

CSE528 Computer Graphics: Theory, Algorithms, and Applications

Hong Qin

Department of Computer Science

State University of New York at Stony Brook (Stony
Brook University)

Stony Brook, New York 11794--4400

Tel: (631)632-8450; Fax: (631)632-8334

qin@cs.sunysb.edu

<http://www.cs.sunysb.edu/~qin>

Transformation and Viewing

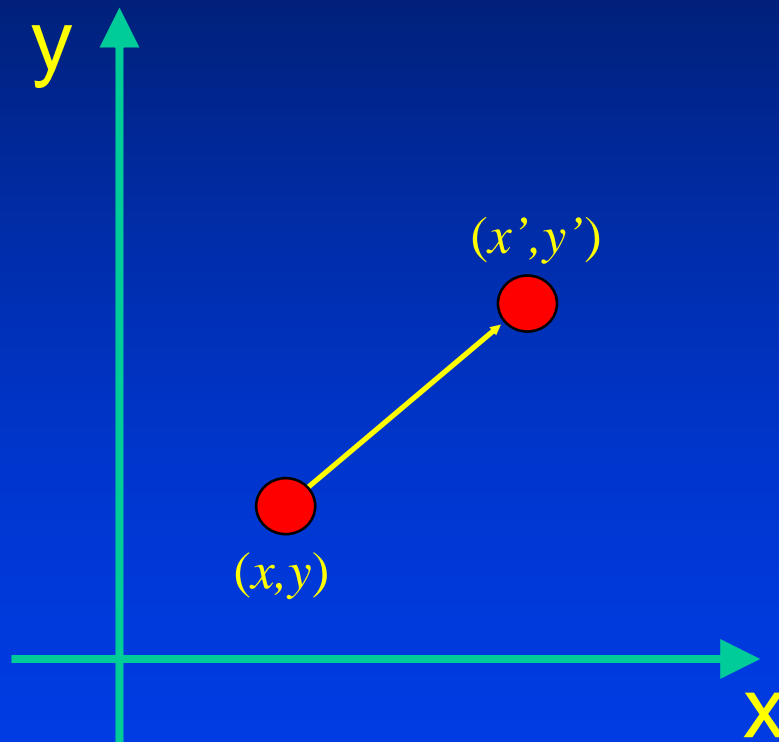
2D Geometric Transformations

- Translation
- Rotation
- Scaling
- Shear
- Homogenous Coordinates
- Matrix Representations
- Composite Transformations

Translation

- $x' = x + t_x$
- $y' = y + t_y$

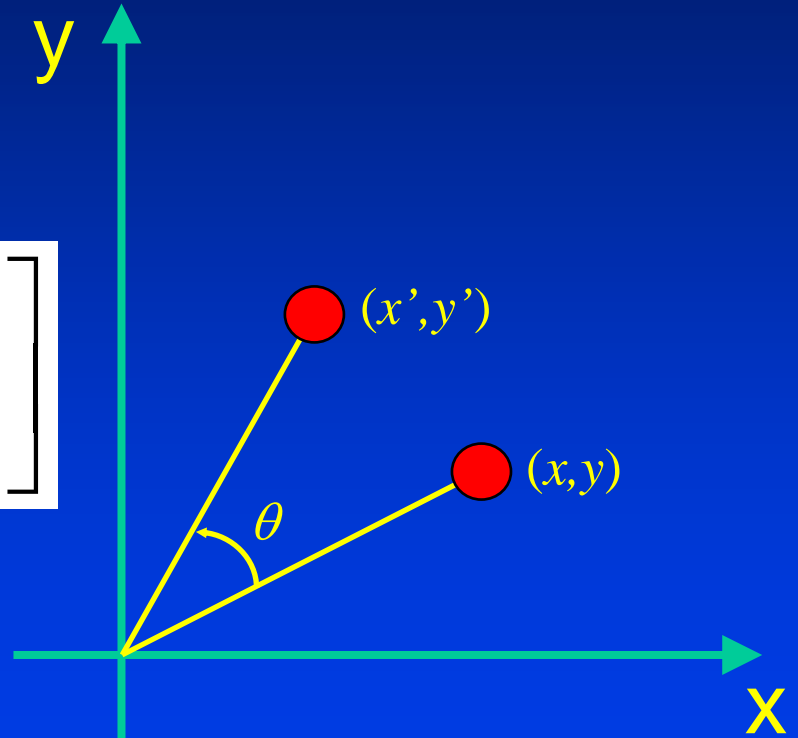
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$



Rotation

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

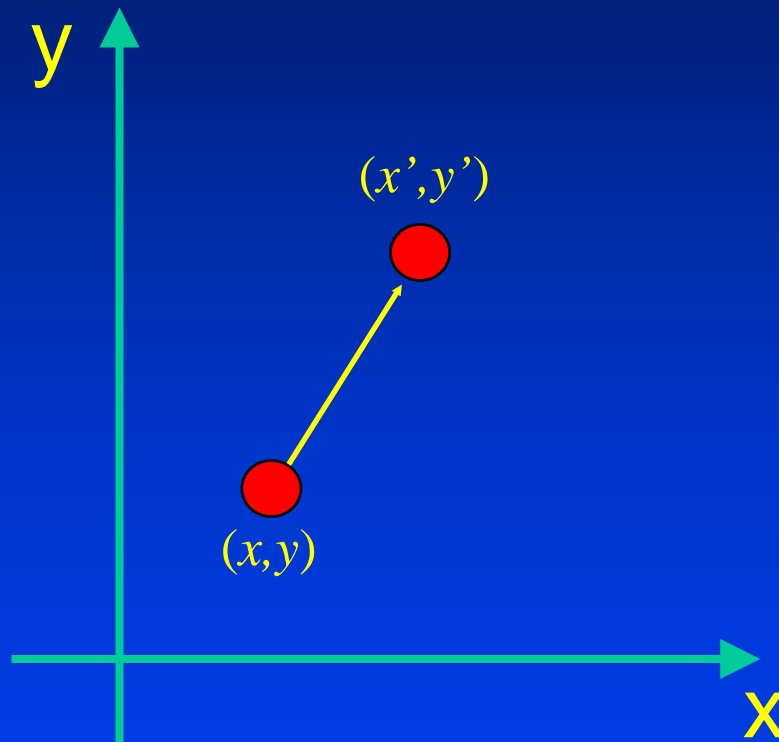
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$



Scaling

- $x' = S_x \cdot x$
- $y' = S_y \cdot y$

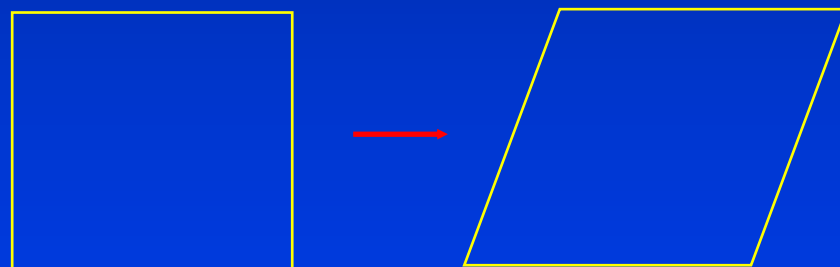
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$



Shear

- $x' = x + h_x \cdot y$
- $y' = y$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & h_x \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$



Homogenous Coordinates

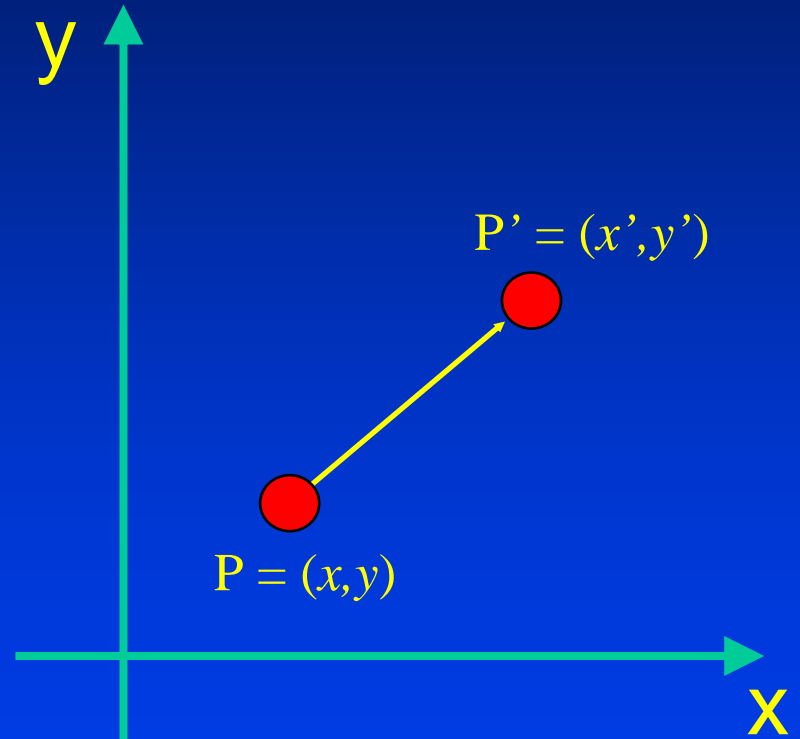
- Each position (x, y) is represented as $(x, y, 1)$.
- All transformations can be represented as matrix multiplication.
- Composite transformation becomes easier.

