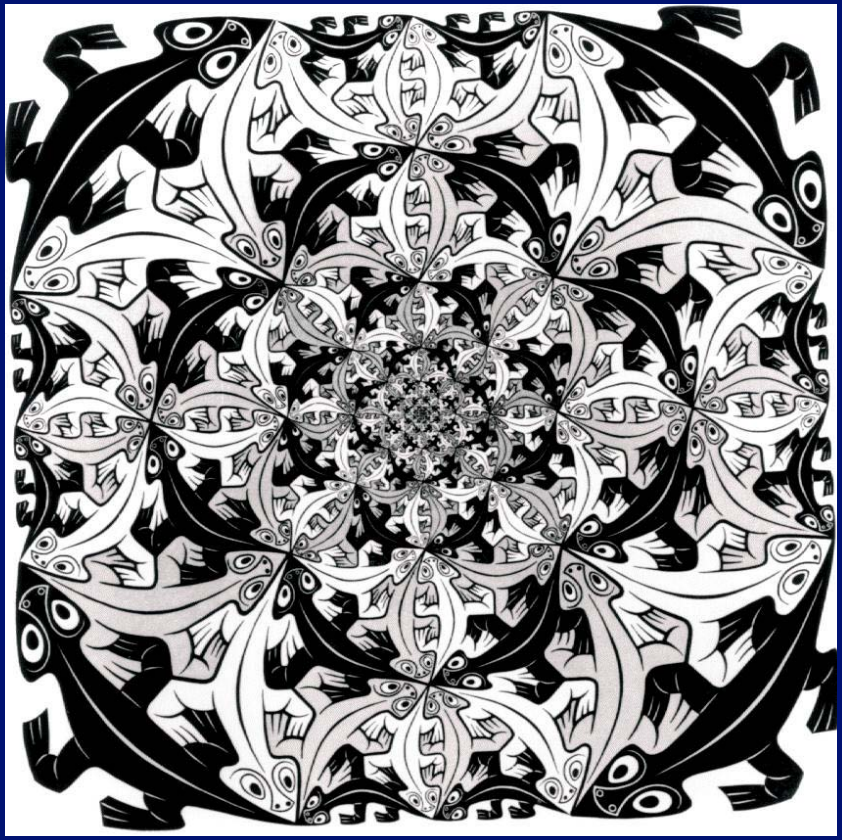
CS164: 2D and 3D Transformations

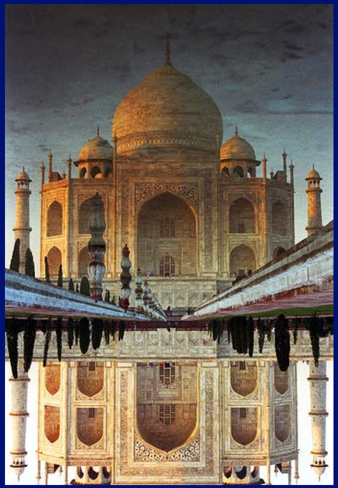
Leonidas Guibas
Computer Science Dept.
Stanford University



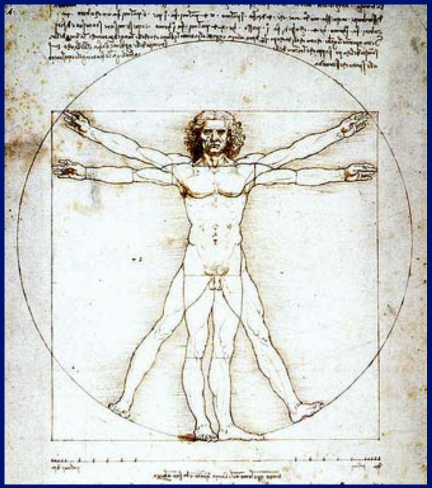
M.C. Escher – Smaller and Smaller (1956)

Transformations are everywhere --- in science, engineering, art …

Symmetries







Symmetry and regularity is related to art, aesthetics, and our sense of beauty. It is also prevalent for economic, manufacturing and other efficiency reasons.

“Symmetry is a complexity-reducing concept [...]; seek it everywhere.”

Alan J. Perlis

Transformations in Graphics, Robotics and Vision

In Modeling (Graphics)

- Allows definitions of objects in own coordinate systems
- Allows use of object definition multiple times in a scene

In Kinematics (Robotics)

- Allows specification of multi-link kinematic chains

In Multi-View Geometry (Vision)

- Allows derivation of fundamental relationships between images of the same scene from different cameras

Overview

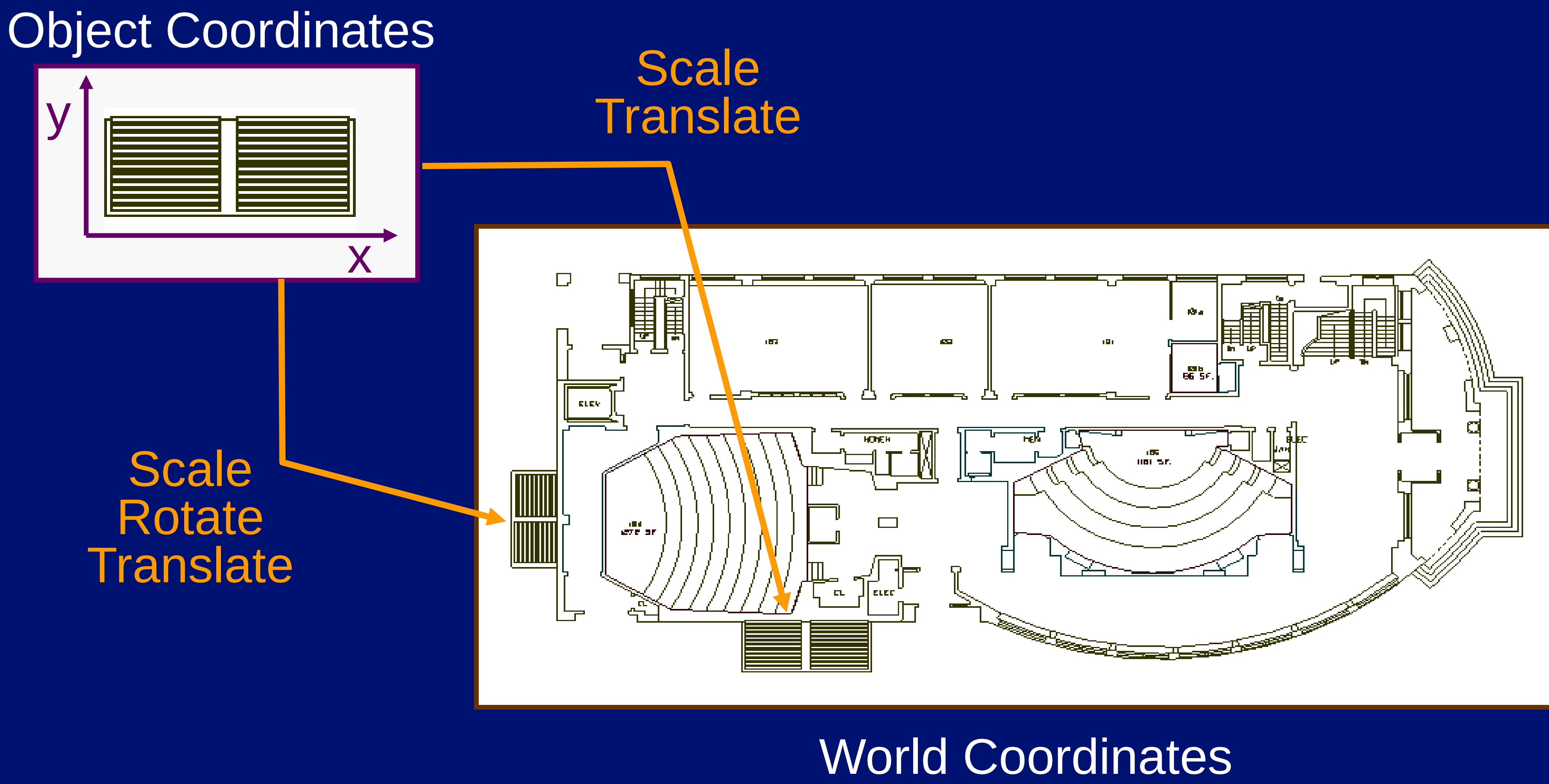
2D Transformations

- Basic 2D transformations
- Matrix representation
- Matrix composition

3D Transformations

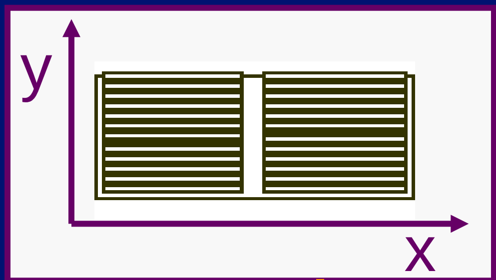
- Basic 3D transformations
- Matrix forms

