Computer Graphics : Viewing in 3-D

By Naeem Iqbal ch.

Transformations in 3-D

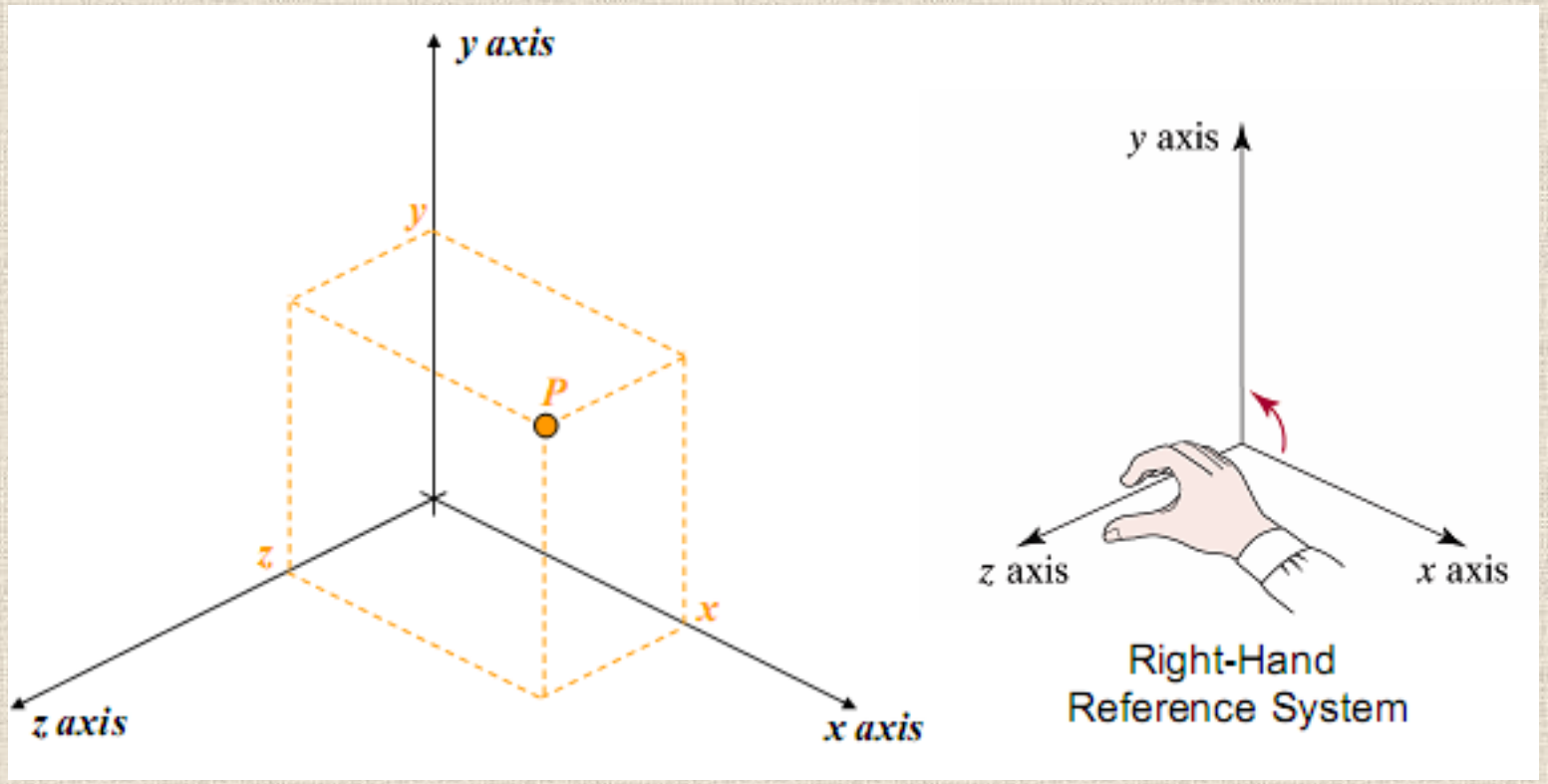
- How do transformations in 3-D work?
- 3-D homogeneous coordinates and matrix based transformations

Projections

- History
- Geometrical Constructions
- Types of Projection
- Projection in Computer Graphics

