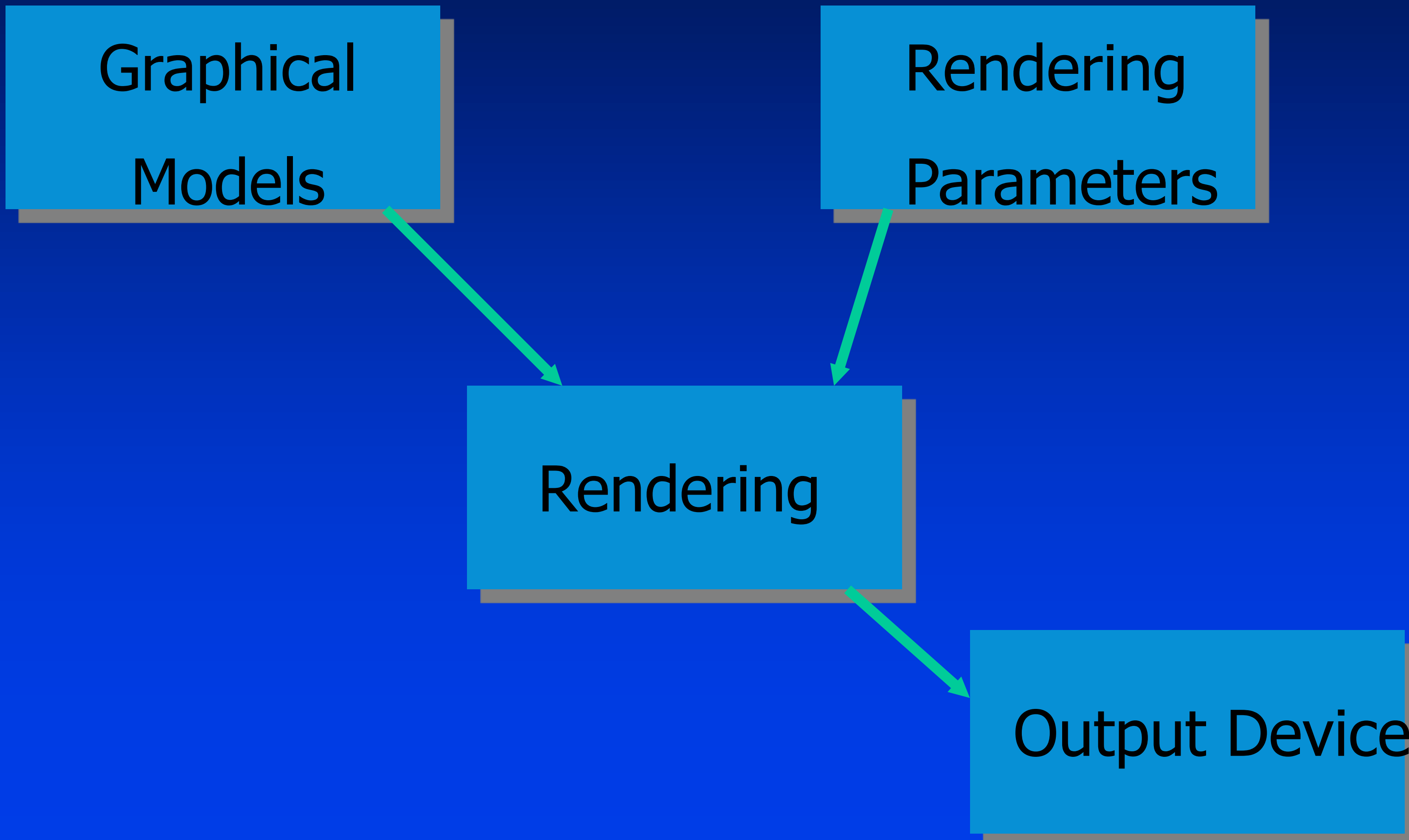
CSE528 Computer Graphics: Theory, Algorithms, and Applications

Hong Qin
Department of Computer Science
Stony Brook University (SUNY at Stony Brook)
Stony Brook, New York 11794-2424
Tel: (631)632-8450; Fax: (631)632-8334
qin@cs.stonybrook.edu
http://www.cs.stonybrook.edu/~qin

Computer Graphics Systems



Output Devices

- Vector Devices
  - Lasers (for example)

- Raster Devices
  - CRT, LCD, bitmaps, etc.

  - Most output devices are 2D
  - Can you name any 3D output devices?

Graphical Models

- 2D and 3D objects
  - Triangles, quadrilaterals, polygons
  - Spheres, cones, boxes
- Surface characteristics
  - Color, reaction to light
  - Texture, material properties
- Composite objects
  - Other objects and their relationships to each other
- Lighting, fog, etc.
- Much, much, more…

Rendering

- Conversion of 3D model to 2D image
  – Determine where the surfaces “project” to
  – Determine what every screen pixel might see
  – Determine the color of each surface

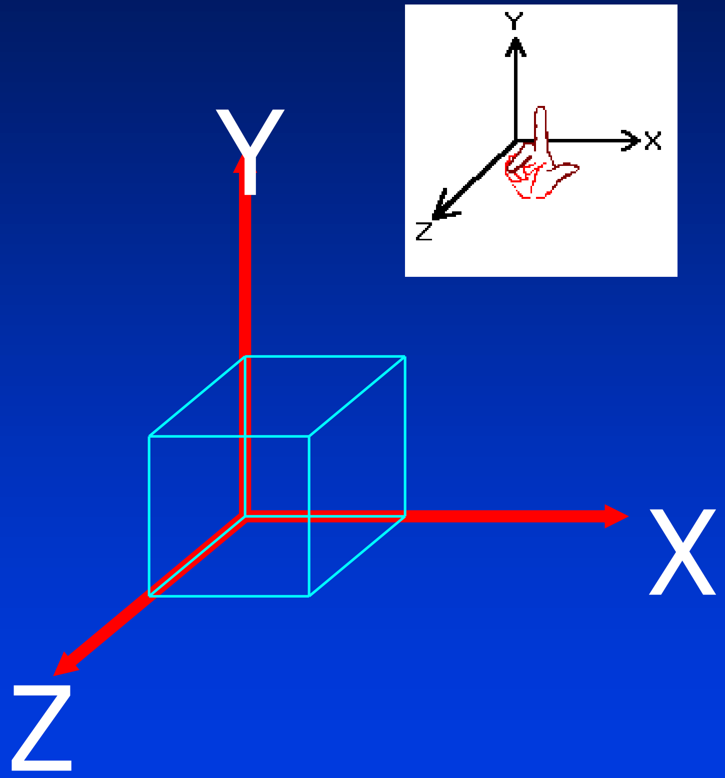
Rendering Parameters

- Camera parameters
  - Location
  - Orientation
  - Focal length

2D Graphics vs. 3D Graphics

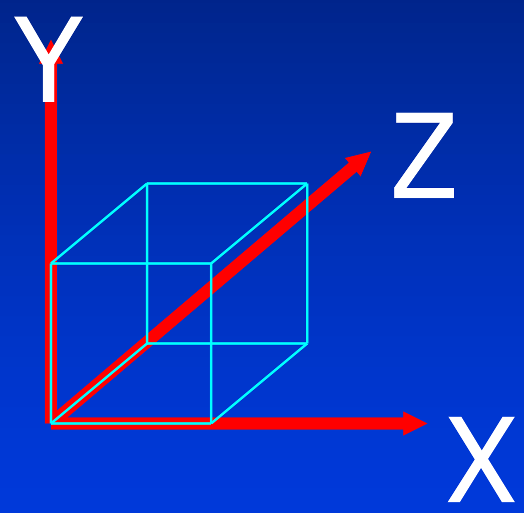
- 2D
  - X, Y - 2 dimensions only
  - We won't spend time on 2D graphics in this course
- 3D
  - X, Y, and Z
  - Space
- Rendering is typically the conversion of 3D to 2D

3D Coordinate Systems



Right-Hand Coordinate System

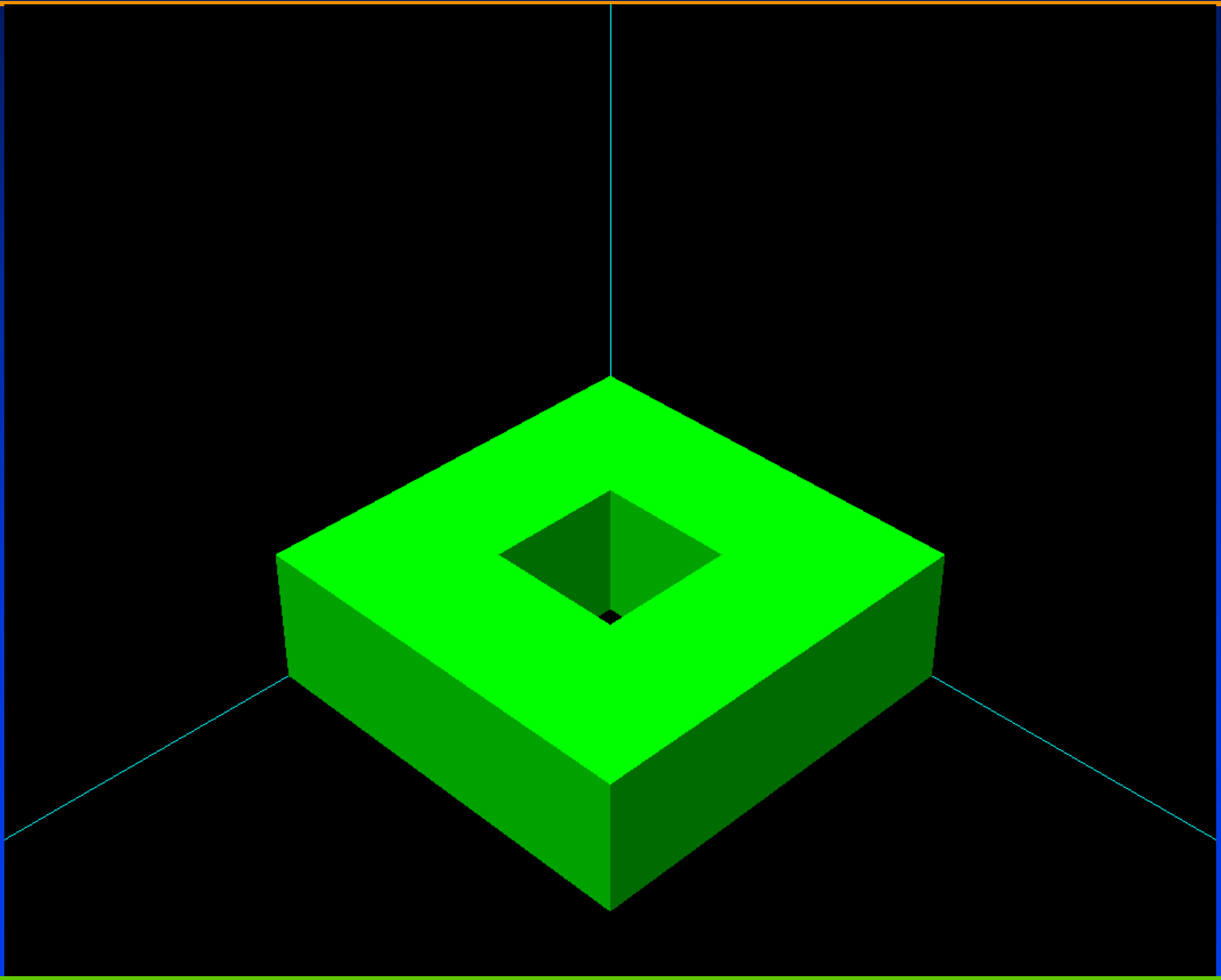
OpenGL uses this!



Left-Hand Coordinate System

Direct3D uses this!

