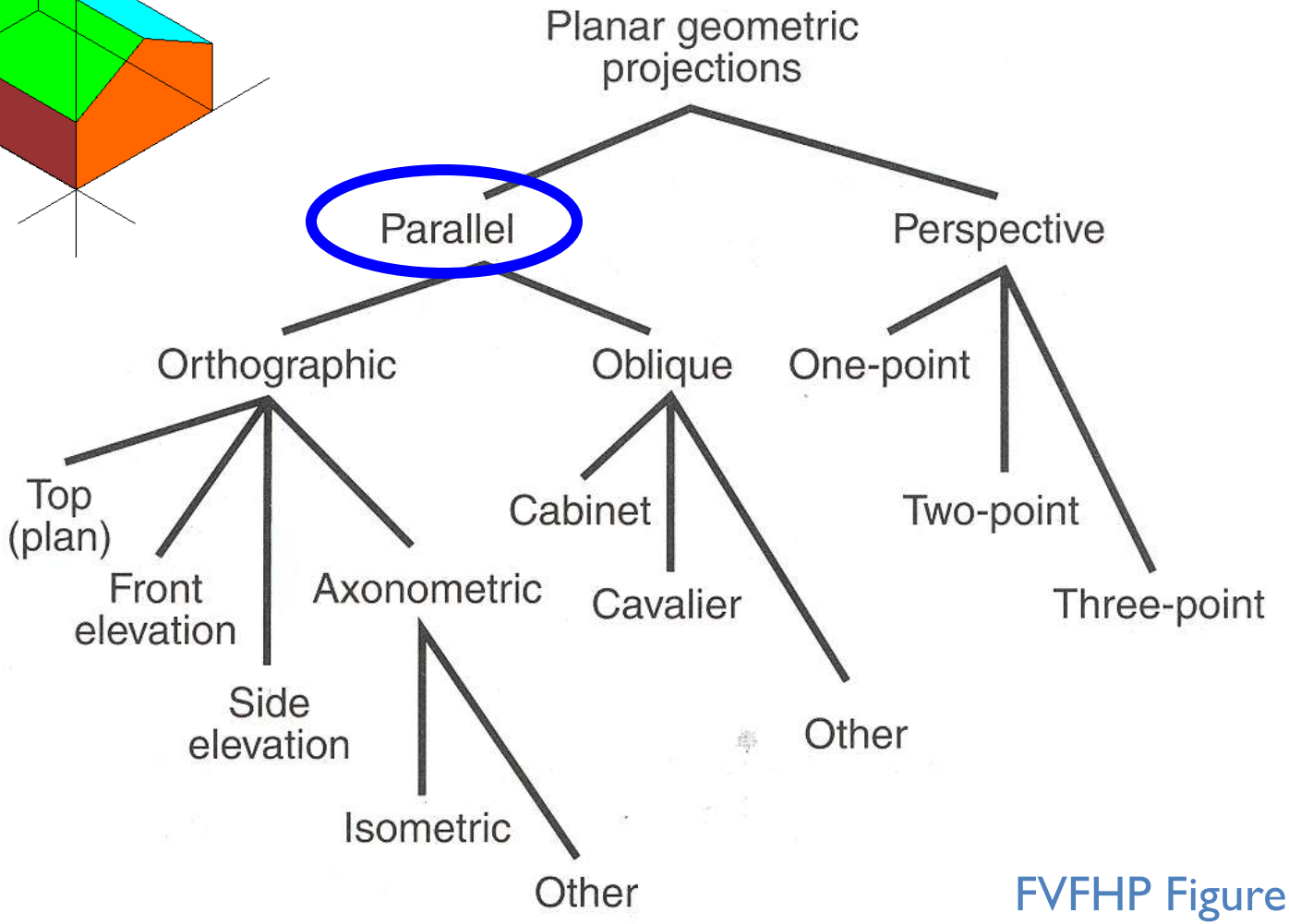
Projection

- General definition:
  - Transform points in *n*-space to *m*-space (*m*<*n*)
- In computer graphics:
  - Map 3D camera coordinates to 2D screen coordinates
  - Projection is formed by the intersection of certain lines with the view plane



Taxonomy of Projections

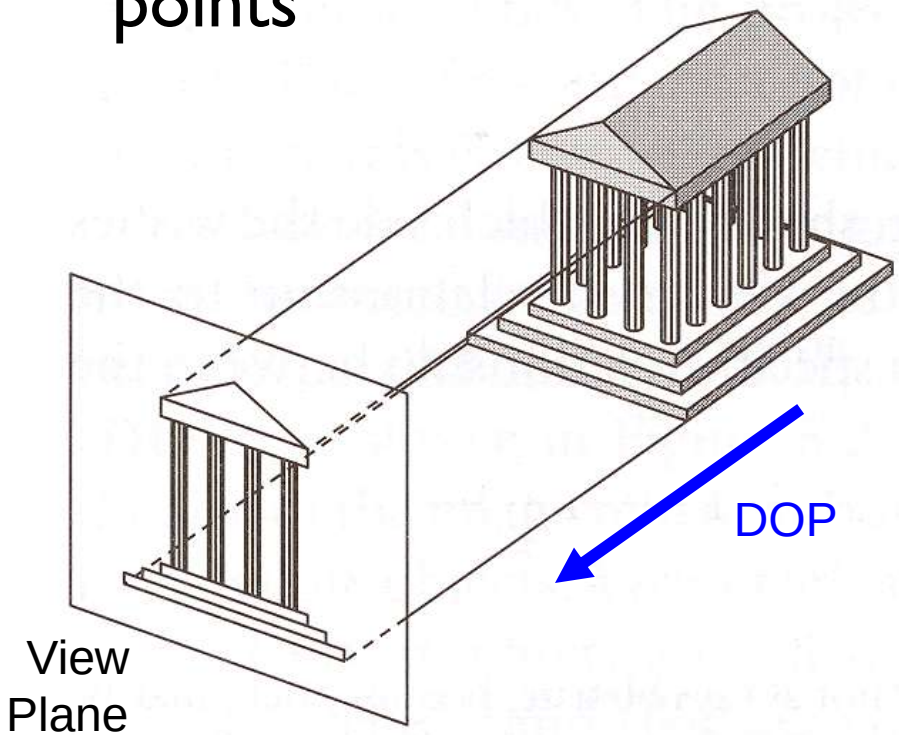




FVFHP Figure 6.10

Parallel Projection

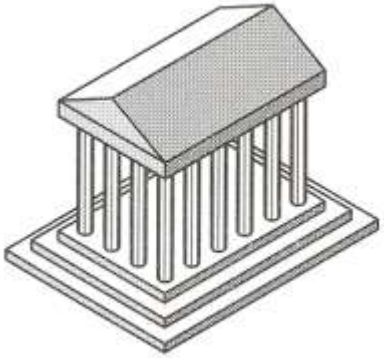
- Center of projection is at infinity
  - Direction of projection (DOP) same for all points

