**Dr. George Karraz, Ph. D.**

# Viewing in 3-D

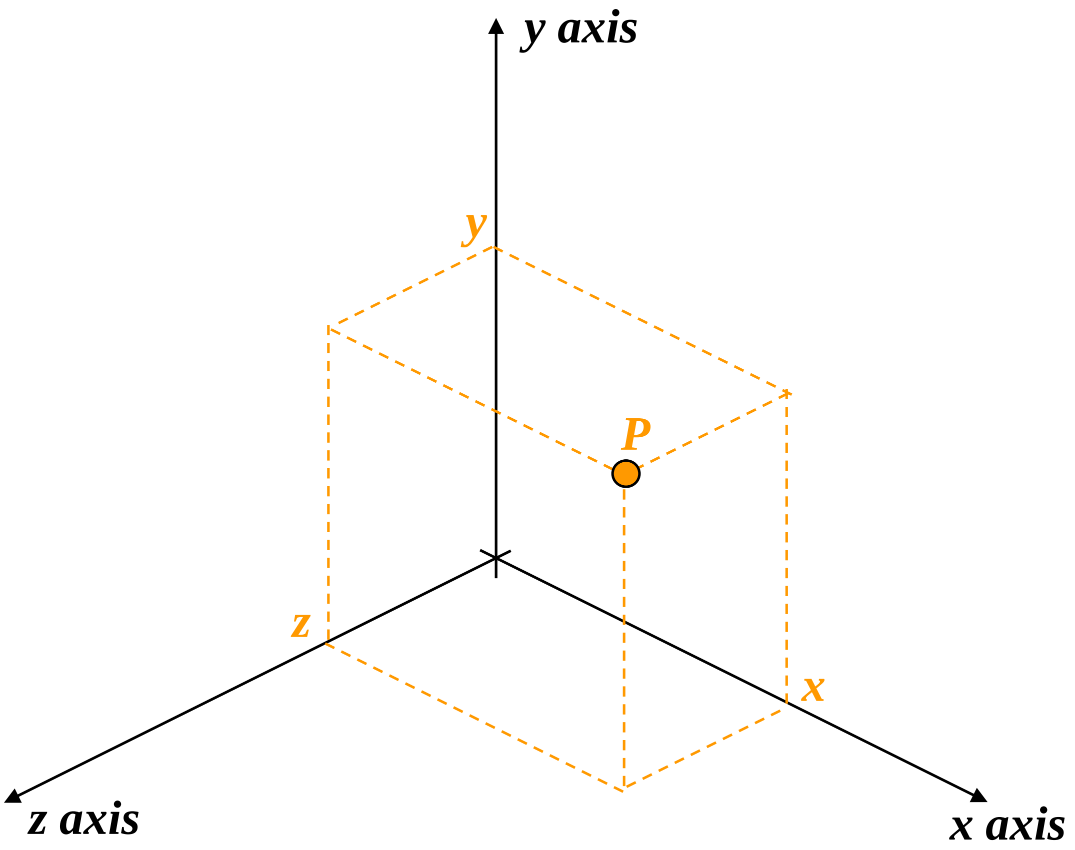
Projections

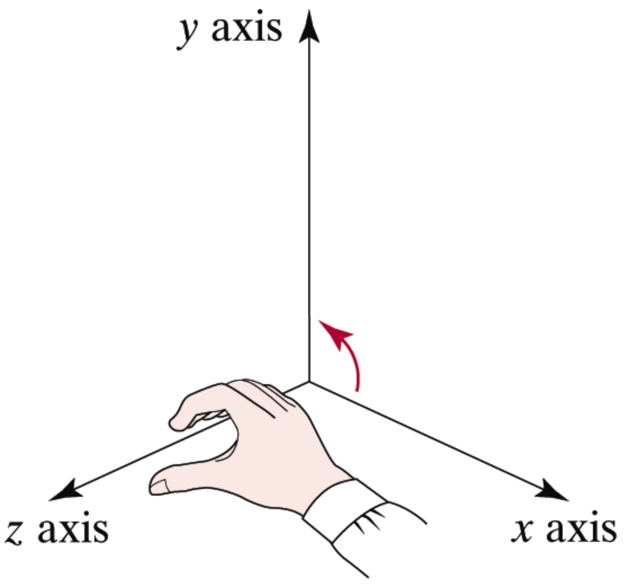
# Outline

- Review of transformations in 3-D
  - 3-D Coordinate Spaces
  - How do transformations in 3-D work?
  - 3-D homogeneous coordinates and matrix based transformations
- Projections
  - Types of Projection
  - Parallel Projections
  - Perspective projecton

# 3-D Coordinate Spaces

Remember what we mean by a 3-D coordinate space



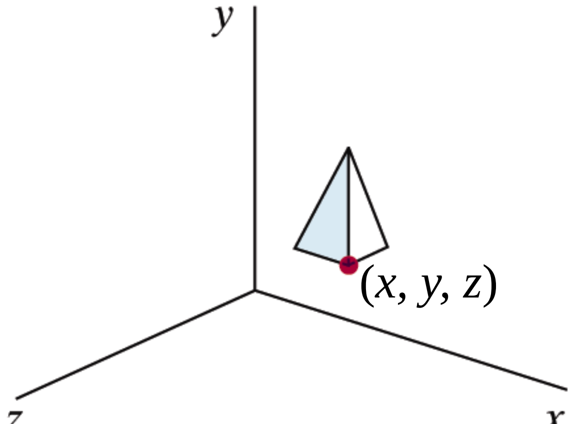




Right-Hand Reference System

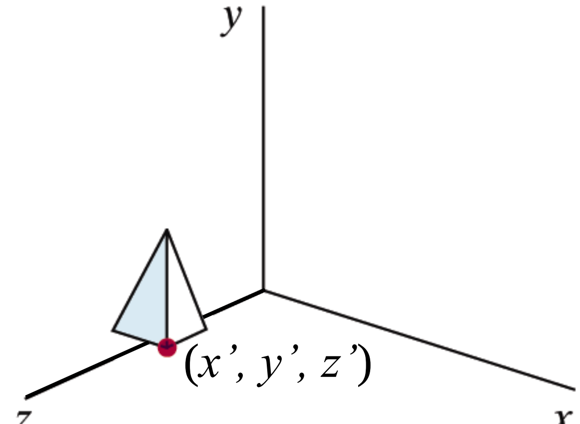
# Translations In 3-D

To translate a point in three dimensions by $dx$, $dy$ and $dz$ simply calculate the new points as follows:

$$x' = x + dx \qquad y' = y + dy \qquad z' = z + dz$$




Original Position

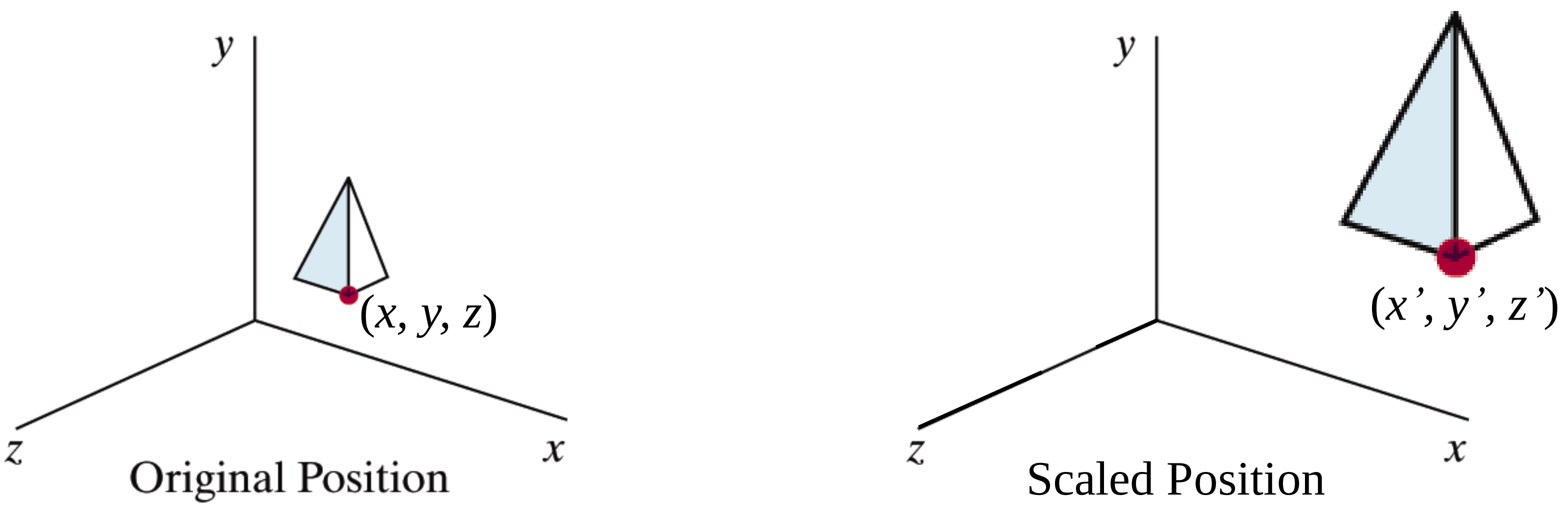



Original Position

# Scaling In 3-D

To sale a point in three dimensions by *sx*, *sy* and *sz* simply calculate the new points as follows:

$$x' = sx*x \qquad y' = sy*y \qquad z' = sz*z$$

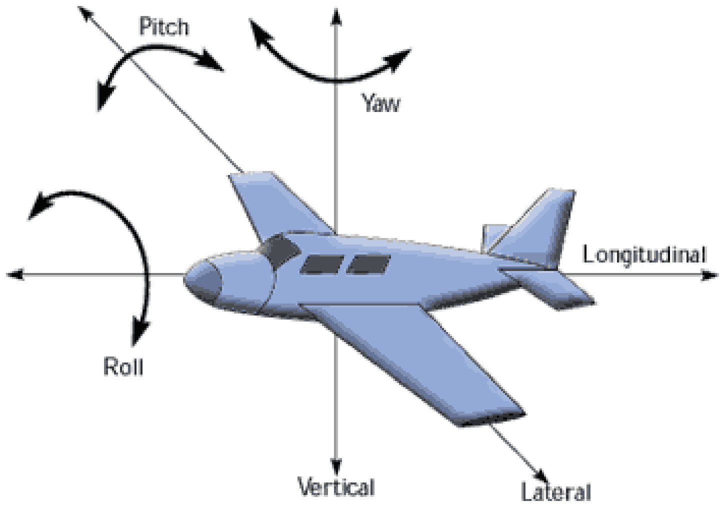


# Rotations In 3-D

When we performed rotations in two dimensions we only had the choice of rotating about the $Z$ axis