2D Modeling Transformations

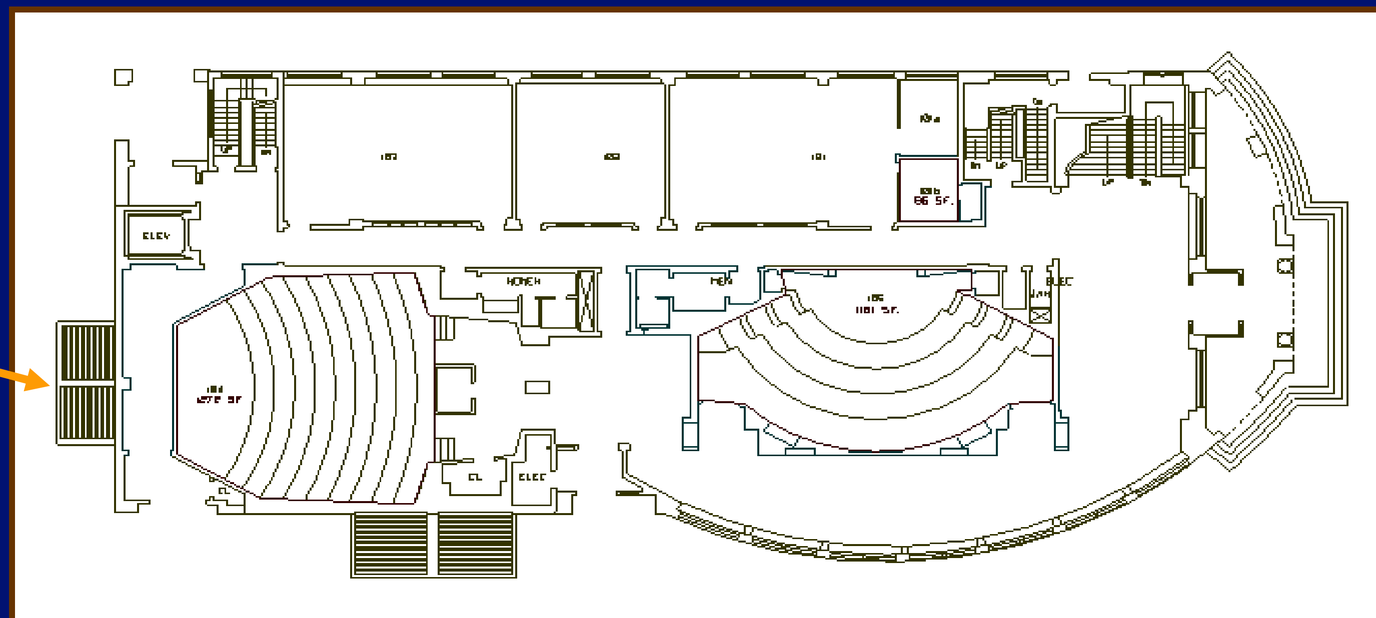


2D Modeling Transformations

Object Coordinates



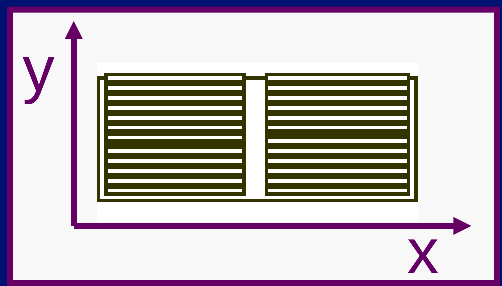
Let’s look at this in detail…



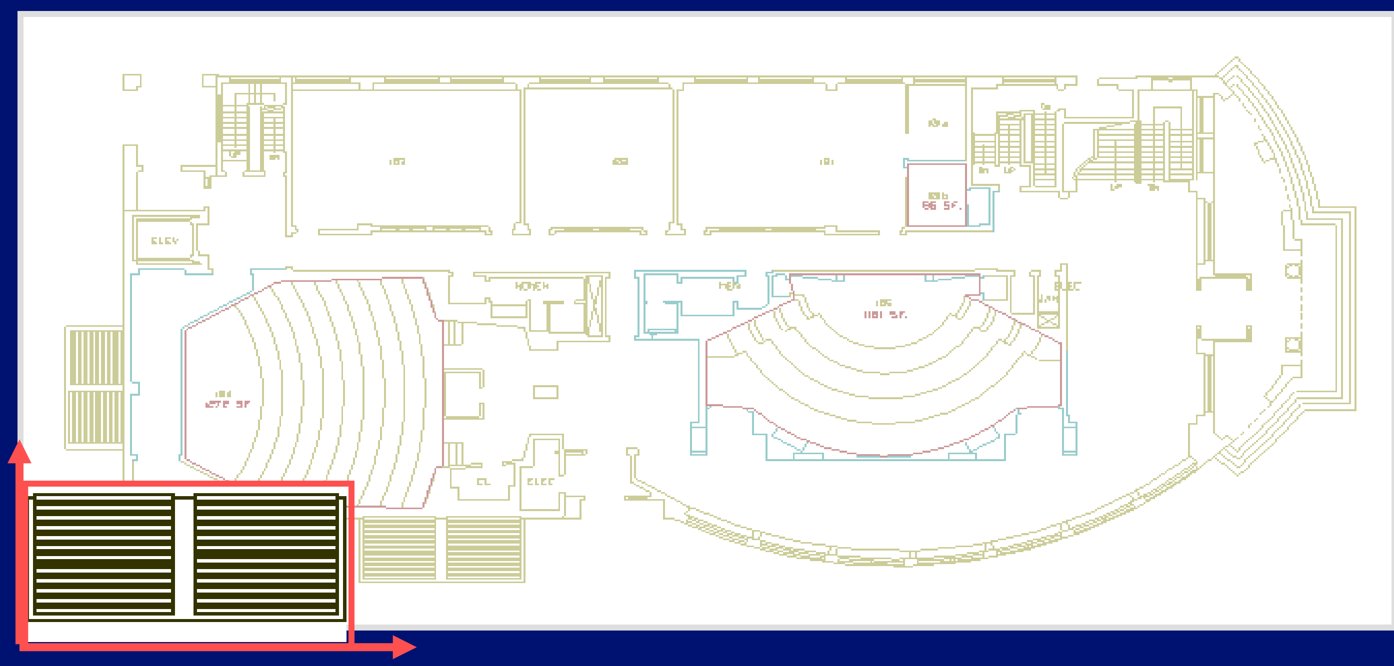
World Coordinates

2D Modeling Transformations

Object Coordinates

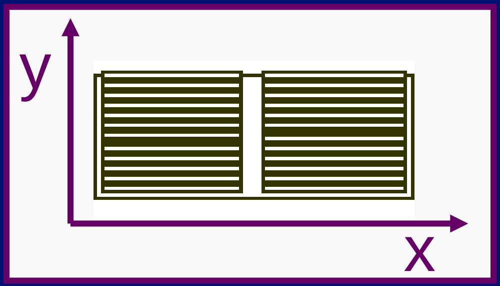


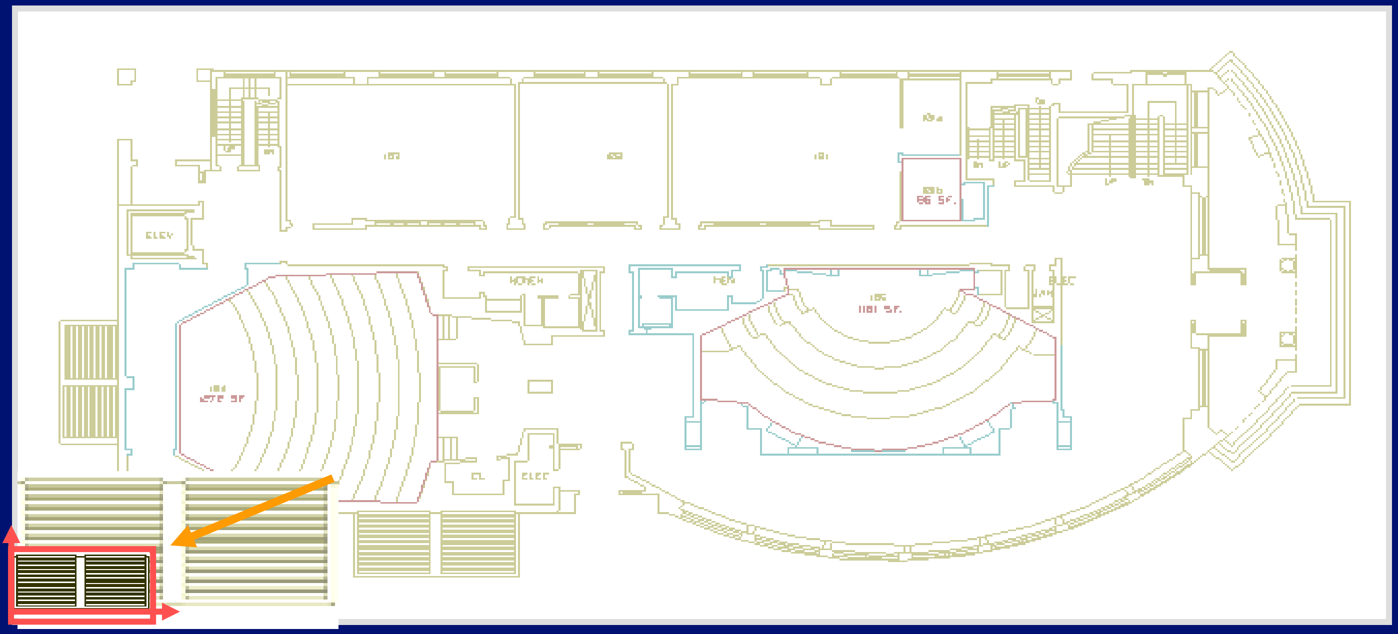
Initial location at (0, 0) with x- and y-axes aligned