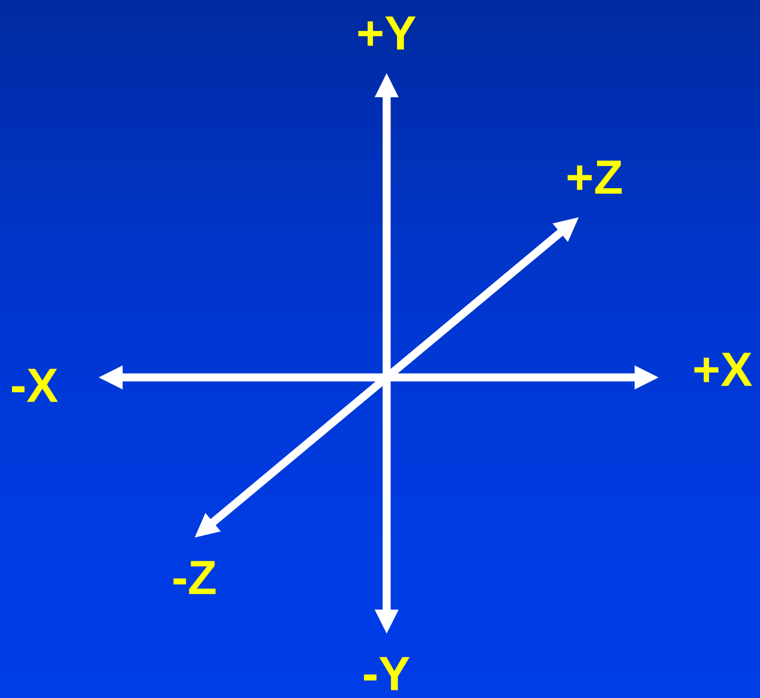
How to Model/Render This?

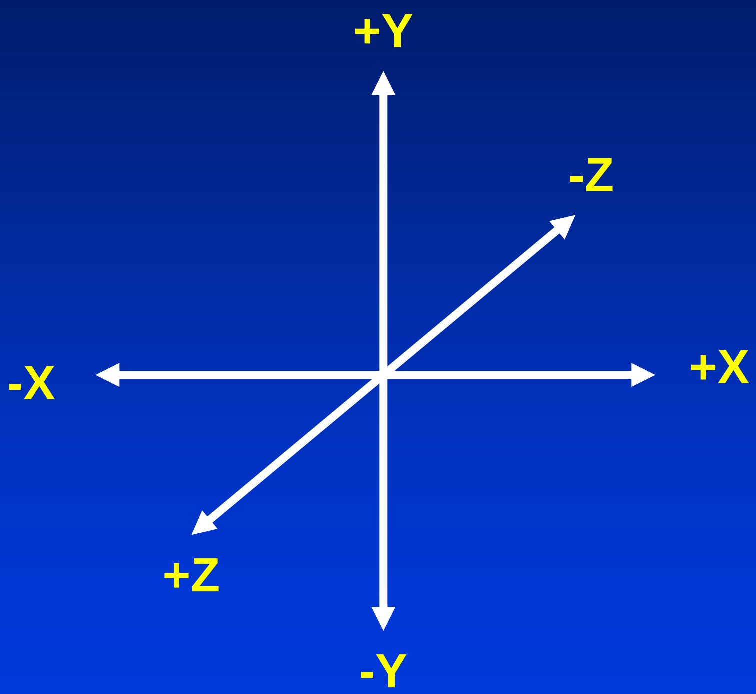


Transformation and Viewing

Cartesian Coordinate System

Left Handed





Right Handed

Euclidean Space

- Scalars
- Points: P = (x,y,z)
- Vectors: **V** = [x,y,z]
  - Magnitude or distance $||V|| = \sqrt{(x^2+y^2+z^2)}$
  - Direction
  - No position
- Position vector
  - Think of as magnitude and distance relative to a point, usually the origin of the coordinate system

Review of Common Vector Operations in 3D

- Addition of vectors
  - $\mathbf{V_1}+\mathbf{V_2} = [x_1,y_1,z_1] + [x_2,y_2,z_2] = [x_1+x_2, y_1+y_2, z_1+z_2]$
- Multiply a scalar times a vector
  - $s\mathbf{V} = s[x,y,z] = [sx,sy,sz]$
- Dot Product
  - $\mathbf{V_1}\bullet\mathbf{V_2} = [x_1,y_1,z_1]\bullet[x_2,y_2,z_2] = [x_1x_2+y_1y_2+z_1z_2]$
  - $\mathbf{V_1}\bullet\mathbf{V_2} = ||V_1||\,||v_2||\cos\beta$ where β is the angle between V_1 and V_2
- Cross Product of two vectors
  - $\mathbf{V_1}\times\mathbf{V_2} = [x_1,y_1,z_1]\times[x_2,y_2,z_2] = [y_1z_2-y_2z_1, x_2z_1-x_1z_2, x_1y_2-x_2y_1]$
    $= - V_2\times V_1$
  - Results in a vector that is orthogonal to the plane defined by V_1 and V_2

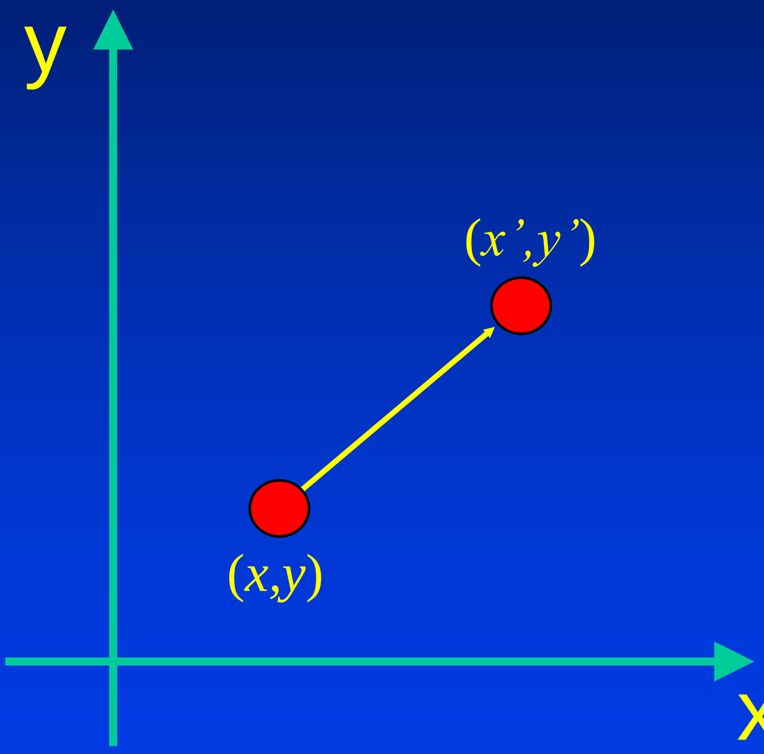
2D Geometric Transformations

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

Translation

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

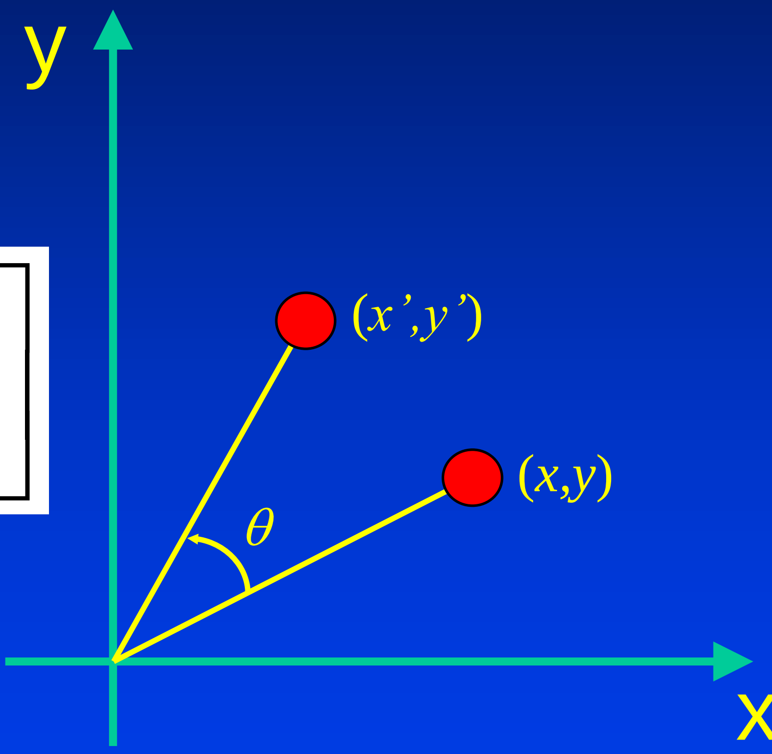
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$



Rotation

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

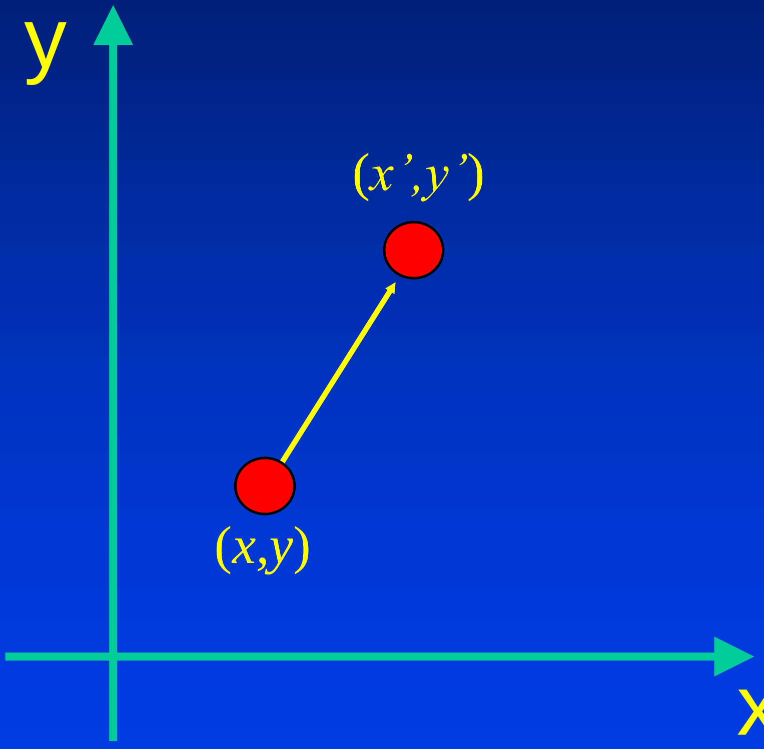
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$



Scaling

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

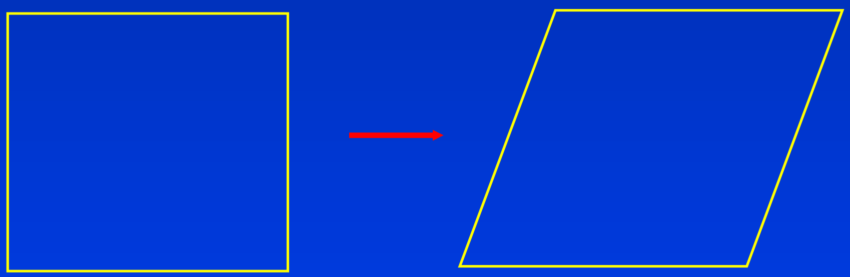
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$