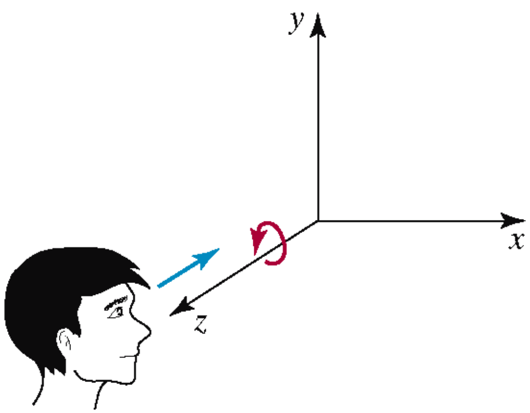
In the case of three dimensions we have more options

- Rotate about $X$ – pitch
- Rotate about $y$ – yaw
- Rotate about $Z$ - roll

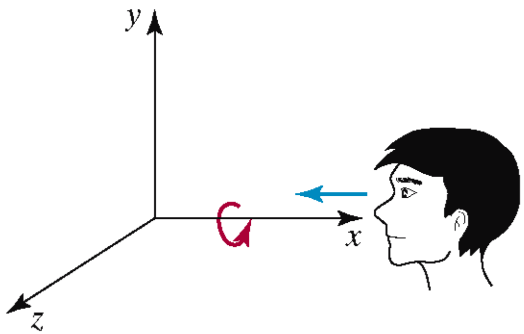


# Rotations In 3-D

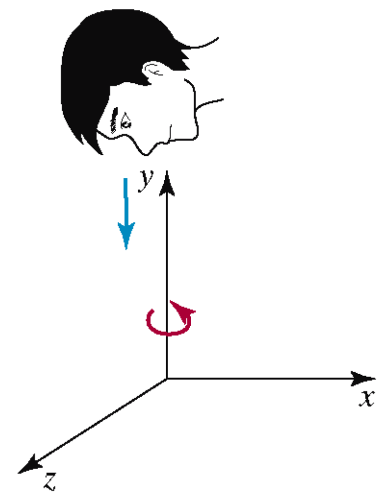
The equations for the three kinds of rotations in 3-D are as follows:




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




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



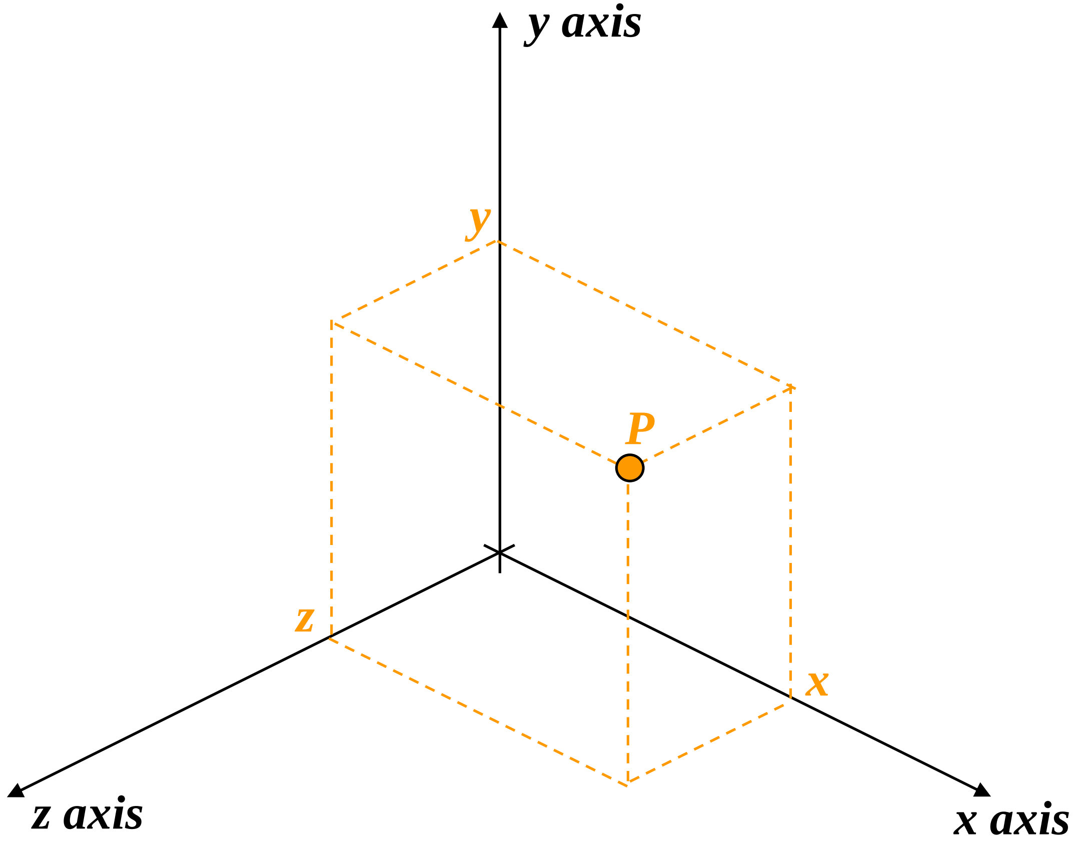

$$x' = z \cdot \sin\theta + x \cdot \cos\theta$$
$$y' = y$$
$$z' = z \cdot \cos\theta - x \cdot \sin\theta$$

# Homogeneous Coordinates In 3-D

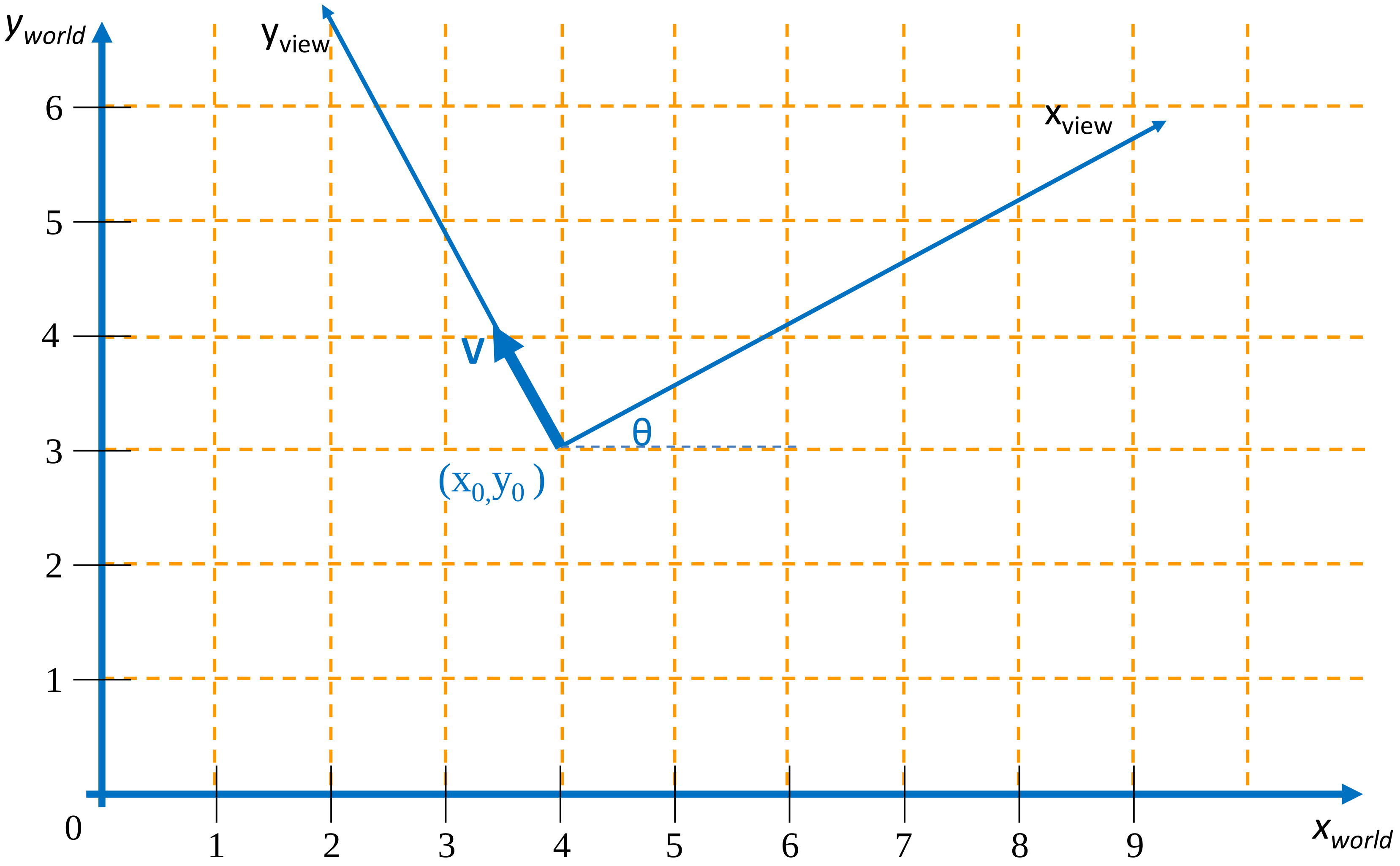
Similar to the 2-D situation we can use homogeneous coordinates for 3-D transformations - 4 coordinate column vector