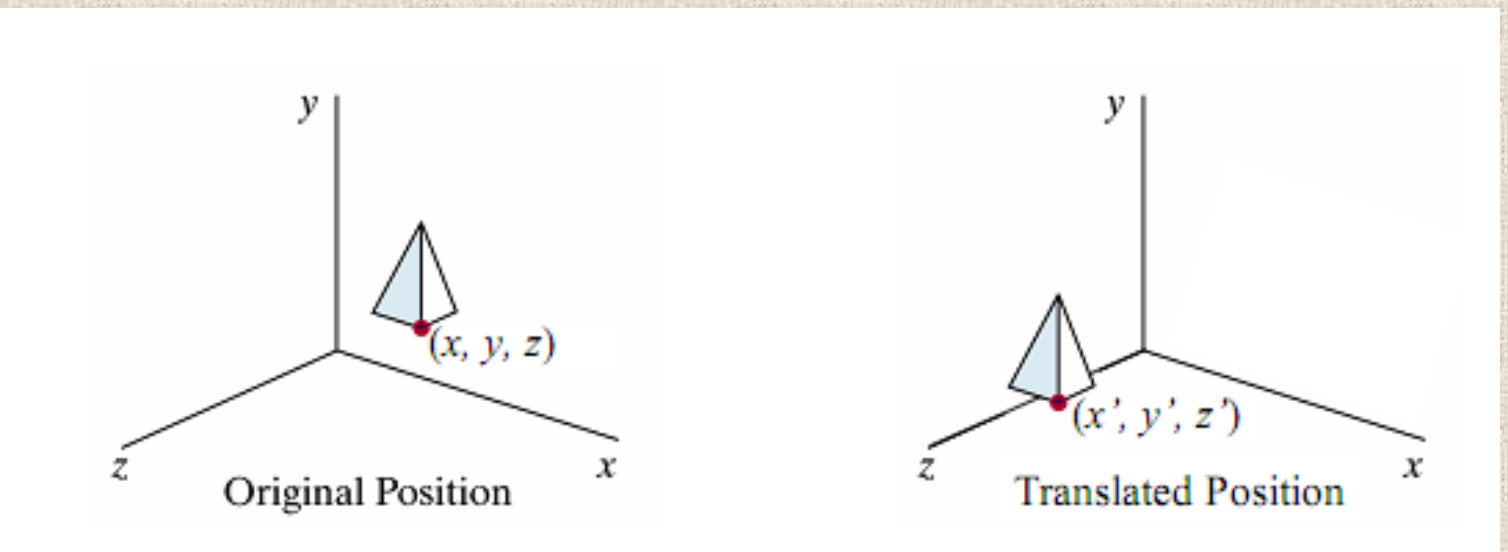
3-D Coordinate Spaces

- Remember what we mean by a 3-D
- coordinate space



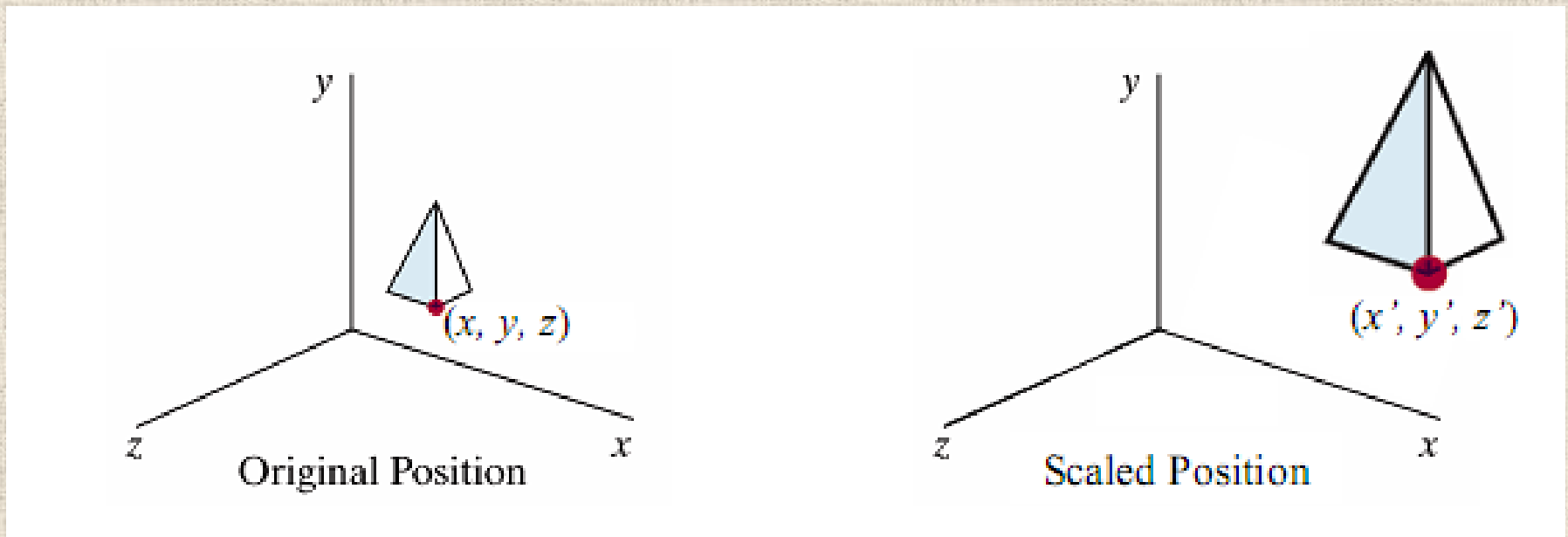
Translations In 3-D

- To translate a point in three dimensions by
- dx, dy and dz simply calculate the new points as follows:
  - x' = x + dx y' = y + dy z' = z + dz



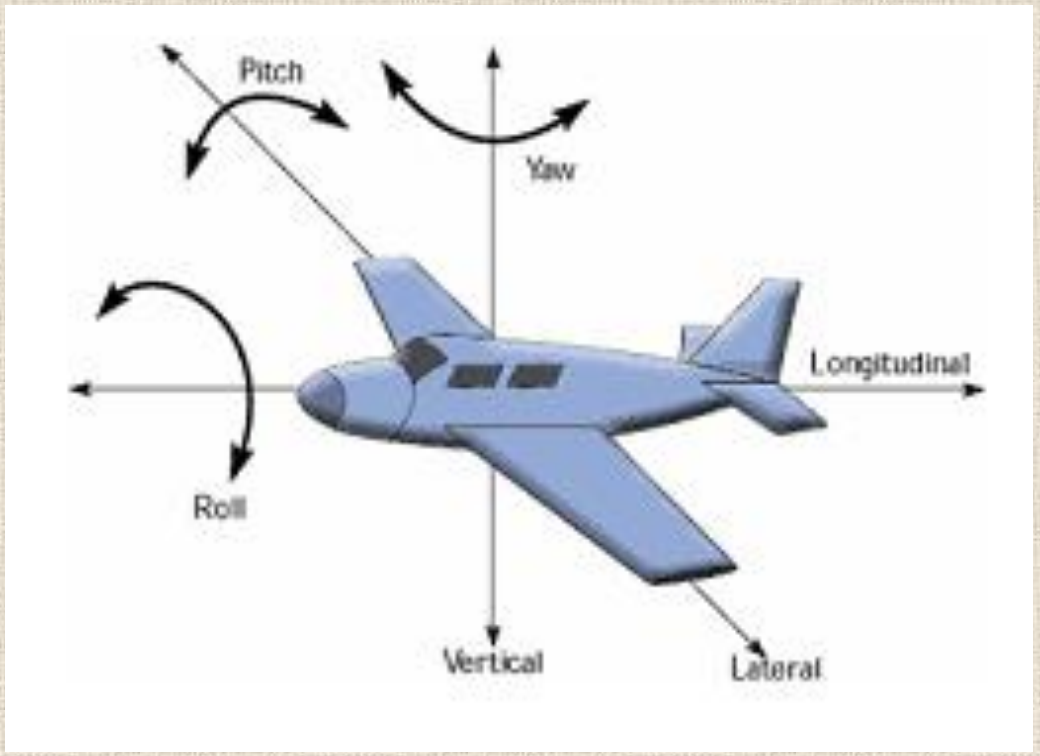
Scaling In 3-D

- To sale a point in three dimensions by sx, sy & sz simply calculate the new points as follows:
  - x’ = sx*x y’ = sy*y z’ = sz*z