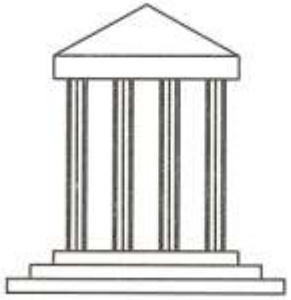


Angel Figure 5.4

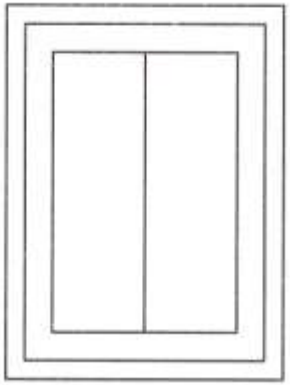
Orthographic Projections

- DOP perpendicular to view plane

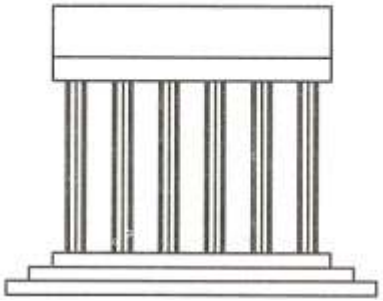




Front



Top



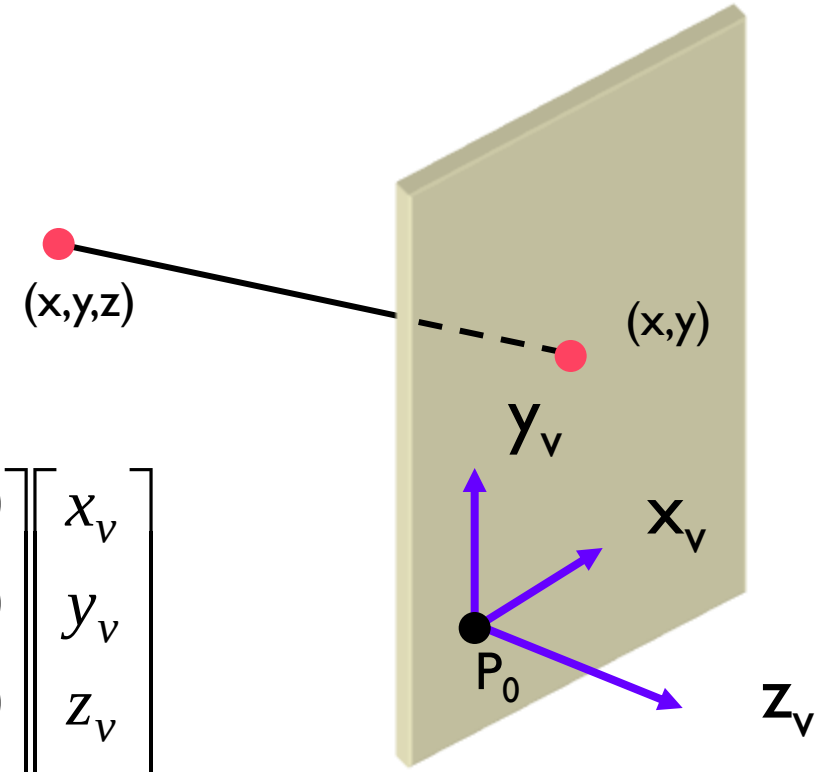
Side

Angel Figure 5.5

Orthographic Projections

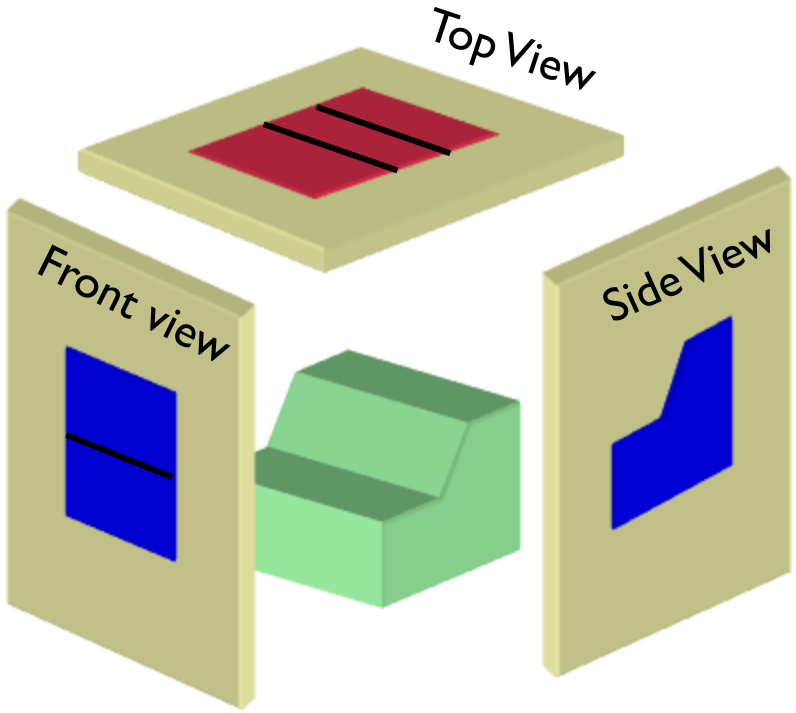
- Since the viewing plane is aligned with (x_v,y_v), orthographic projection is performed by:

$$\begin{bmatrix} x_p \\ y_p \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} x_v \\ y_v \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_v \\ y_v \\ z_v \\ 1 \end{bmatrix}$$



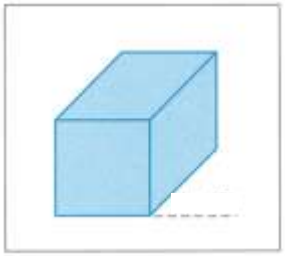
Orthographic Projections

- Lengths and angles of faces parallel to the viewing planes are preserved.
- **Problem**: 3D nature of projected objects is difficult to deduce.

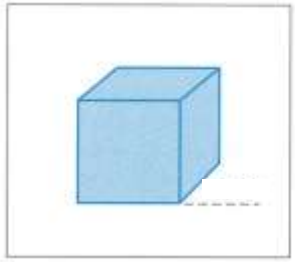


Oblique Projections

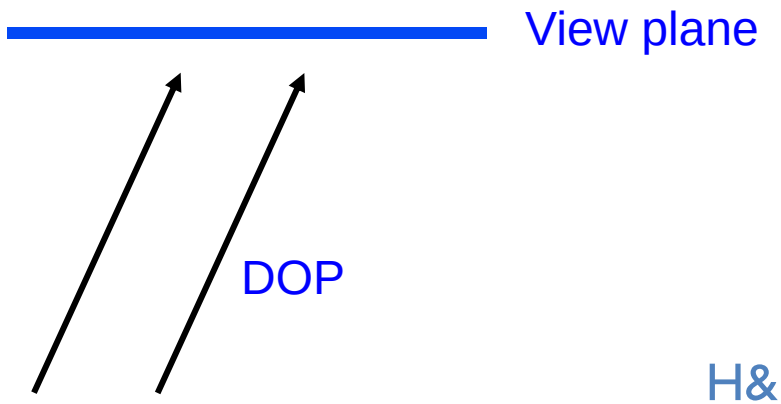
- DOP not perpendicular to view plane



Cavalier
(DOP at 45°)



Cabinet
(DOP at 63.4°)



H&B Figure 12.24

Oblique Projections

$$\begin{bmatrix} x_p \\ y_p \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & a\cos\phi & 0 \\ 0 & 1 & a\sin\phi & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_v \\ y_v \\ z_v \\ 1 \end{bmatrix} = \begin{bmatrix} x_v + z_v a\cos\phi \\ y_v + z_v a\sin\phi \\ 0 \\ 1 \end{bmatrix}$$