All transformations can then be represented as matrices

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



# Transformations between two coordinate systems ($x_{world}, y_{world}$) and ($x_{view}, y_{view}$)



# Transformations between coordinate systems

- Consider two Cartesian systems $x_{world}$, $y_{world}$ and ($x_{view}$, $y_{view}$), with the coordinate origins at (0,0) and ($x_0$, $y_0$) and with an orientation angle theta θ between the x and $x_{view}$ axes.
- To transform object descriptions from *xy* coordinates to $x_{view}$ $y_{view}$ coordinates, we need to set up a transformation that superimposes the $x_{view}$ $y_{view}$ axes onto the *xy* axes. This is done in two steps:
  1. Translate so that the origin ($x_0$, $y_0$) of the $x_{view}$ $y_{view}$ system is moved to the origin of the xy system.
  2. Rotate the $x_{view}$ axis onto the x axis.

# Transformations between coordinate systems Contd.

- Translation of the coordinate origin is expressed with the matrix operation

$$T(-x_0,-y_0)=\begin{bmatrix}1 & 0 & -x_0\\ 0 & 1 & -y_0\\ 0 & 0 & 1\end{bmatrix}$$

- To get the axes of the two systems into coincidence, we then perform the clockwise rotation

$$R(-\theta)=\begin{bmatrix}\cos\theta & \sin\theta & 0\\ -\sin\theta & \cos\theta & 0\\ 0 & 0 & 1\end{bmatrix}$$

# Alternate method

- An alternate method for giving the orientation of the second coordinate system is to specify a vector **V** that indicates the direction for the positive $y_{view}$ axis

- A unit vector in the $y_{view}$ direction can then be obtained as **v** = **V**/|**V**| = $(v_x, v_y)$
- And we obtain the unit vector **u** along the $x_{view}$ axis by rotating **v** $90^o$ clockwise → **u** = $(u_x, u_y) = (v_y, -v_x)$

# Rotation matrix as elements of a set of orthogonal unit vectors

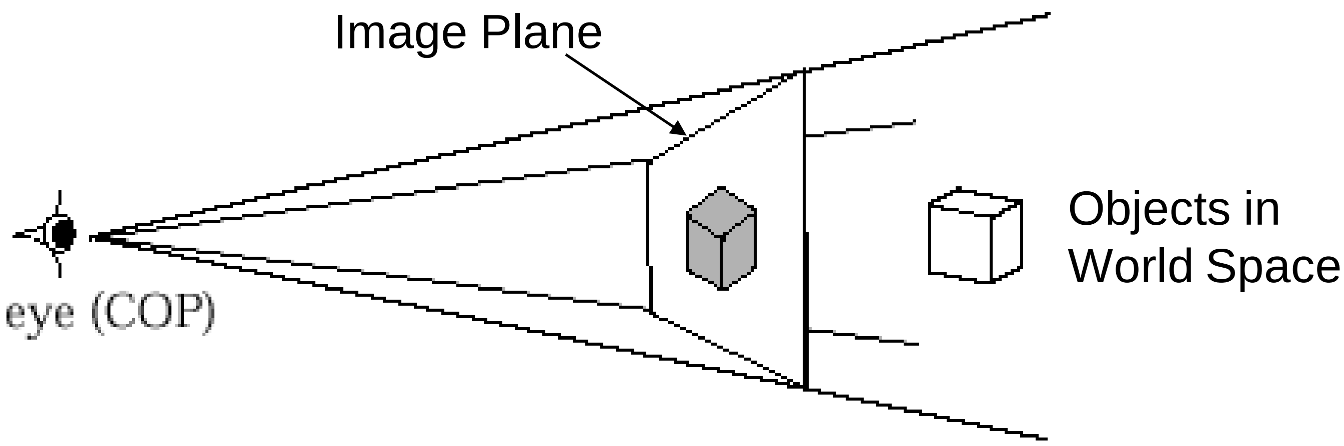
- The elements of any rotation matrix could be expressed as elements of a set of orthogonal unit vectors. Therefore, the matrix to rotate the r'y' system into coincidence with the xy system can be written as

$$R = \begin{bmatrix} u_x & u_y & 0 \\ v_x & v_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

# 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?

# Projections

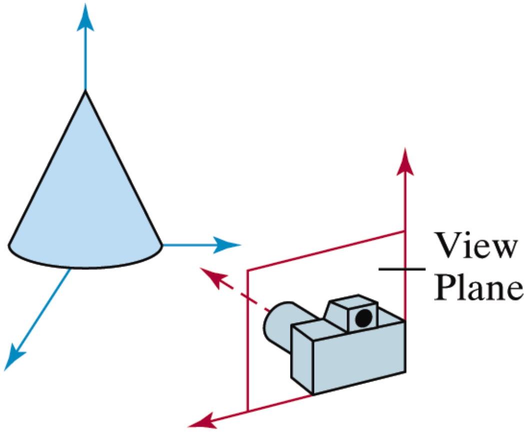



Figure 7-1

Coordinate reference for obtaining a selected view of a three-dimensional scene.

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

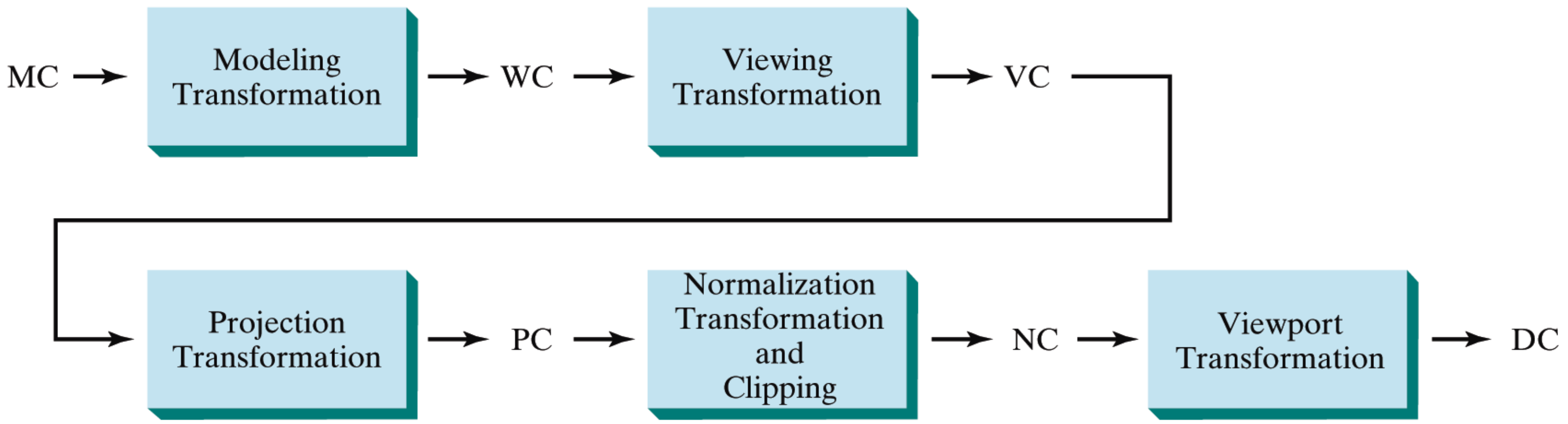



Figure 7-11

General three-dimensional transformation pipeline, from modeling coordinates to world coordinates to viewing coordinates to projection coordinates to normalized coordinates and, ultimately, to device coordinates.