Shear

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

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



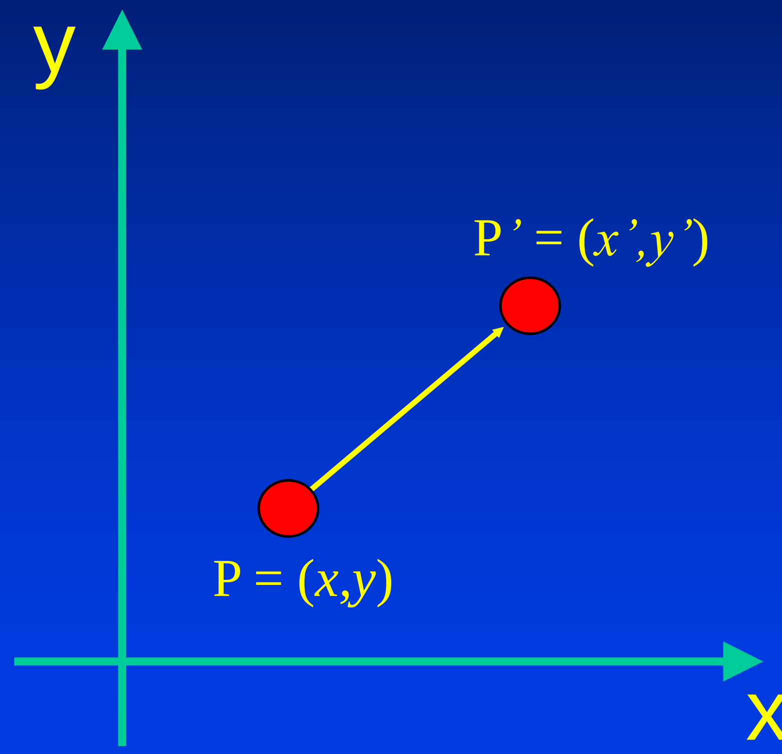
Homogenous Coordinates

- 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

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

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

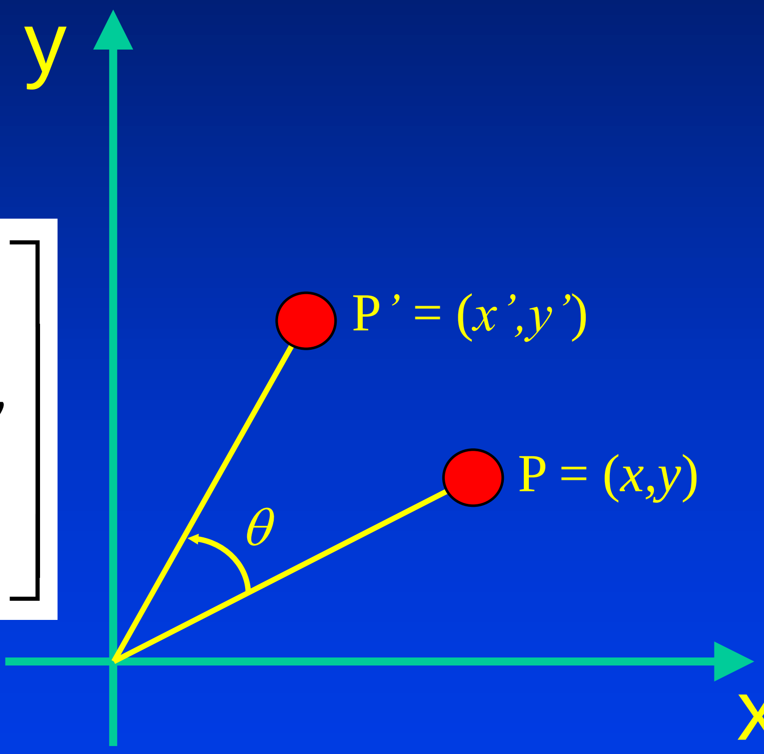


$$\mathbf{P'} = \mathbf{T}(t_x, t_y) \cdot \mathbf{P}$$

Rotation in Homogenous Coordinates

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

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



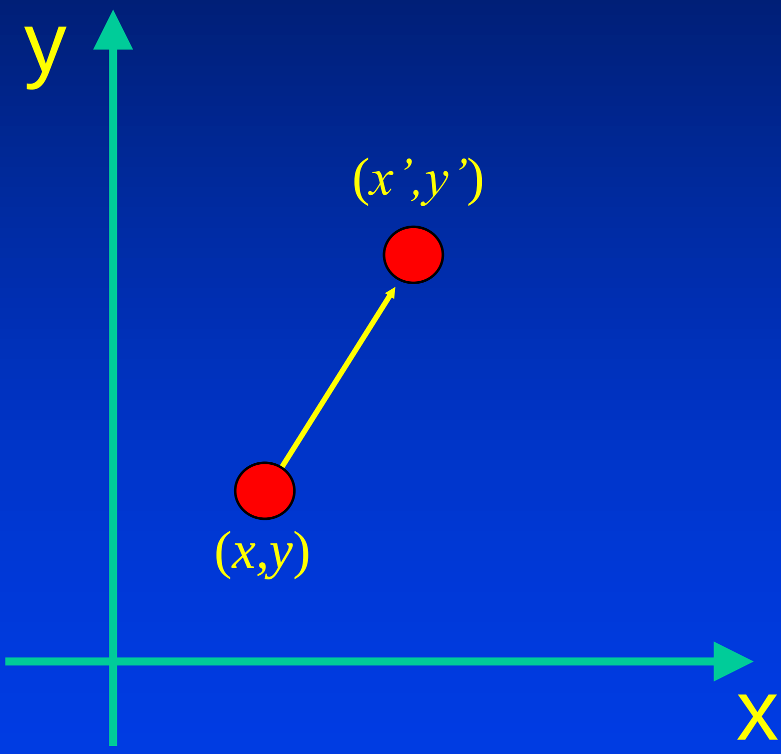
$$\mathbf{P}' = \mathbf{R}(\theta) \cdot \mathbf{P}$$

Scaling in Homogenous Coordinates

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

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

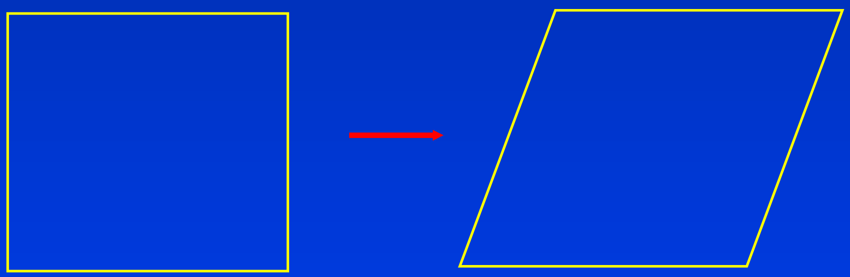
$$\mathbf{P}' = \mathbf{S}(s_x, s_y) \cdot \mathbf{P}$$



Shear in Homogenous Coordinates

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

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



$$\mathbf{P'} = \mathbf{SH}_x \bullet \mathbf{P}$$

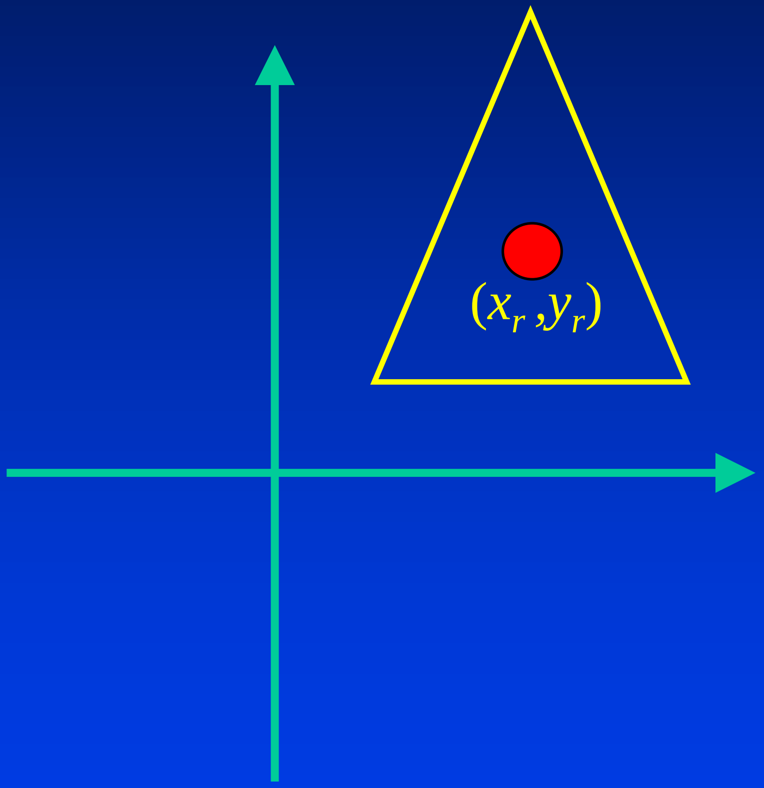
2D Geometric Transformations

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

2D Geometric Transformations

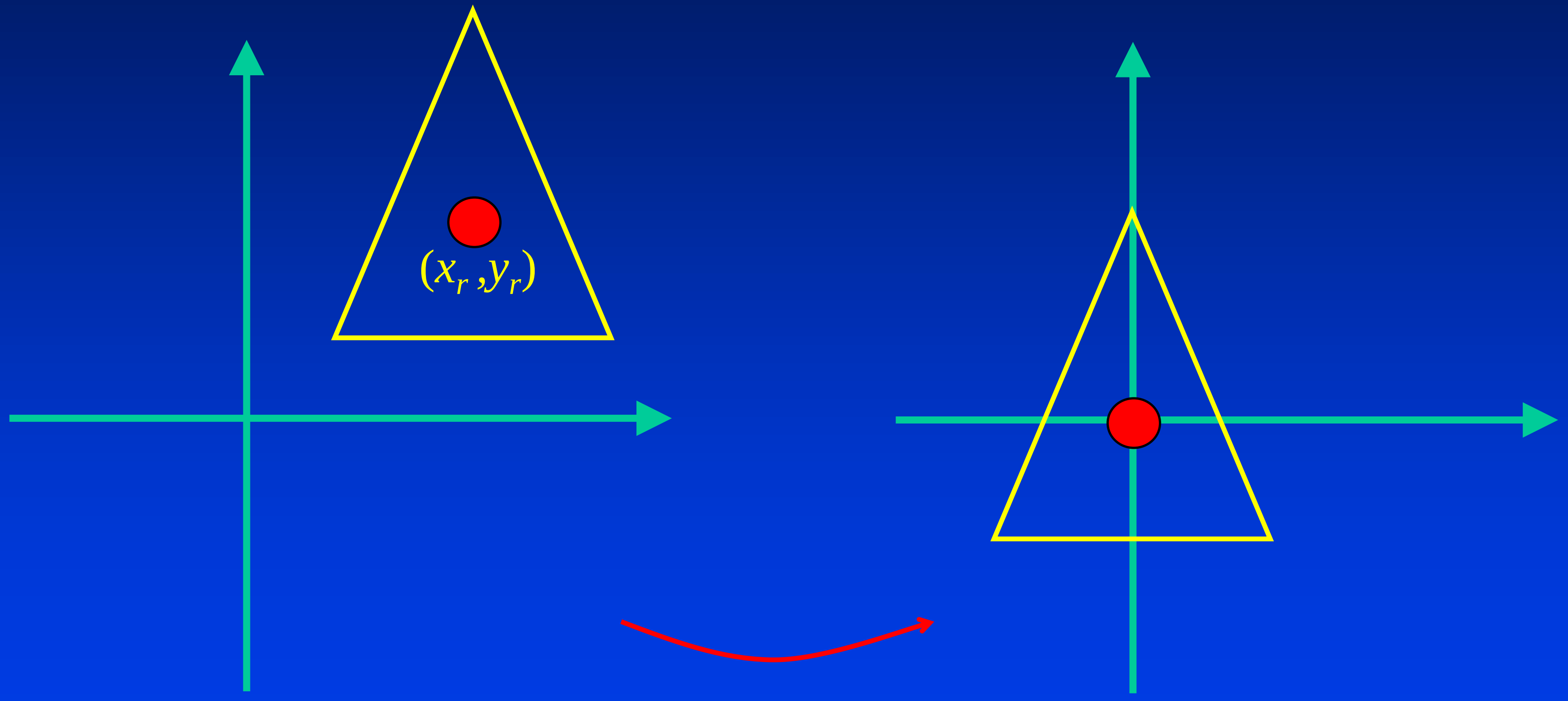
- Translation
- Rotation
- Scaling
- Shear
- Homogenous Coordinates
- Composite Transformations
  - Rotation about a fixed point