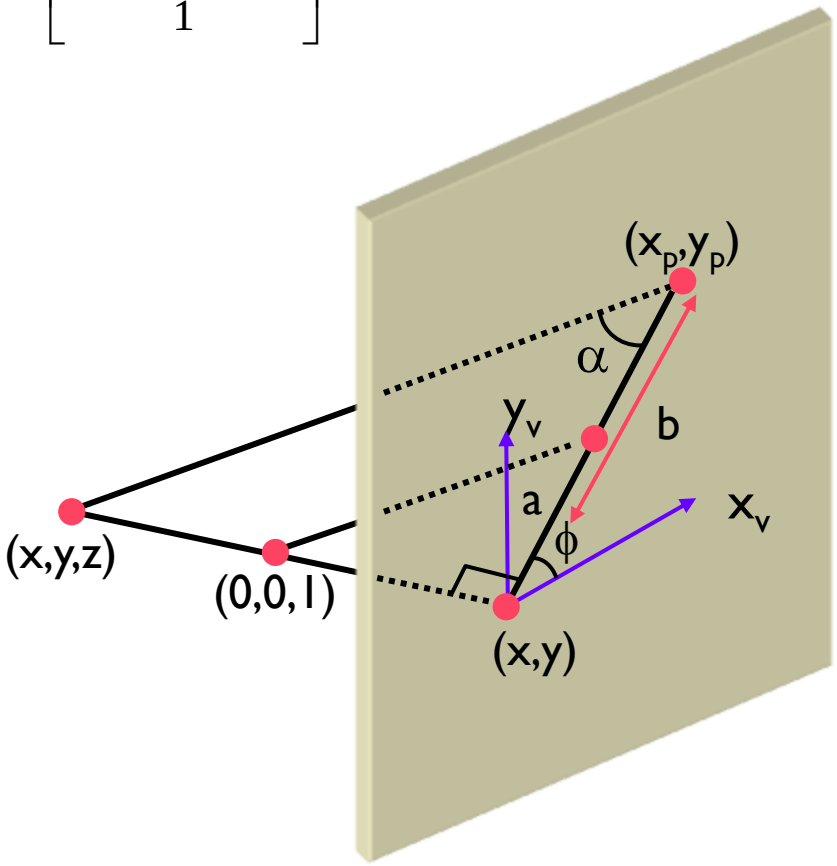
$l/a=\tan(\alpha)$

$z/b= l/a$

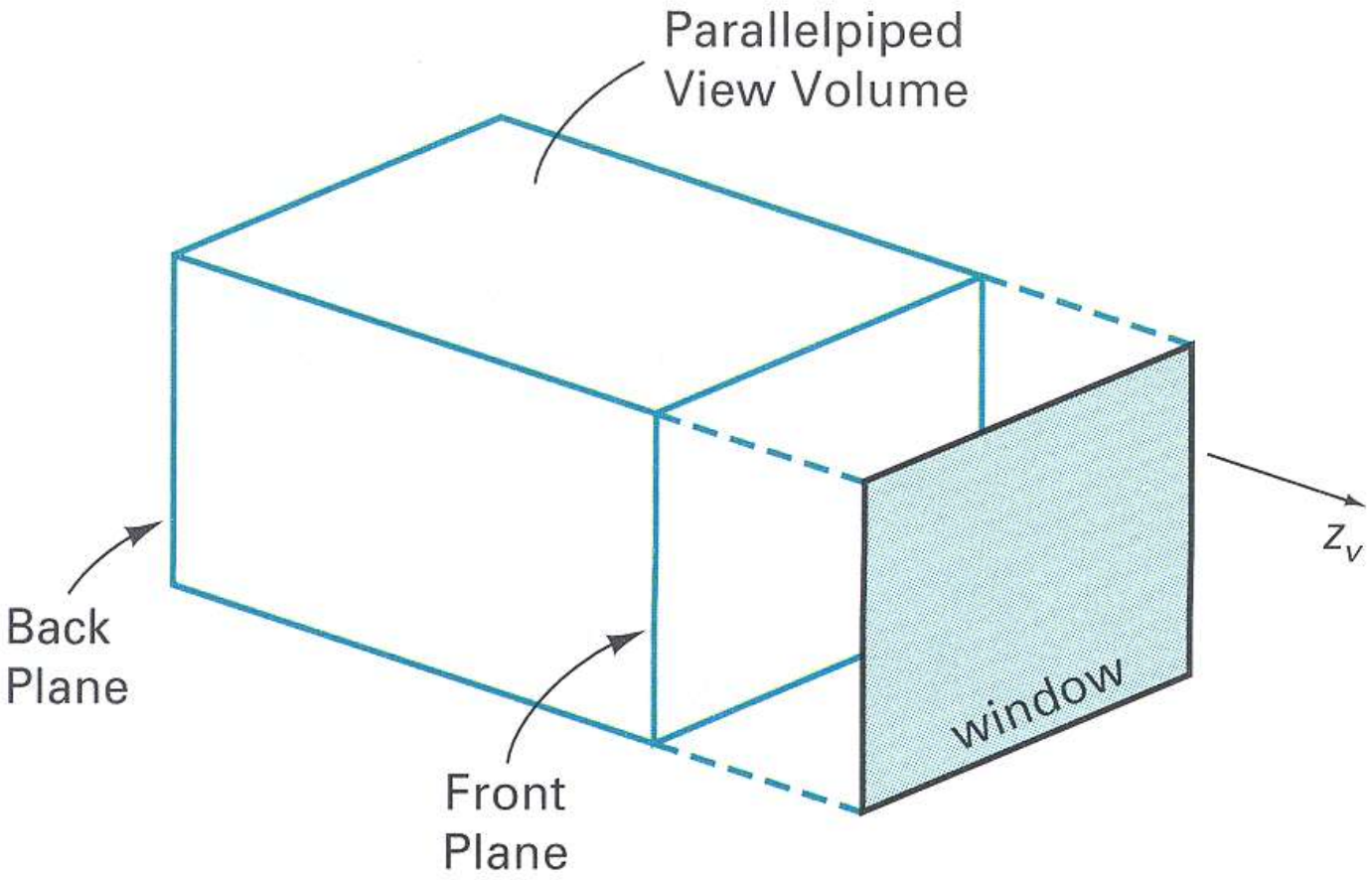
$b=za$

$x_p=z\cdot a\cdot\cos(\phi)$

$y_p=z\cdot a\cdot\sin(\phi)$

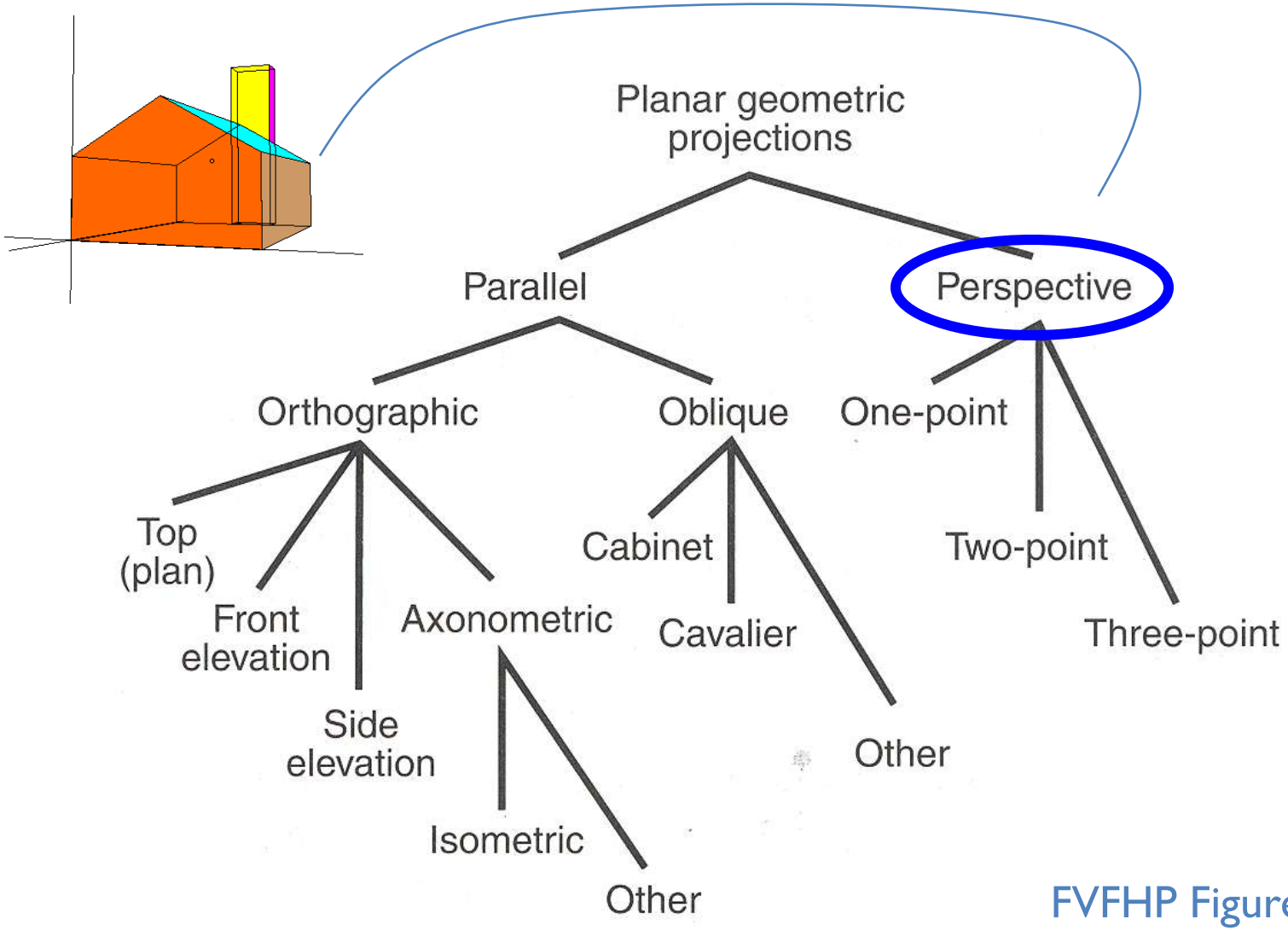


Parallel Projection View Volume



H&B Figure 12.30

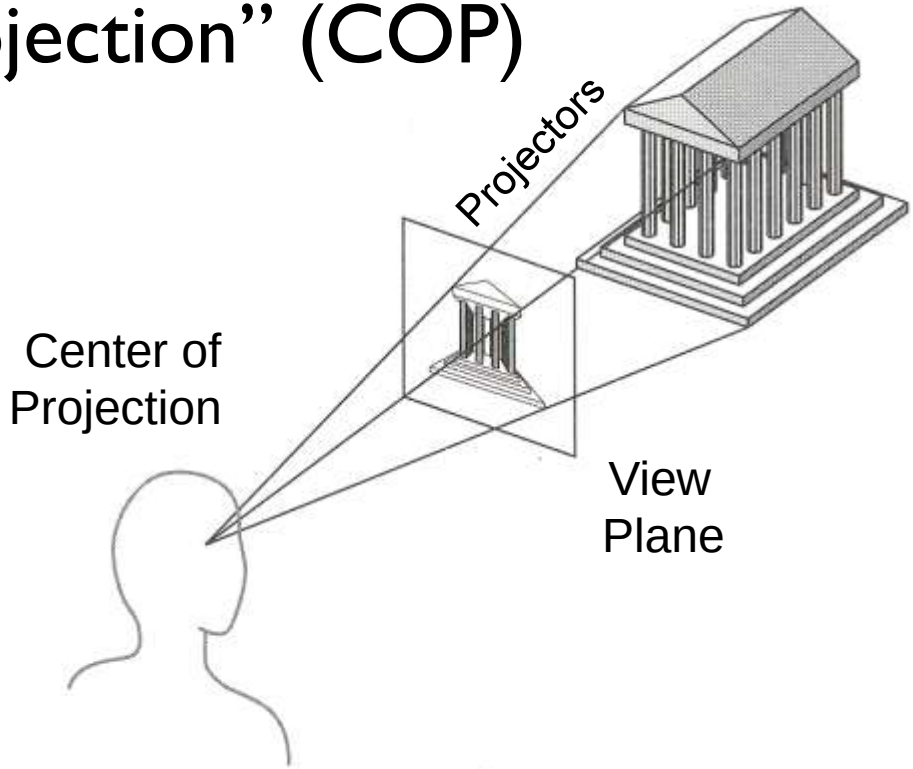
Taxonomy of Projections



FVFHP Figure 6.10

Perspective Projection

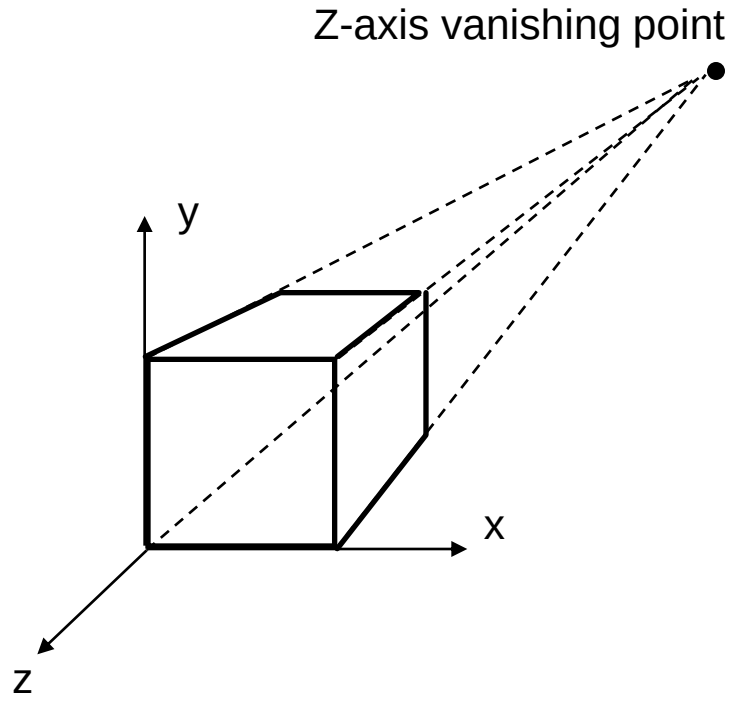