2D Modeling Transformations

Object Coordinates

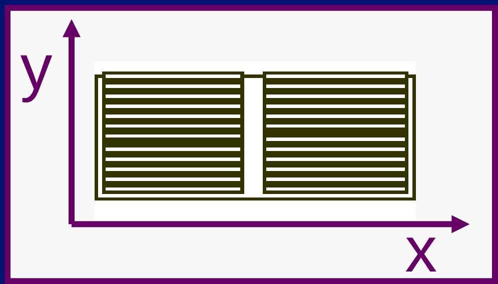




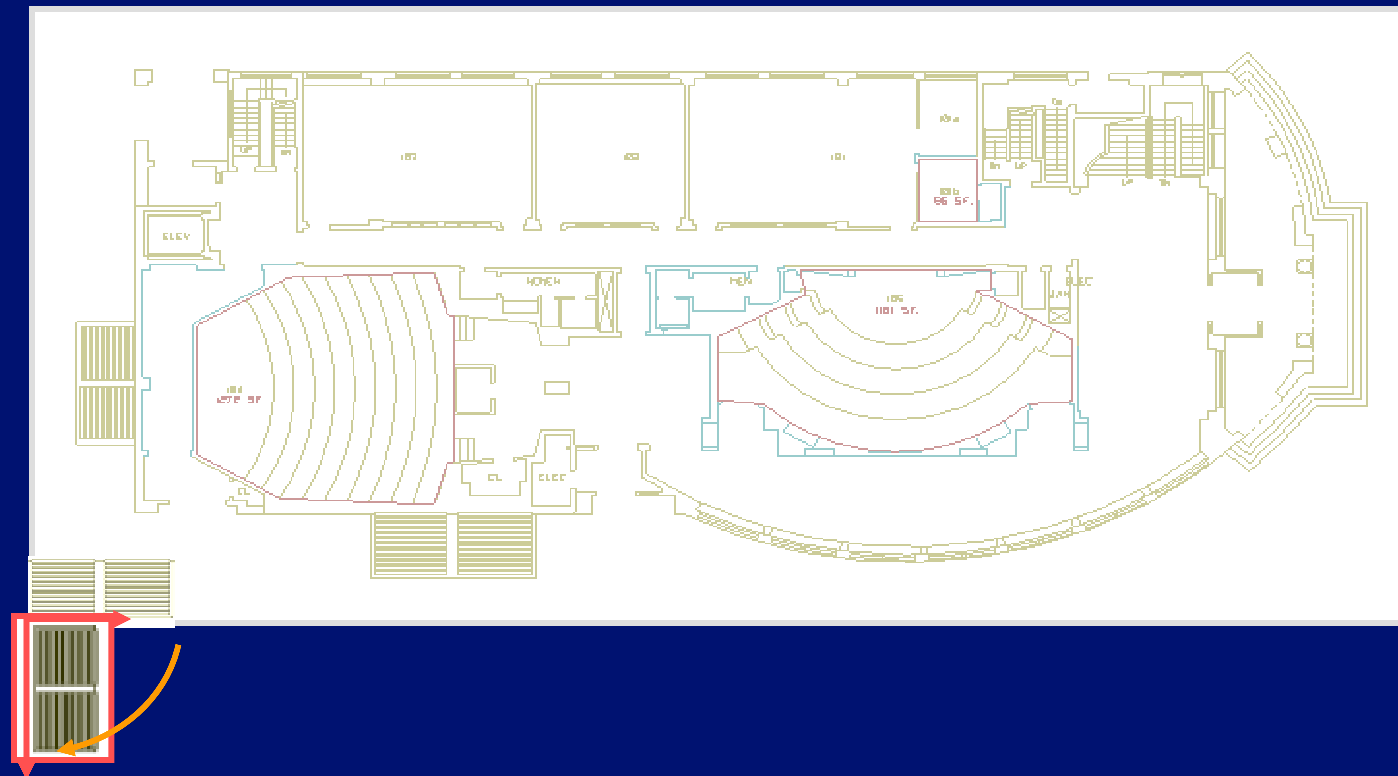
Scale .3, .3
Rotate -90
Translate 5, 3

2D Modeling Transformations

Object Coordinates

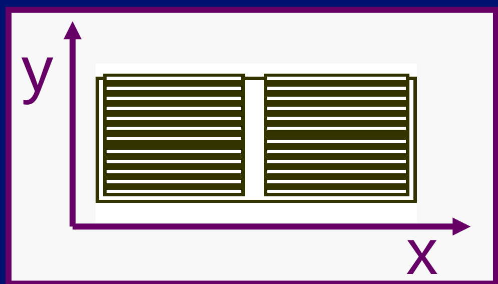


Scale .3, .3
Rotate -90
Translate 5, 3

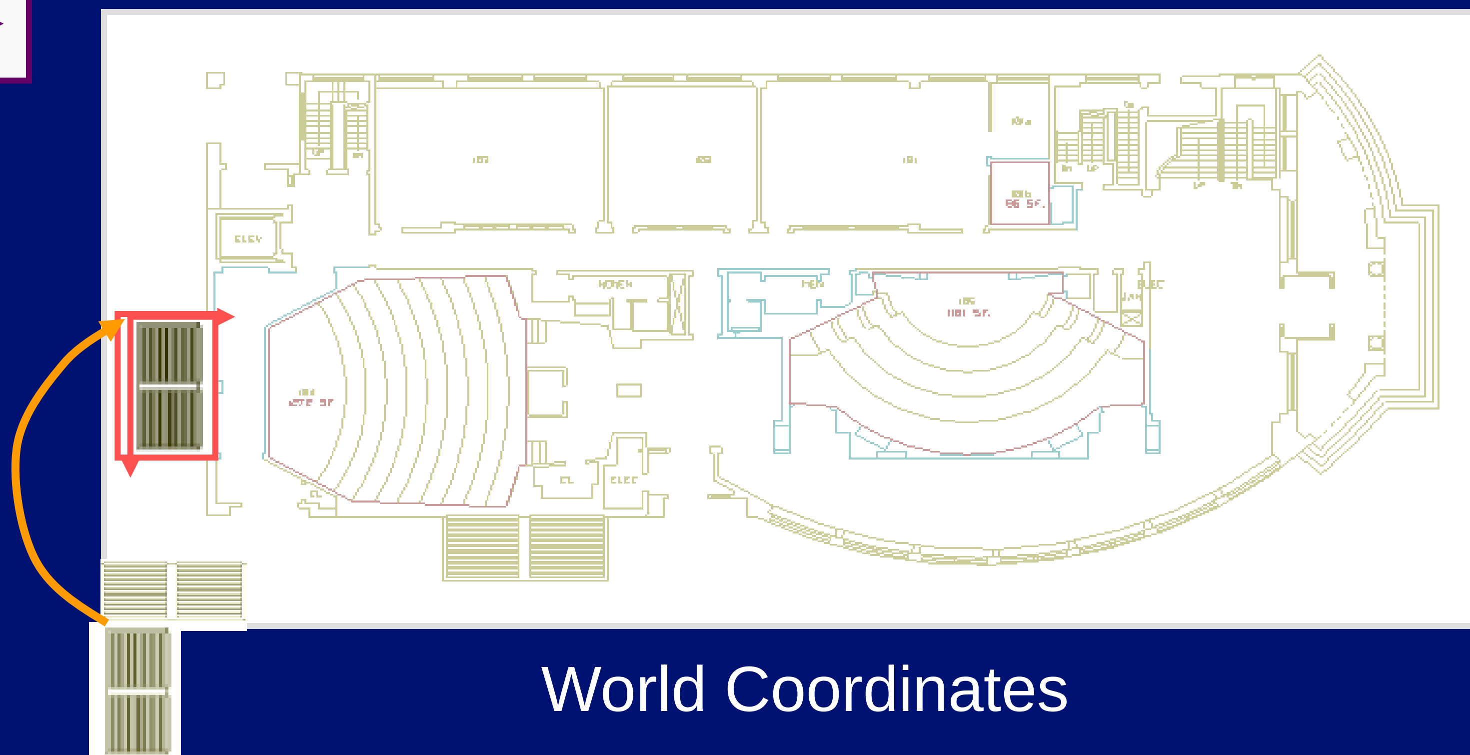


2D Modeling Transformations

Object Coordinates



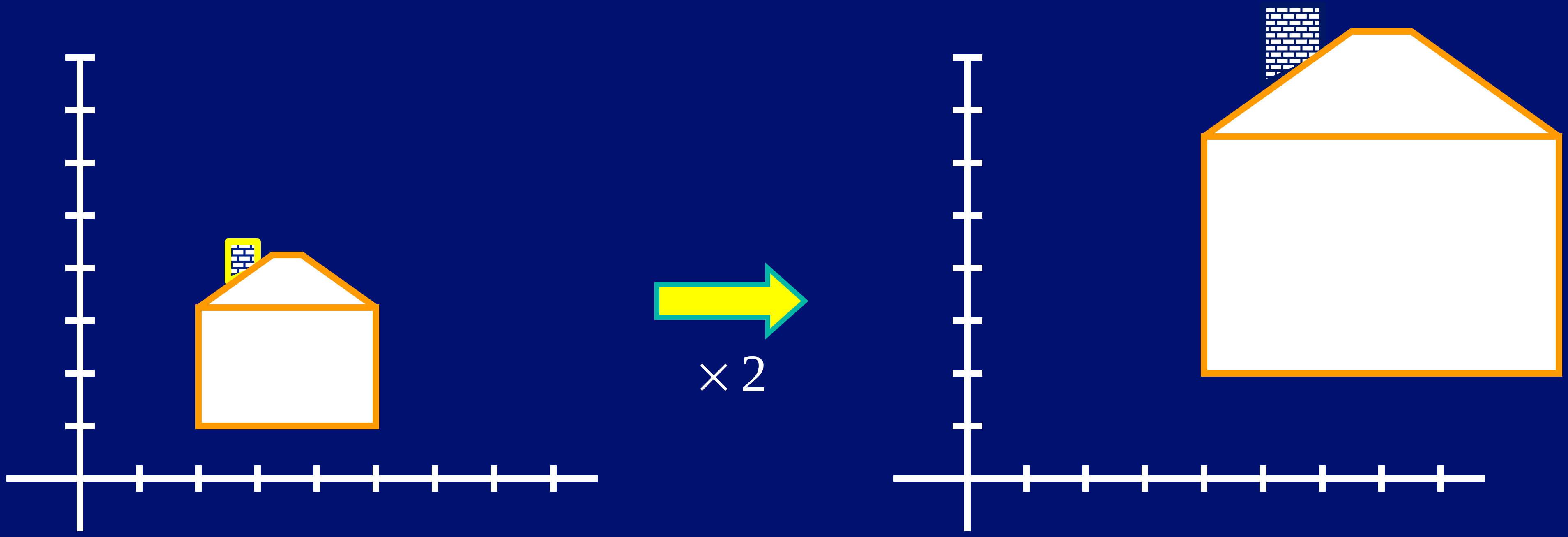
Scale .3, .3
Rotate -90
Translate 5, 3



World Coordinates

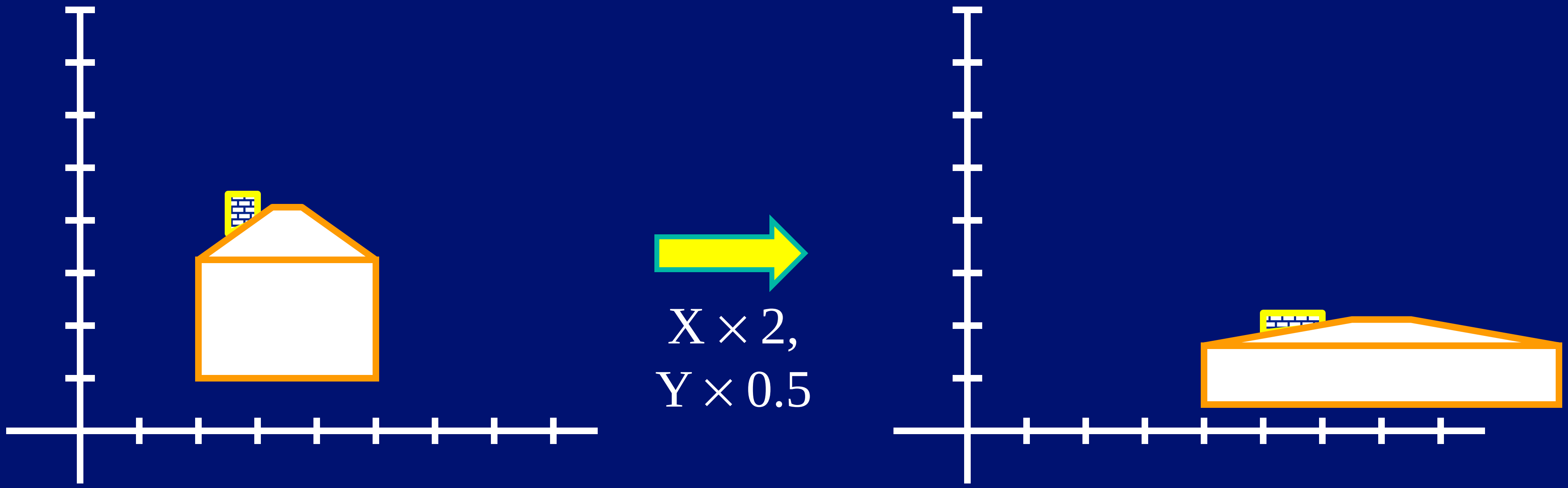
Scaling

- **Scaling** *means multiplying each coordinate by a scalar*
- **Uniform scaling** *means this scalar is the same for all coordinates:*



Scaling

Non-uniform scaling***: different scalars per coordinate:***



How can we represent this in matrix form?

Scaling

Scaling operation:

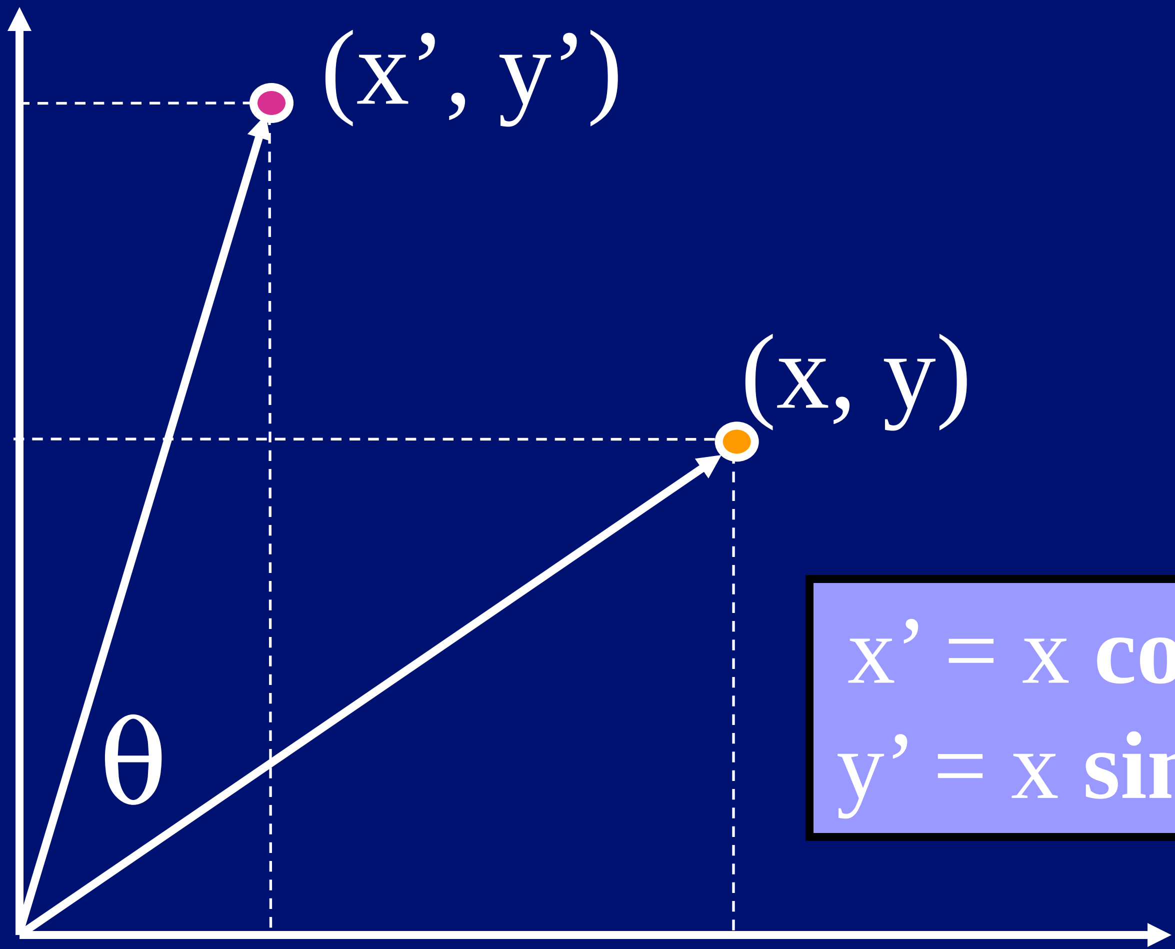
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} ax \\ by \end{bmatrix}$$

Or, in matrix form:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \underbrace{\begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}}_{\text{scaling matrix}} \begin{bmatrix} x \\ y \end{bmatrix}$$