Translation in Homogenous Coordinates

- $x' = x + t_x$
- $y' = y + t_y$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

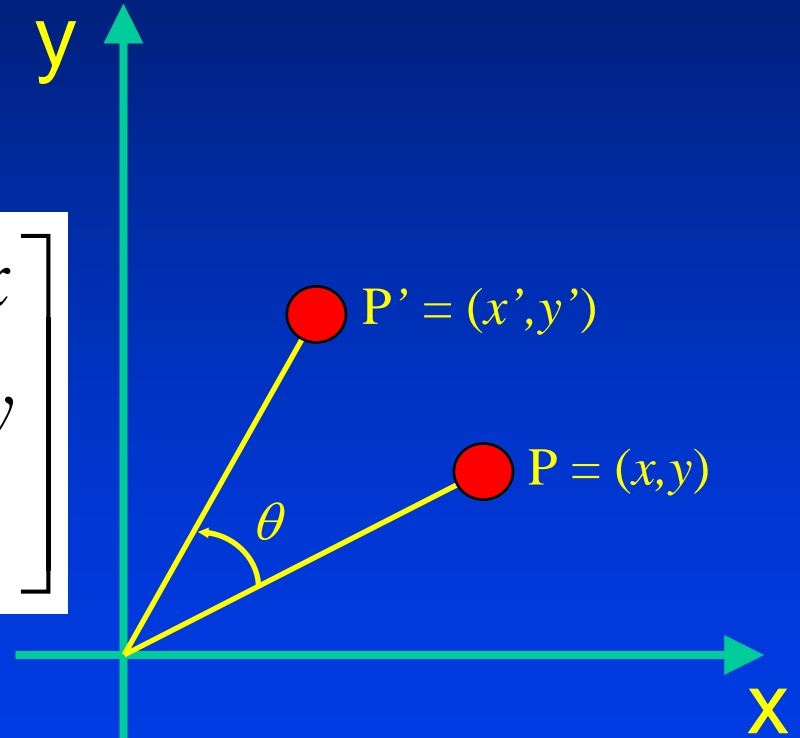
$$\mathbf{P}' = \mathbf{T}(t_x, t_y) \cdot \mathbf{P}$$



Rotation in Homogenous Coordinates

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

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$



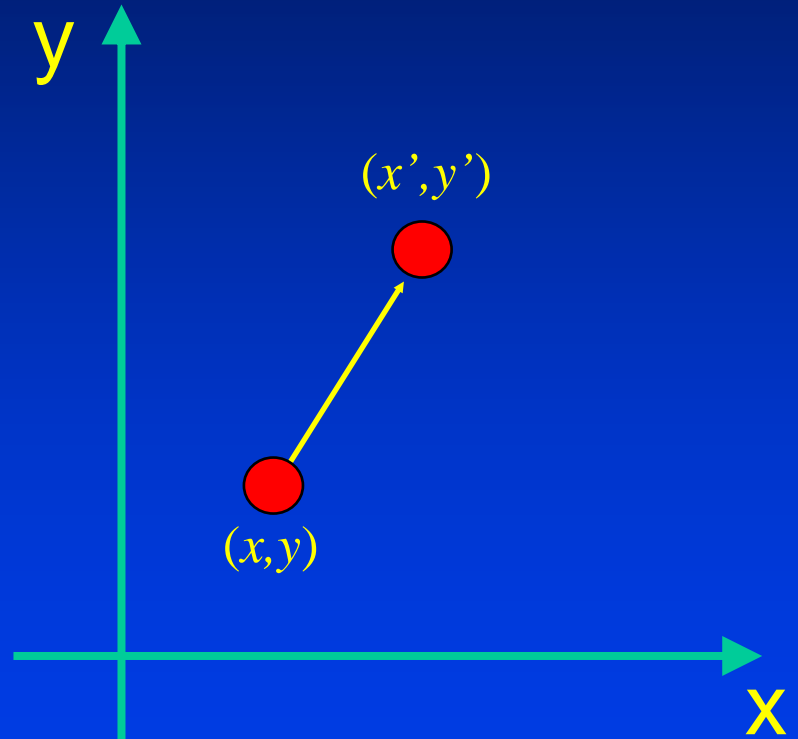
$$\mathbf{P}' = \mathbf{R}(\theta) \cdot \mathbf{P}$$

Scaling in Homogenous Coordinates

- $x' = s_x \cdot x$
- $y' = s_y \cdot y$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

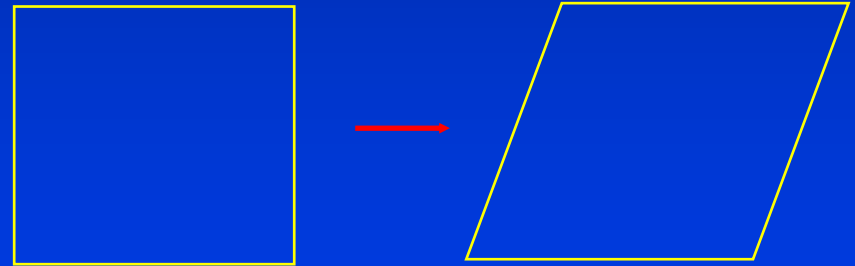
$$\mathbf{P}' = \mathbf{S}(s_x, s_y) \cdot \mathbf{P}$$



Shear in Homogenous Coordinates

- $x' = x + h_x \cdot y$
- $y' = y$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & h_x & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$



$$\mathbf{P}' = \mathbf{S} \mathbf{H}_x \cdot \mathbf{P}$$

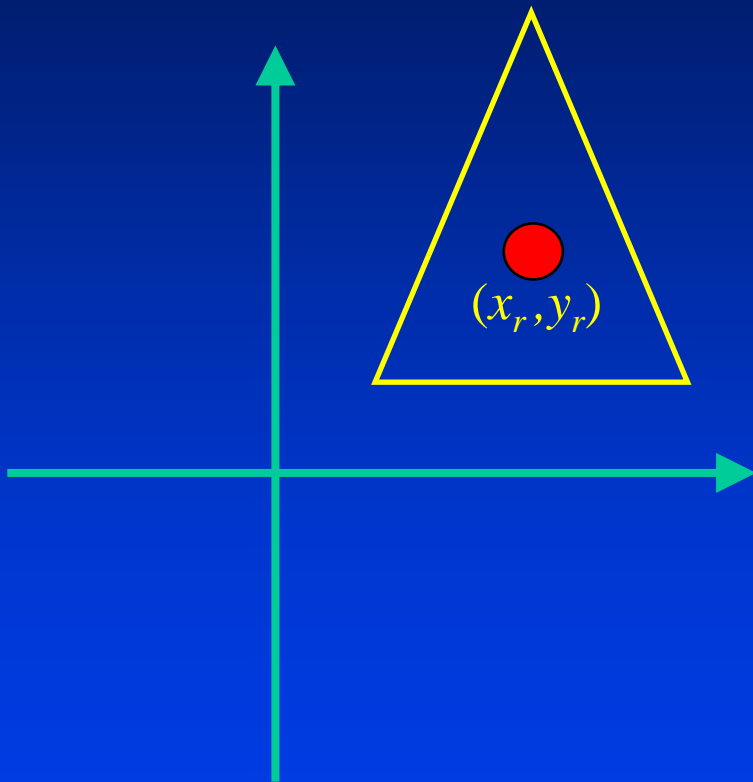
2D Geometric Transformations

- Translation
- Rotation
- Scaling
- Shear
- Homogenous Coordinates
- Composite Transformations