Rotations In 3-D

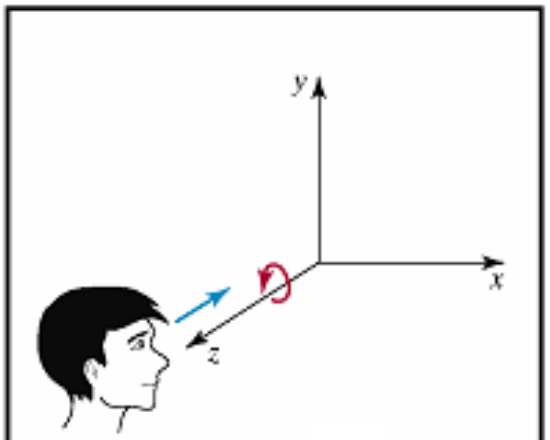
- When we performed rotations in two dimensions we only had the choice of rotating about the z axis
- In the case of three dimensions we have more options



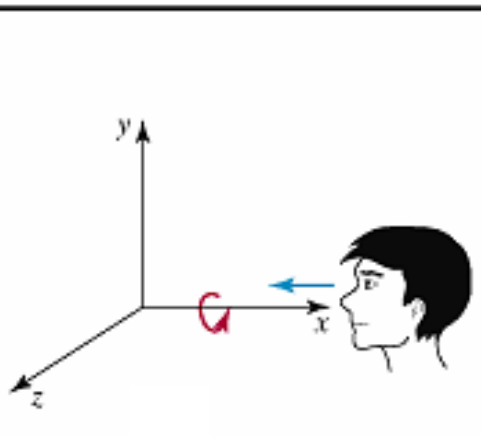
- Rotate about x – **pitch**
- Rotate about y –**yaw**
- Rotate about z -**roll**

Rotations In 3-D (cont...)

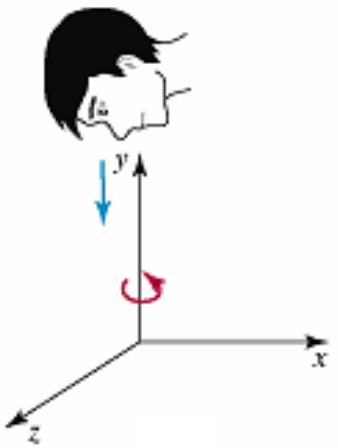
- The equations for the three kinds of rotations in 3-D are as follows:



$$x' = x \cdot \cos\theta - y \cdot \sin\theta$$
$$y' = x \cdot \sin\theta + y \cdot \cos\theta$$
$$z' = z$$



$$x' = x$$
$$y' = y \cdot \cos\theta - z \cdot \sin\theta$$
$$z' = y \cdot \sin\theta + z \cdot \cos\theta$$

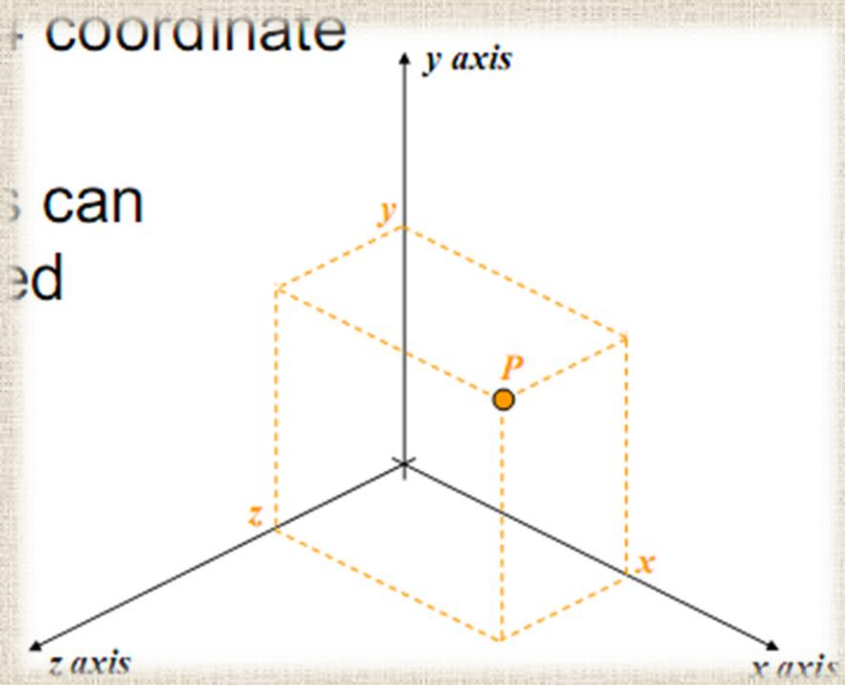


$$x' = z \cdot \sin\theta + x \cdot \cos\theta$$
$$y' = y$$
$$z' = z \cdot \cos\theta - x \cdot \sin\theta$$

Homogeneous Coordinates In 3-D

- Similar to the 2-D situation we can use homogeneous coordinates for 3-D transformations
- - 4 coordinate column vector
- All transformations can then be represented as matrices

$$P(x, y, z) = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$



3D Transformation Matrices

Translation by dx, dy, dz

$$\begin{bmatrix} 1 & 0 & 0 & dx \\ 0 & 1 & 0 & dy \\ 0 & 0 & 1 & dz \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} sx & 0 & 0 & 0 \\ 0 & sy & 0 & 0 \\ 0 & 0 & sz & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Scaling by sx, sy, sz

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Rotate About X-Axis

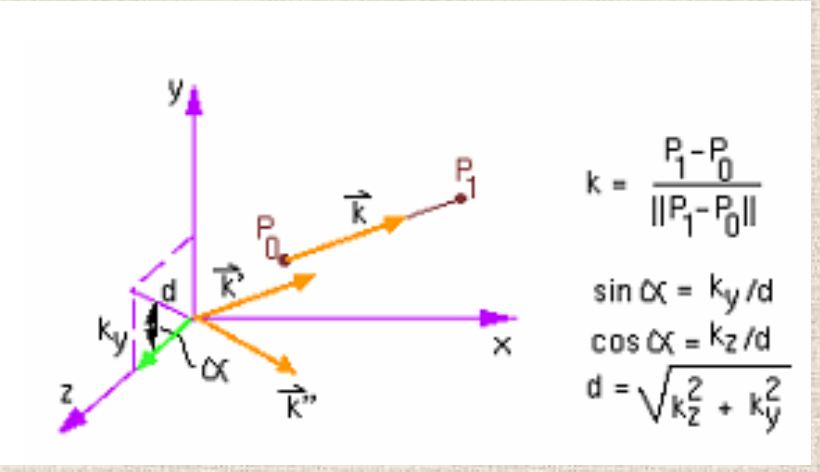
$$\begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Rotate About Y-Axis