Rotation About a Fixed Point



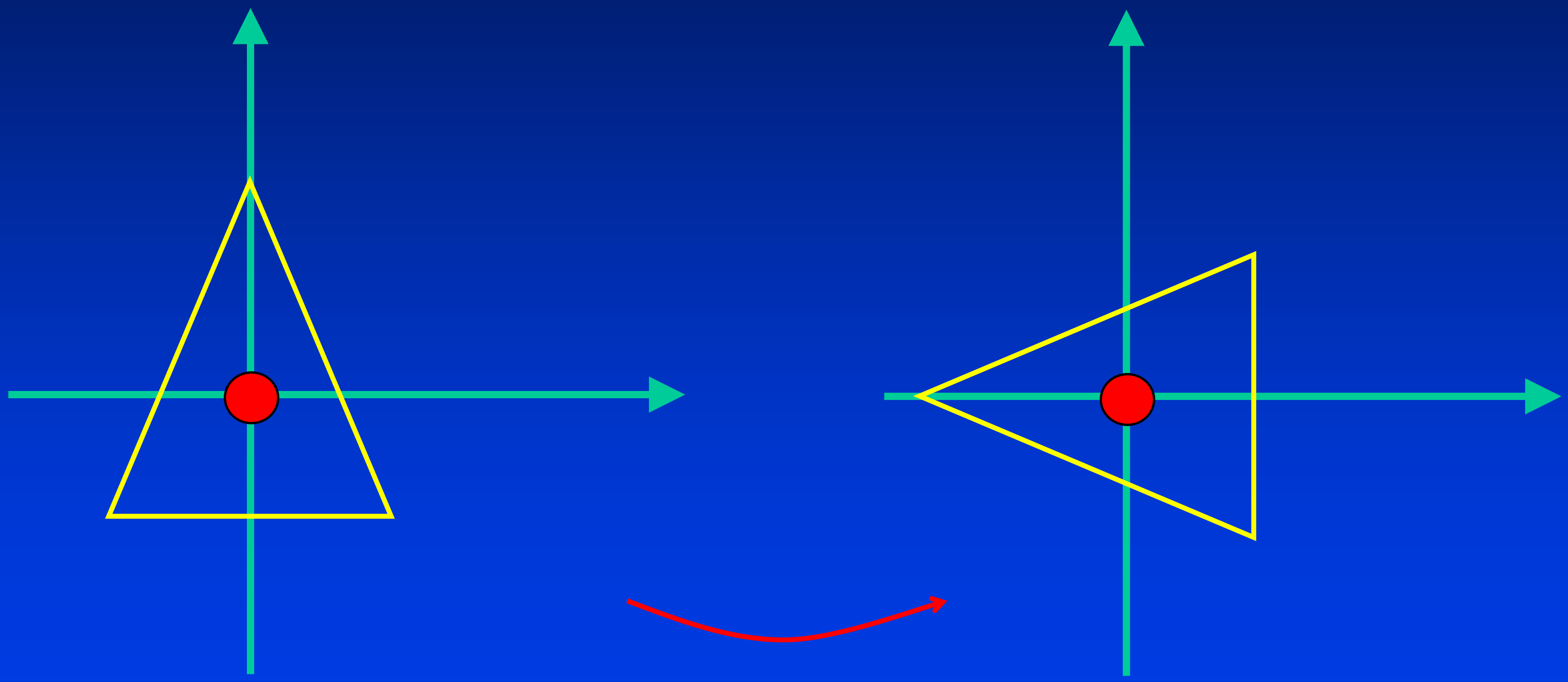
1. **Translate the object to the origin.**
2. **Rotate around the origin.**
3. **Translate the object back.**

Rotation About a Fixed Point



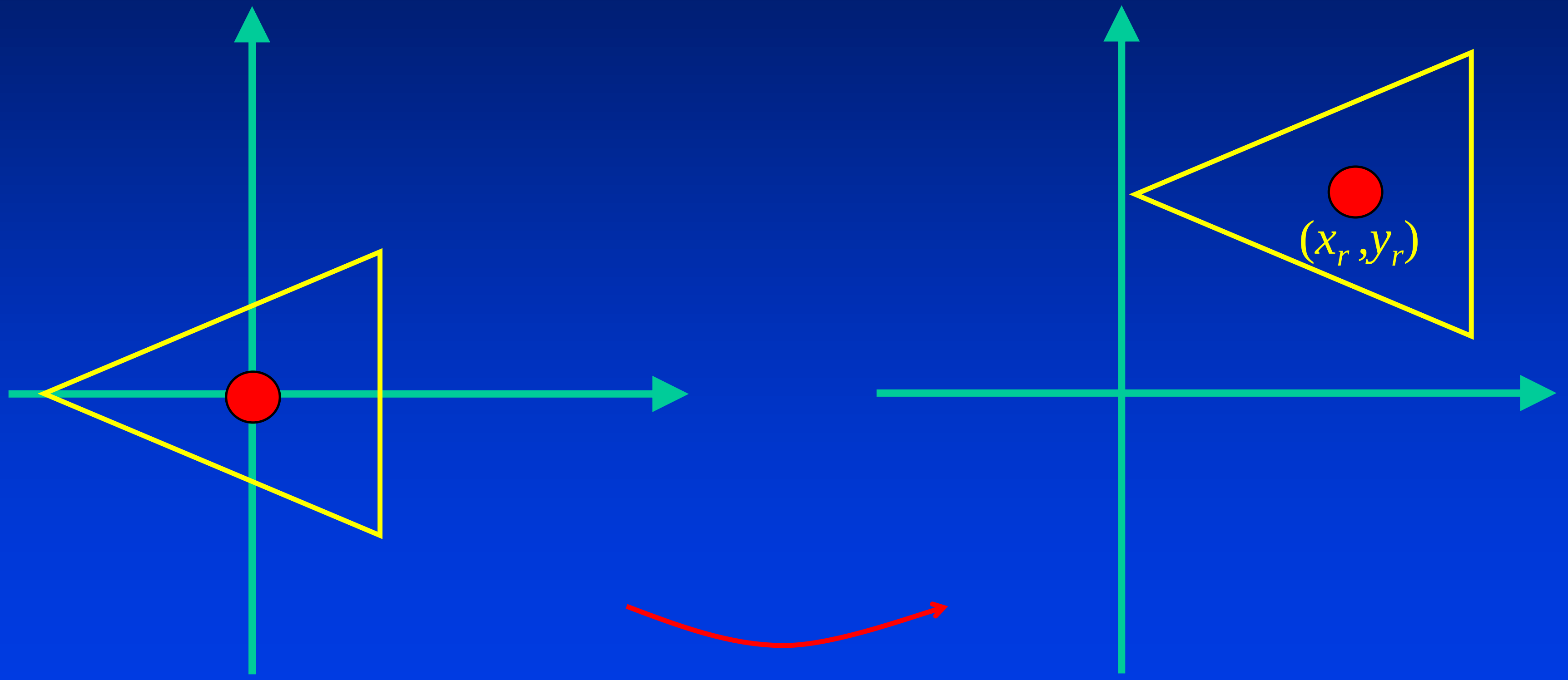
1. Translate the object to the origin

Rotation About a Fixed Point



2. Rotate about origin

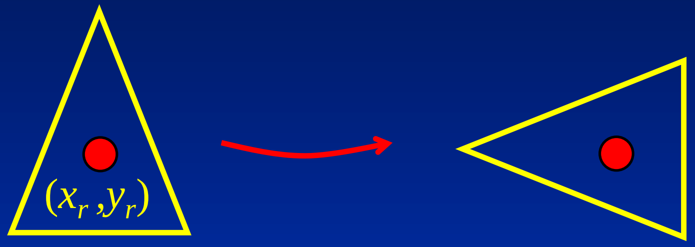
Rotation About a Fixed Point



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.**



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

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

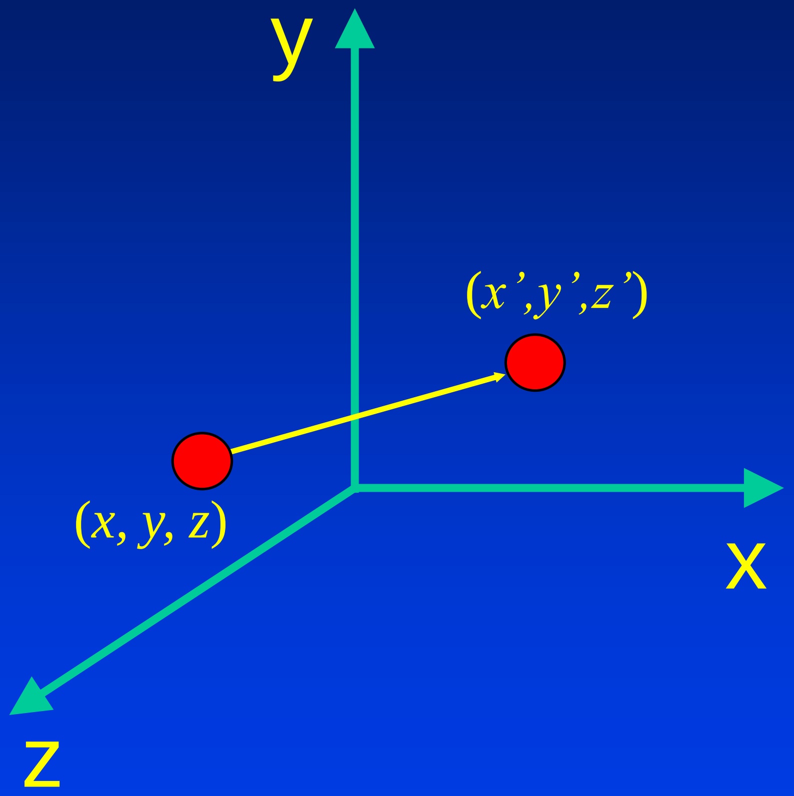
$$\mathbf{P}' = \mathbf{T}(x_r, y_r) \bullet \mathbf{R}(\theta) \bullet \mathbf{T}(-x_r, -y_r) \bullet \mathbf{P}$$

3D Geometric Transformations

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

Translation

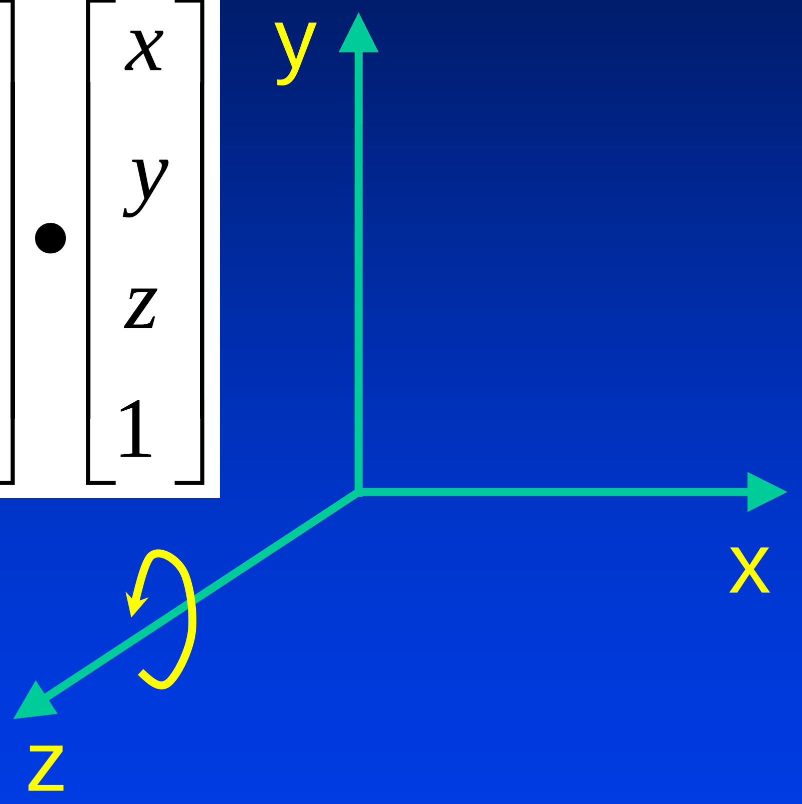
$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$



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

Rotation about z-axis

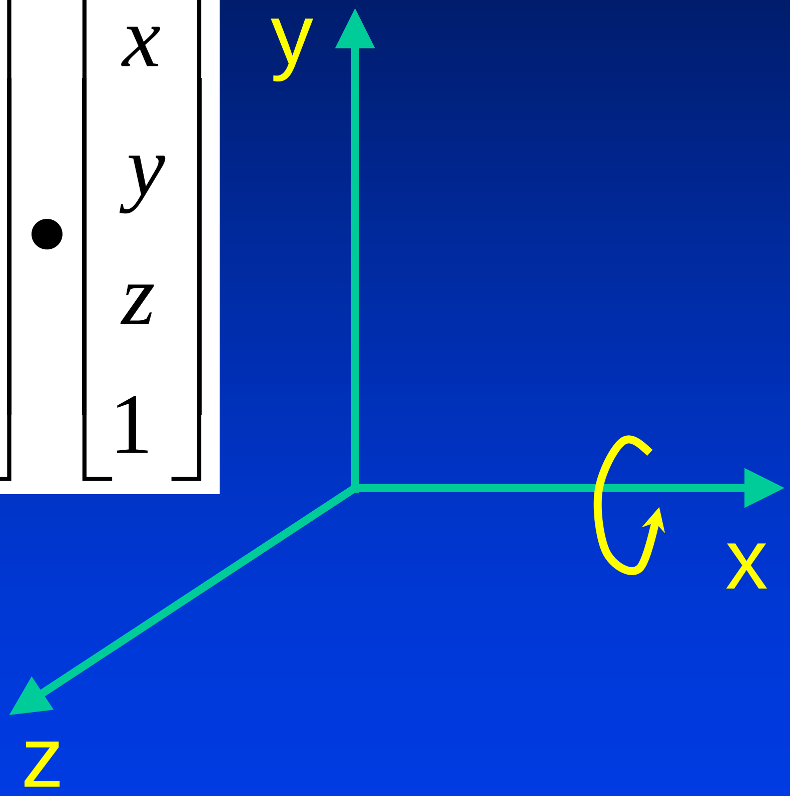
$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$