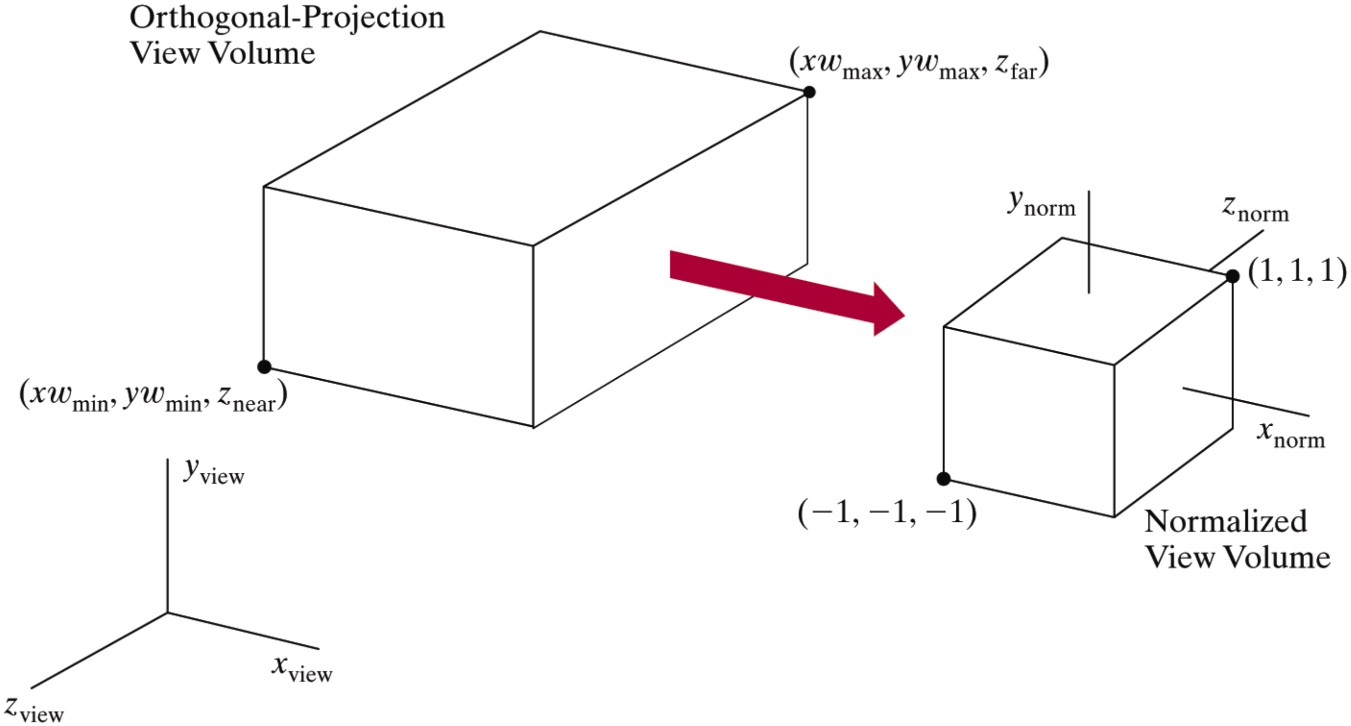


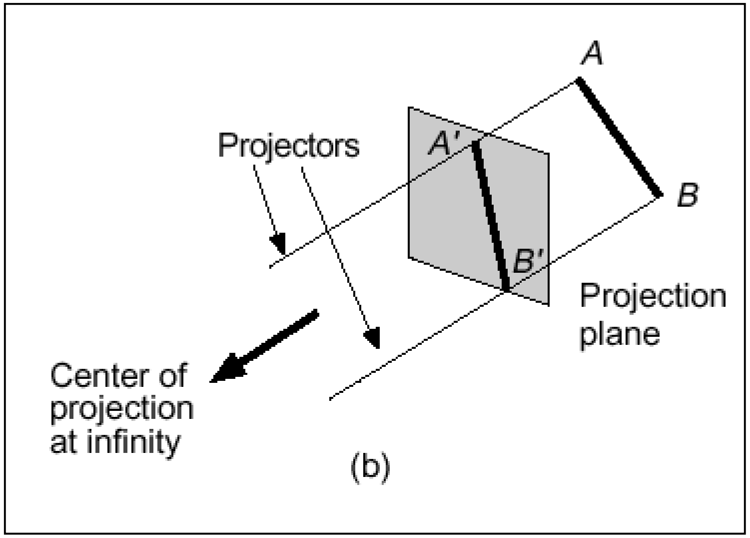

Figure 7-31

Normalization transformation from an orthogonal-projection view volume to the symmetric normalization cube within a left-handed reference frame.

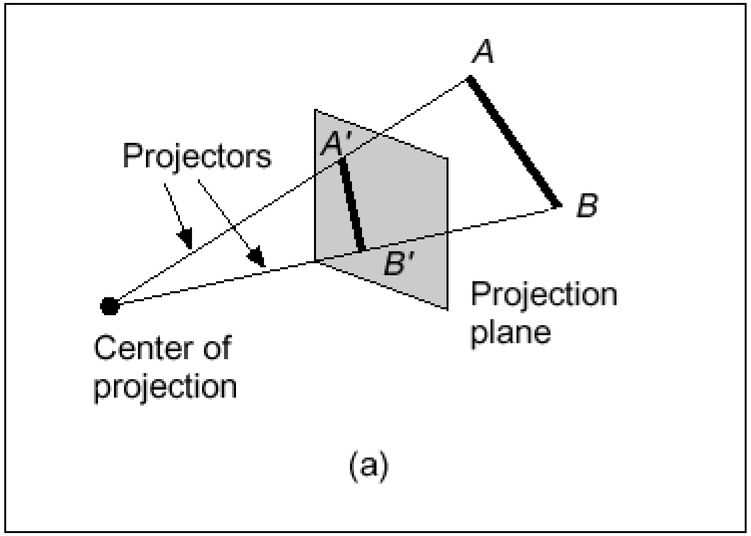
# Types Of Projections

There are two broad classes of projection:

- **Parallel:** Typically used for architectural and engineering drawings (maintain relative proportions of the object)
- **Perspective:** Realistic looking and used in computer graphics




Parallel Projection




Perspective Projection

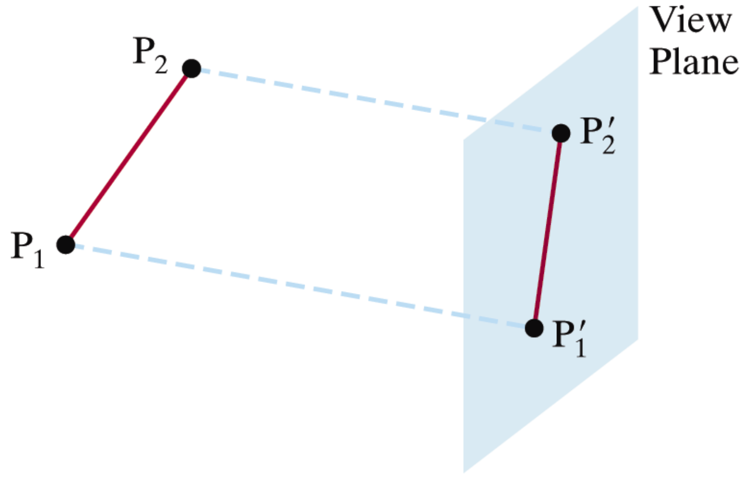



Figure 7-22

Parallel projection of a line segment onto a view plane.

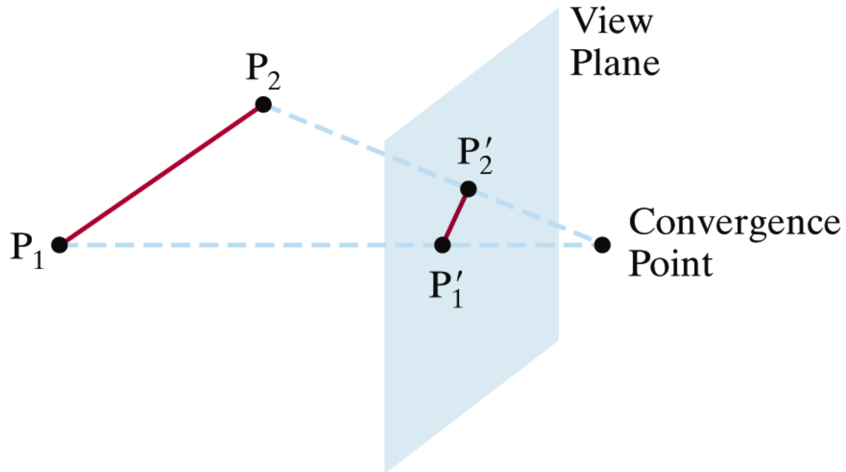
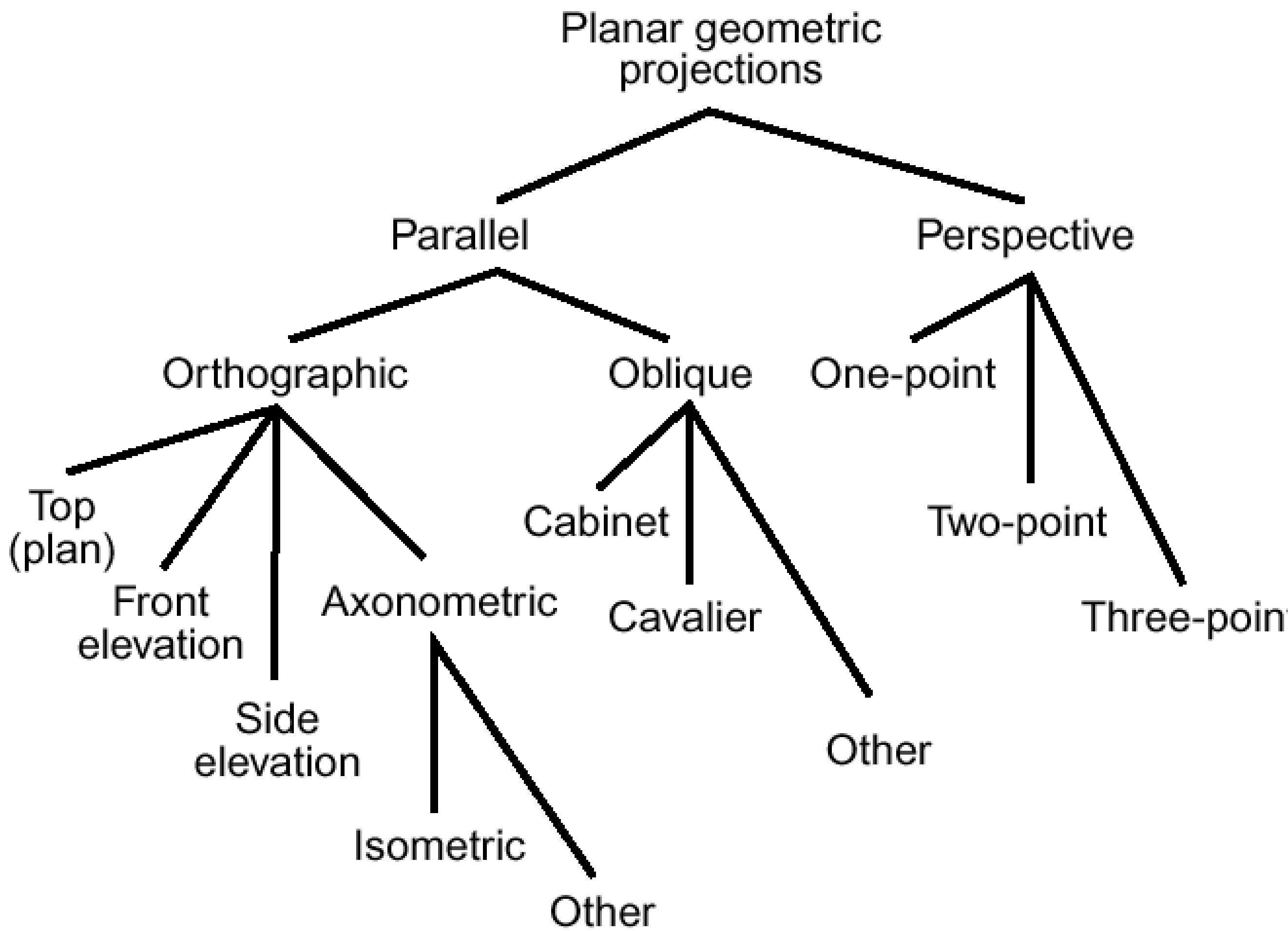



Figure 7-23

Perspective projection of a line segment onto a view plane.