$$\begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Rotate About Z-Axis

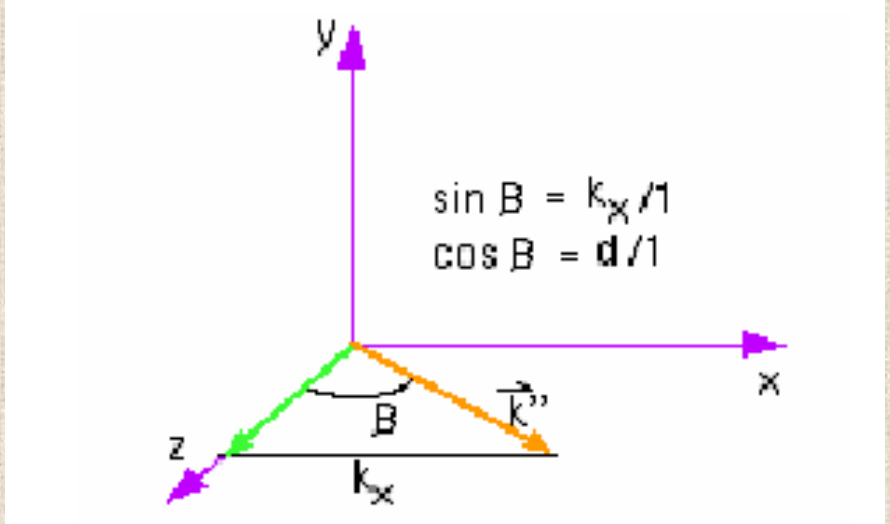
3D Rotation About Arbitrary Axis



trans(-P0): translate axis k to origin

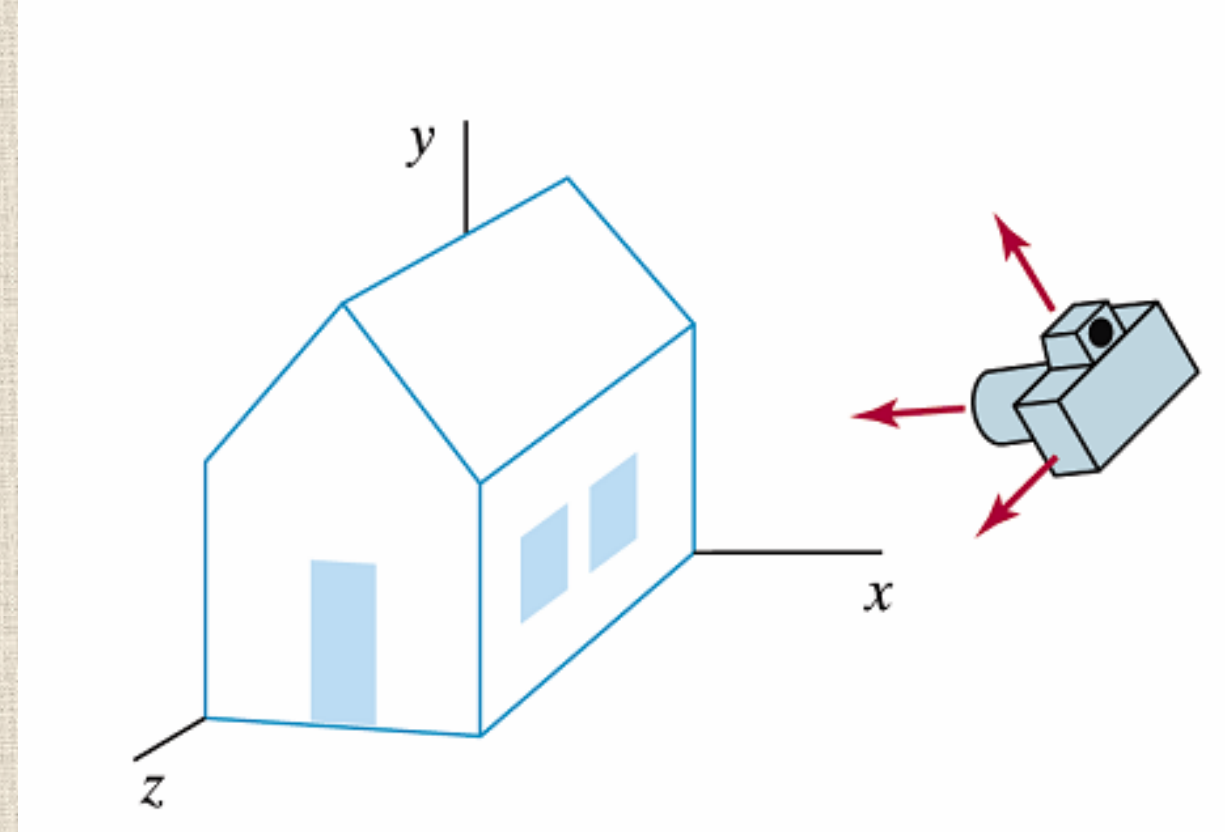
- rot(x,alpha): rotate about x-axis to bring axis k' to lie in xz plane.
- The amount of rotation is determined by looking at the projection on the yz plane.
- Alpha need not actually be calculated; it's sine and cosine can be evaluated directly

3D Rotation About Arbitrary Axis



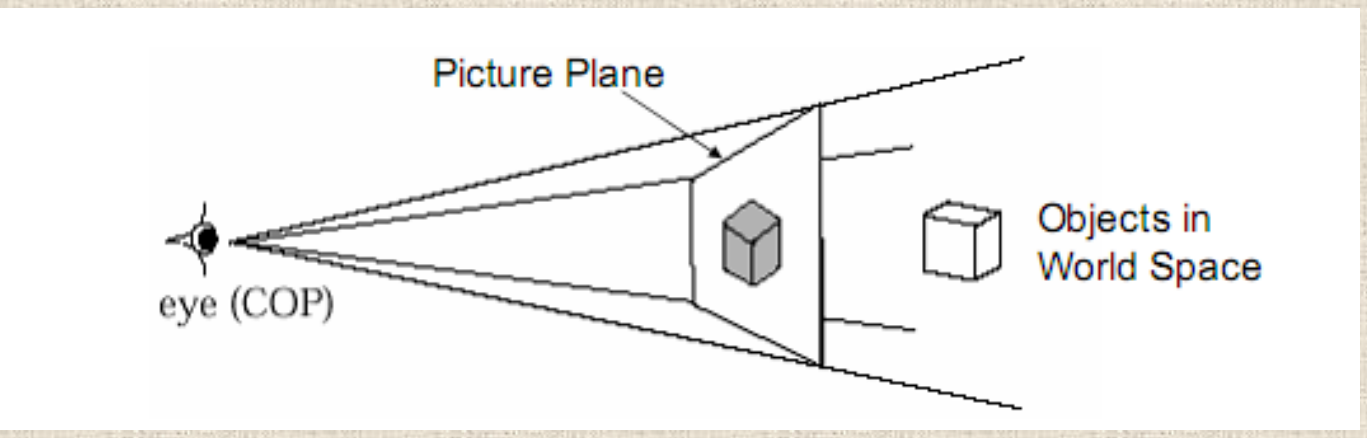
- rot(y,-beta): rotate about y- axis to align axis k" with z- axis.
- As in the previous step, we need not actually calculate beta.
- rot(z, theta): perform the desired rotation to the object.
- reverse all the other steps.