$$P' = R_z(\theta) \bullet P$$

Rotation about x-axis

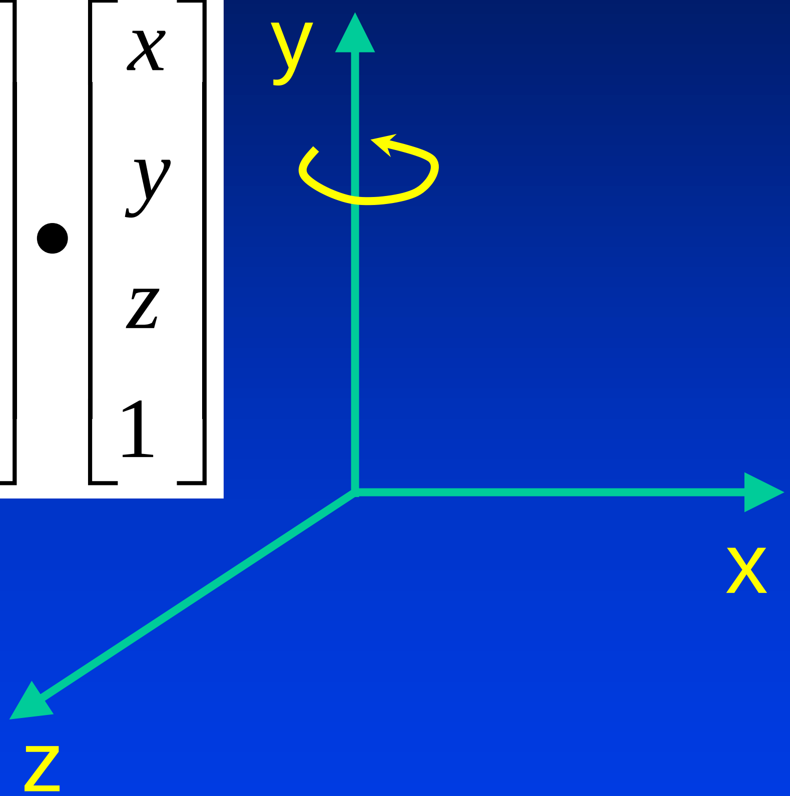
$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$



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

Rotation about y-axis

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



$$\mathbf{P'} = \mathbf{R_y(\theta)} \bullet \mathbf{P}$$

Rotation About a Fixed Point

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

$$\mathbf{P}' = \mathbf{T}\,(x_r\ , y_{r,}\ , z_r) \bullet \mathbf{R1} * \mathbf{R2} * \mathbf{R3} \bullet \mathbf{T}\,(-x_r\ , -y_{r,}\ , -z_r) \bullet \mathbf{P}$$

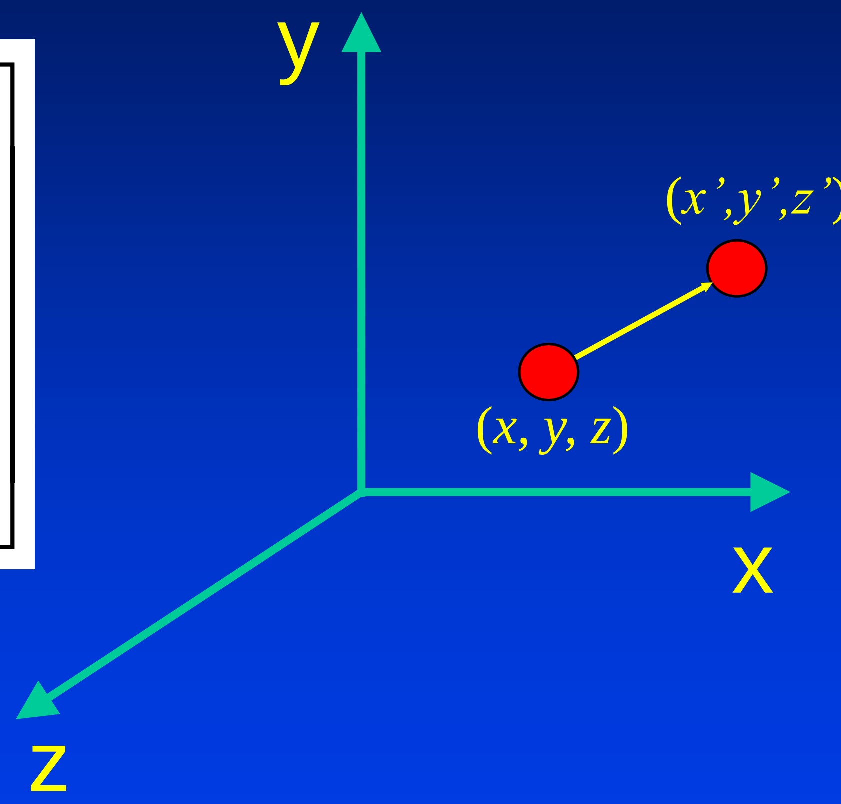
$$\mathbf{Ri} = \mathbf{R_x}(\theta_{\mathbf{x,i}}) \bullet \mathbf{R_y}(\theta_{\mathbf{y,i}}) \bullet \mathbf{R_z}(\theta_{\mathbf{z,i}})$$

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

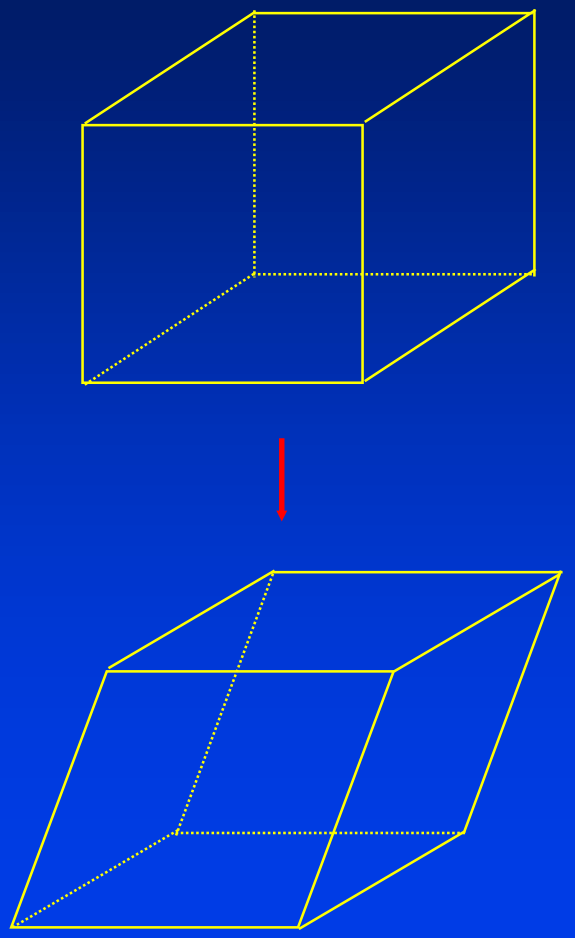
$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$



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

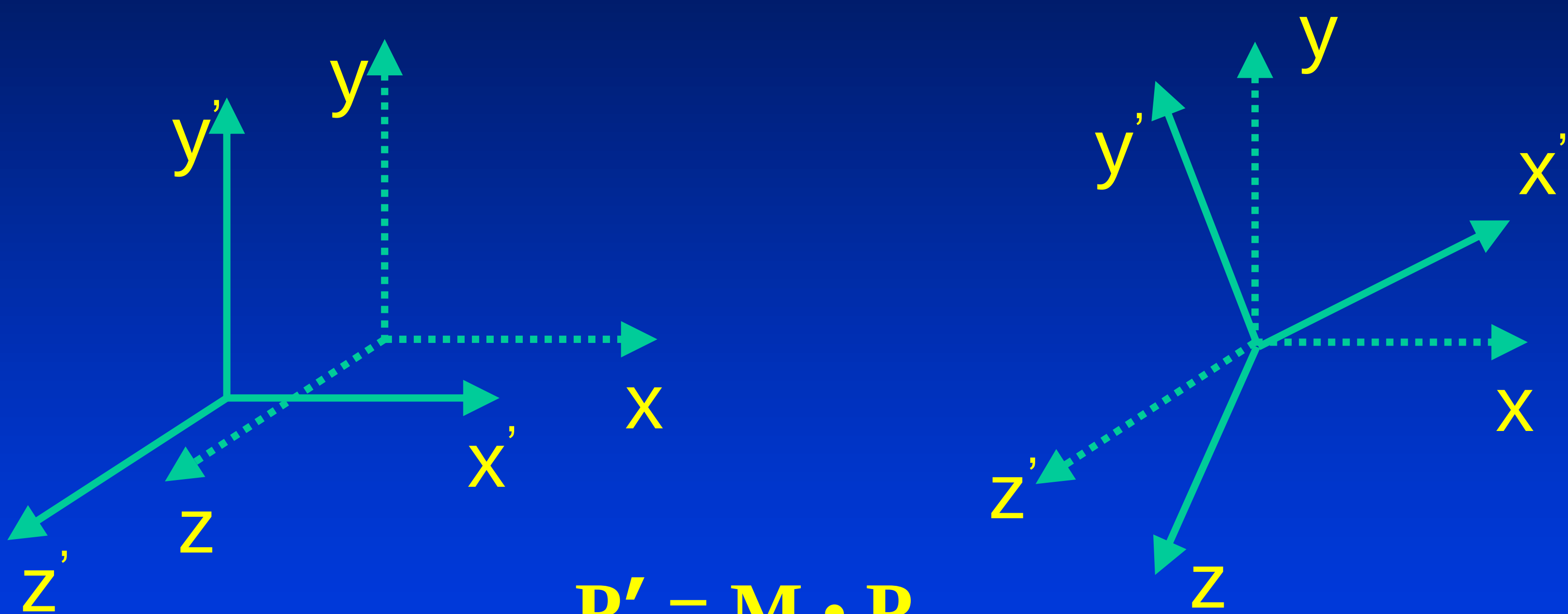
Shear

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



$$\mathbf{P'} = \mathbf{SH}_{xy} \bullet \mathbf{P}$$

Change in Coordinate Systems



$$P' = M \cdot P$$

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

Multiple Coordinate Systems

- If ONLY Translation is involved between the two systems

$$\begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} \rightarrow \begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix}$$

$$\begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix} = \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} + (-\vec{\mathrm{v}})$$

Multiple Coordinate Systems

- What if there is Rotation involved

$$\begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} = \vec{\mathrm{i}}x_1 + \vec{\mathrm{j}}y_1 + \vec{\mathrm{k}}z_1 = \begin{bmatrix} \vec{\mathrm{i}} & \vec{\mathrm{j}} & \vec{\mathrm{k}} \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix}$$

$$\begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix} = \vec{\mathrm{l}}x_2 + \vec{\mathrm{m}}y_2 + \vec{\mathrm{n}}z_2 = \begin{bmatrix} \vec{\mathrm{l}} & \vec{\mathrm{m}} & \vec{\mathrm{n}} \end{bmatrix} \begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix}$$

Multiple Coordinate Systems

- If Rotation is involved

$$\begin{bmatrix} i & j & k \end{bmatrix} = \begin{bmatrix} l & m & n \end{bmatrix} \begin{bmatrix} & & \\ & & \\ & & \end{bmatrix}$$

Multiple Coordinate Systems

$$\begin{bmatrix} i & j & k \end{bmatrix} = \begin{bmatrix} l & m & n \end{bmatrix} \begin{bmatrix} i \bullet l & j \bullet l & k \bullet l \\ i \bullet m & j \bullet m & k \bullet m \\ i \bullet n & j \bullet n & k \bullet n \end{bmatrix}$$

Multiple Coordinate Systems

- Change of bases

$$[\mathrm{i} \quad \mathrm{j} \quad \mathrm{k}] \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} = [\mathrm{l} \quad \mathrm{m} \quad \mathrm{n}] \begin{bmatrix} \mathrm{i} \bullet \mathrm{l} & \mathrm{j} \bullet \mathrm{l} & \mathrm{k} \bullet \mathrm{l} \\ \mathrm{i} \bullet \mathrm{m} & \mathrm{j} \bullet \mathrm{m} & \mathrm{k} \bullet \mathrm{m} \\ \mathrm{i} \bullet \mathrm{n} & \mathrm{j} \bullet \mathrm{n} & \mathrm{k} \bullet \mathrm{n} \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix}$$

$$= [\mathrm{l} \quad \mathrm{m} \quad \mathrm{n}] \begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix}$$