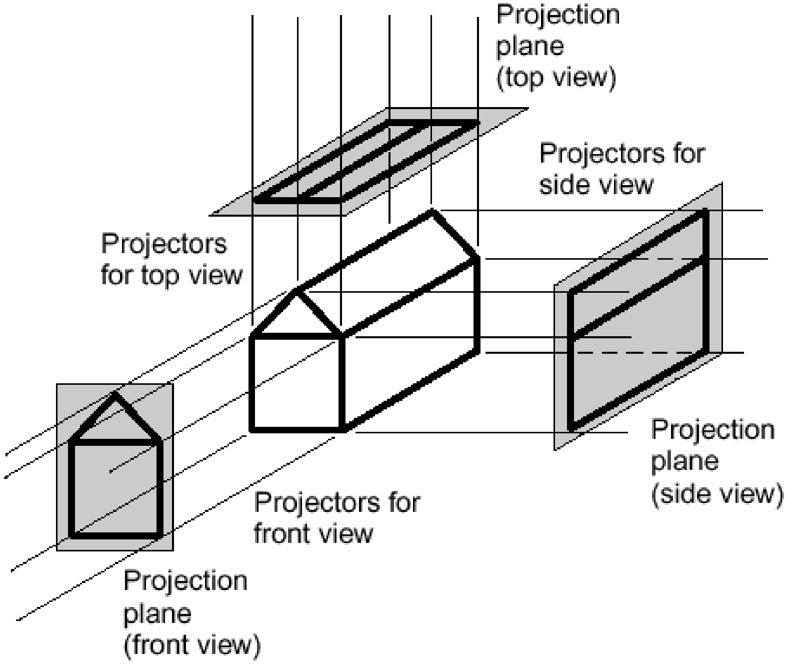
# Types Of Projections

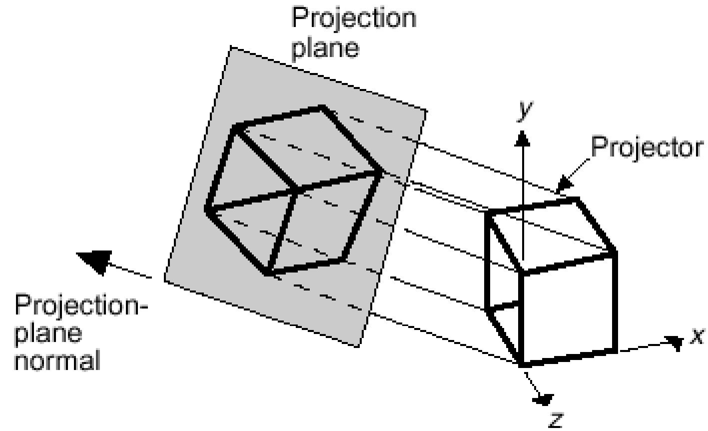
For anyone who did engineering or technical drawing