Remember The Big Idea



What Are Projections?

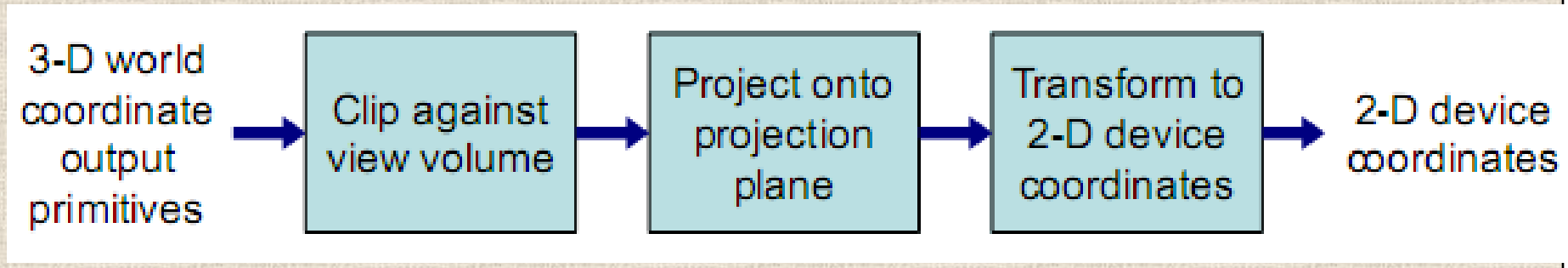
- Our 3-D scenes are all specified in 3-D world coordinates
- To display these we need to generate a 2-D image - project objects onto a picture plane



So how do we figure out these projections?

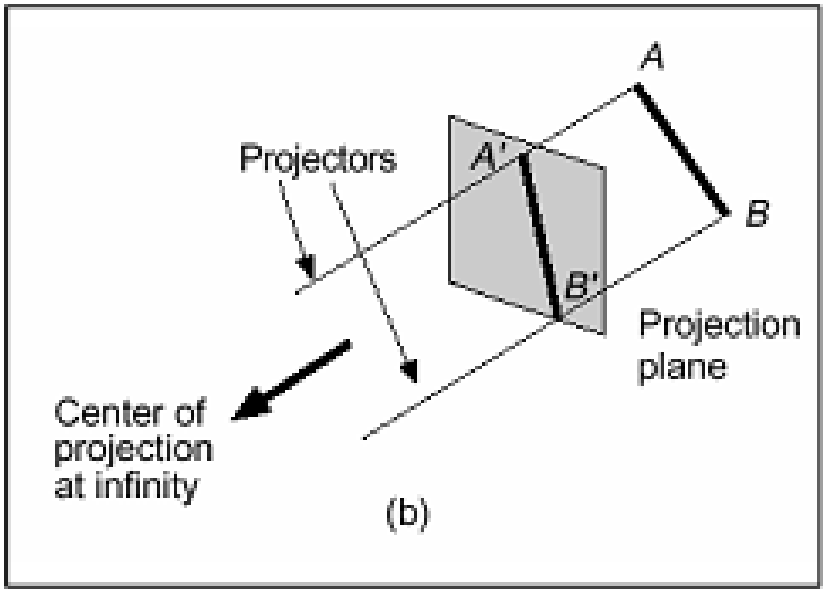
Converting From 3-D To 2-D

- Projection is just one part of the process of converting from 3-D world coordinates to a 2-D image

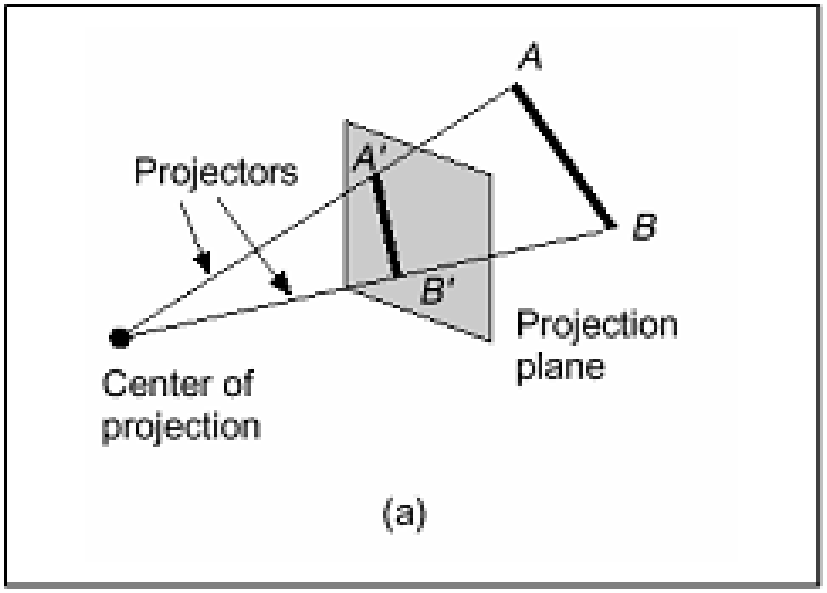


Types Of Projections

- There are two broad classes of projection:
  - **Parallel**: Typically used for architectural and engineering drawings
  - **Perspective**: Realistic looking and used in computer