Changes of Bases

$$\mathbf{i} = \mathbf{l}(\mathbf{i} \bullet \mathbf{l}) + \mathbf{m}(\mathbf{i} \bullet \mathbf{m}) + \mathbf{n}(\mathbf{i} \bullet \mathbf{n})$$

$$\mathbf{j} = \mathbf{l}(\mathbf{j} \bullet \mathbf{l}) + \mathbf{m}(\mathbf{j} \bullet \mathbf{m}) + \mathbf{n}(\mathbf{j} \bullet \mathbf{n})$$

$$\mathbf{k} = \mathbf{l}(\mathbf{k} \bullet \mathbf{l}) + \mathbf{m}(\mathbf{k} \bullet \mathbf{m}) + \mathbf{n}(\mathbf{k} \bullet \mathbf{n})$$

Changes of Bases

$$\begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix} = \begin{bmatrix} & & \\ & & \\ & & \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix}$$

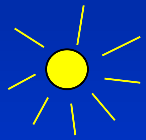
Homogeneous Representations

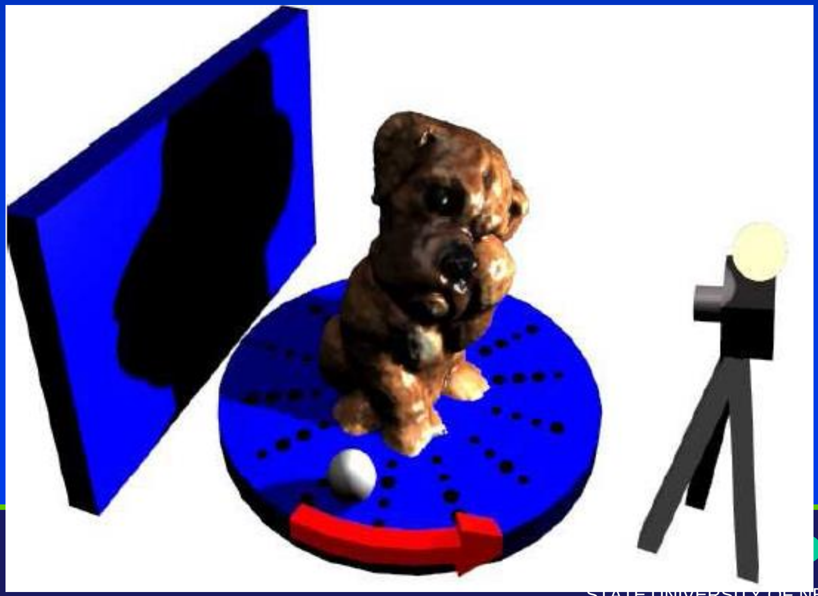
$$\begin{bmatrix} x_2 \\ y_2 \\ z_2 \\ 1 \end{bmatrix} = \begin{bmatrix} \mathrm{i} \bullet \mathrm{l} & \mathrm{j} \bullet \mathrm{l} & \mathrm{k} \bullet \mathrm{l} & \mathrm{v}_\mathrm{x} \\ \mathrm{i} \bullet \mathrm{m} & \mathrm{j} \bullet \mathrm{m} & \mathrm{k} \bullet \mathrm{m} & \mathrm{v}_\mathrm{y} \\ \mathrm{i} \bullet \mathrm{n} & \mathrm{j} \bullet \mathrm{n} & \mathrm{k} \bullet \mathrm{n} & \mathrm{v}_\mathrm{z} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{bmatrix}$$

Image Formation

- Camera
- Light, shape, reflectance, texture

Image formation



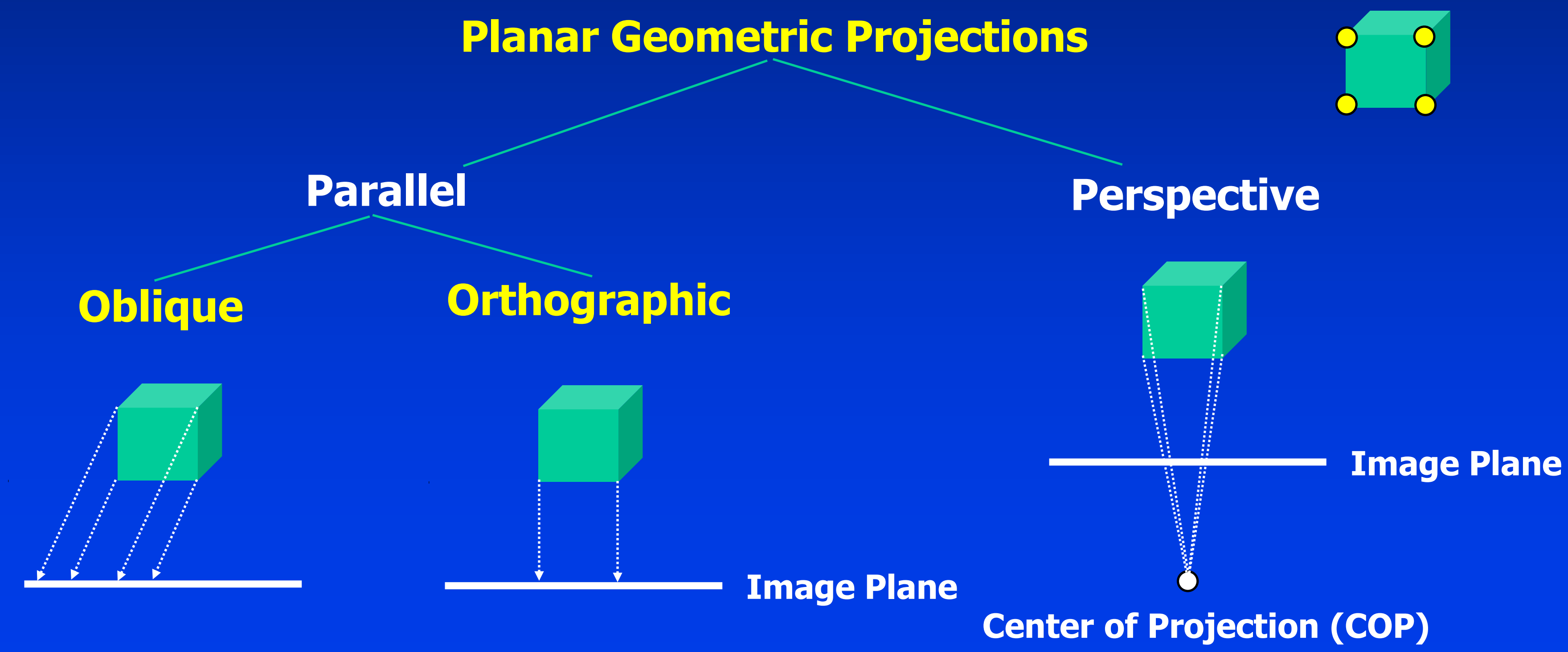


Viewing in 3D

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

Planar Geometric Projections

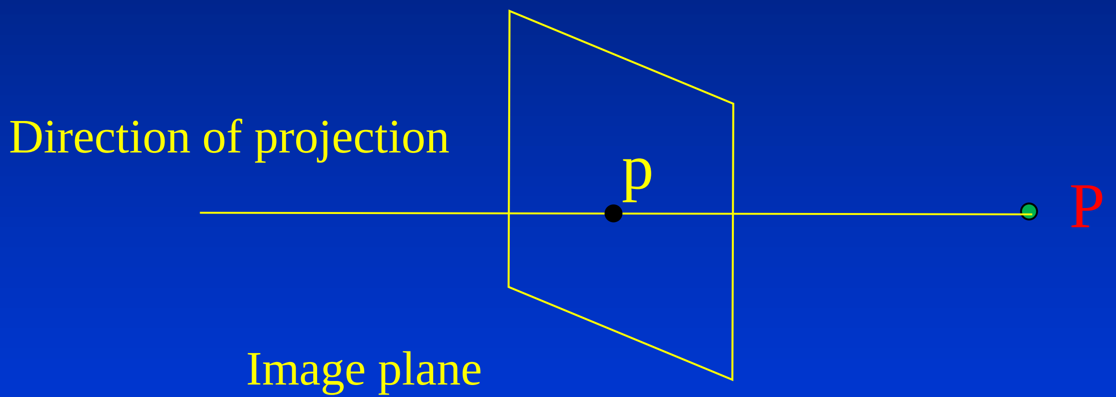
- Maps points from camera coordinate system to the screen (image plane of the virtual camera).



Orthographic Camera Model

Infinite Projection matrix - last row is (0,0,0,1)

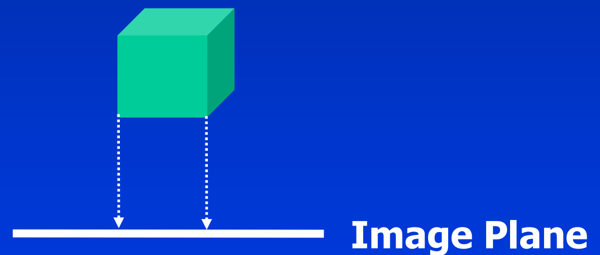
Good Approximations – object is far from the camera (relative to its size)



$$P_{orth} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Parallel Orthographic Projection

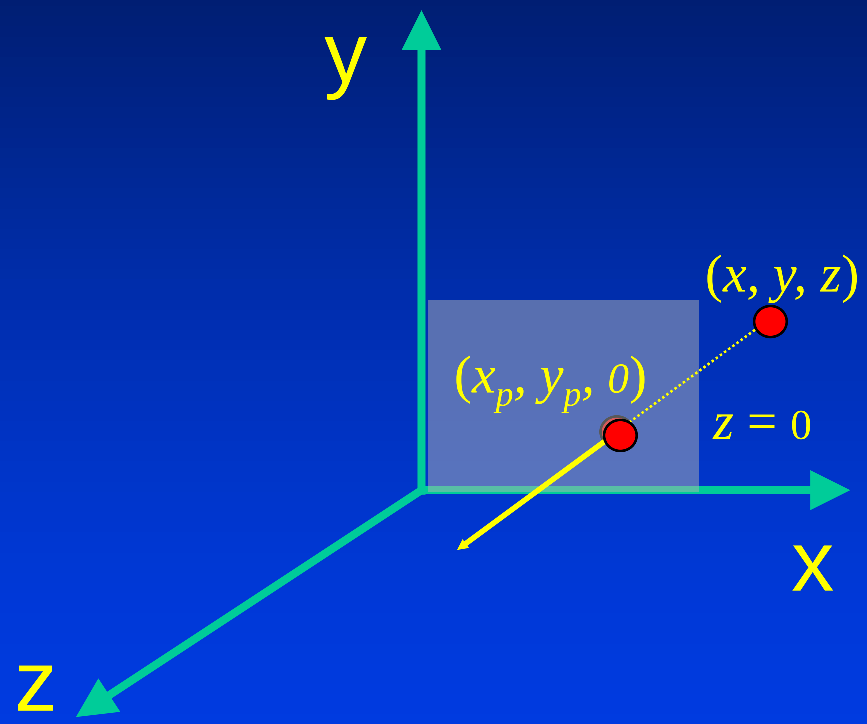
- Preserves X and Y coordinates.
- Preserves both distances and angles.