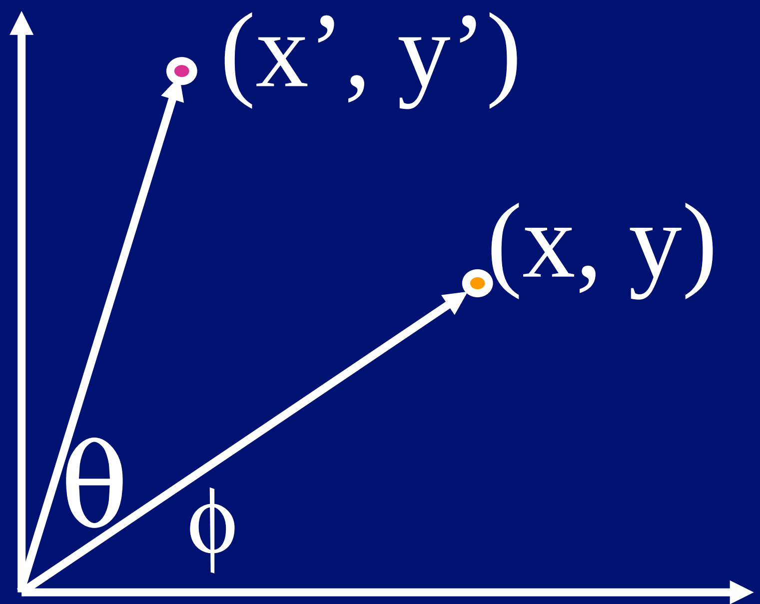
scaling matrix

2-D Rotation



$$x' = x\,\mathbf{cos}(\theta) - y\,\mathbf{sin}(\theta)$$
$$y' = x\,\mathbf{sin}(\theta) + y\,\mathbf{cos}(\theta)$$

2-D Rotation



$x = r \cos(\phi)$

$y = r \sin(\phi)$

$x' = r \cos(\phi + \theta)$

$y' = r \sin(\phi + \theta)$

Trig. Identities…

$x' = r \cos(\phi) \cos(\theta) - r \sin(\phi) \sin(\theta)$

$y' = r \sin(\phi) \sin(\theta) + r \cos(\phi) \cos(\theta)$

Substitute…

$x' = x\ \mathbf{cos}(\theta) - y\ \mathbf{sin}(\theta)$

$y' = x\ \mathbf{sin}(\theta) + y\ \mathbf{cos}(\theta)$

2-D Rotation

This is easy to capture in matrix form:

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

Even though sin(θ) and cos(θ) are nonlinear transcendental functions of θ, for fixed θ:

- ***x' is a linear combination of x and y***
- ***y' is a linear combination of x and y***

Basic 2D Transformations

Translation:

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

Scale:

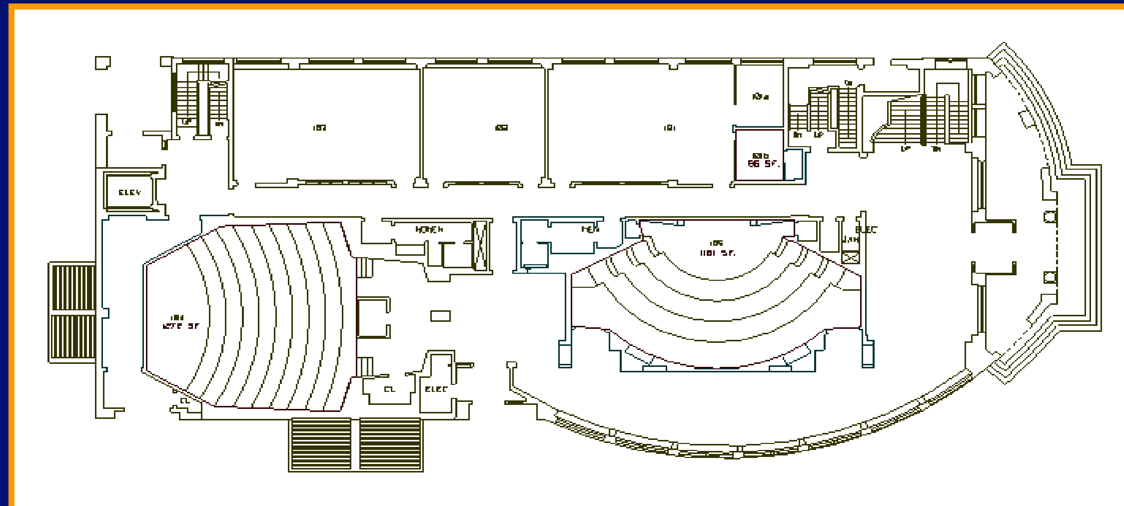
- $x' = x * s_x$
- $y' = y * s_y$

Shear:

- $x' = x + h_x*y$
- $y' = y + h_y*x$

Rotation:

- $x' = x*\cos\Theta - y*\sin\Theta$
- $y' = x*\sin\Theta + y*\cos\Theta$



Transformations
can be combined
(with simple algebra)

Basic 2D Transformations

Translation:

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

Scale:

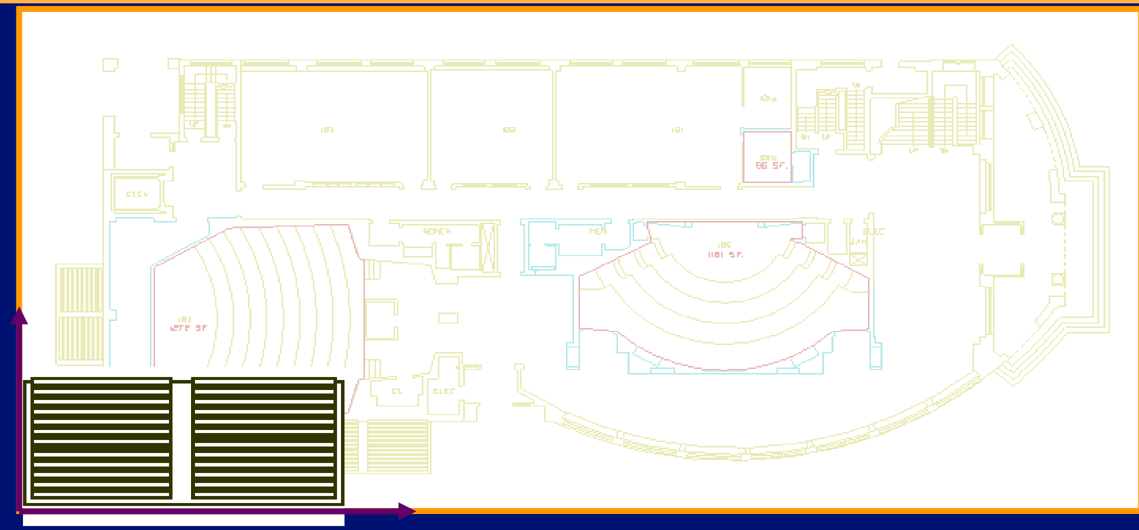
- $x' = x * s_x$
- $y' = y * s_y$

Shear:

- $x' = x + h_x * y$
- $y' = y + h_y * x$

Rotation:

- $x' = x * \cos\Theta - y * \sin\Theta$
- $y' = x * \sin\Theta + y * \cos\Theta$



Basic 2D Transformations

Translation:

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

Scale:

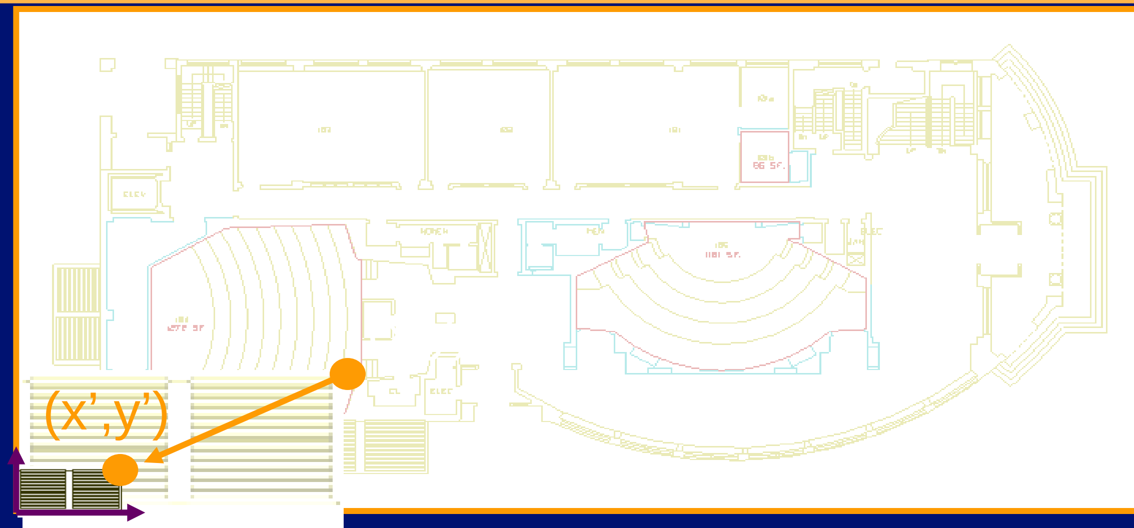
- $x' = x * s_x$
- $y' = y * s_y$

Shear:

- $x' = x + h_x*y$
- $y' = y + h_y*x$

Rotation:

- $x' = x*\cos\Theta - y*\sin\Theta$
- $y' = x*\sin\Theta + y*\cos\Theta$



$x' = x*s_x$
$y' = y*s_y$

Basic 2D Transformations

Translation:

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

Scale:

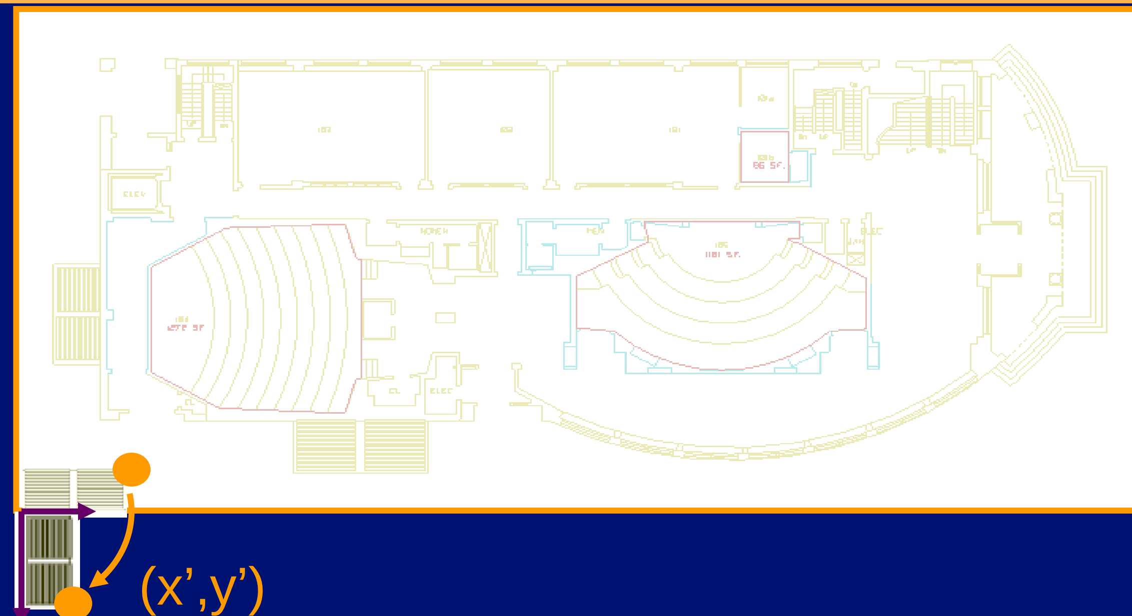
- $x' = x * s_x$
- $y' = y * s_y$

Shear:

- $x' = x + h_x*y$
- $y' = y + h_y*x$

Rotation:

- $x' = x*\cos\Theta - y*\sin\Theta$
- $y' = x*\sin\Theta + y*\cos\Theta$



$$x' = (x*s_x)*\cos\Theta - (y*s_y)*\sin\Theta$$
$$y' = (x*s_x)*\sin\Theta + (y*s_y)*\cos\Theta$$

Basic 2D Transformations

Translation:

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

Scale:

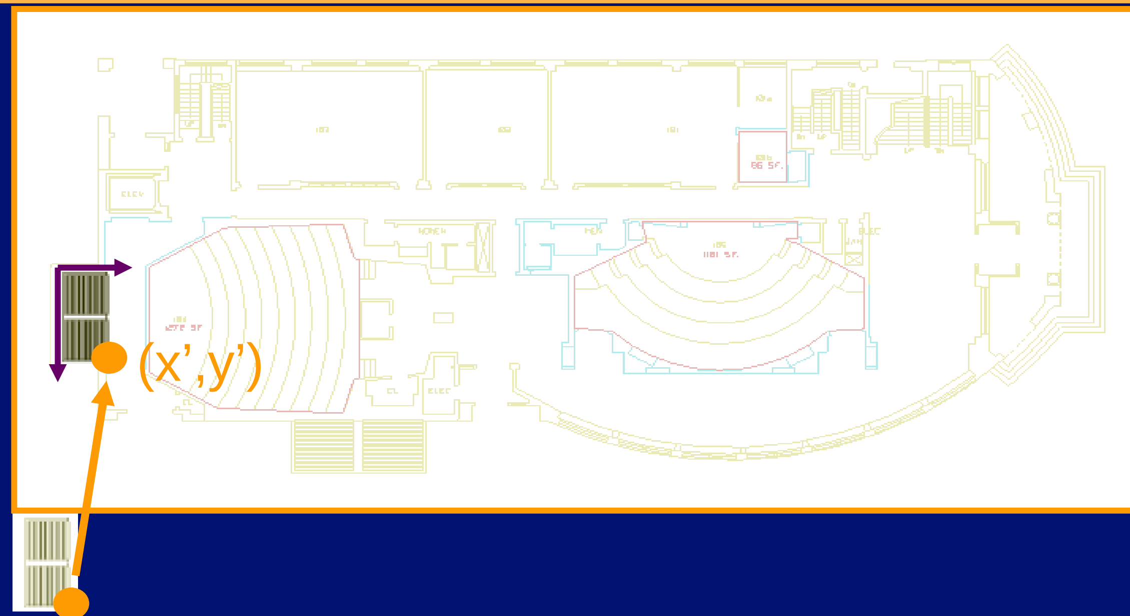
- $x' = x * s_x$
- $y' = y * s_y$

Shear:

- $x' = x + h_x*y$
- $y' = y + h_y*x$

Rotation:

- $x' = x*\cos\Theta - y*\sin\Theta$
- $y' = x*\sin\Theta + y*\cos\Theta$



$$x' = ((x*s_x)*\cos\Theta - (y*s_y)*\sin\Theta) + t_x$$
$$y' = ((x*s_x)*\sin\Theta + (y*s_y)*\cos\Theta) + t_y$$

Basic 2D Transformations

Translation:

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

Scale:

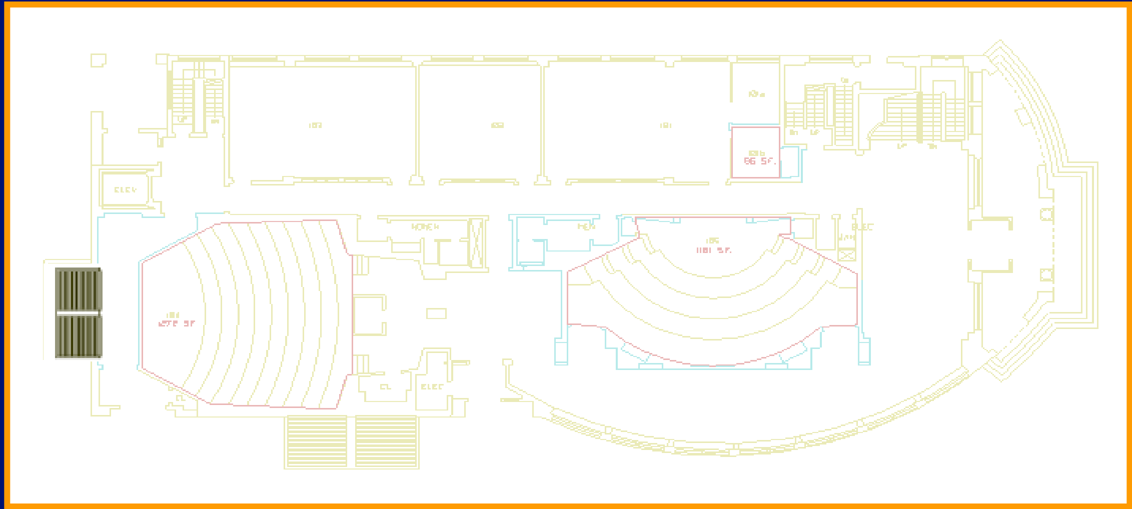
- $x' = x * s_x$
- $y' = y * s_y$

Shear:

- $x' = x + h_x * y$
- $y' = y + h_y * x$

Rotation:

- $x' = x * \cos\Theta - y * \sin\Theta$
- $y' = x * \sin\Theta + y * \cos\Theta$



$$x' = ((x * s_x) * \cos\Theta - (y * s_y) * \sin\Theta) + t_x$$
$$y' = ((x * s_x) * \sin\Theta + (y * s_y) * \cos\Theta) + t_y$$

Overview

2D Transformations

- Basic 2D transformations
- Matrix representation
- Matrix composition

3D Transformations

- Basic 3D transformations
- Same as 2D

Matrix Representation

Represent points as column vectors and 2D transformations as matrices

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

Left multiply matrix by column vector
⇔ apply transformation to point

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$$x' = ax + by$$
$$y' = cx + dy$$

Matrix Representation

Transformations composed by matrix multiplications

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} e & f \\ g & h \end{bmatrix} \begin{bmatrix} i & j \\ k & l \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Matrices are a convenient and efficient way to represent a sequence of transformations

2x2 Matrices

What types of transformations can be represented with a 2x2 matrix?

2D Identity?

$$x' = x$$
$$y' = y$$

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

2D Scale around (0,0)?

$$x' = s_x * x$$
$$y' = s_y * y$$

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

2x2 Matrices

What types of transformations can be represented with a 2x2 matrix?

2D Rotate around (0,0)?

$$x' = \cos\Theta * x - \sin\Theta * y$$
$$y' = \sin\Theta * x + \cos\Theta * y$$

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

2D Shear?

$$x' = x + sh_x * y$$
$$y' = sh_y * x + y$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & sh_x \\ sh_y & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

2x2 Matrices

What types of transformations can be represented with a 2x2 matrix?

2D Mirror about the Y axis?

$$x' = -x \\ y' = y$$

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

2D Mirror about the origin (0,0)?

$$x' = -x \\ y' = -y$$

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

2x2 Matrices

What types of transformations can be represented with a 2x2 matrix?

2D Translation?

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

NO!

Only linear 2D transformations can be represented with a 2x2 matrix

Linear Transformations

Linear transformations are combinations of …

- Scale,
- Rotation,
- Shear, and
- Mirror

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Properties of linear transformations:

- Satisfies: $T(s_1\mathbf{p}_1 + s_2\mathbf{p}_2) = s_1T(\mathbf{p}_1) + s_2T(\mathbf{p}_2)$
- Origin maps to origin
- Lines map to lines
- Parallel lines remain parallel
- Ratios are preserved
- Closed under composition

Homogeneous Coordinates

Q: How can we represent translation as a 3x3 matrix?

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

Homogeneous Coordinates

Homogeneous coordinates

- represent coordinates in 2 dimensions with a 3-vector

$$\begin{bmatrix} x \\ y \end{bmatrix} \xrightarrow{\text{homogeneous coords}} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Homogeneous coordinates seem unintuitive, but they make many operations in graphics, robotics, and vision much easier

Homogeneous Coordinates

More generally, 2D points are given by three coordinates

$$p = \begin{bmatrix} x \\ y \\ w \end{bmatrix} \qquad X = \frac{x}{w}, Y = \frac{y}{w}$$

but with the convention

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} \lambda x \\ \lambda y \\ \lambda w \end{bmatrix}, \lambda \neq 0$$

Homogeneous Coordinates

Q: How can we represent translation as a 3x3 matrix?

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

A: Using the rightmost column:

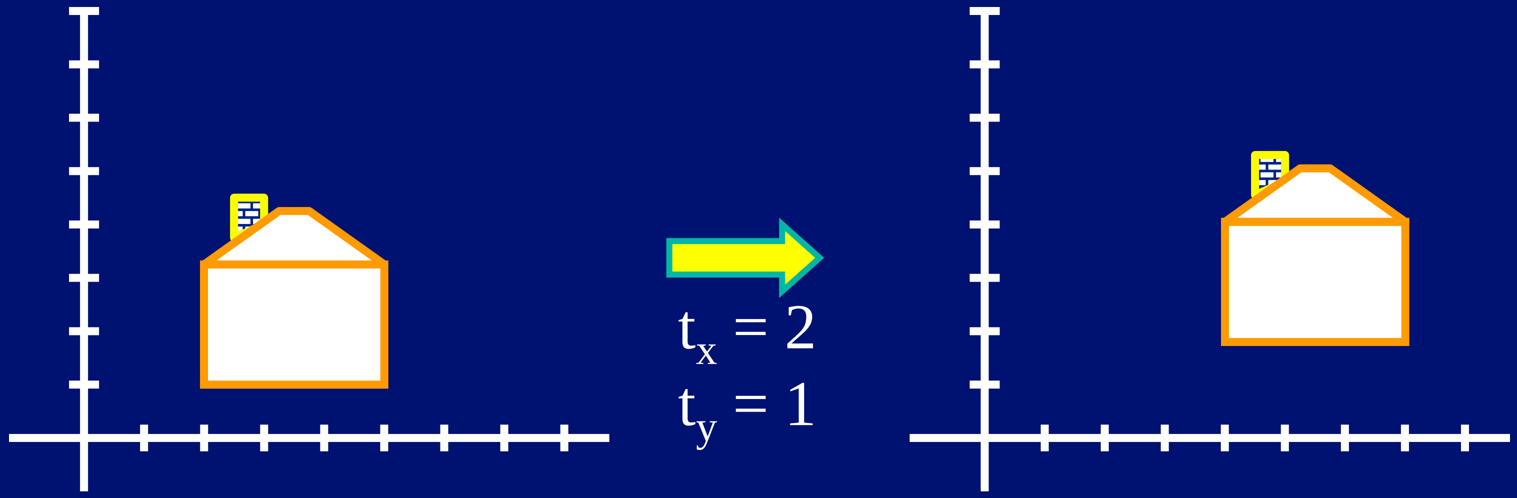
$$\mathbf{Translation} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

Translation

Example of translation

Homogeneous Coordinates

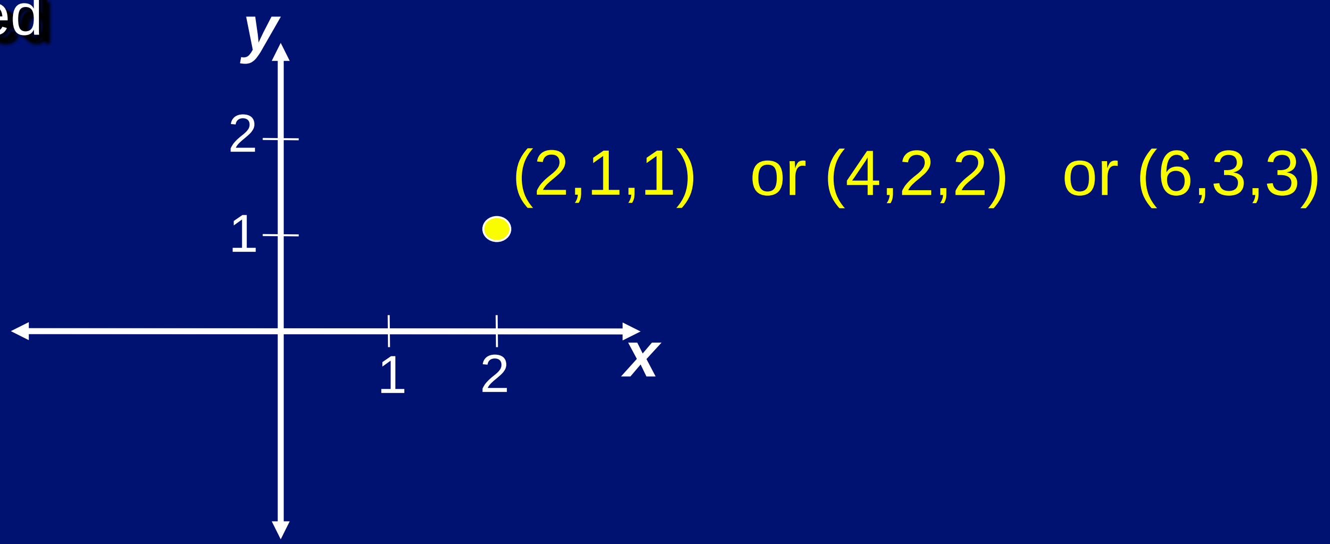
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x + t_x \\ y + t_y \\ 1 \end{bmatrix}$$