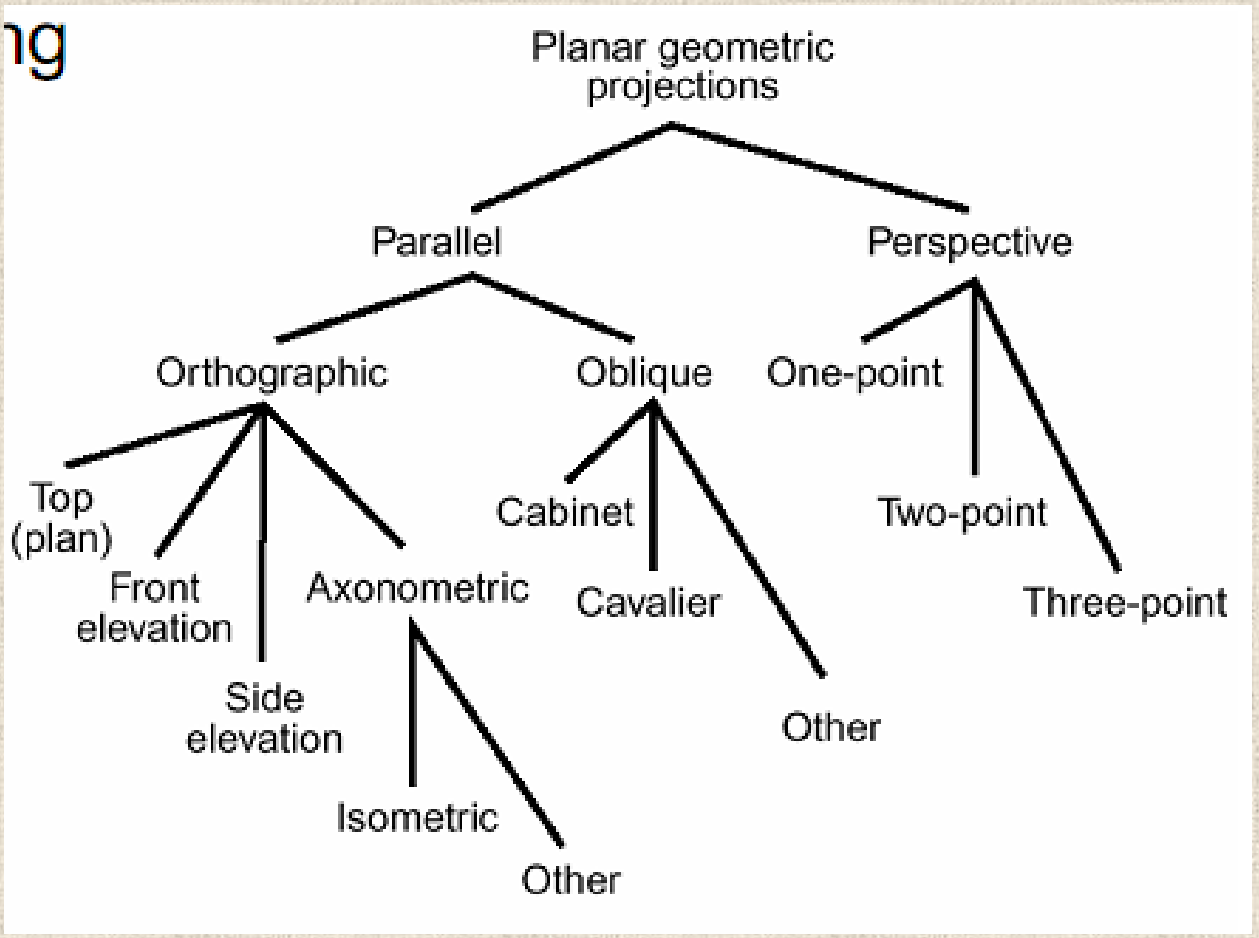
Parallel Projection



Perspective Projection

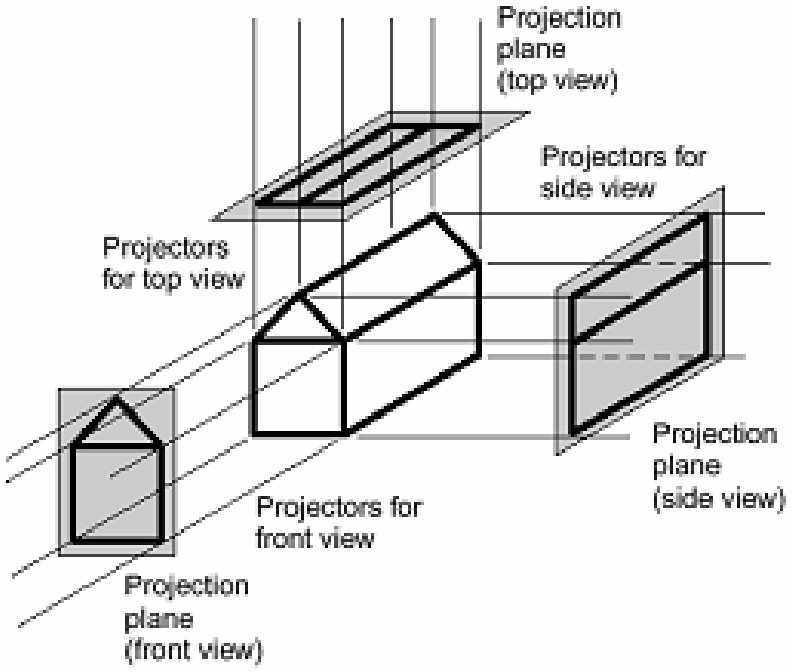
Types Of Projections (cont...)