2D Geometric Transformations

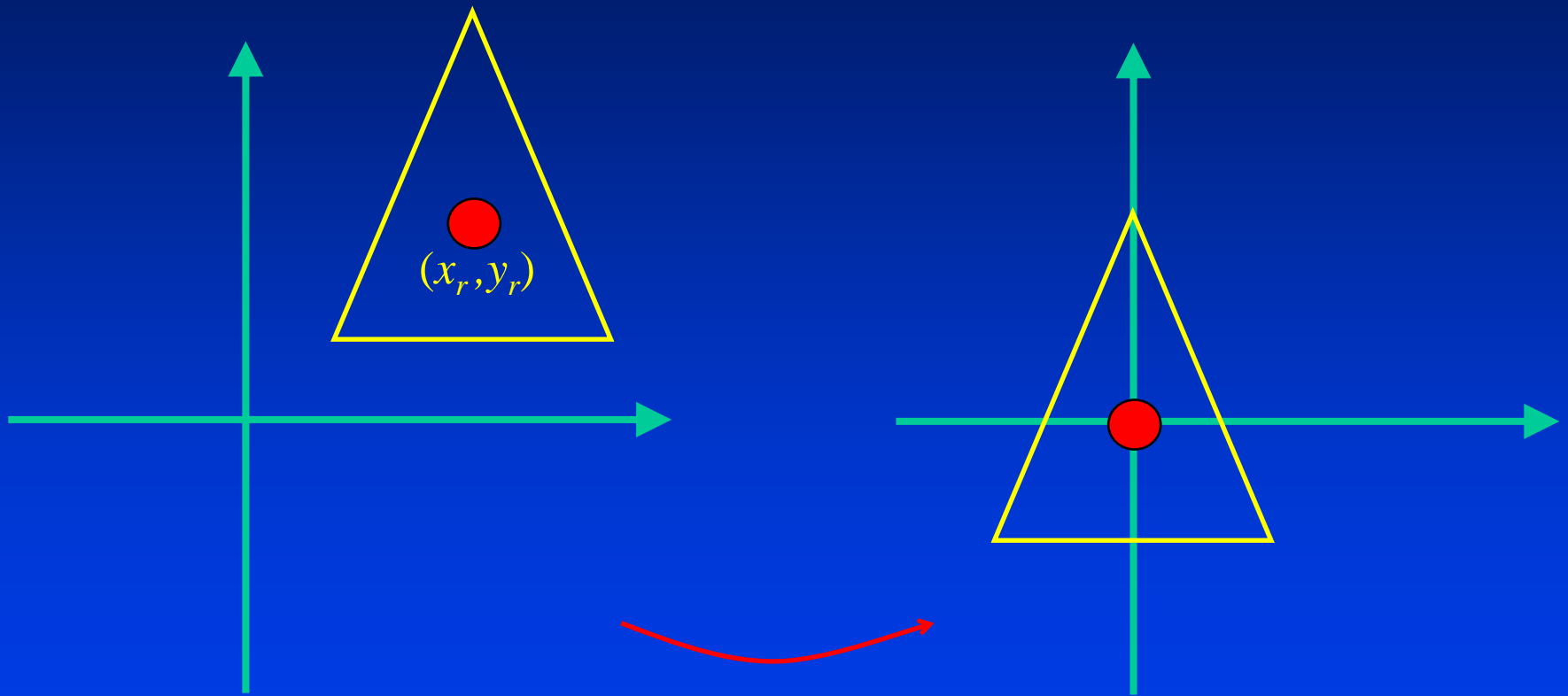
- Translation
- Rotation
- Scaling
- Shear
- Homogenous Coordinates
- Composite Transformations
 - Rotation about a fixed point

Rotation About a Fixed Point



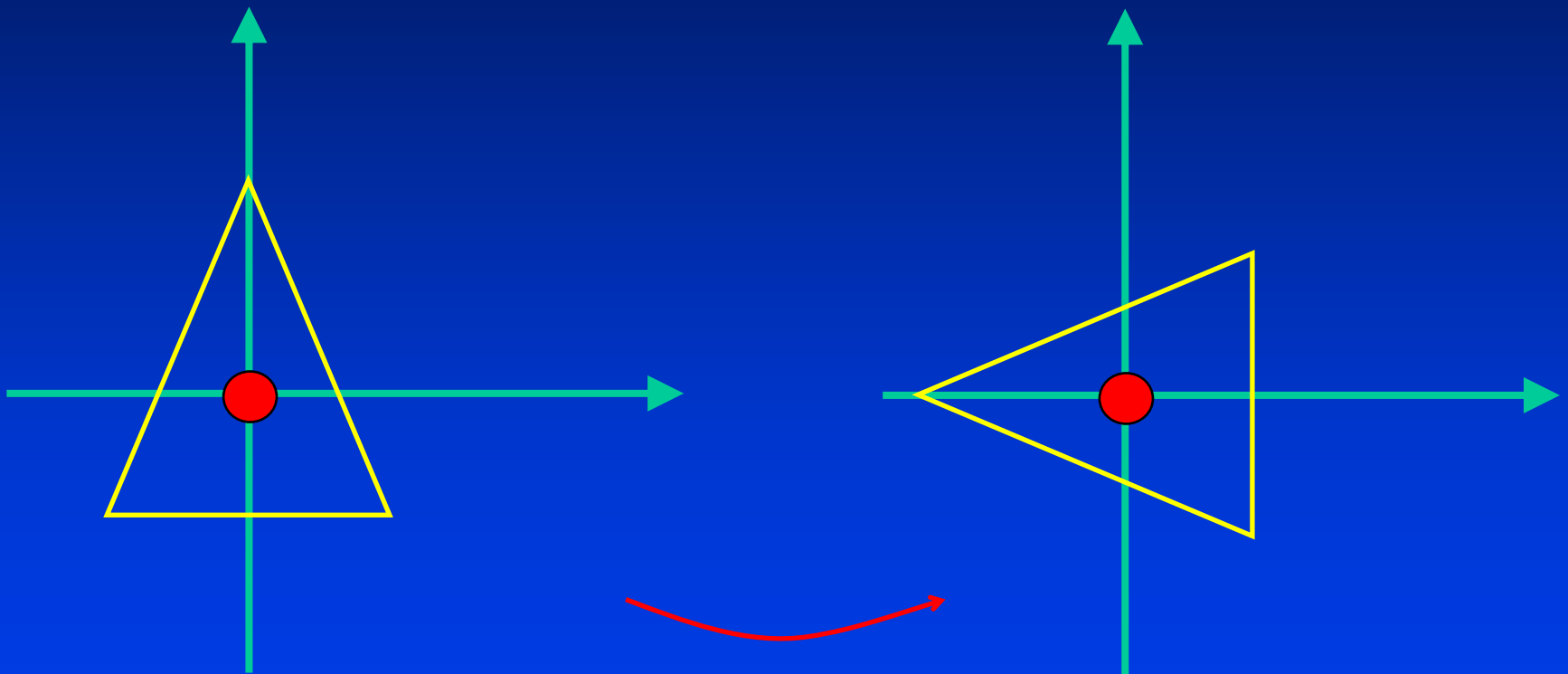
1. Translate the object to the origin.
2. Rotate around the origin.
3. Translate the object back.