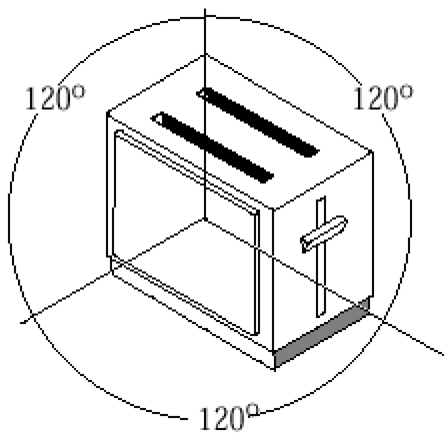


# Parallel Projections

Some examples of parallel projections

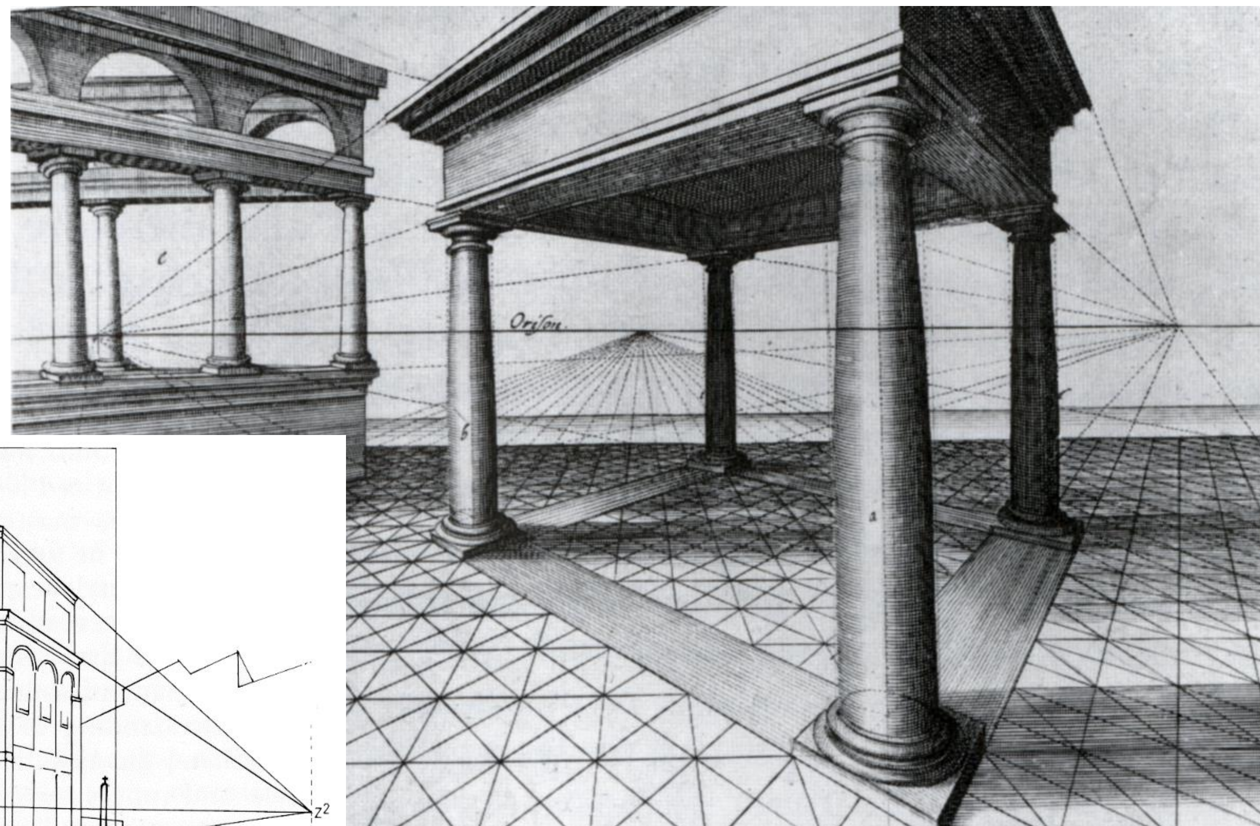


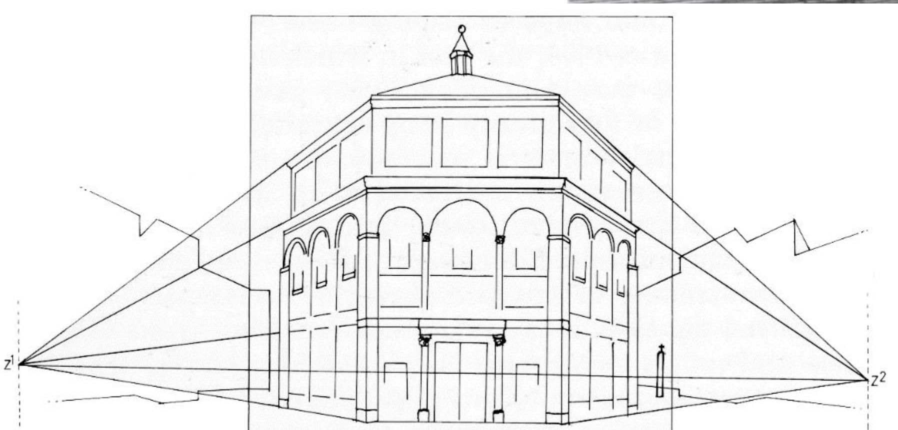



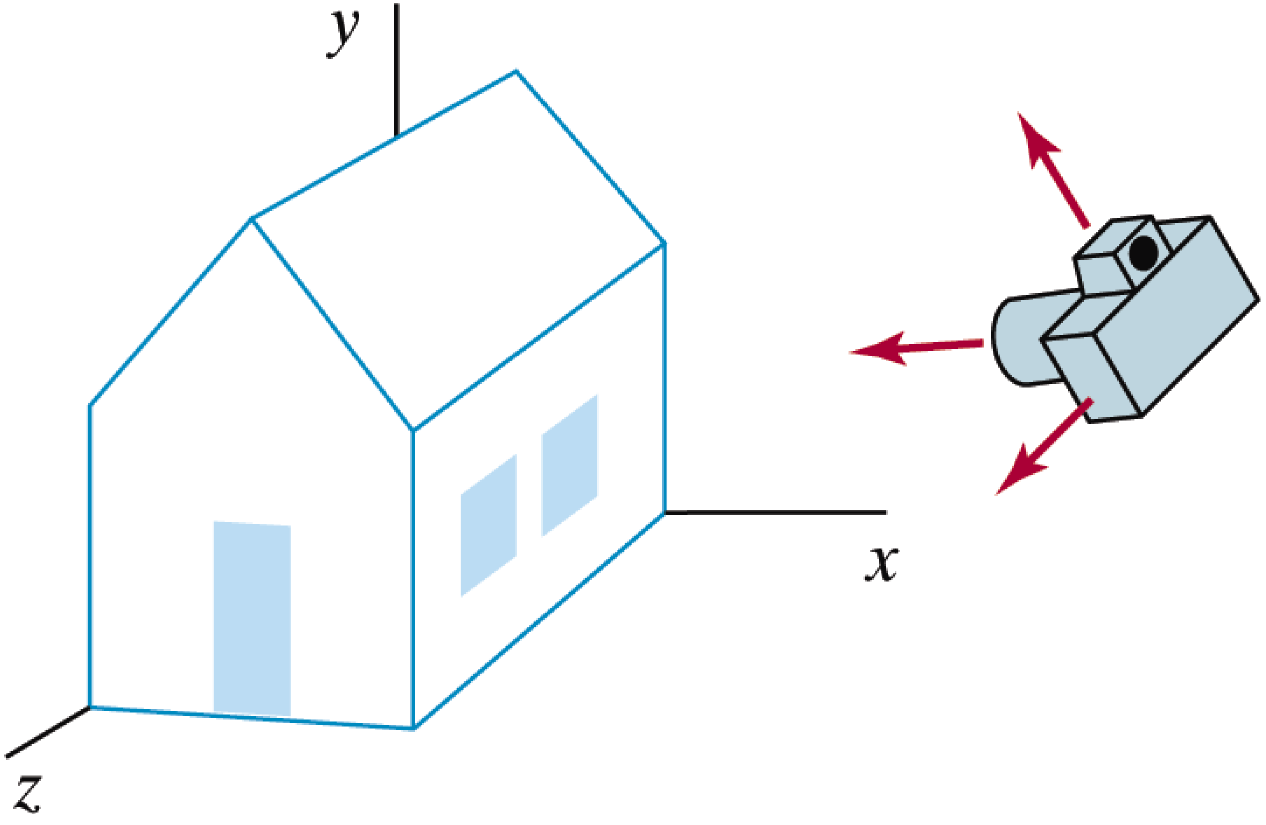
# Perspective Projections

Perspective projections are much more realistic than parallel projections

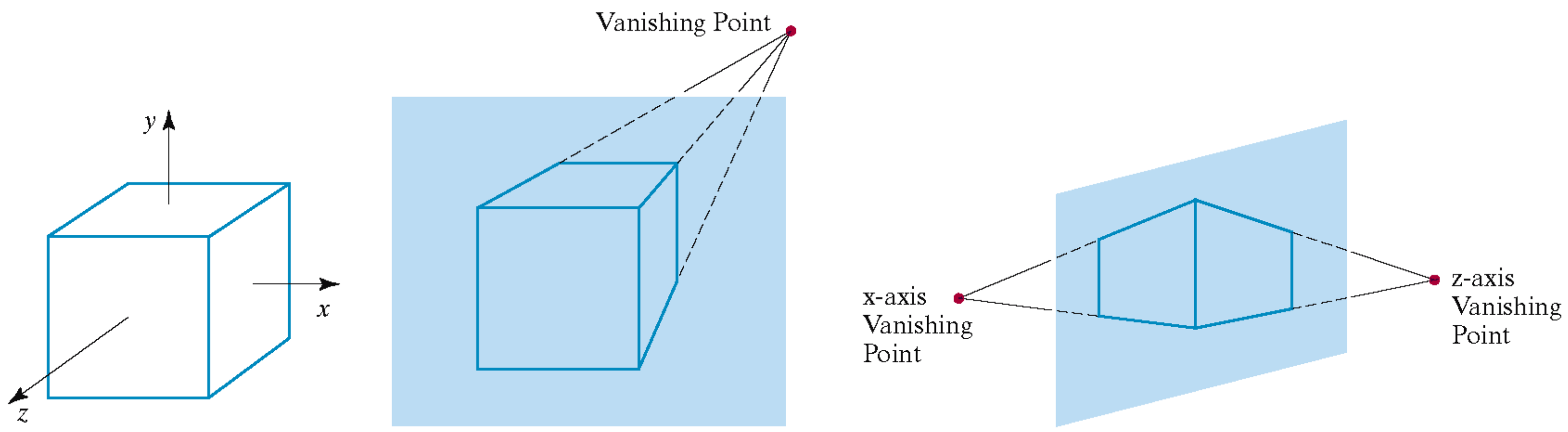


# Perspective Projections



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




(a) Principle Axes for Cube

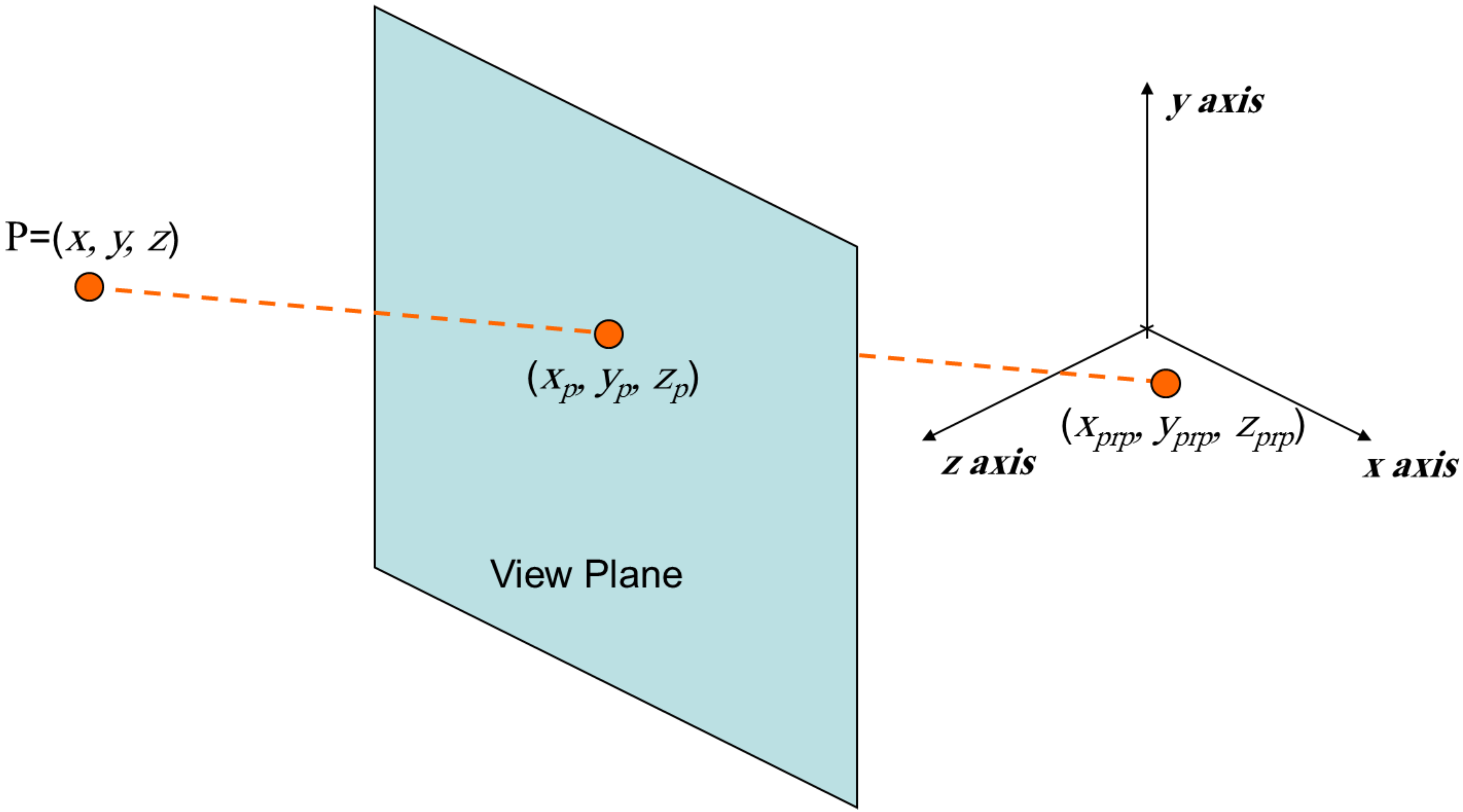
(b) One-Point Perspective Projection

(c) Two-Point Perspective Projection

Figure 7-44

Principal vanishing points for perspective-projection views of a cube. When the cube in (a) is projected to a view plane that intersects only the *z* axis, a single vanishing point in the *z* direction (b) is generated. When the cube is projected to a view plane that intersects both the *z* and *x* axes, two vanishing points (c) are produced.