Homogeneous Coordinates

Add a 3rd coordinate to every 2D point

- (x, y, w) represents a point at location (x/w, y/w)
- (x, y, 0) represents a point at infinity
- (0, 0, 0) is not allowed



Convenient coordinate system to represent many useful transformations

Basic 2D Transformations

Basic 2D transformations as 3x3 matrices

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

Translate

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

Scale

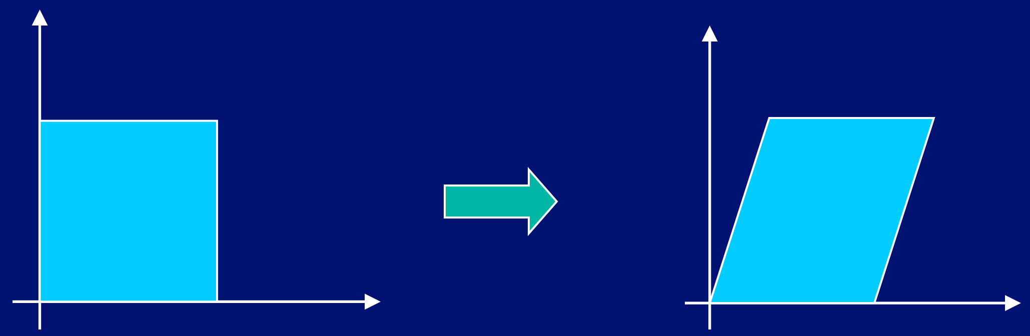
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\Theta & -\sin\Theta & 0 \\ \sin\Theta & \cos\Theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Rotate

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

Shear

x-Shearing



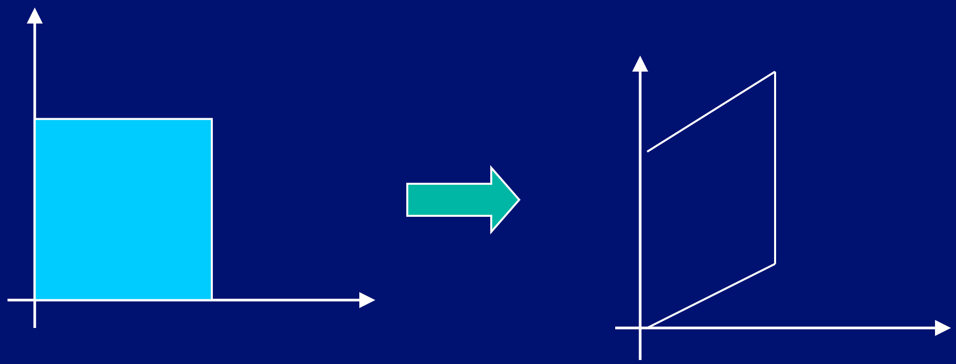
Y coordinates are unaffected, but x coordinates are translated linearly with y

That is:

- y’ = y
- x’ = x + y * h

$$\begin{vmatrix} x \\ y \\ 1 \end{vmatrix} = \begin{vmatrix} 1 & h & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{vmatrix} * \begin{vmatrix} x \\ y \\ 1 \end{vmatrix}$$

y-Shearing



$$\begin{vmatrix} x \\ y \\ 1 \end{vmatrix} = \begin{vmatrix} 1 & 0 & 0 \\ g & 1 & 0 \\ 0 & 0 & 1 \end{vmatrix} * \begin{vmatrix} x \\ y \\ 1 \end{vmatrix}$$

Interesting Facts:

- A 2D rotation can be expressed by a combination of shears
- Any 2D shearing can be expressed by rotations and scales
- Shearing will not change the area of the object

Affine Transformations

Affine transformations are combinations of …

- Linear transformations, and
- Translations

$$\begin{bmatrix} x' \\ y' \\ w \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

Properties of affine transformations:

- Origin does not necessarily map to origin
- Lines map to lines
- Parallel lines remain parallel
- Ratios are preserved
- Closed under composition

Projective Transformations

Projective transformations …

- Affine transformations, and
- Projective warps

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

Properties of projective transformations:

- Origin does not necessarily map to origin
- Lines map to lines
- Parallel lines do not necessarily remain parallel
- Ratios are not preserved (but cross-ratios are)
- Closed under composition

Overview

2D Transformations

- Basic 2D transformations
- Matrix representation
- Matrix composition

3D Transformations

- Basic 3D transformations
- Same as 2D

Matrix Composition

Transformations can be combined by matrix multiplication

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \left(\begin{bmatrix} 1 & 0 & tx \\ 0 & 1 & ty \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\Theta & -\sin\Theta & 0 \\ \sin\Theta & \cos\Theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} sx & 0 & 0 \\ 0 & sy & 0 \\ 0 & 0 & 1 \end{bmatrix} \right) \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

p' = $T(t_x,t_y)$ $R(\Theta)$ $S(s_x,s_y)$ **p**

Matrix Composition

Matrices are a convenient and efficient way to represent a sequence of transformations

- General purpose representation
- Graphics hardware matrix multiply

$$\mathbf{p}' = (T * (R * (S*\mathbf{p})\ )\ )$$
$$\mathbf{p}' = (T*R*S) * \mathbf{p}$$

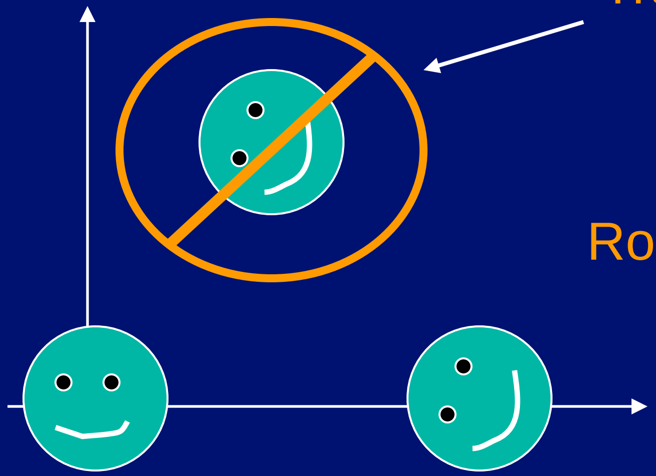
$$\mathbf{p}' = T * R * S * \mathbf{p}$$

"Global" ⟷ "Local"

Matrix composition is associative

Order of Transformations Matters

Example: rotation and translation do not commute



Translate (5,0) and then Rotate 60 degree

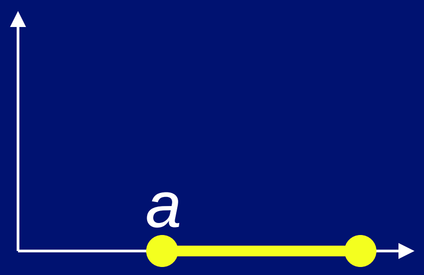
OR

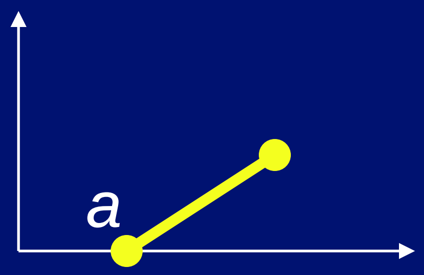
Rotate 60 degree and then translate (5,0)??

Matrix Composition

What if we want to rotate about a general point?

- Ex: Rotate line segment by 45 degrees about endpoint *a*

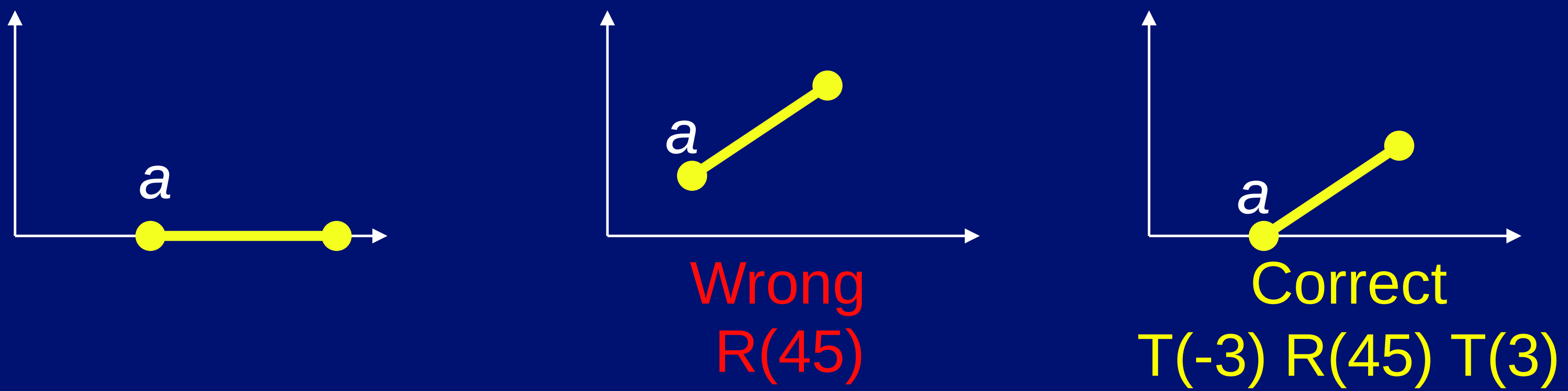




Multiplication Order – Wrong Way

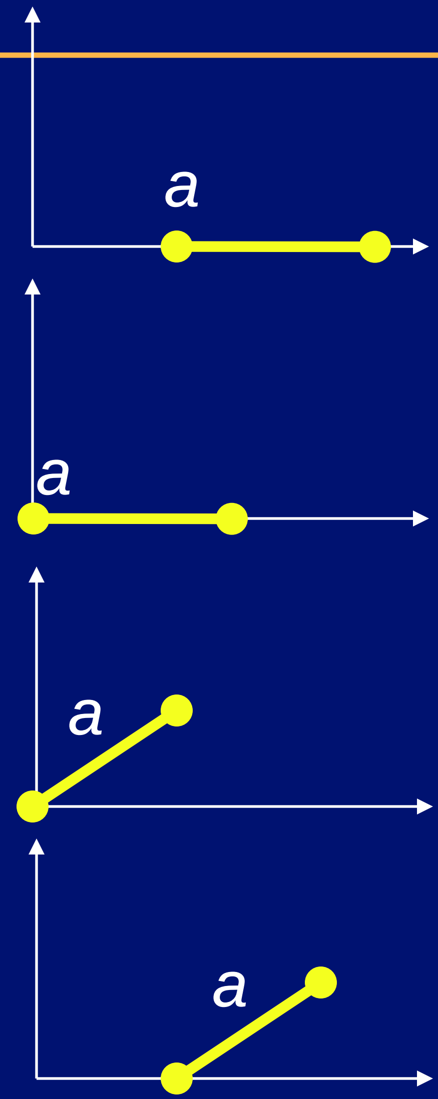
Our line is defined by two endpoints

- Applying a rotation of 45 degrees, R(45), affects both points
- We could try to translate both endpoints to return endpoint *a* to its original position, but by how much?