Rotation About a Fixed Point



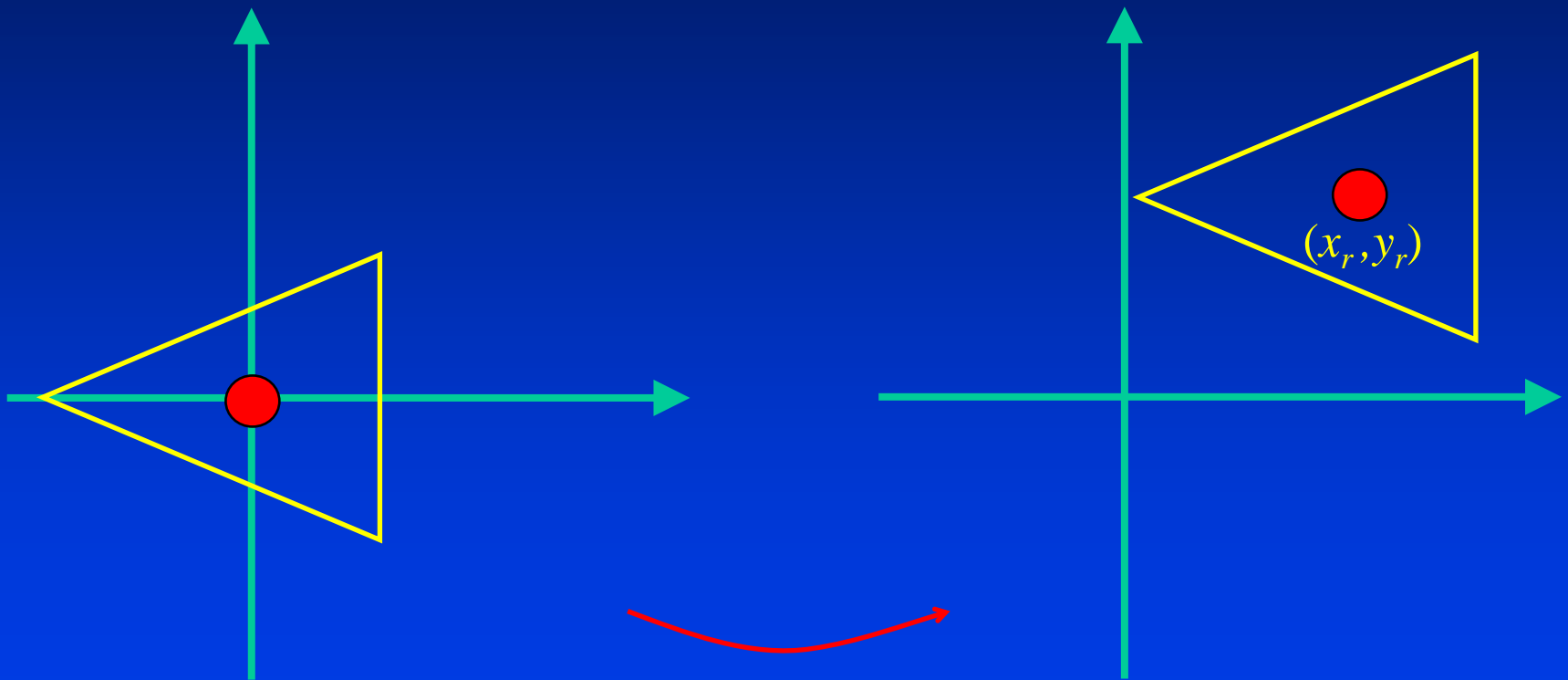
1. Translate the object to the origin

Rotation About a Fixed Point



2. Rotate about origin

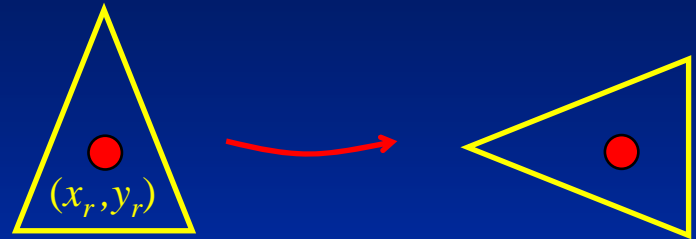
Rotation About a Fixed Point



3. Translate the object back

Rotation About a Fixed Point

1. **Translate the object to the origin.**
2. **Rotate around the origin.**
3. **Translate the object back.**



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

$$\mathbf{T}(x_r, y_r) \bullet \mathbf{R}(\theta) \bullet \mathbf{T}(-x_r, -y_r)$$

$$\mathbf{P}' = \mathbf{T}(x_r, y_r) \bullet \mathbf{R}(\theta) \bullet \mathbf{T}(-x_r, -y_r) \bullet \mathbf{P}$$

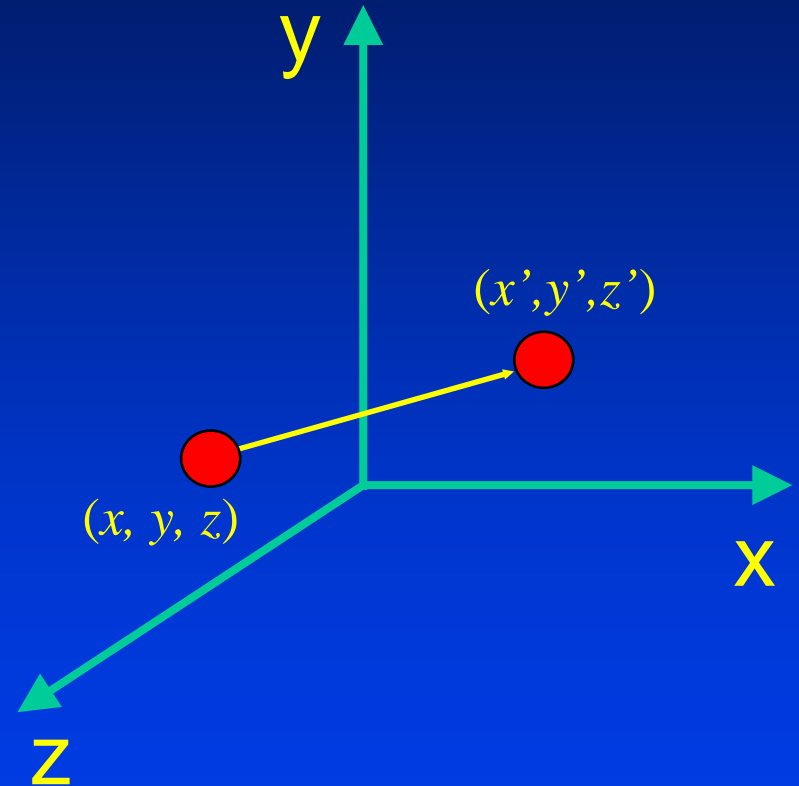
3D Geometric Transformations

- **Basic 3D Transformations**
 - Translation
 - Rotation
 - Scaling
 - Shear
- **Composite 3D Transformations**
- **Change of Coordinate systems**

Translation

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

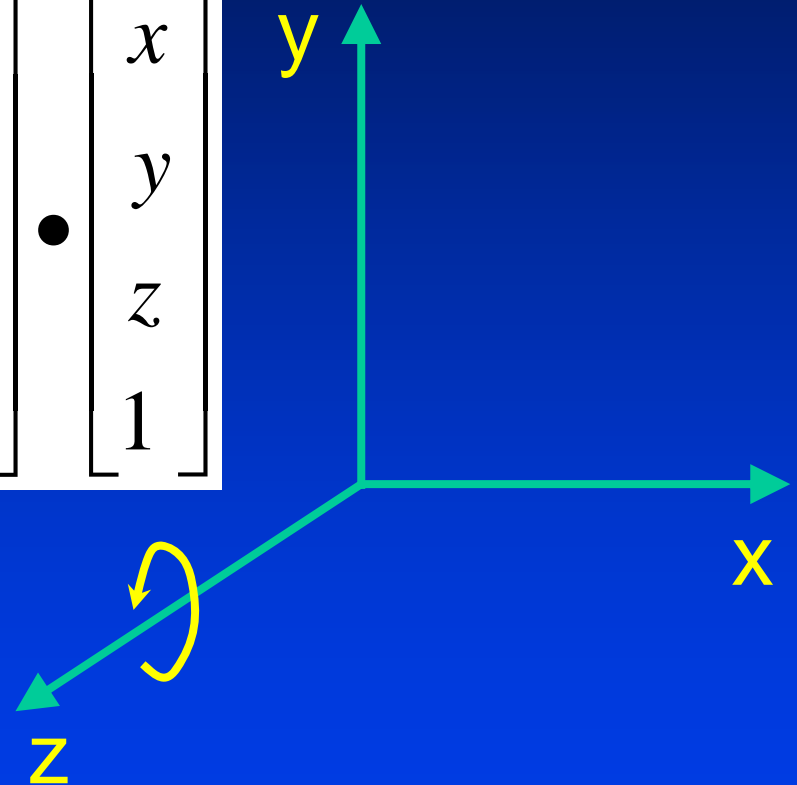
$$\mathbf{P}' = \mathbf{T} \cdot \mathbf{P}$$



Rotation about z-axis

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta & 0 & 0 \\ \sin \theta & \cos \theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

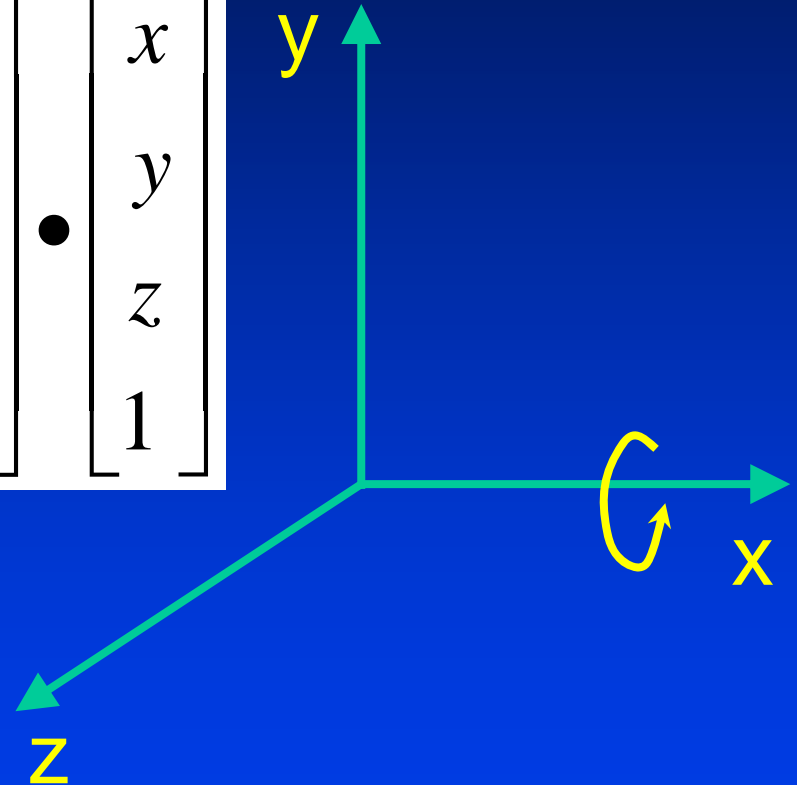
$$\mathbf{P}' = \mathbf{R}_z(\theta) \cdot \mathbf{P}$$