- For anyone who did engineering or technical drawing

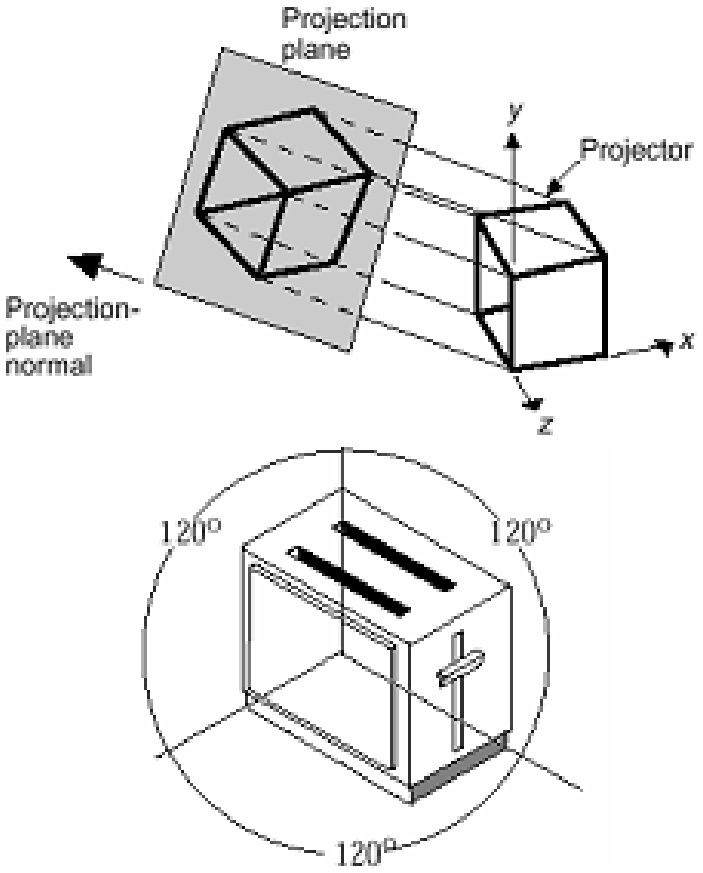


Parallel Projections

- Some examples of parallel projections



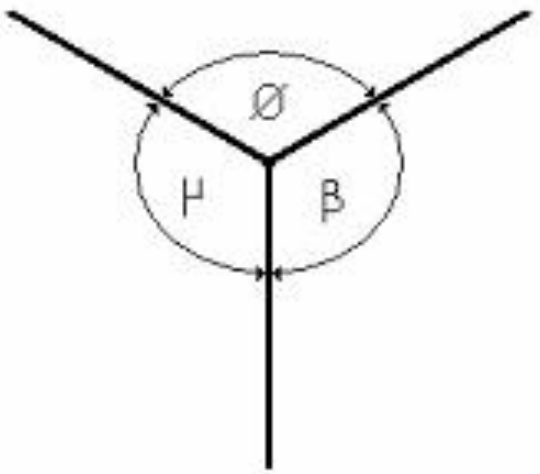
Orthographic Projection

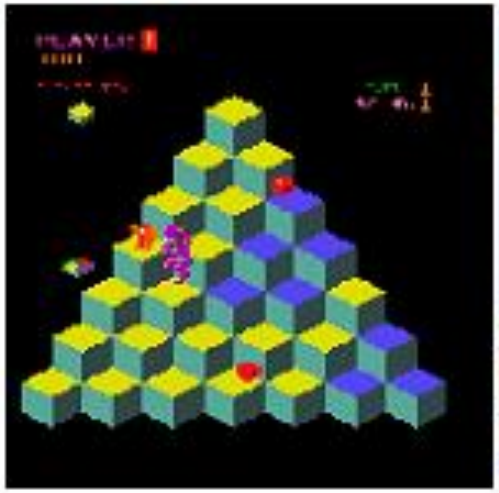


Isometric Projection

Isometric Projections

- In isometric projection the angles between the projection of the axes are equal i.e. 120º.