Parallel Orthographic Projection

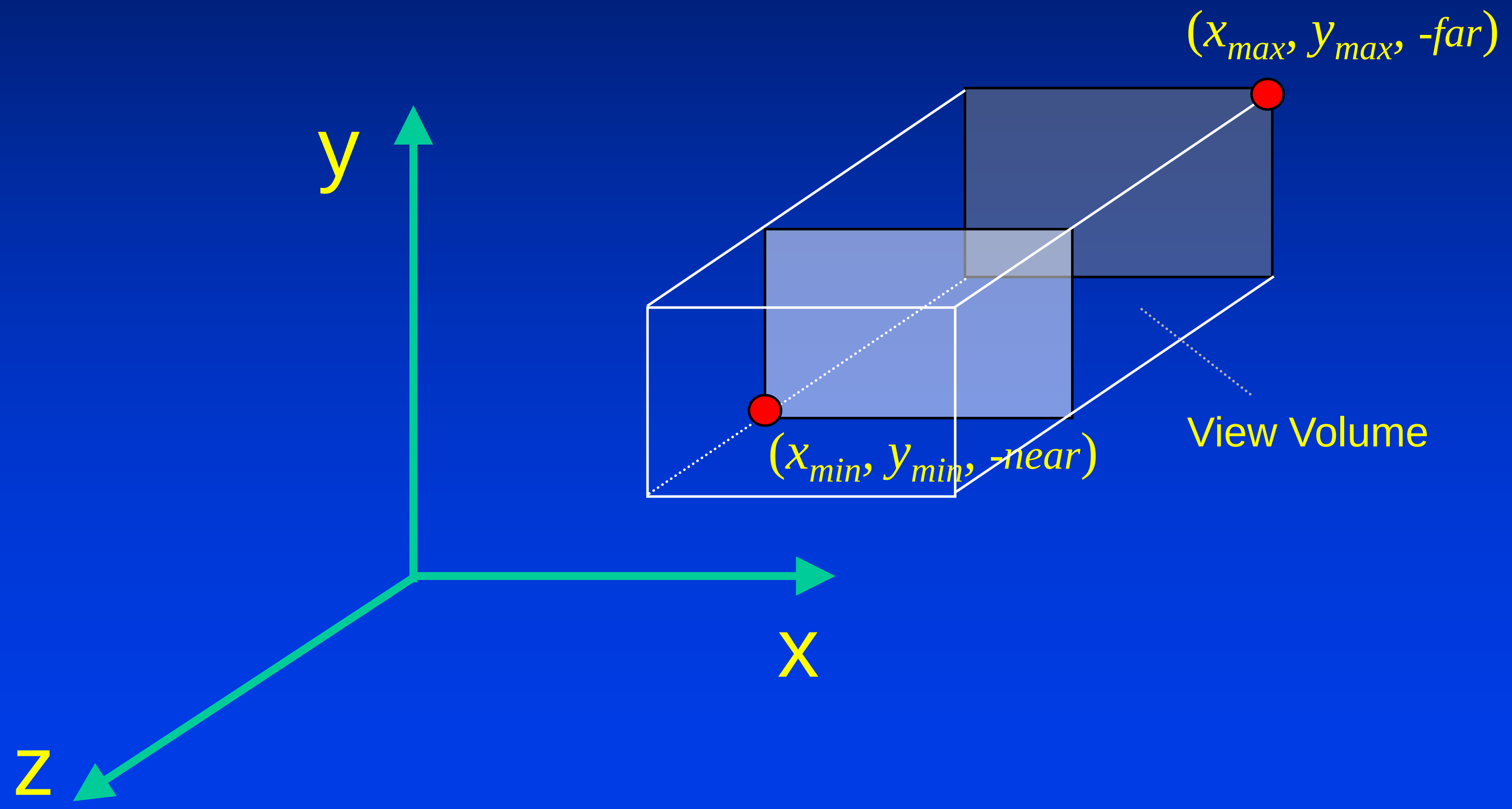
- $x_p = x$
- $y_p = y$
- $z_p = 0$

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

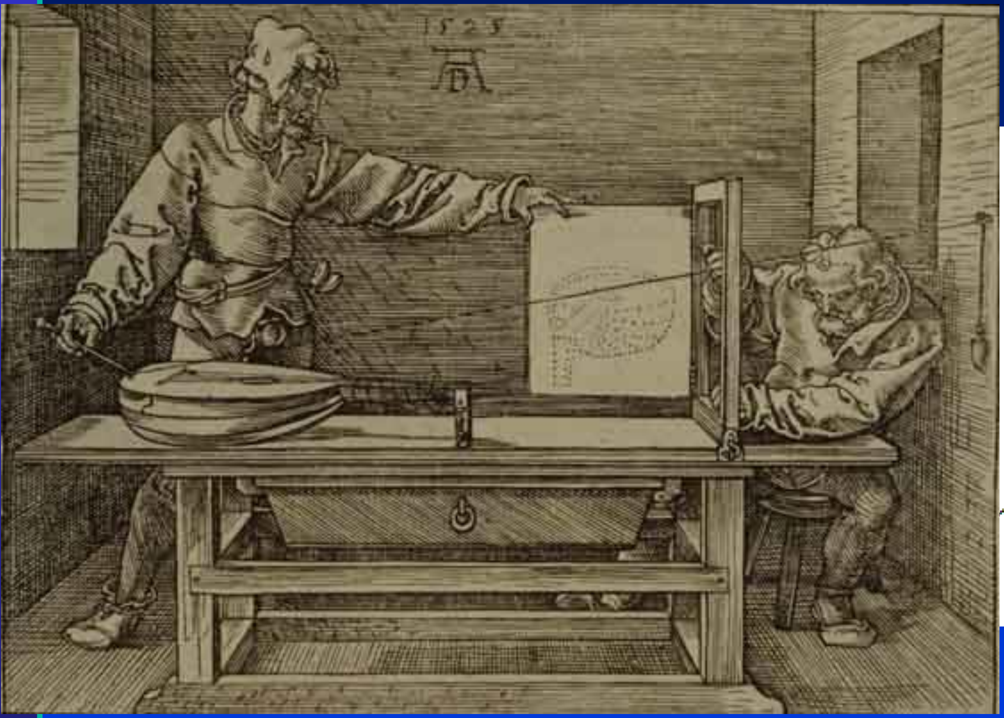


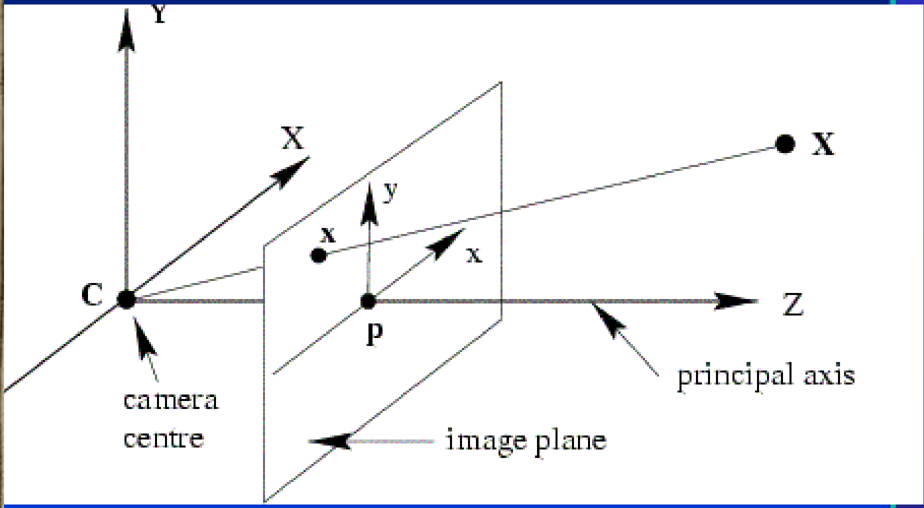
Defining the Parallel View Volume

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



Projective Camera Model



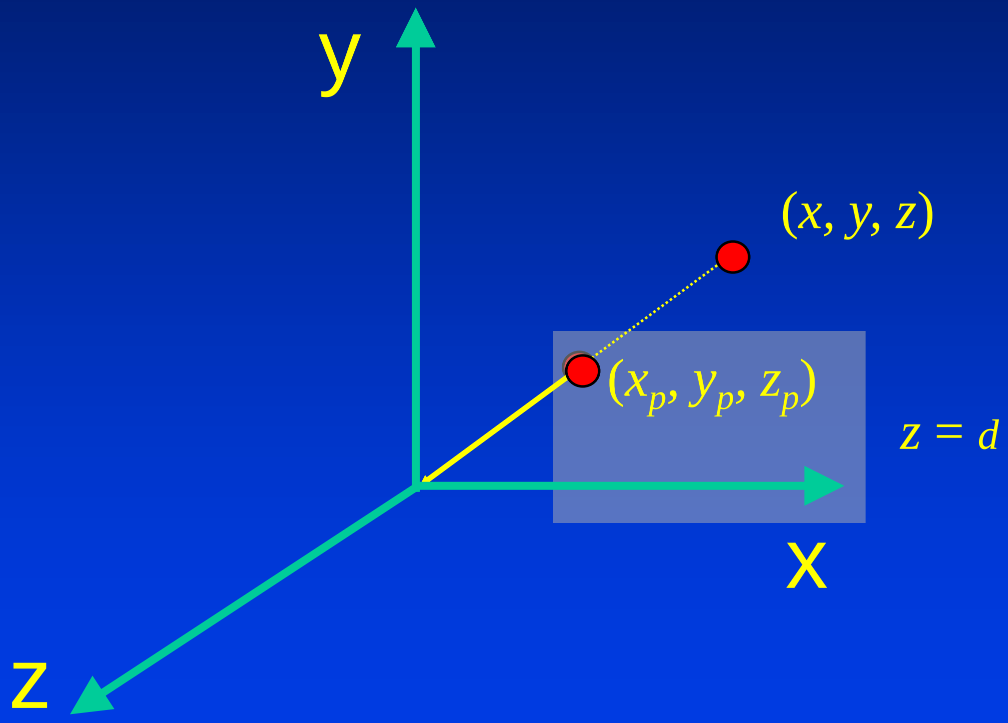


$\mathbf{x} = P\mathbf{X} \quad P: 3 \times 4$ **Projection matrix**

$\mathbf{x} = K[R \quad \mathbf{t}]\mathbf{X} \quad K: 3 \times 3$ **Camera matrix (int. parameters)**

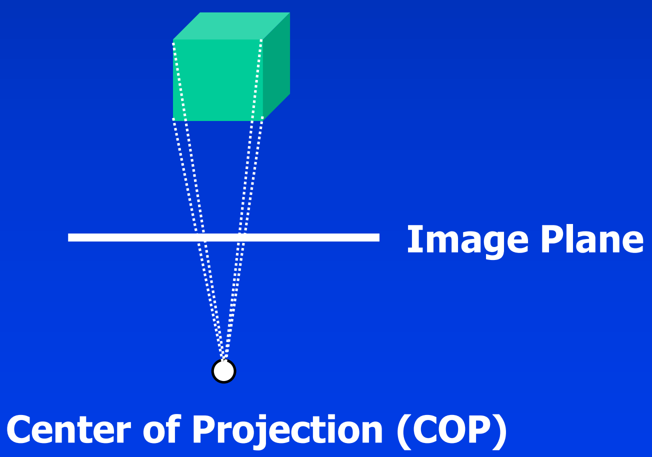
$R, \mathbf{t}$ **Rotation, translation (ext. parameters)**