Rotation about x-axis

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \theta & -\sin \theta & 0 \\ 0 & \sin \theta & \cos \theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

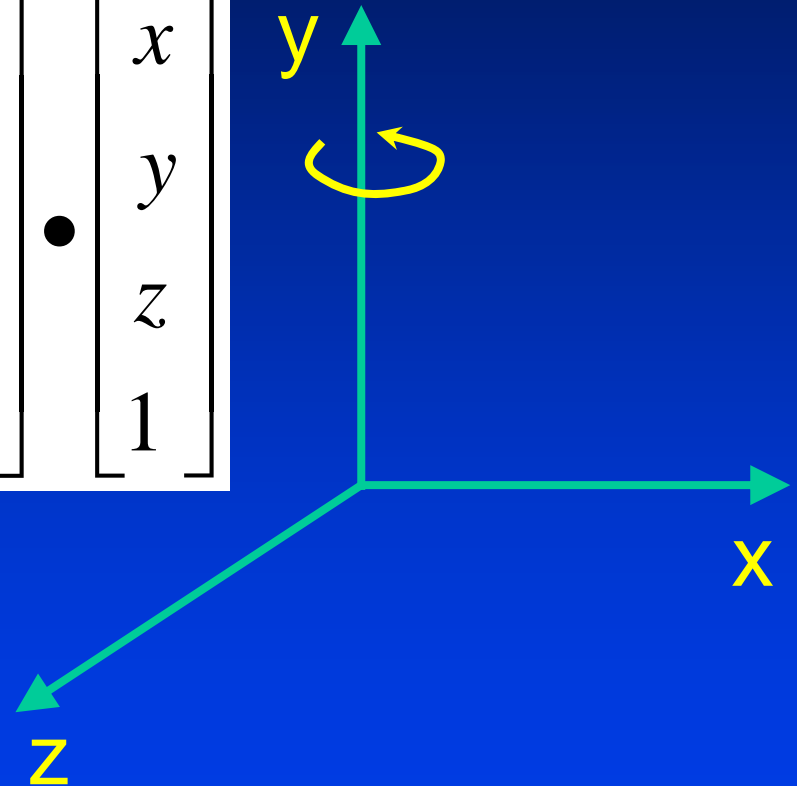
$$\mathbf{P}' = \mathbf{R}_x(\theta) \cdot \mathbf{P}$$



Rotation about y-axis

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos \theta & 0 & \sin \theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin \theta & 0 & \cos \theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$\mathbf{P}' = \mathbf{R}_y(\theta) \cdot \mathbf{P}$$



Rotation About a Fixed Point

1. Translate the object to the origin.
2. Rotate about the three axis, respectively.
3. Translate the object back.

$$\mathbf{P}' = \mathbf{T}(x_r, y_r, z_r) \cdot \mathbf{R1} * \mathbf{R2} * \mathbf{R3} \cdot \mathbf{T}(-x_r, -y_r, -z_r) \cdot \mathbf{P}$$

$$\mathbf{Ri} = \mathbf{R}_x(\theta_{x,i}) \cdot \mathbf{R}_y(\theta_{y,i}) \cdot \mathbf{R}_z(\theta_{z,i})$$

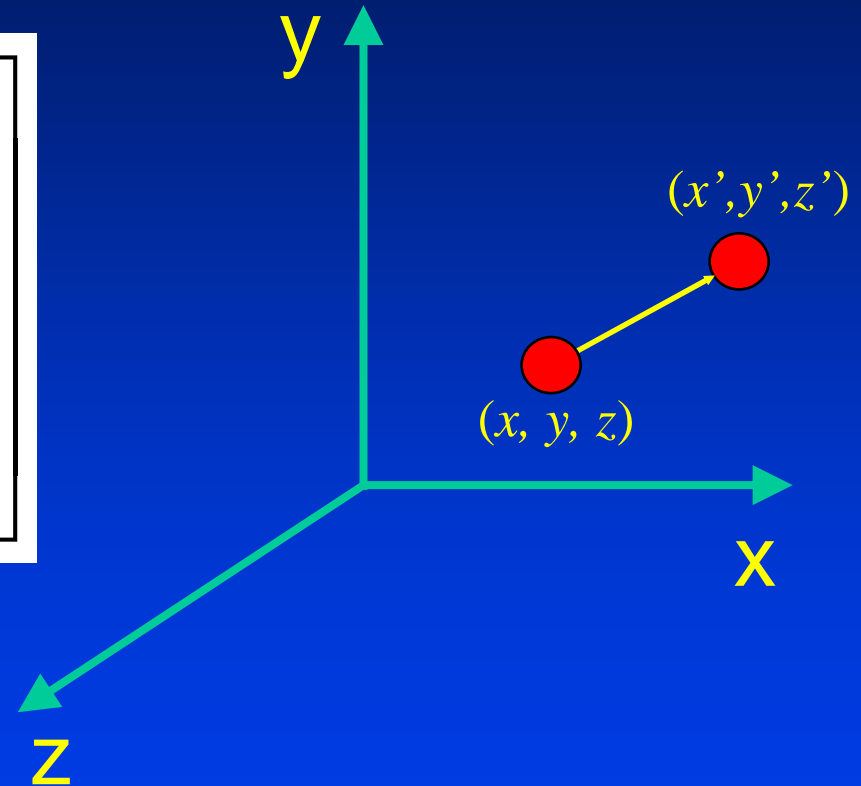
Rotation with Arbitrary Direction

1. We will have to translate an arbitrary vector so that its starting point starts from the origin
2. We will have to rotate w.r.t. x-axis so that this vector stays on x-z plane
3. We will then rotate w.r.t. y-axis so that this vector aligns with z-axis
4. We will then rotate w.r.t. z-axis
5. Reverse (3), (2), and (1)

Scaling

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

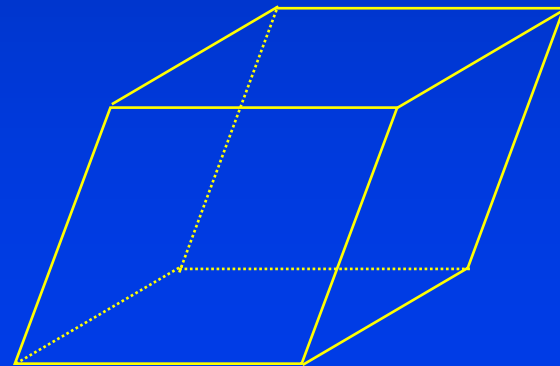
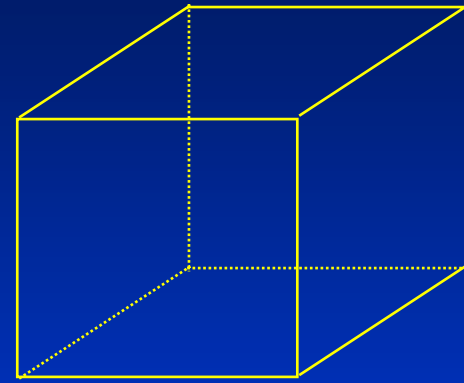
$$\mathbf{P}' = \mathbf{S} \cdot \mathbf{P}$$