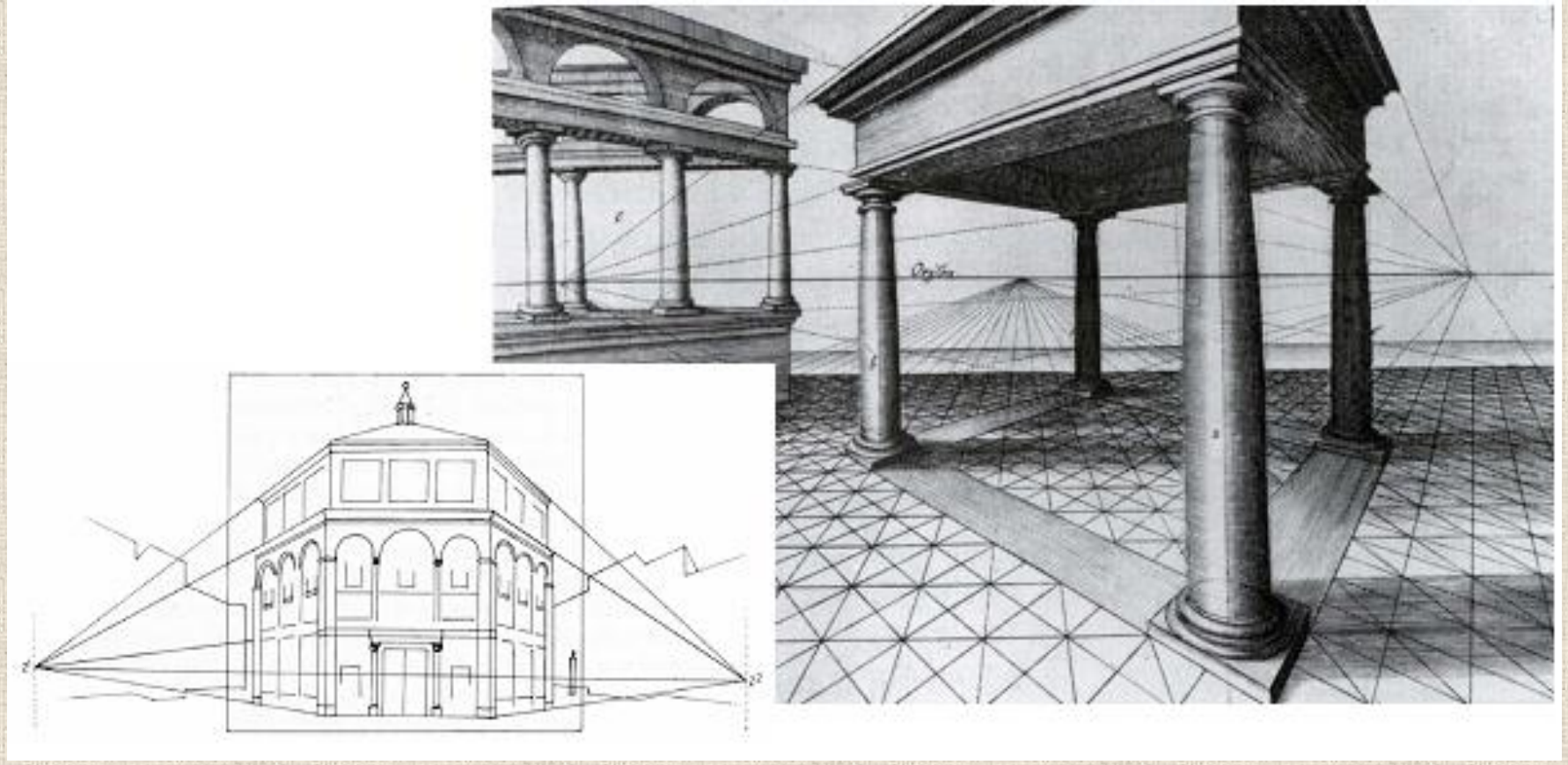


Q*Bert

Sim City

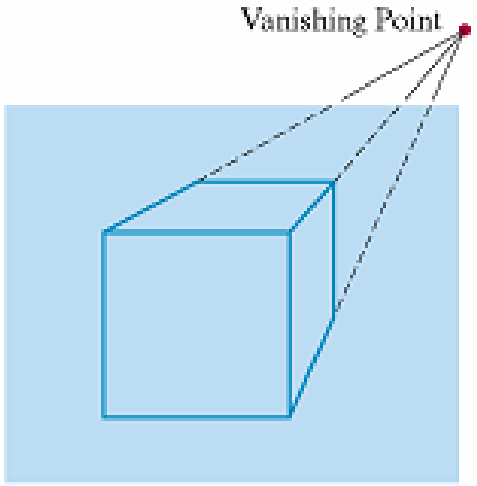
Perspective Projections

- Perspective projections are much more realistic than parallel projections

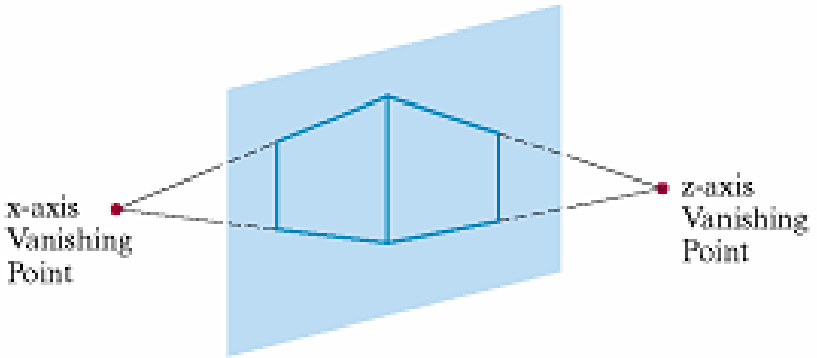


Perspective Projections

- There are a number of different kinds of perspective views
- The most common are one-point and two point perspectives

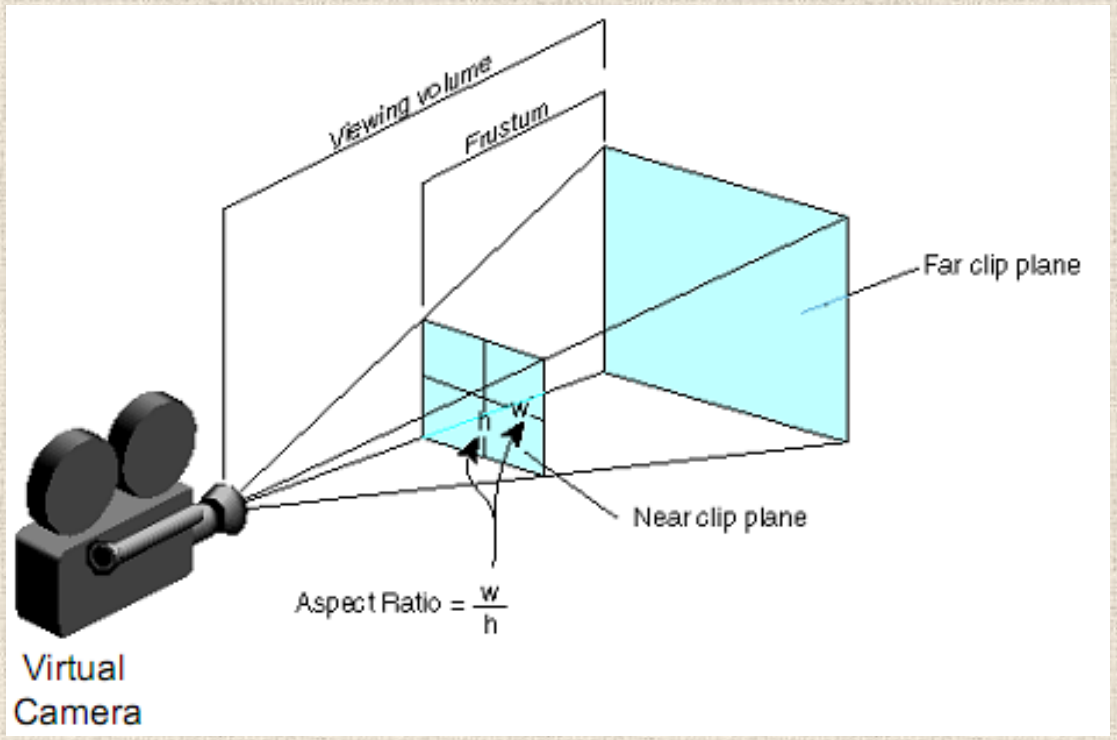


One Point Perspective Projection



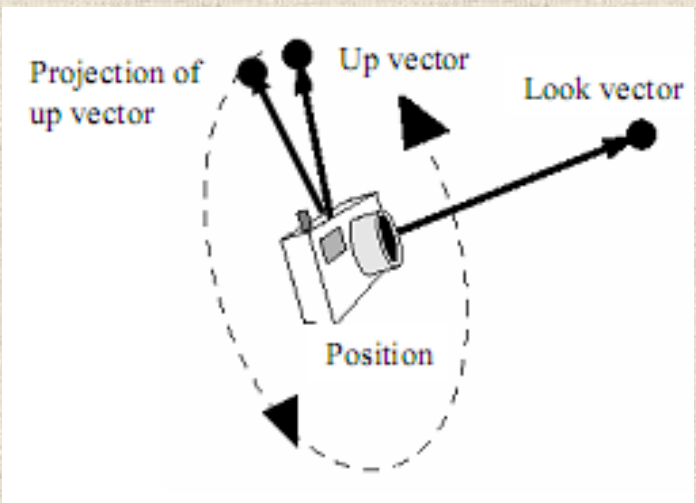
Two-Point Perspective Projection

Elements Of A Perspective Projection



The Up And Look Vectors

- The **look vector** indicates the direction in which the camera is pointing
- The **up vector** determines how the camera is rotated
  - For example, is the camera held vertically or horizontally



Summary

In this part of today's lecture we looked at:

- Transformations in 3-D
  - Very similar to those in 2-D
- Projections
  - 3-D scenes must be projected onto a 2-D image Plane
  - Lots of ways to do this
    - Parallel projections
    - Perspective projections
  - The virtual camera