Perspective Projection



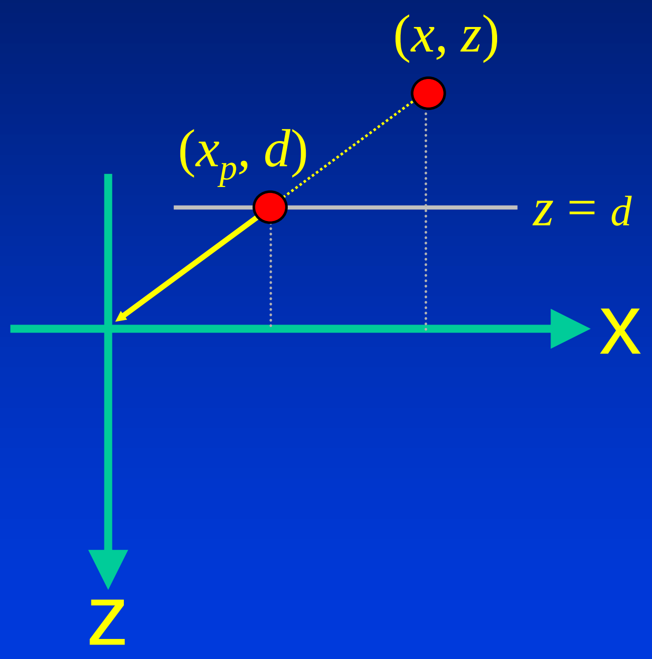
Perspective Projection

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



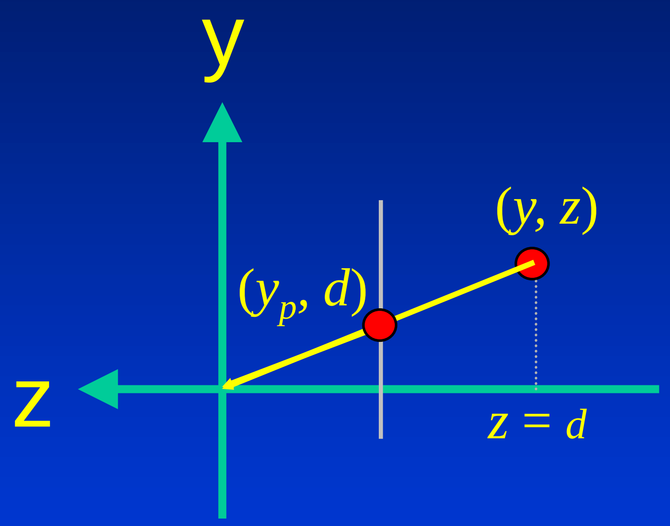
Perspective Projection

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



Perspective Projection

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



Perspective Projection

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

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

Viewing in 3D

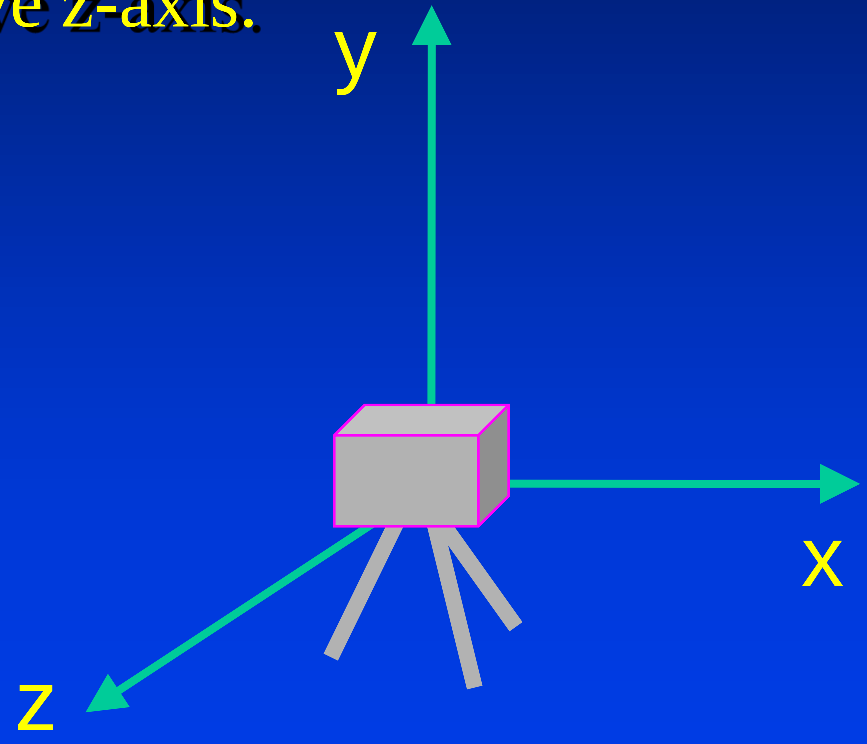
- Planar Geometric Projections
- Parallel Orthographic Projections
- Perspective Projections
- Projections in OpenGL

Viewing in 3D

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

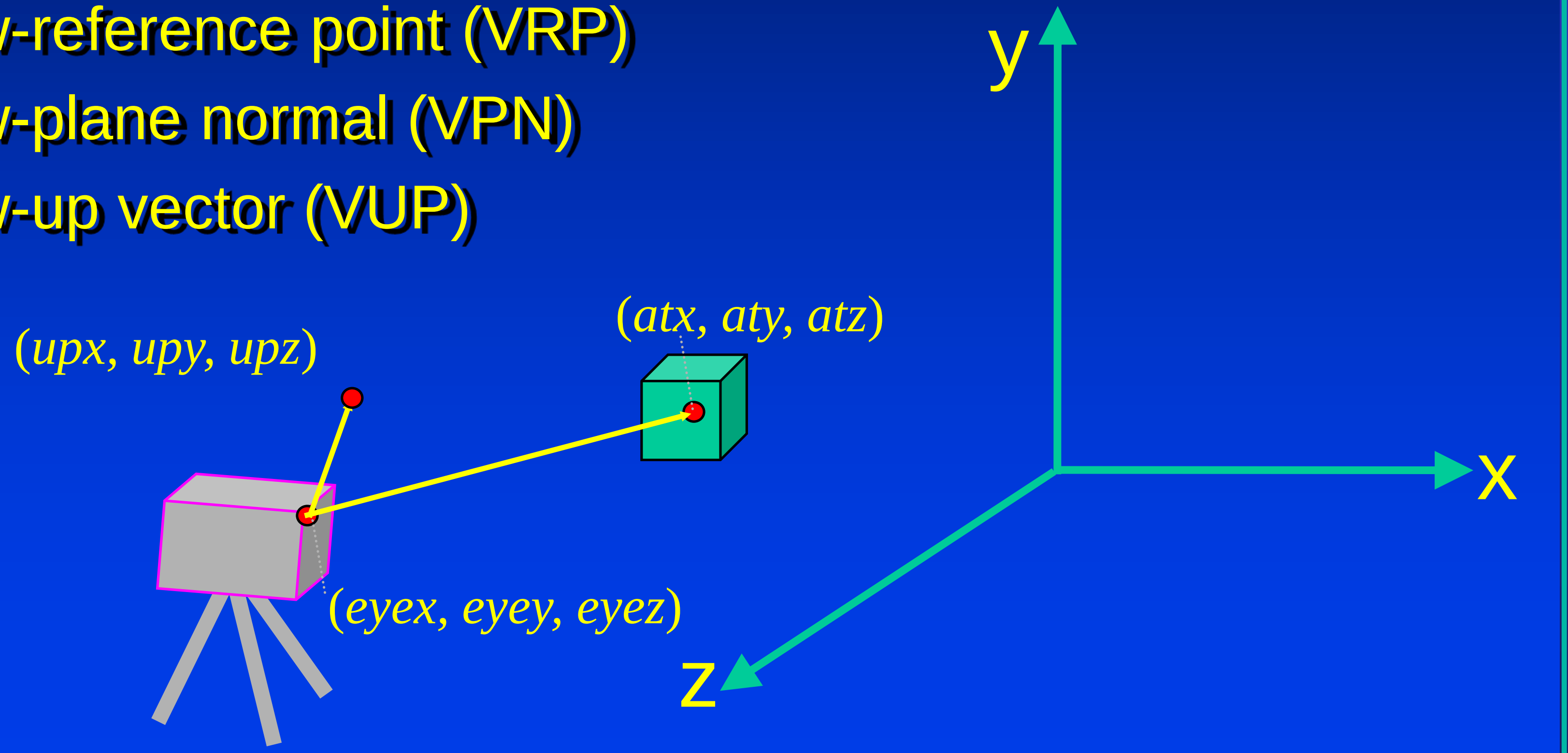
Positioning the Camera

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



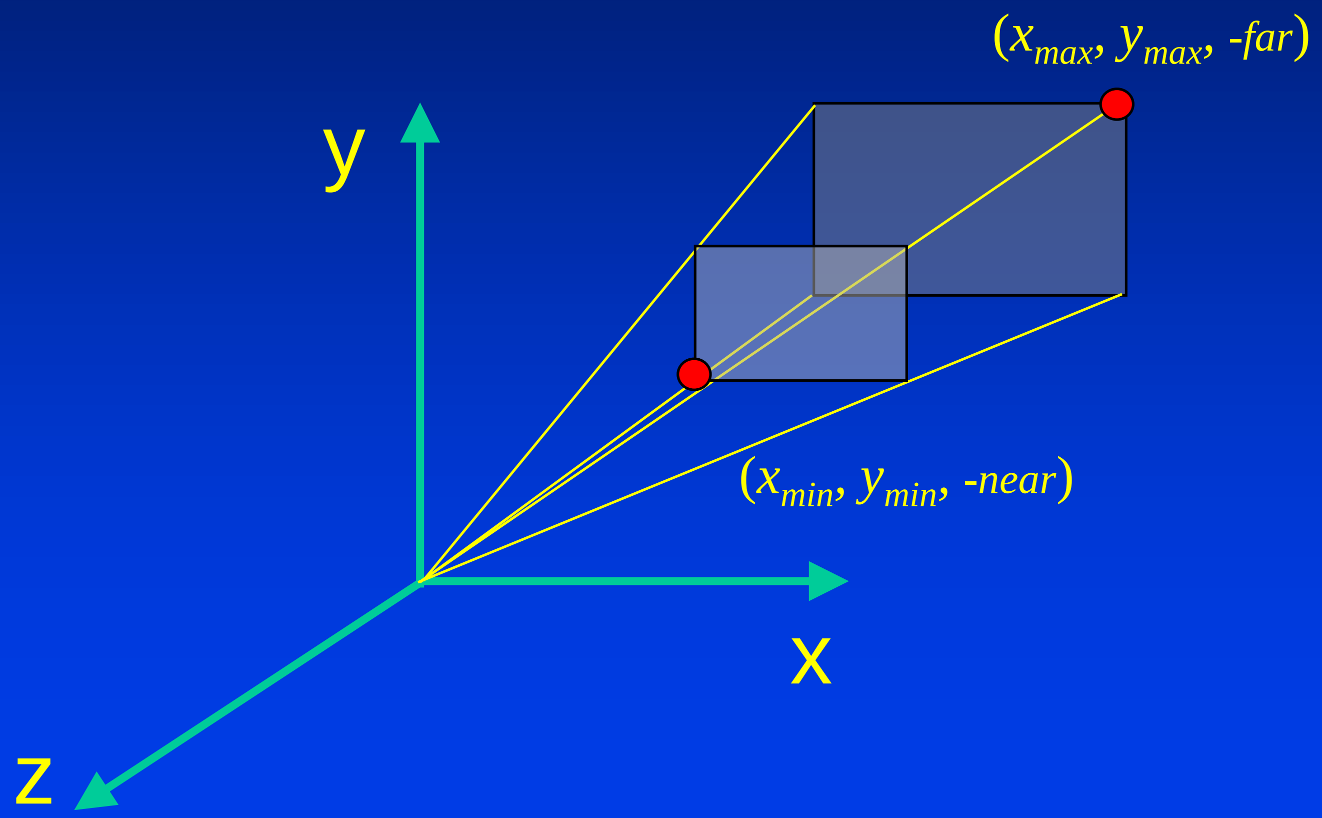
Positioning the Camera

- 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)



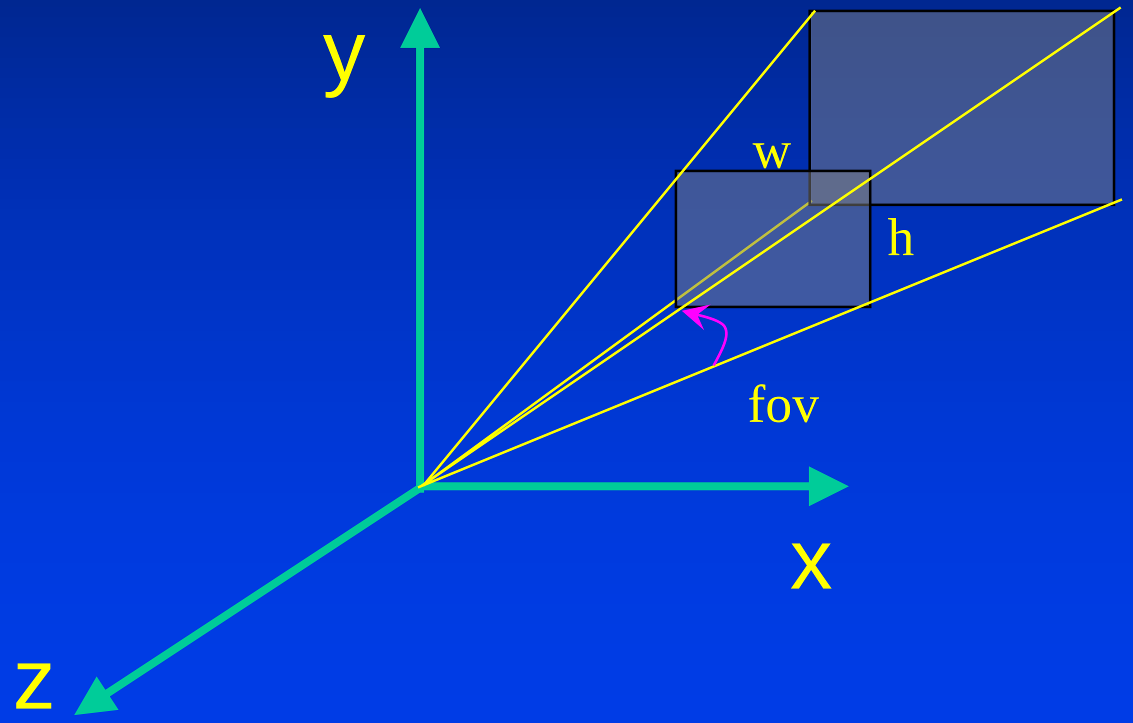
Defining the Perspective View Volume

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



Defining the Perspective View Volume

gluPerspective(fovy, aspect, near, far)



Taking a Picture with a Camera

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