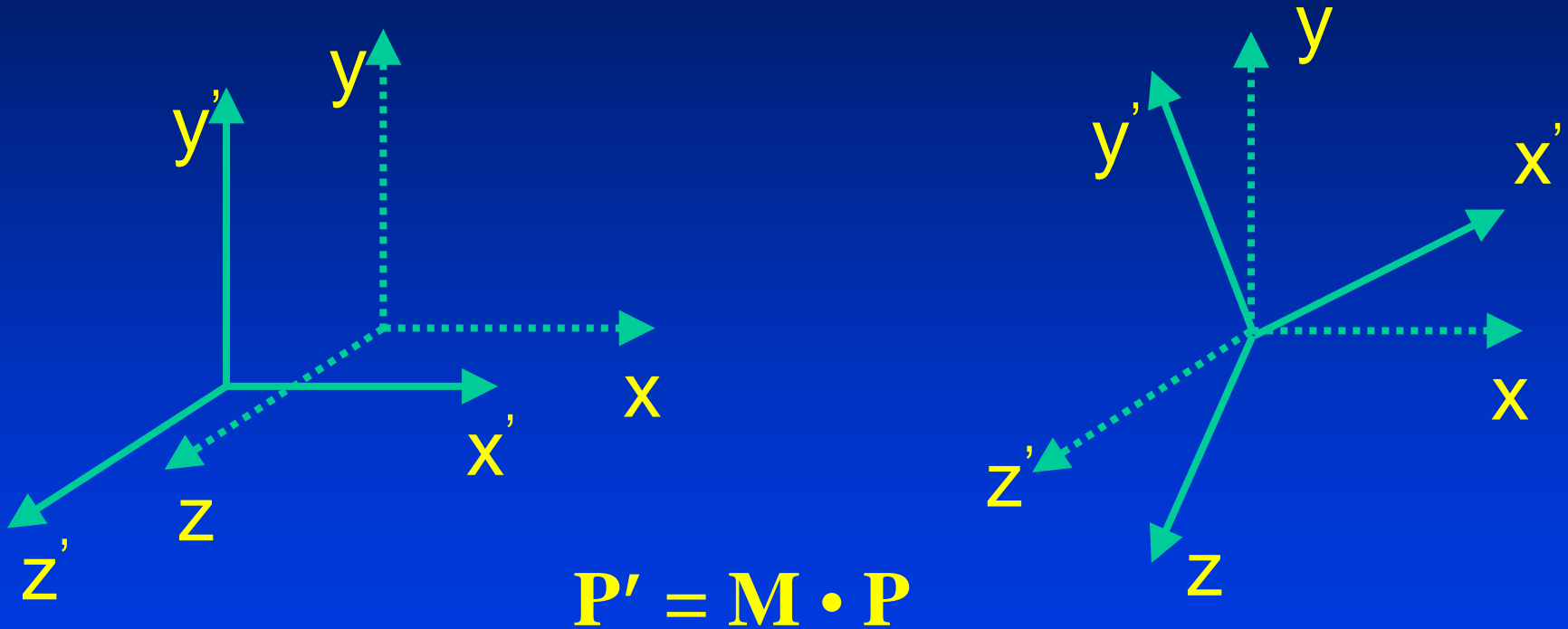
Shear

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & h_x & 0 \\ 0 & 1 & h_y & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$\mathbf{P}' = \mathbf{S} \mathbf{H}_{xy} \cdot \mathbf{P}$$



Change in Coordinate Systems



M can be a combination of translation, rotation and scaling.

Taking a Picture with a Camera

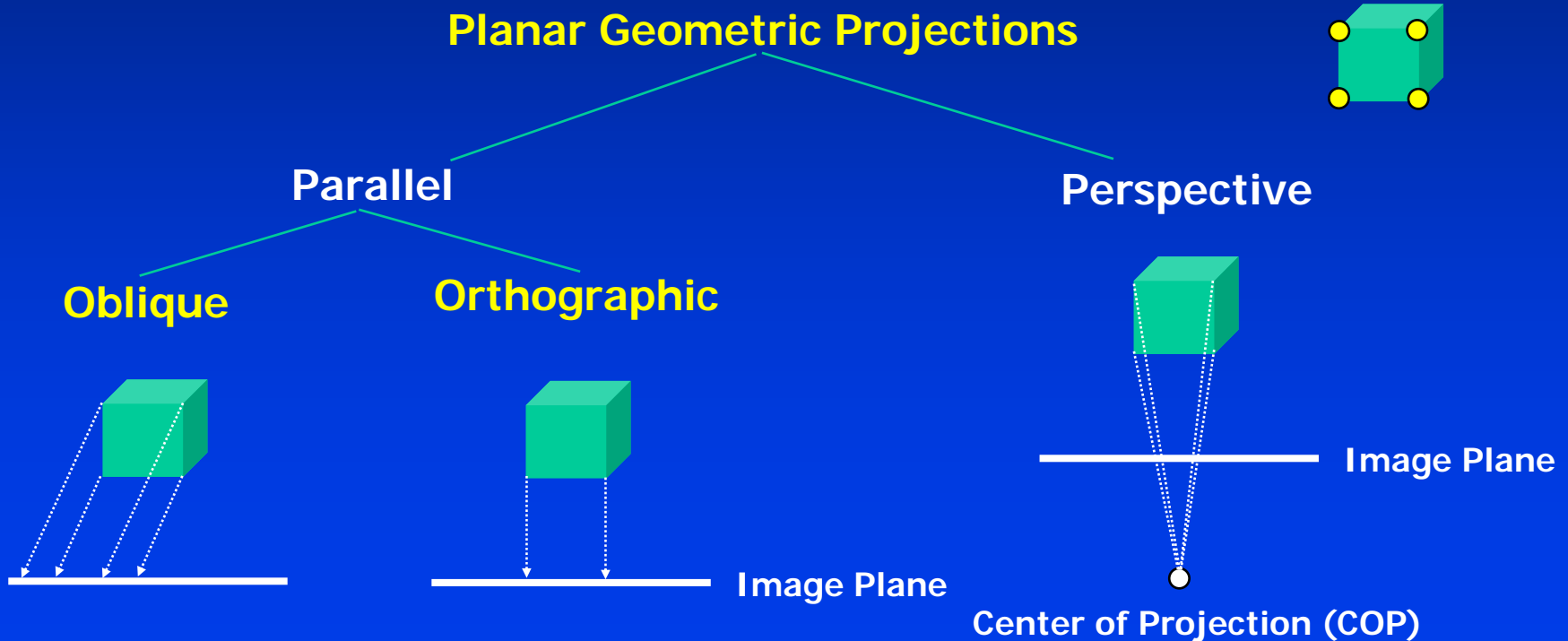
- Geometric Coordinate Systems: Local, World, Viewing
- Graphics Rendering Pipeline
- ModelView
 - Matrix operations on models
- World coordinates to Viewing coordinates
 - Matrix operations (models or cameras)
- Projection with a camera

Viewing in 3D

- Planar Geometric Projections
- Parallel Orthographic Projections
- Perspective Projections
- Projections in OpenGL
- Clipping

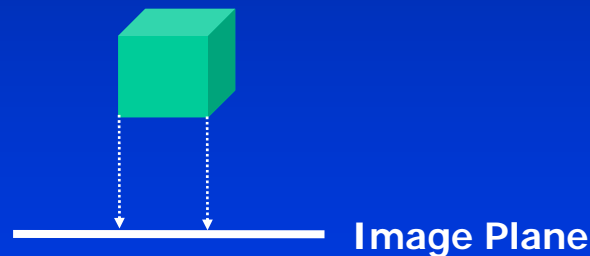
Planar Geometric Projections

- Maps points from camera coordinate system to the screen (image plane of the virtual camera).



Parallel Orthographic Projection

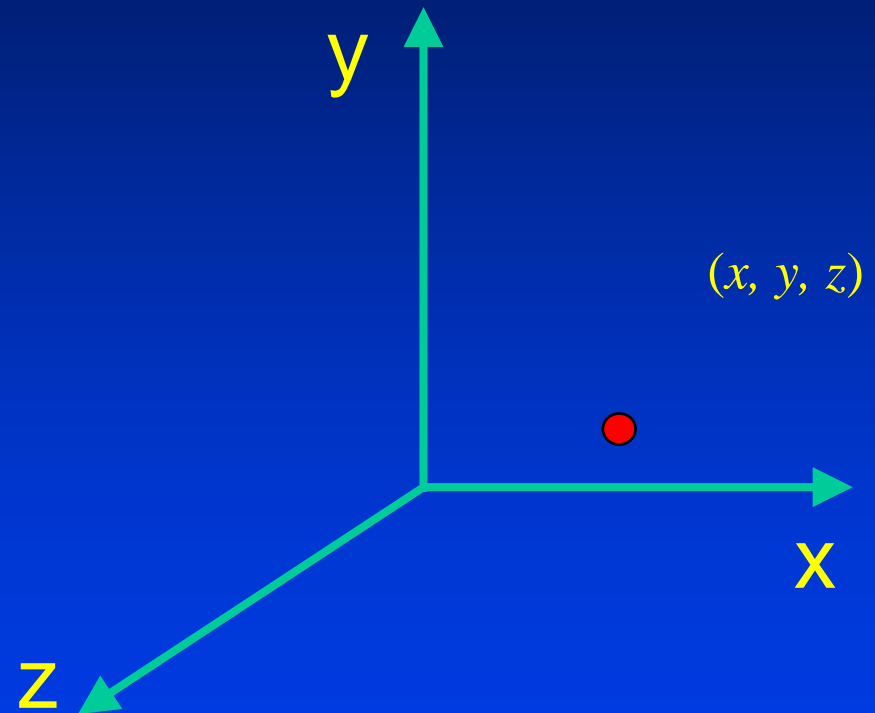
- Preserves X and Y coordinates.
- Preserves both distances and angles.