- Map points onto "view plane" along "projectors" emanating from "center of projection" (COP)



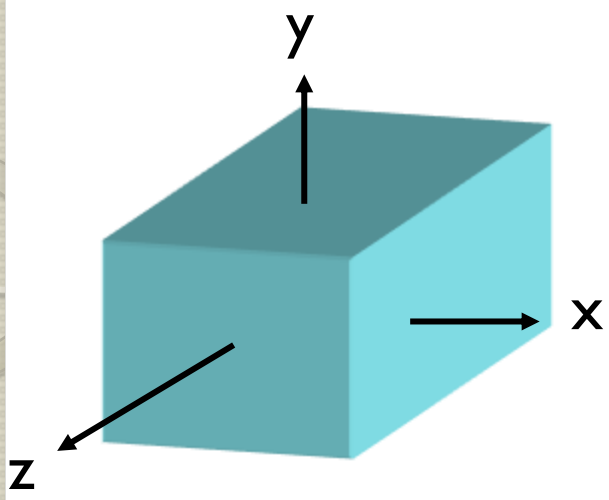
Angel Figure 5.9

Perspective Projection

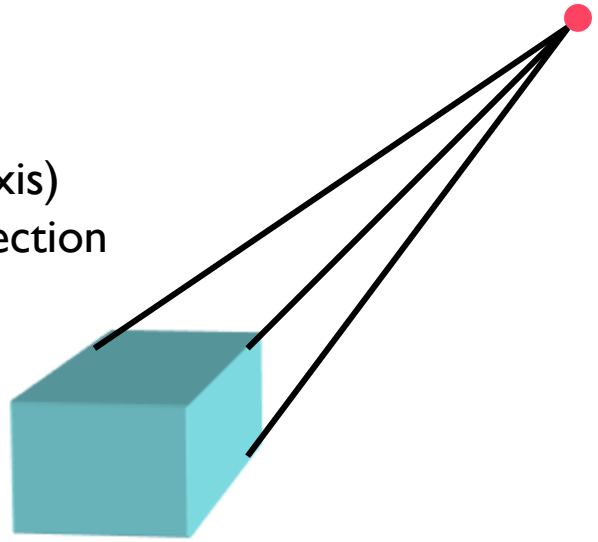
- Vanishing points → The projection of a point at infinity
- There can be up to 3 axis vanishing points



Perspective Projection



One point (z axis)
perspective projection

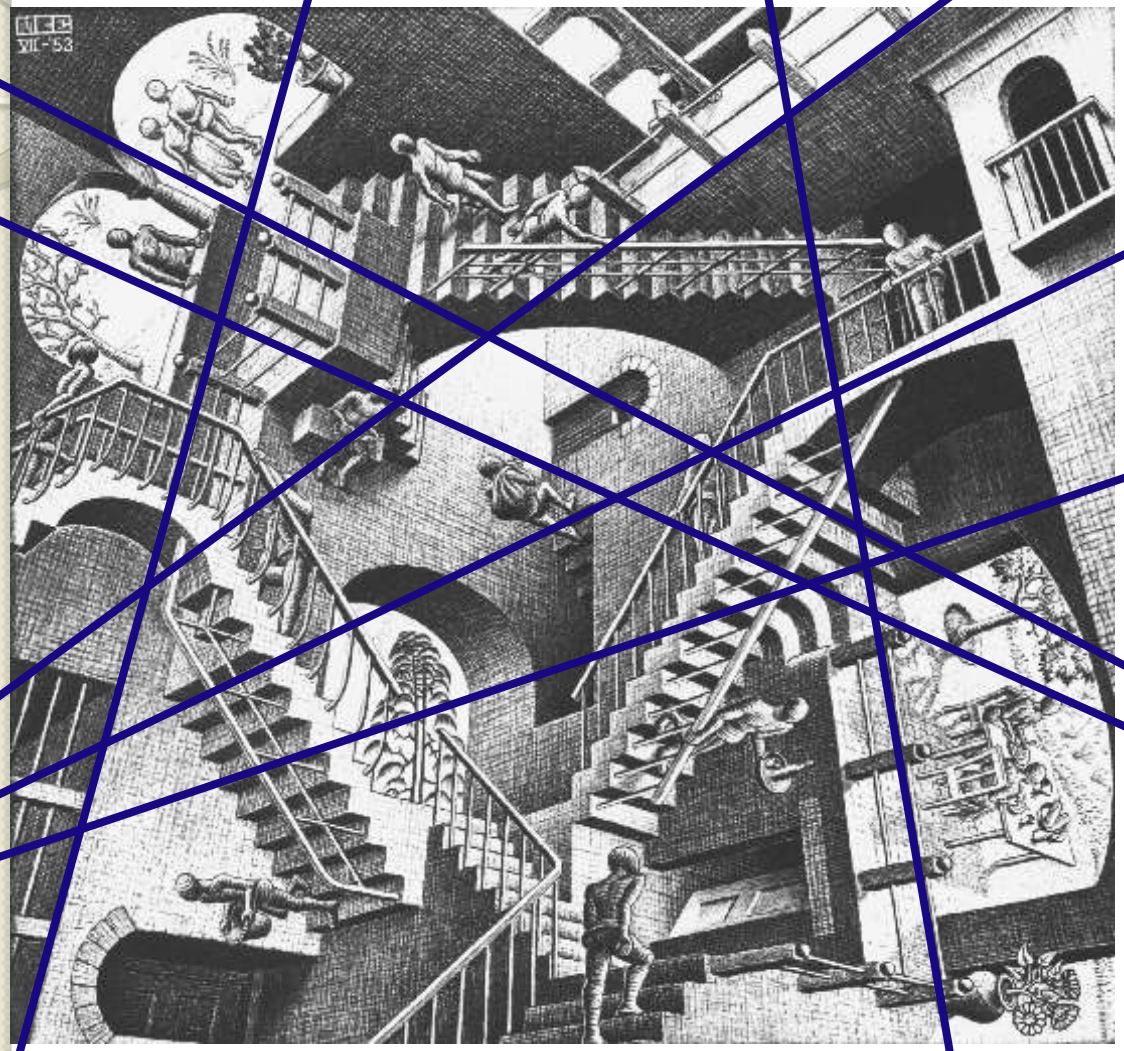


x axis
vanishing point.

z axis
vanishing point.