# Perspective projection of point P(x,y,z)



# Projection Calculations

Any point along the projector ($x'$, $y'$, $z'$) can be given as:

$$x' = x - (x - x_{prp})u$$

$$y' = y - (y - y_{prp})u \qquad 0 \le u \le 1$$

$$z' = z - (z - z_{prp})u$$

When $u$ = 0 we are at P, while when $u$ = 1 we are at the *Projection Reference Point*

# Projection Calculations

- At the view plane $z' = z_{vp}$ so we can solve the $z'$ equation for $u$:

$$u = \frac{z_{vp} - z}{z_{prp} - z}$$

# Projection Calculations

Armed with this we can restate the equations for $x'$ and $y'$ for general perspective:

$$x_{vp} = x\left(\frac{z_{prp} - z_{vp}}{z_{prp} - z}\right) + x_{prp}\left(\frac{z_{vp} - z}{z_{prp} - z}\right)$$

$$y_{vp} = y\left(\frac{z_{prp} - z_{vp}}{z_{prp} - z}\right) + y_{prp}\left(\frac{z_{vp} - z}{z_{prp} - z}\right)$$

# Projection Calculations: special case

To simplify the perspective calculation, the projection reference point could be limited to positions along the zview axis, then

$x_{prp} = y_{prp} = 0$:

$$x_{vp} = x\left(\frac{z_{prp} - z_{vp}}{z_{prp} - z}\right)$$

$$y_{vp} = y\left(\frac{z_{prp} - z_{vp}}{z_{prp} - z}\right)$$

# Projection Calculations

Because the $x$ and $y$ coordinates of a projected point are expressed in terms of $z$ we need to do a little work to generate a perspective transformation matrix

**First** we use a homogeneous representation to give $x_{vp}$ and $y_{vp}$ as:

where:

$$x_{vp} = \frac{x_h}{h} \qquad y_{vp} = \frac{y_h}{h}$$

$$h = z_{prp} - z$$

# Perspective Projection Transformation

From the previous equations for $x_{vp}$ and $y_{vp}$ we can see that:

$$x_h = x\left(z_{prp} - z_{vp}\right) + x_{prp}\left(z_{vp} - z\right)$$
$$y_h = y\left(z_{prp} - z_{vp}\right) + y_{prp}\left(z_{vp} - z\right)$$

# Perspective Projection Transformation

- Now we can set up a transformation matrix, that only contains perspective parameters, to convert a spatial position to homogeneous coordinates
- First we calculate the homogeneous coordinates using the perspective-transformation matrix:

$$P_h = M_{pers} \cdot P$$

- where $P_h$ is the homogeneous point $(x_h, y_h, z_h, h)$ and P is the coordinate position $(x, y, z, 1)$

# Perspective Projection Transformation

The following is the perspective projection matrix which arises:

$$M_{pers} = \begin{bmatrix} z_{prp} - z_{vp} & 0 & -x_{prp} & x_{prp} z_{prp} \\ 0 & z_{prp} - z_{vp} & -y_{prp} & y_{prp} z_{prp} \\ 0 & 0 & s_z & t_z \\ 0 & 0 & -1 & z_{prp} \end{bmatrix}$$

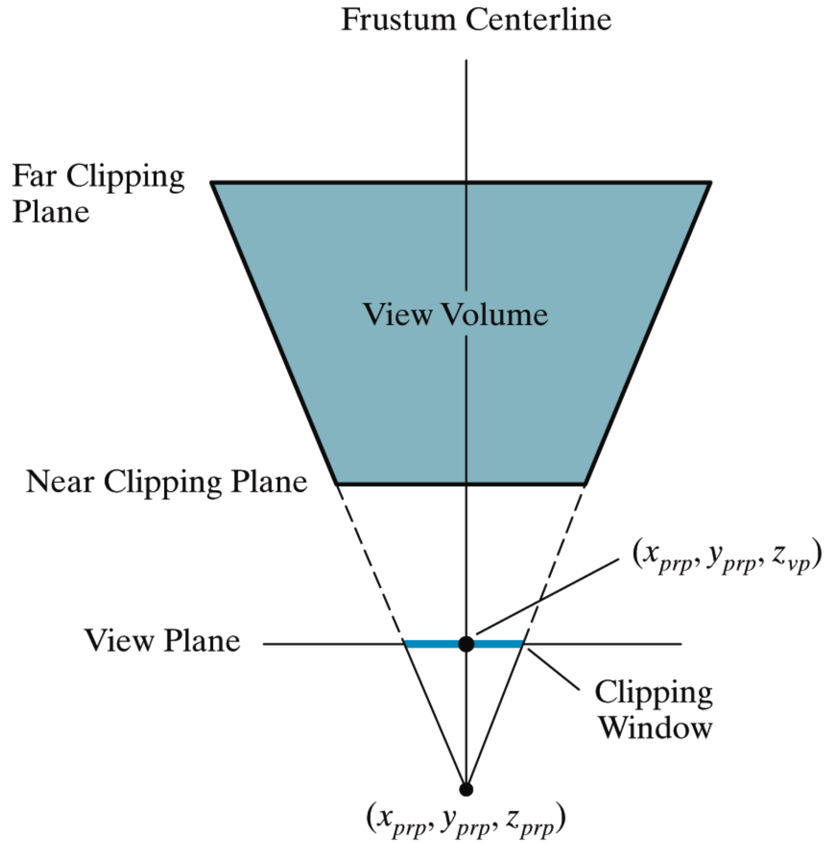
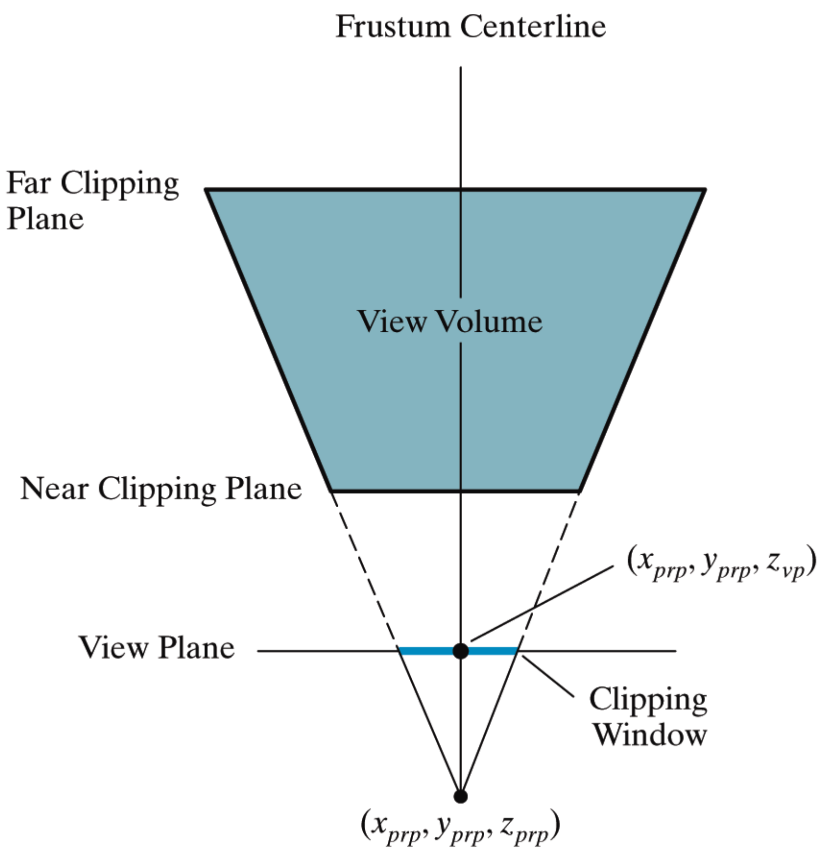



Figure 7-47

A symmetric perspective-projection frustum view volume, with the view plane between the projection reference point and the near clipping plane. This frustum is symmetric about its centerline when viewed from above, below, or either side.

# Setting Up A Perspective Projection

- A perspective projection can be set up by specifying the position and size of the view plane and the position of the projection reference point
- However, this can be kind of awkward