Parallel Orthographic Projection

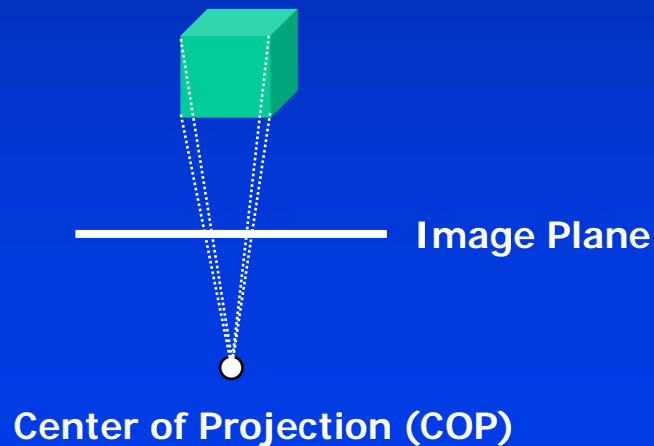
- $x_p = x$
- $y_p = y$
- $z_p = 0$

$$\begin{bmatrix} x_p \\ y_p \\ z_p \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

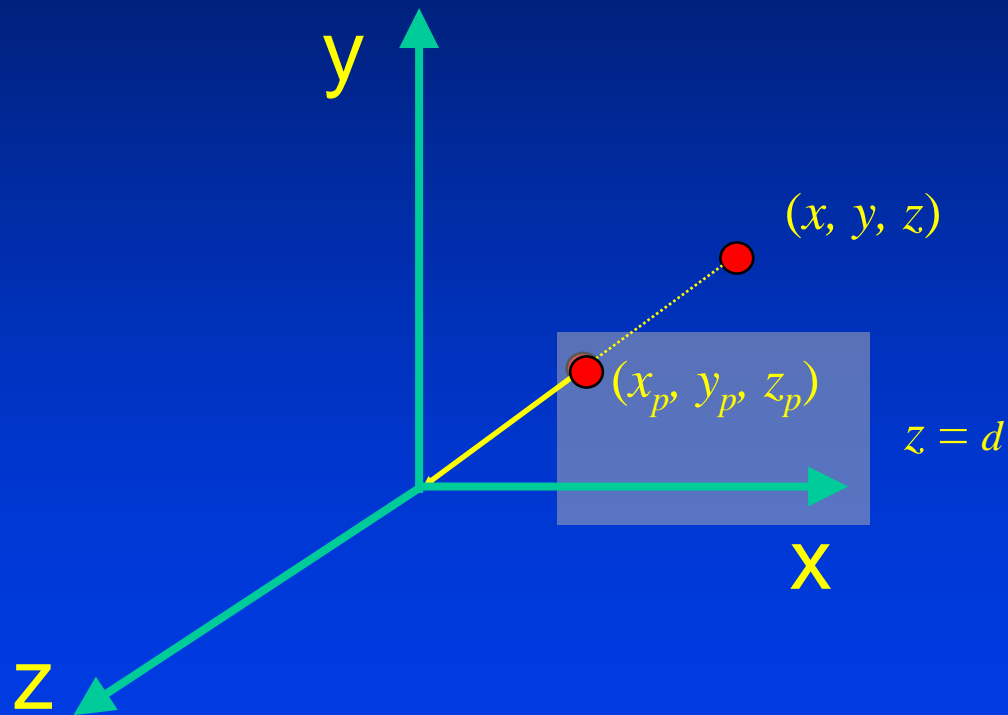


Perspective Projection

- Only preserves parallel lines that are parallel to the image plane.
- Line segments are shorten by distance.

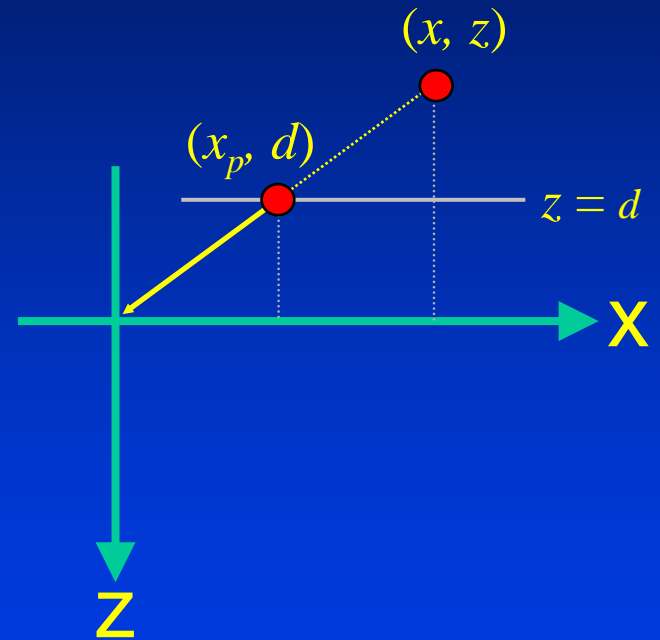


Perspective Projection



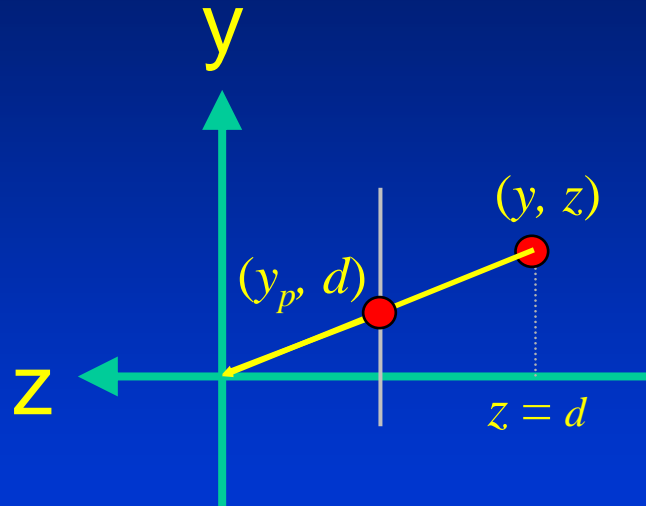
Perspective Projection

- $z_p = d$
- $x_p = (x \cdot d) / z$



Perspective Projection

- $z_p = d$
- $y_p = (y \cdot d) / z$



Perspective Projection

- $x_p = (x \cdot d) / z = x / (z/d)$
- $y_p = (y \cdot d) / z = y / (z/d)$
- $z_p = d = z / (z/d)$

$$\begin{bmatrix} x_h \\ y_h \\ z_h \\ h \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/d & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \quad \begin{bmatrix} x_p \\ y_p \\ z_p \\ 1 \end{bmatrix} = \begin{bmatrix} 1/h & 0 & 0 & 0 \\ 0 & 1/h & 0 & 0 \\ 0 & 0 & 1/h & 0 \\ 0 & 0 & 0 & 1/h \end{bmatrix} \begin{bmatrix} x_h \\ y_h \\ z_h \\ h \end{bmatrix}$$

Viewing in 3D

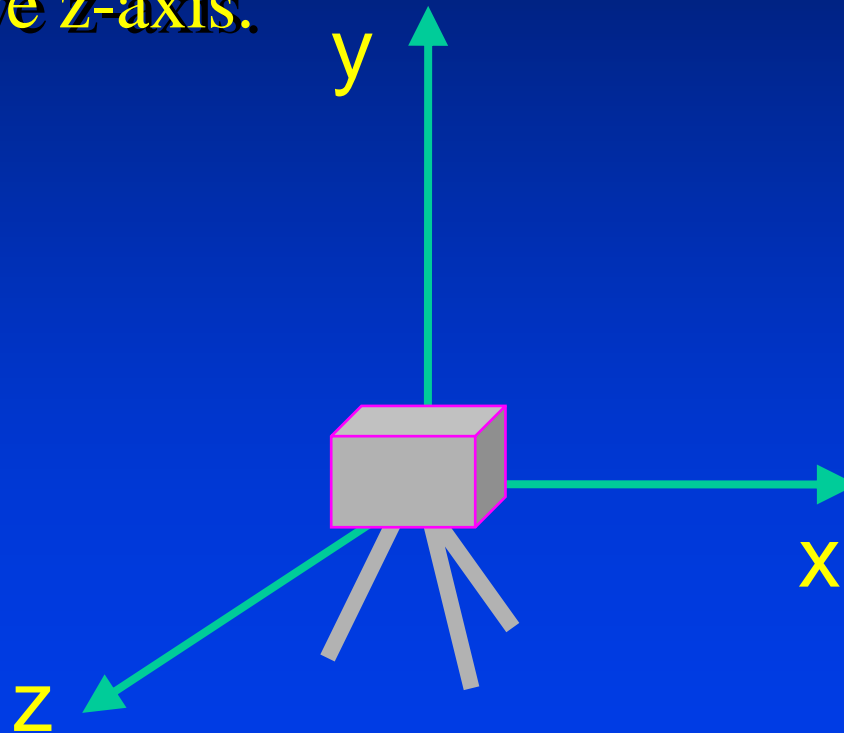
- Planar Geometric Projections
- Parallel Orthographic Projections
- Perspective Projections
- Projections in OpenGL

Viewing in 3D

- Planar Geometric Projections
- Parallel Orthographic Projections
- Perspective Projections
- Projections in OpenGL
 - Positioning of the Camera
 - Define the view volume

Positioning the Camera

- By default, the camera is placed at the origin pointing towards the negative z-axis.