Two points
perspective projection

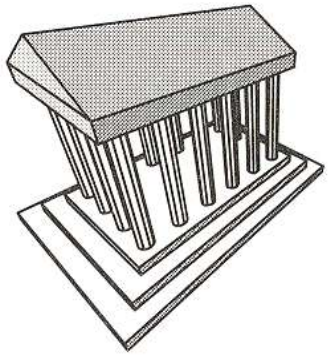
3 point perspective



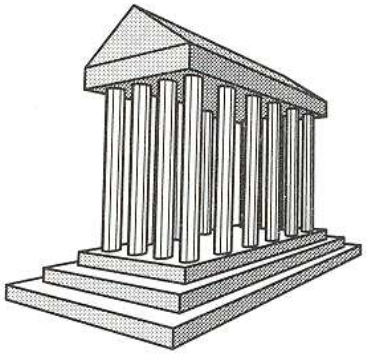
M.C.Escher's "*Relativity*" where 3 worlds co-exist thanks to 3-point perspective.

Perspective Projection

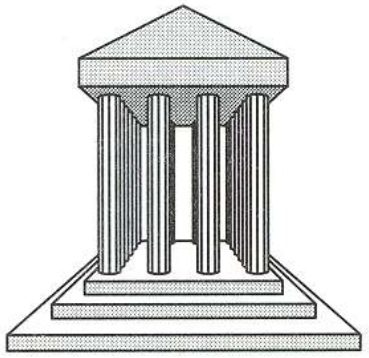
- How many vanishing points?