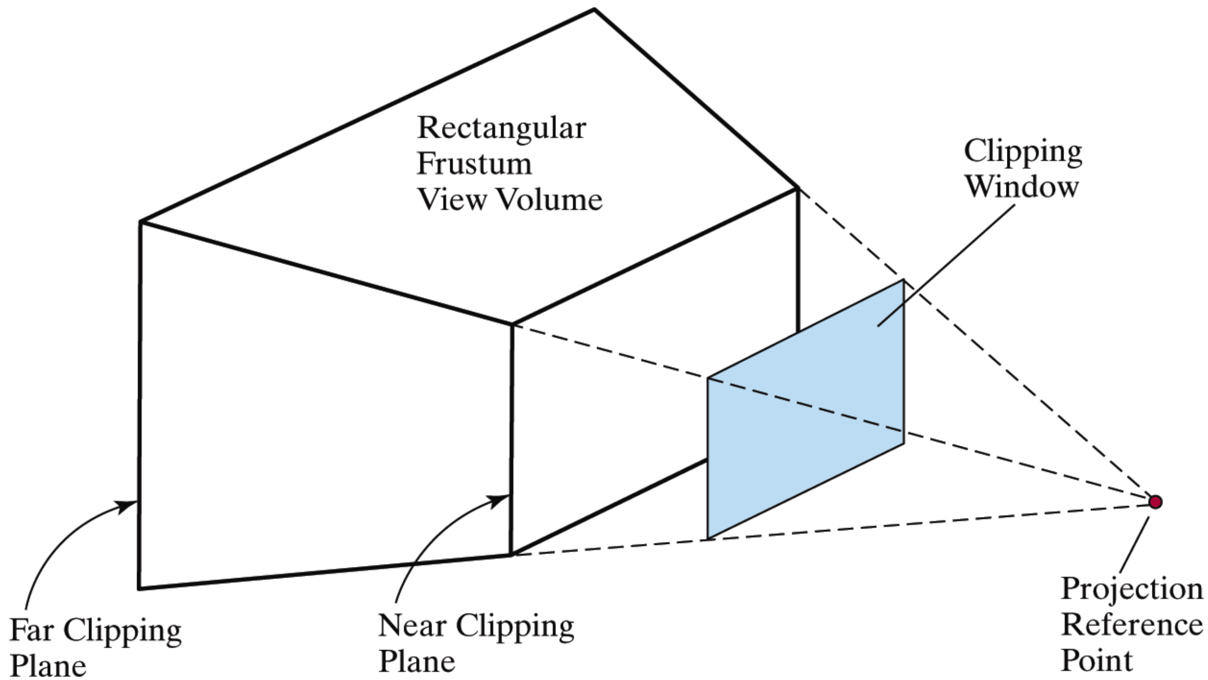


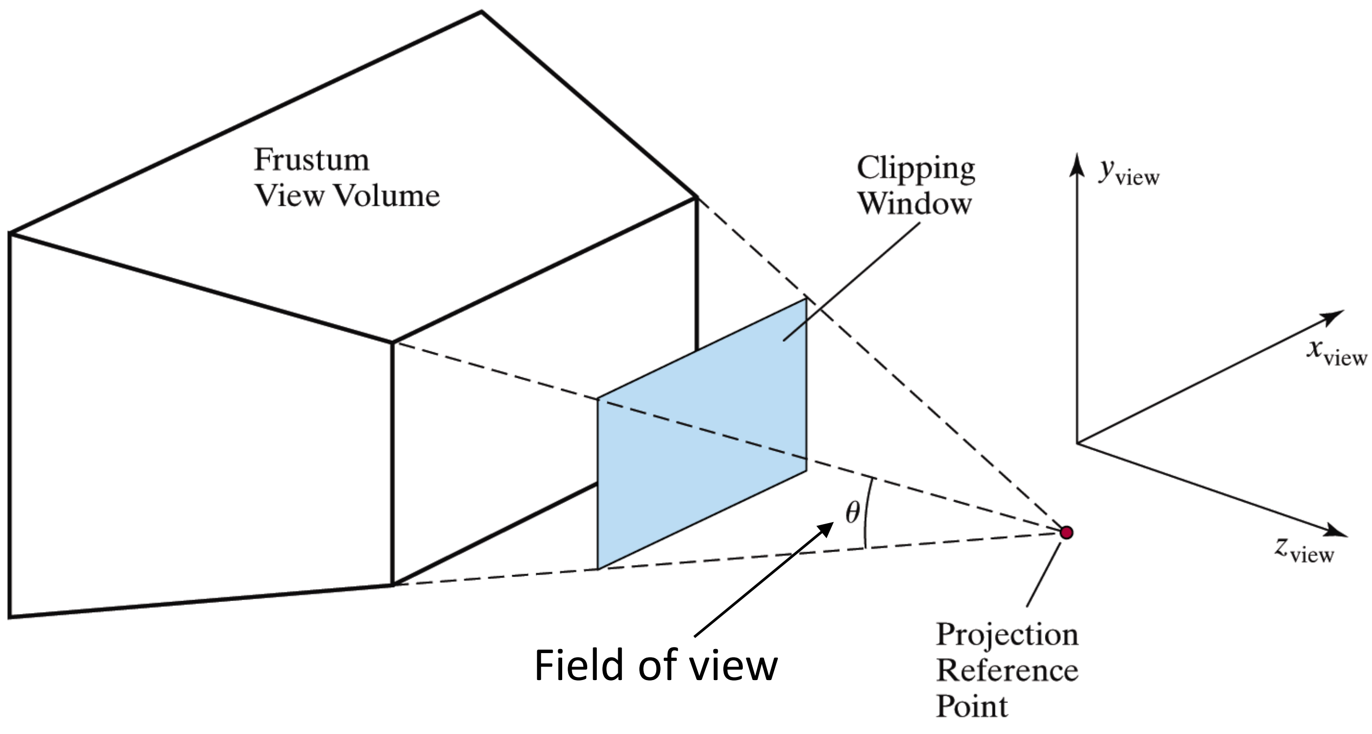

Figure 7-46

A perspective-projection frustum view volume with the view plane "in front" of the near clipping plane.

# Setting Up A Perspective Projection (cont…)

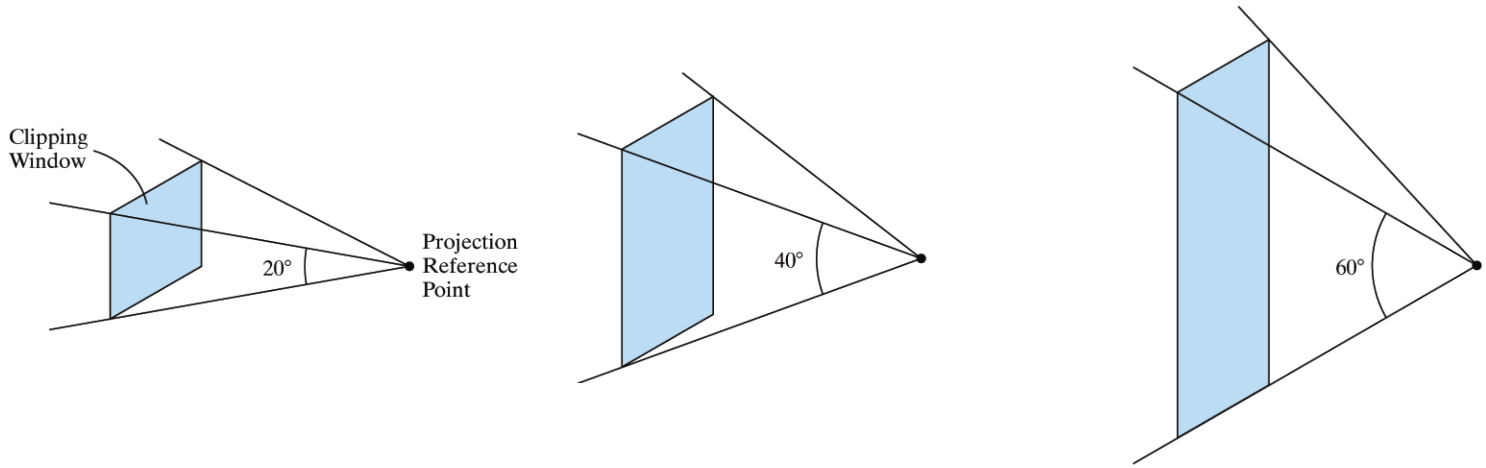
The *field of view* angle can be a more intuitive way to specify perspective projections

This is analogous to choosing a lense for a camera



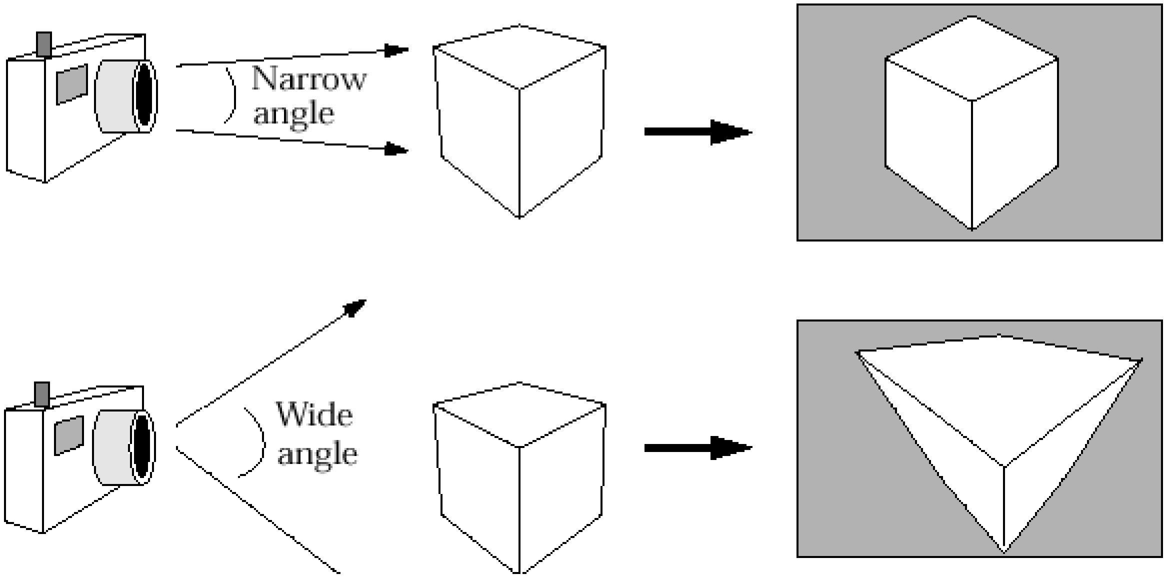
# Setting Up A Perspective Projection (cont…)

- Increasing the field of view angle increases the height of the view plane and so increases *foreshortening*



# Setting Up A Perspective Projection (cont…)

- The amount of foreshortening that is present can greatly affect the appearance of our scenes



# Setting Up A Perspective Projection (cont…)

We need one more thing to specify a perspective projections using the filed of view angle

The aspect ratio gives the ratio between the width sand height of the view plane