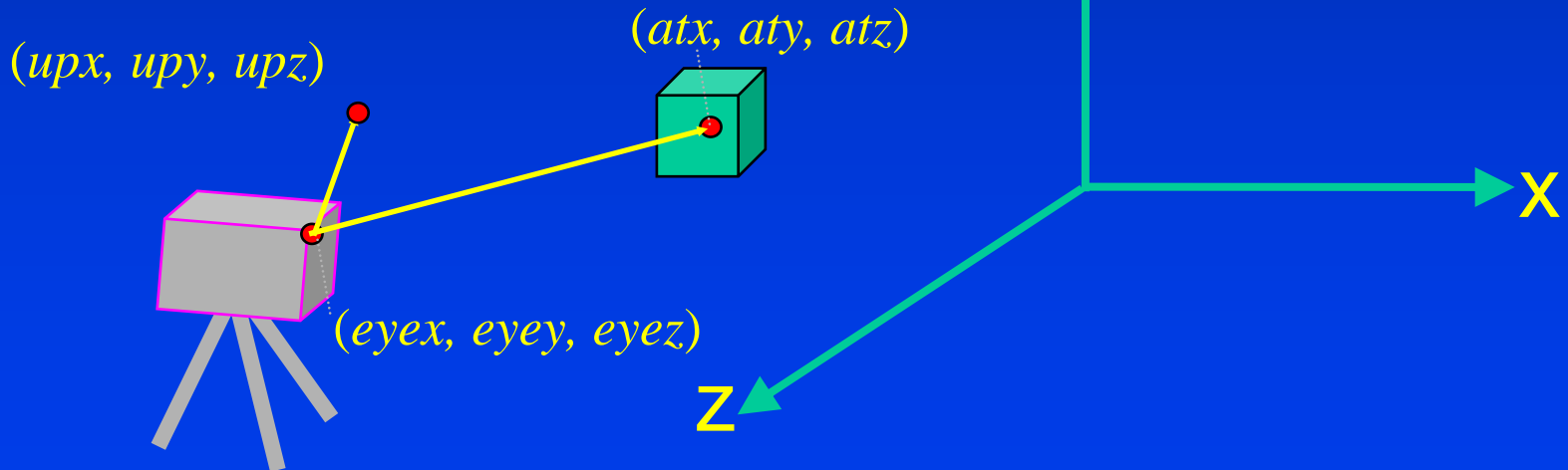


Positioning the Camera

- OpenGL Look-At Function

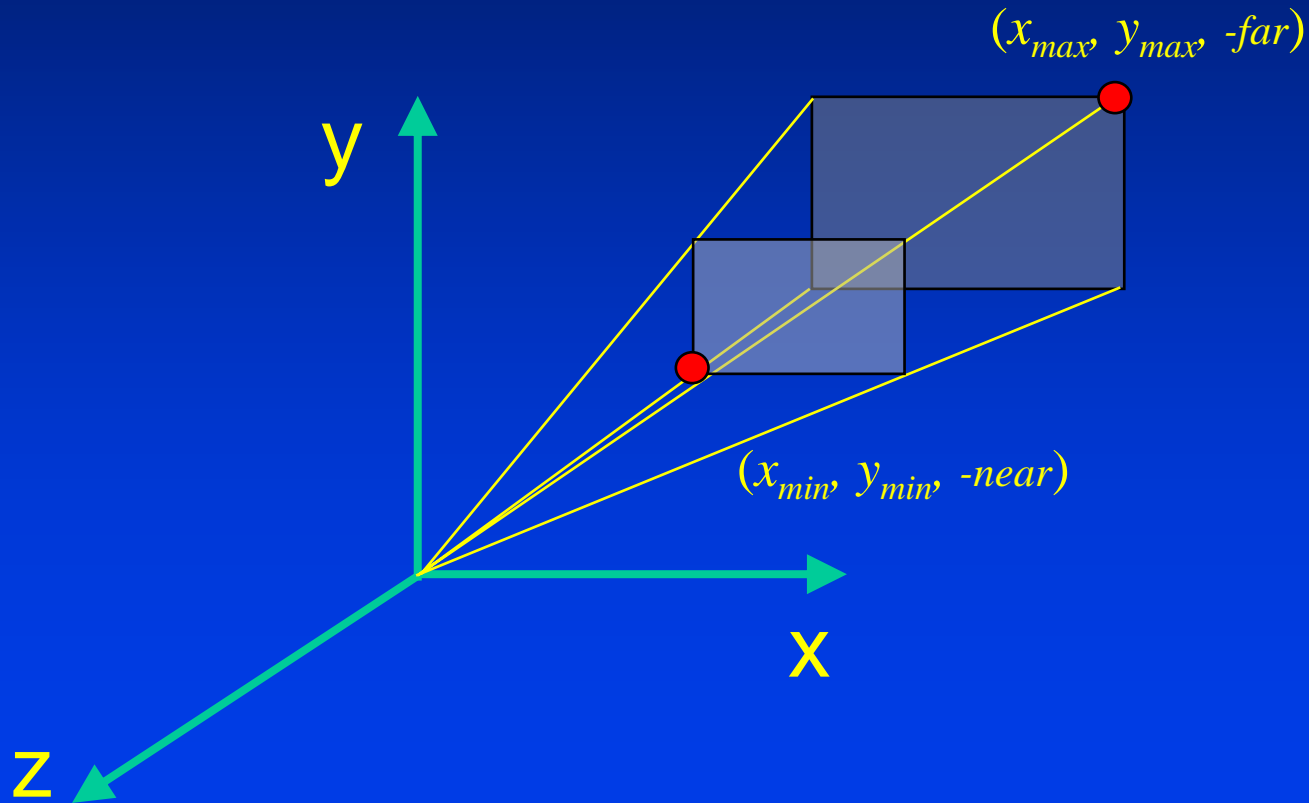
`gluLookAt(eyex, eyey, eyez, atx, aty, atz, upx, upy, upz)`

- View-reference point (VRP)
- View-plane normal (VPN)
- View-up vector (VUP)



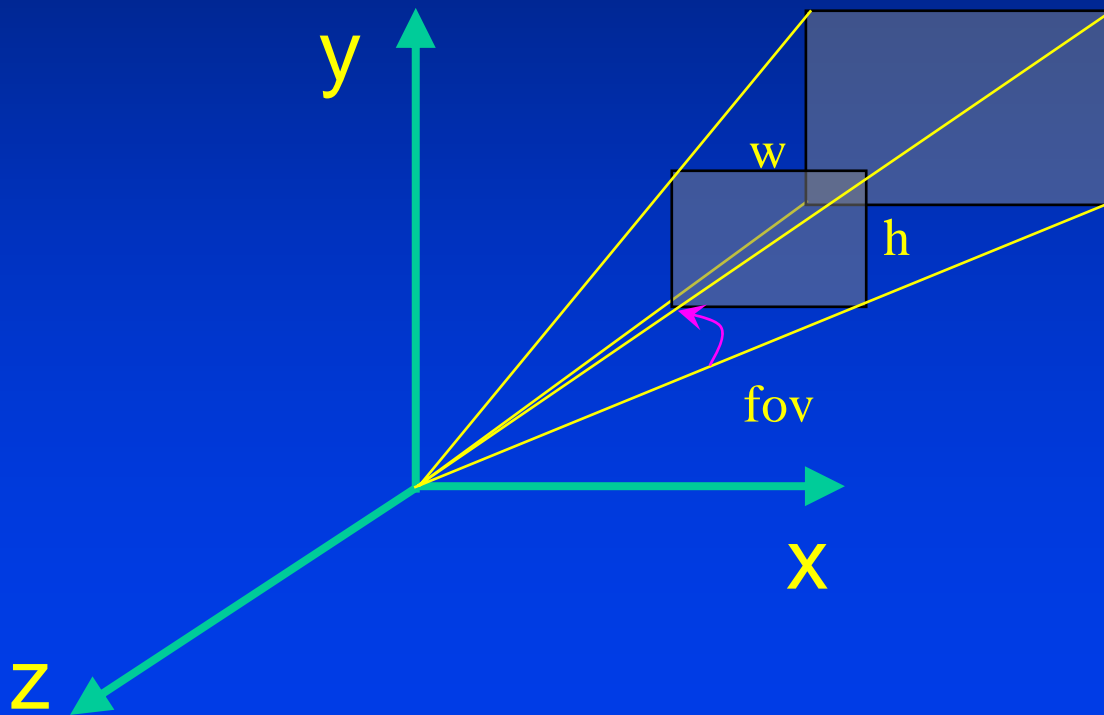
Defining the Perspective View Volume

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



Defining the Perspective View Volume

`gluPerspective(fovy, aspect, near, far)`



Defining the Parallel View Volume

`glOrtho(xmin, xmax, ymin, ymax, near, far)`