3-Point Perspective



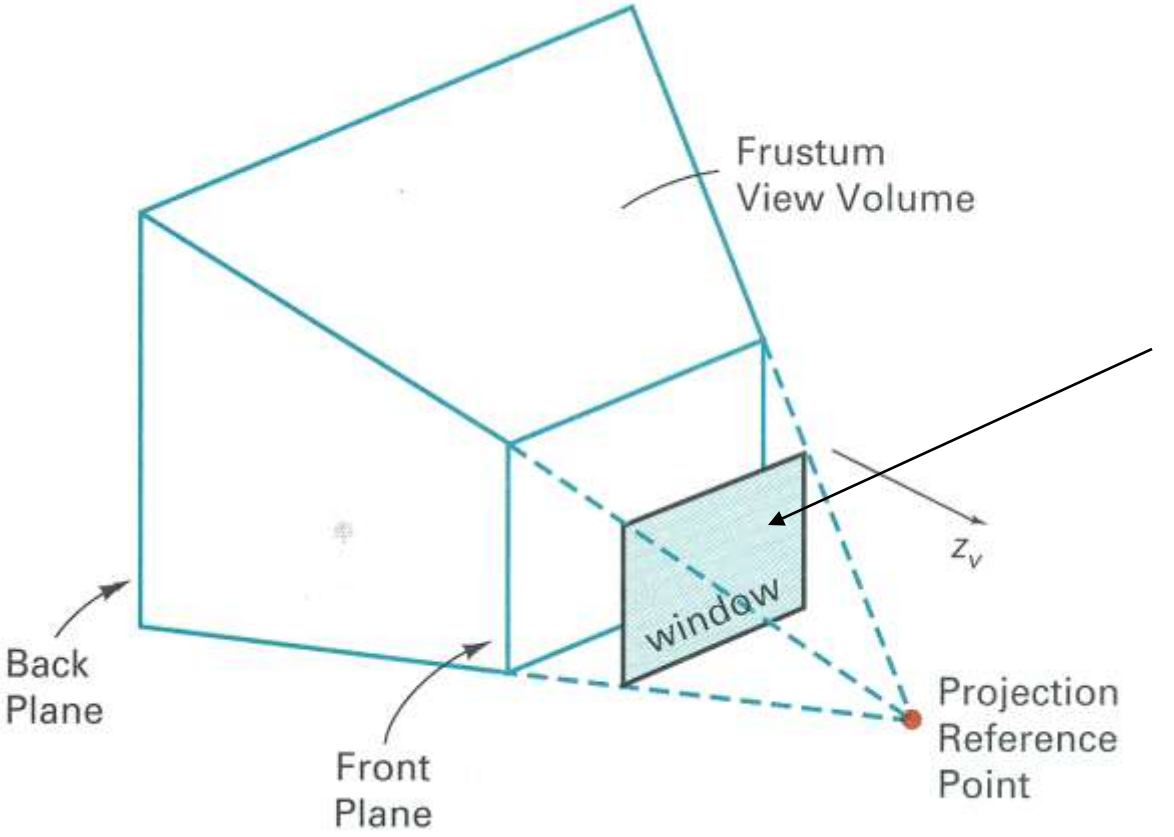
2-Point Perspective



1-Point Perspective

Angel Figure 5.10

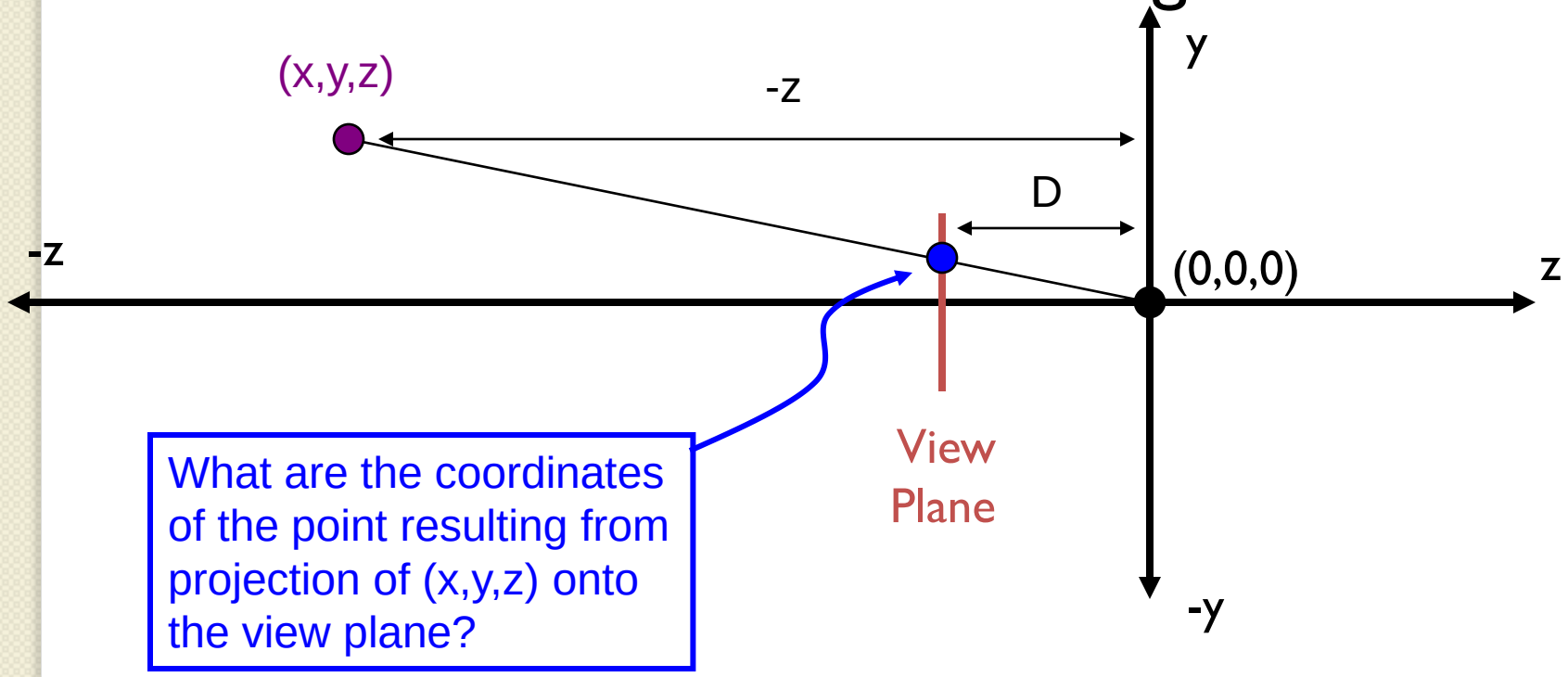
Perspective Projection View Volume



H&B Figure 12.30

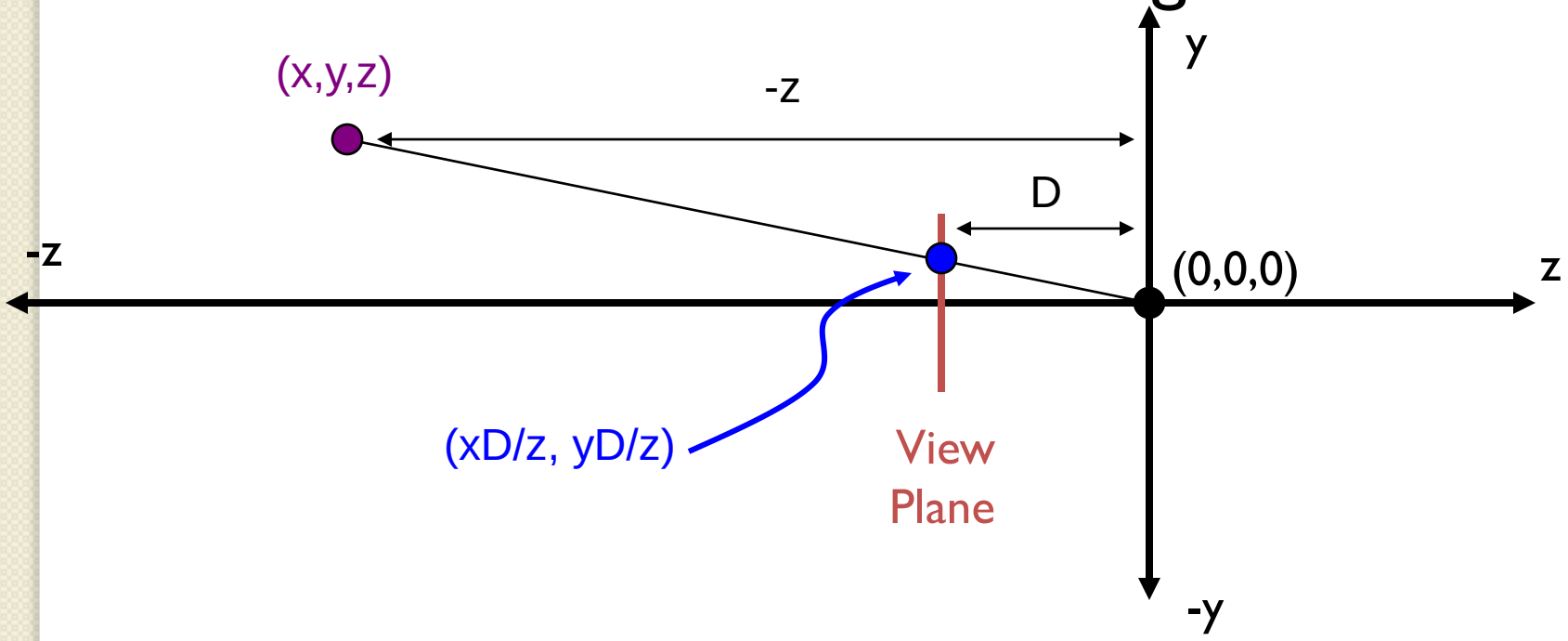
Perspective Projection

- Compute 2D coordinates from 3D coordinates with similar triangles



Perspective Projection

- Compute 2D coordinates from 3D coordinates with similar triangles



Perspective Projection Matrix

- 4x4 matrix representation?

$$x_s = x_c D / z_c$$
$$y_s = y_c D / z_c$$
$$z_s = D$$
$$w_s = 1$$

$$\begin{bmatrix} x_s \\ y_s \\ z_s \\ w_s \end{bmatrix} = \begin{bmatrix} ? & ? & ? & ? \\ ? & ? & ? & ? \\ ? & ? & ? & ? \\ ? & ? & ? & ? \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix}$$

Perspective Projection Matrix

- 4x4 matrix representation?

$$x_s = x_c D / z_c$$
$$y_s = y_c D / z_c$$
$$z_s = D$$
$$w_s = 1$$

$$x' = x_c$$
$$y' = y_c$$
$$z' = z_c$$
$$w' = z_c / D$$

$$\begin{bmatrix} x_s \\ y_s \\ z_s \\ w_s \end{bmatrix} = \begin{bmatrix} ? & ? & ? & ? \\ ? & ? & ? & ? \\ ? & ? & ? & ? \\ ? & ? & ? & ? \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix}$$

Perspective Projection Matrix

- 4x4 matrix representation?

$$x_s = x_c D / z_c$$
$$y_s = y_c D / z_c$$
$$z_s = D$$
$$w_s = 1$$

$$x' = x_c$$
$$y' = y_c$$
$$z' = z_c$$
$$w' = z_c / D$$

$$\begin{bmatrix} x_s \\ y_s \\ z_s \\ w_s \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/D & 0 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix}$$

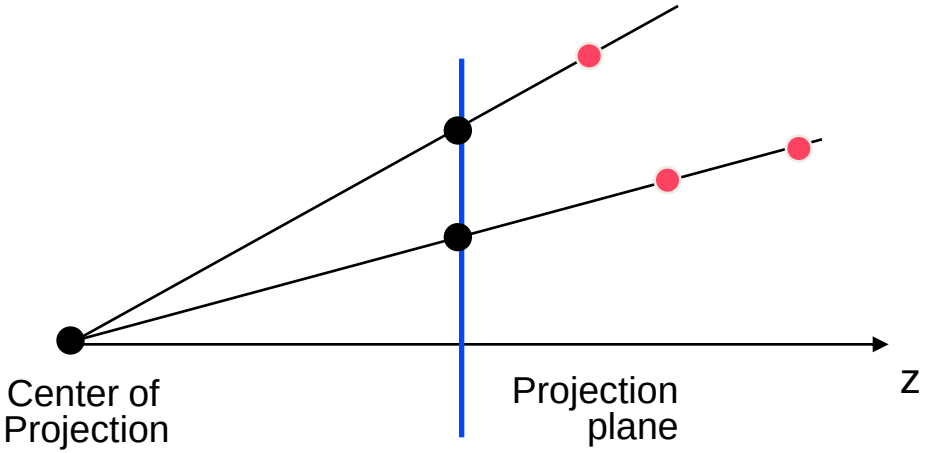
Perspective Projections

- Some observations
  - The matrix representation is singular, therefore a many to one mapping
  - Points on the viewing plane do not change
  - When d goes to infinity, we get an orthogonal projection

Changing Perspective

What is the difference between moving the center of projection and moving the projection plane?

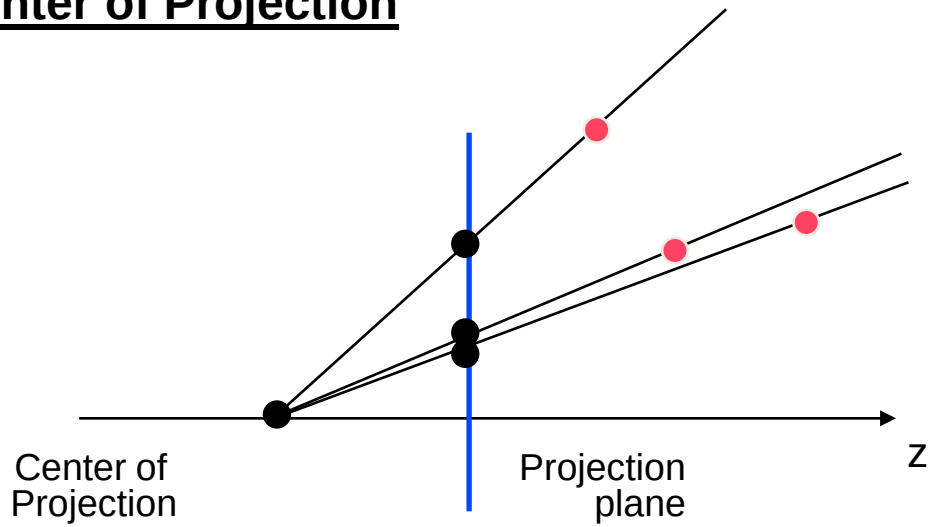
Original



Changing Perspective

What is the difference between moving the center of projection and moving the projection plane?

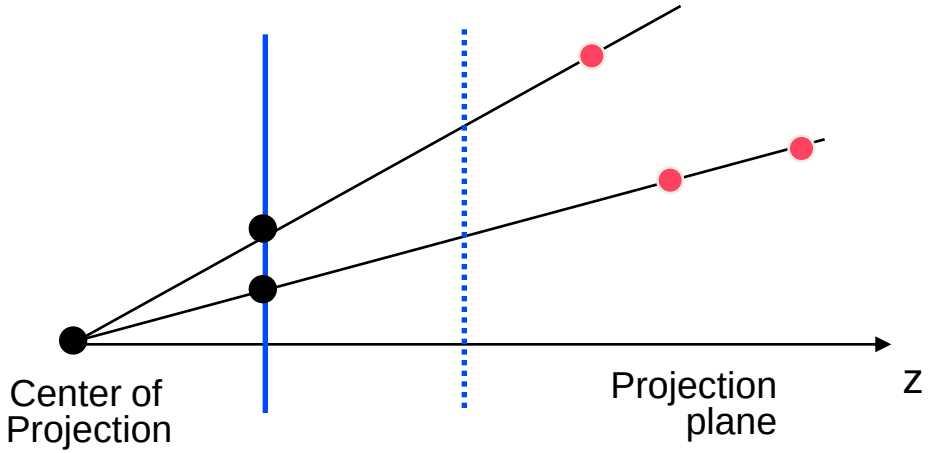
Moving the Center of Projection



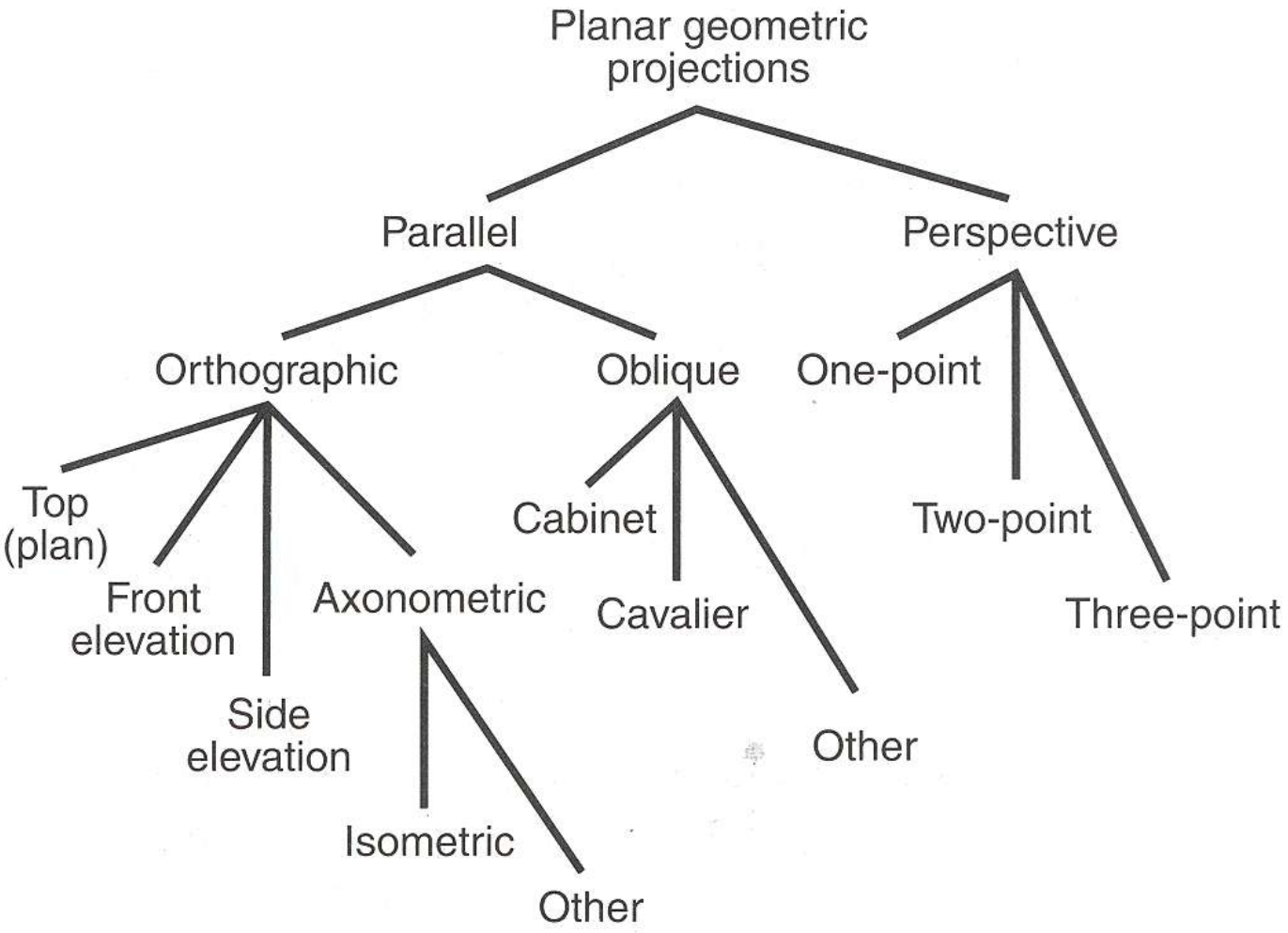
Changing Perspective

What is the difference between moving the center of projection and moving the projection plane?

Moving the Projection Plane

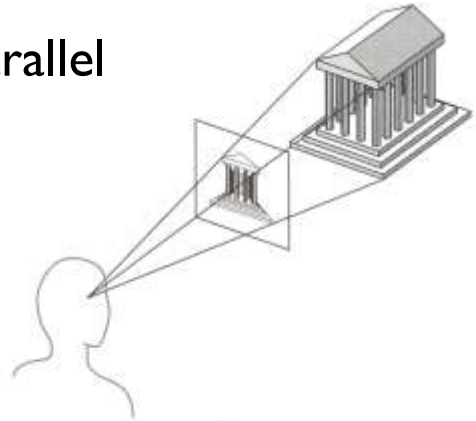


Taxonomy of Projections

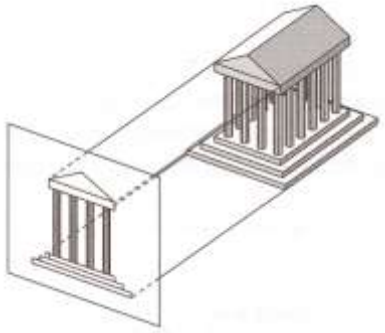


Perspective vs. Parallel

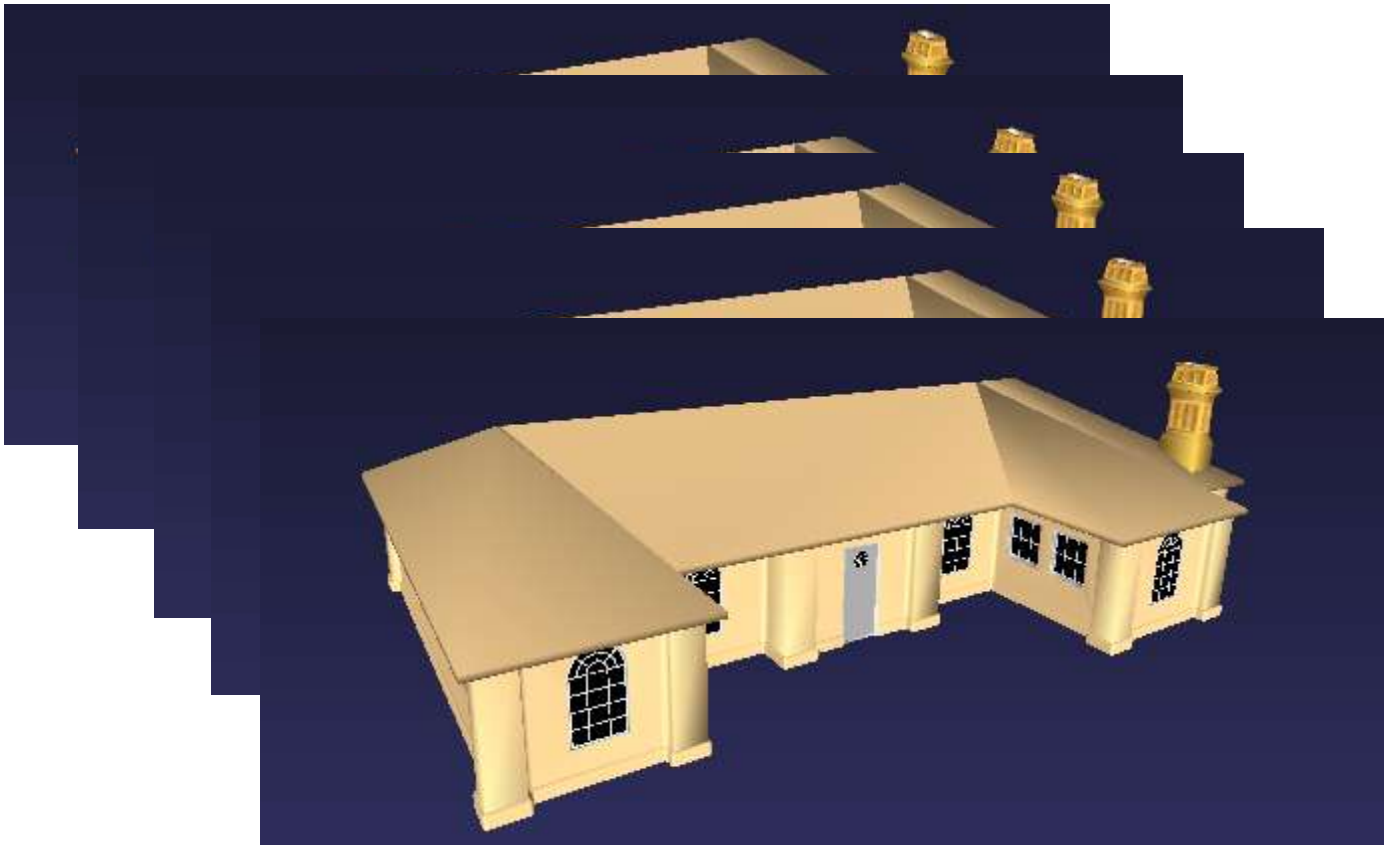
- Perspective projection
  - + Size varies inversely with distance - looks realistic
  - – Distance and angles are not (in general) preserved
  - – Parallel lines do not (in general) remain parallel



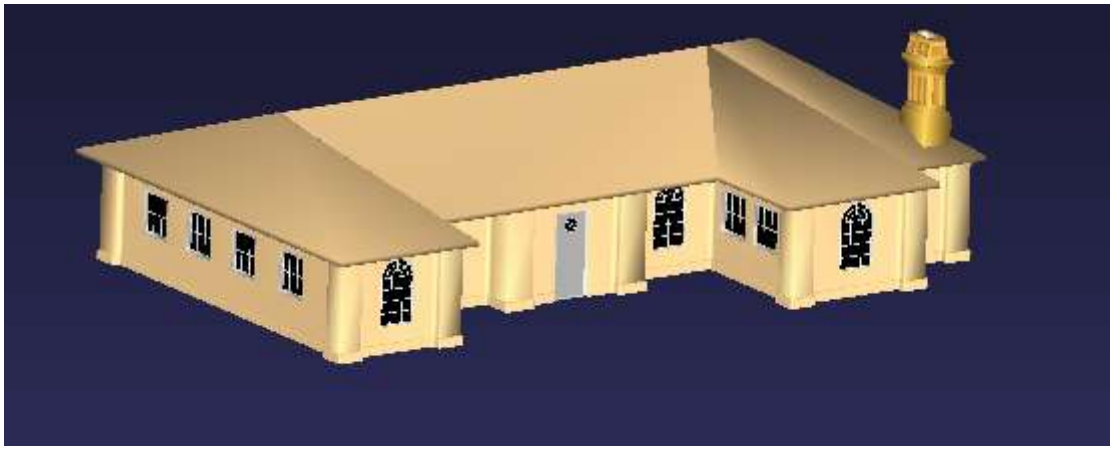
- Parallel projection
  - + Good for exact measurements
  - + Parallel lines remain parallel
  - – Angles are not (in general) preserved
  - – Less realistic looking



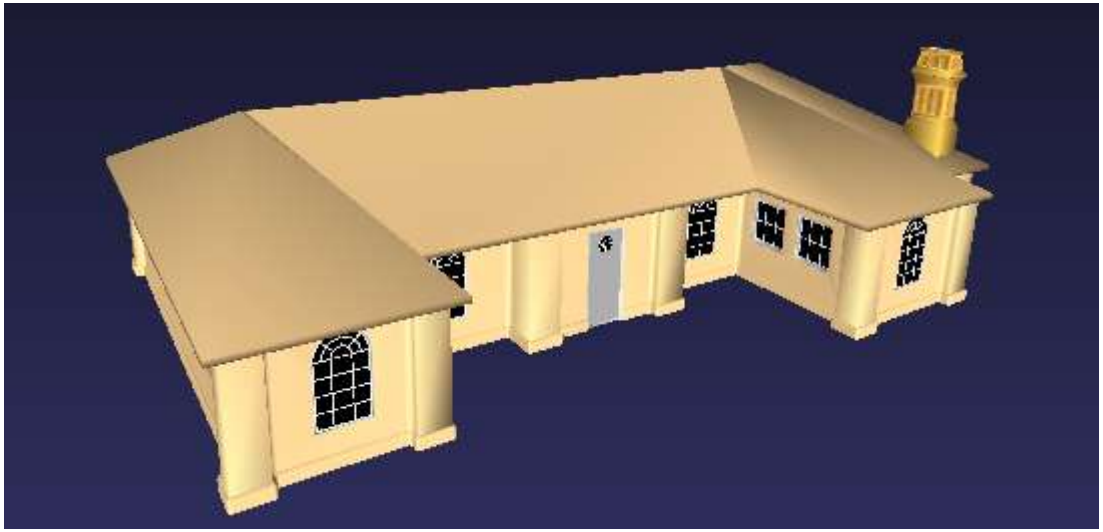
Perspective vs. Parallel



Perspective vs. Parallel

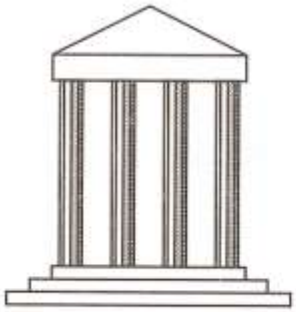


Parallel

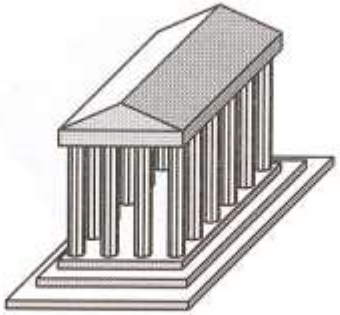


Perspective

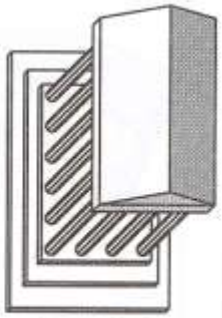
Classical Projections



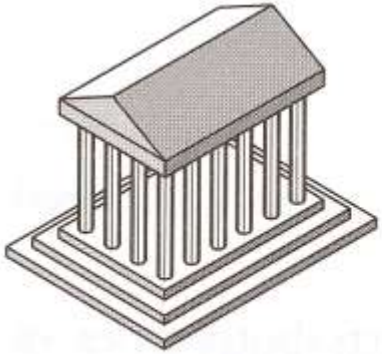
Front elevation



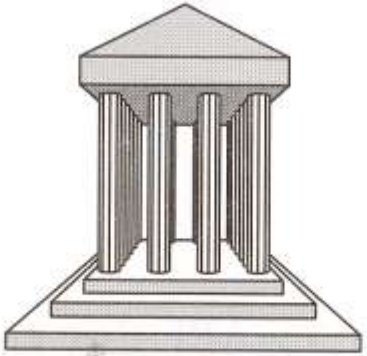
Elevation oblique



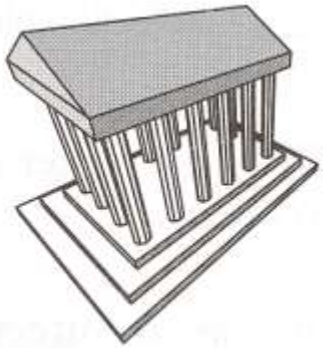
Plan oblique



Isometric



One-point perspective



Three-point perspective

Angel Figure 5.3

Summary

- Camera transformation
  - Map 3D world coordinates to 3D camera coordinates
  - Matrix has camera vectors as rows
- Projection transformation
  - Map 3D camera coordinates to 2D screen coordinates
  - Two types of projections:
    - Parallel
    - Perspective