Multiplication Order - Correct

Isolate endpoint *a* from rotation effects



- First translate line so *a* is at origin: T (-3)
- Then rotate line 45 degrees: R(45)
- Then translate back so *a* is where it was: T(3)

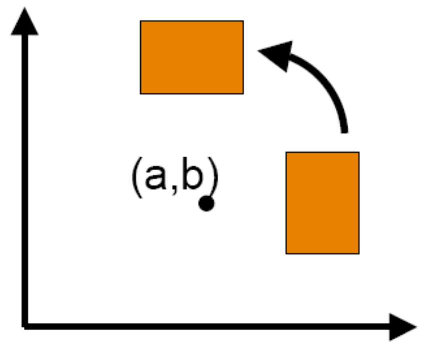
Matrix Composition

Will this sequence of operations work?

$$\begin{bmatrix} 1 & 0 & -3 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(45) & -\sin(45) & 0 \\ \sin(45) & \cos(45) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 3 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} a_x \\ a_y \\ 1 \end{bmatrix} = \begin{bmatrix} a'_x \\ a'_y \\ 1 \end{bmatrix}$$

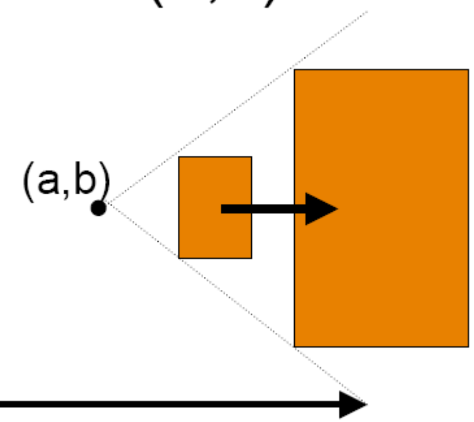
Conjugation: ABA^{-1}

- Rotate by Θ around arbitrary point (a,b)
  - M=T(a,b) * R(Θ) * T(-a,-b)



The trick:
First, translate (a,b) to the origin.
Next, do the rotation about origin.
Finally, translate back.

- Scale by sx,sy around arbitrary point (a,b)
  - M=T(a,b) * S(sx,sy) * T(-a,-b)



(Use the same trick.)

Matrix Composition

After correctly ordering the matrices

Multiply matrices together

*What results is one matrix – **store it (on stack)!***

Multiply this matrix by the vector of each vertex

All vertices easily transformed with one matrix multiply

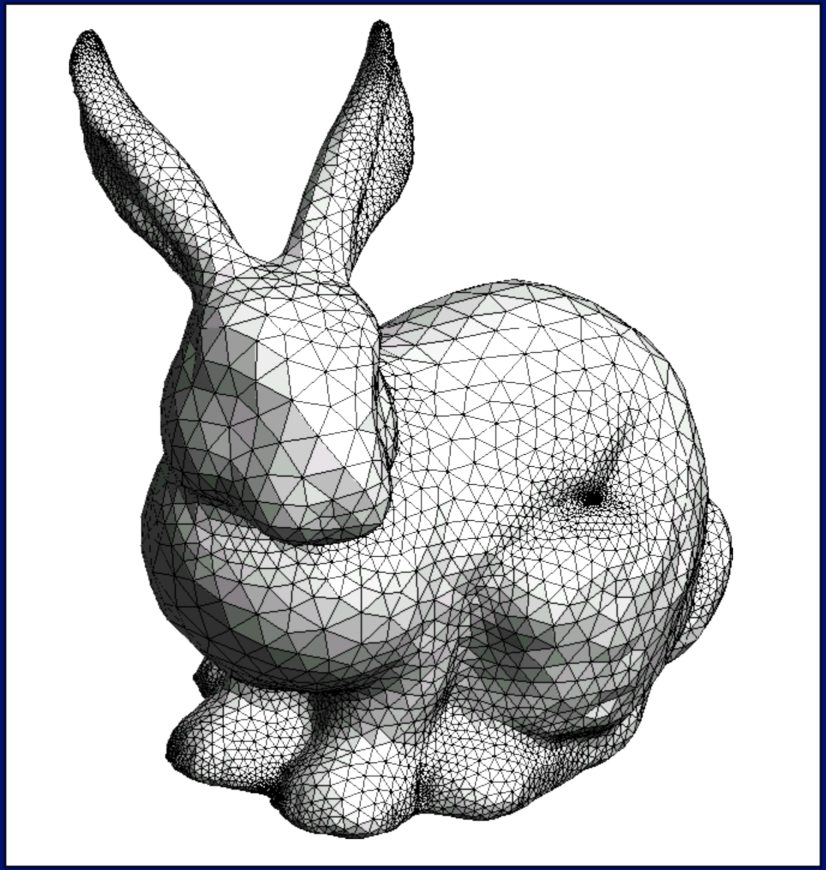
Overview

2D Transformations

- Basic 2D transformations
- Matrix representation
- Matrix composition

3D Transformations

- Basic 3D transformations
- Same as 2D



3D Transformations

Same idea as 2D transformations

- Homogeneous coordinates: (x,y,z,w)
- 4x4 transformation matrices

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

Basic 3D Transformations

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

Identity

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

Scale

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

Translation

$$\begin{bmatrix} x' \\ y' \\ z' \\ w \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix}$$

Mirror about Y-Z plane

Basic 3D Transformations

Rotate around Z axis:

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

Rotate around Y axis:

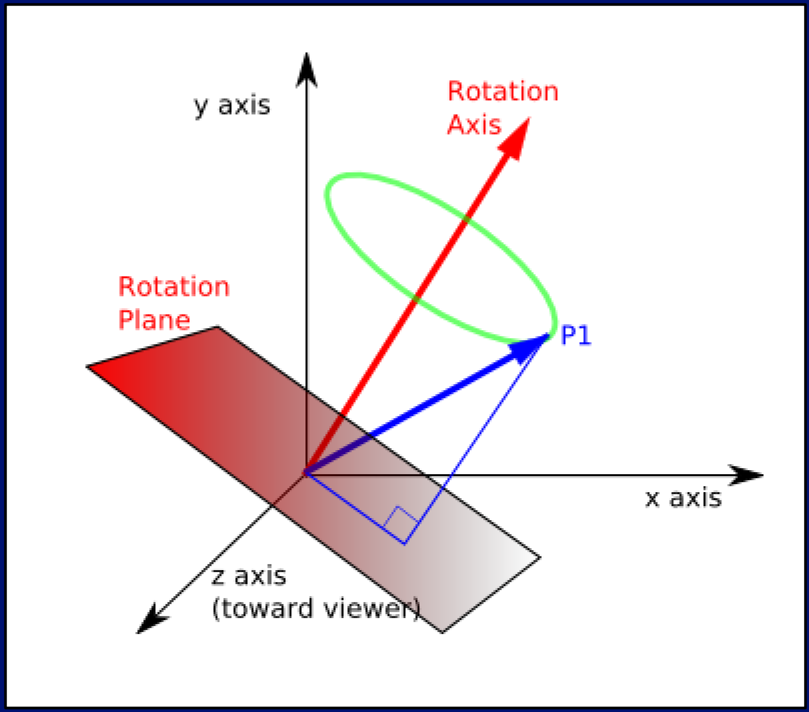
$$\begin{bmatrix} \boldsymbol{x}' \\ \boldsymbol{y}' \\ \boldsymbol{z}' \\ \boldsymbol{w} \end{bmatrix} = \begin{bmatrix} \cos\Theta & 0 & \sin\Theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\Theta & 0 & \cos\Theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \\ \boldsymbol{z} \\ \boldsymbol{w} \end{bmatrix}$$

Rotate around X axis:

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

General 3D Rotations

These require some more study – next time



Inverse Rotations

Q: How do you undo a rotation of θ, $R(\theta)$?

A: Apply the inverse of the rotation… $R^{-1}(\theta) = R(-\theta)$

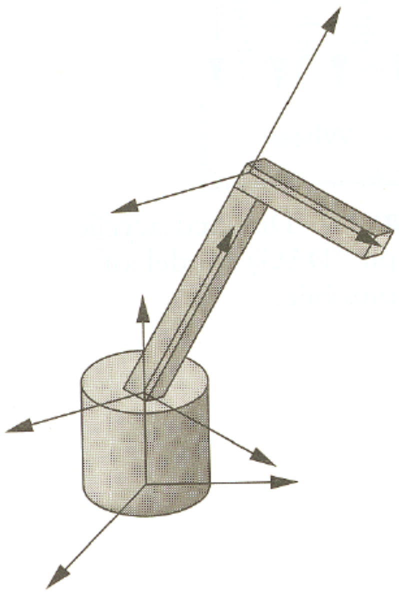
How to construct $R^{-1}(\theta) = R(-\theta)$

- Inside the rotation matrix: $\cos(\theta) = \cos(-\theta)$
  - *The cosine elements of the inverse rotation matrix are unchanged*
- The sign of the sine elements will flip

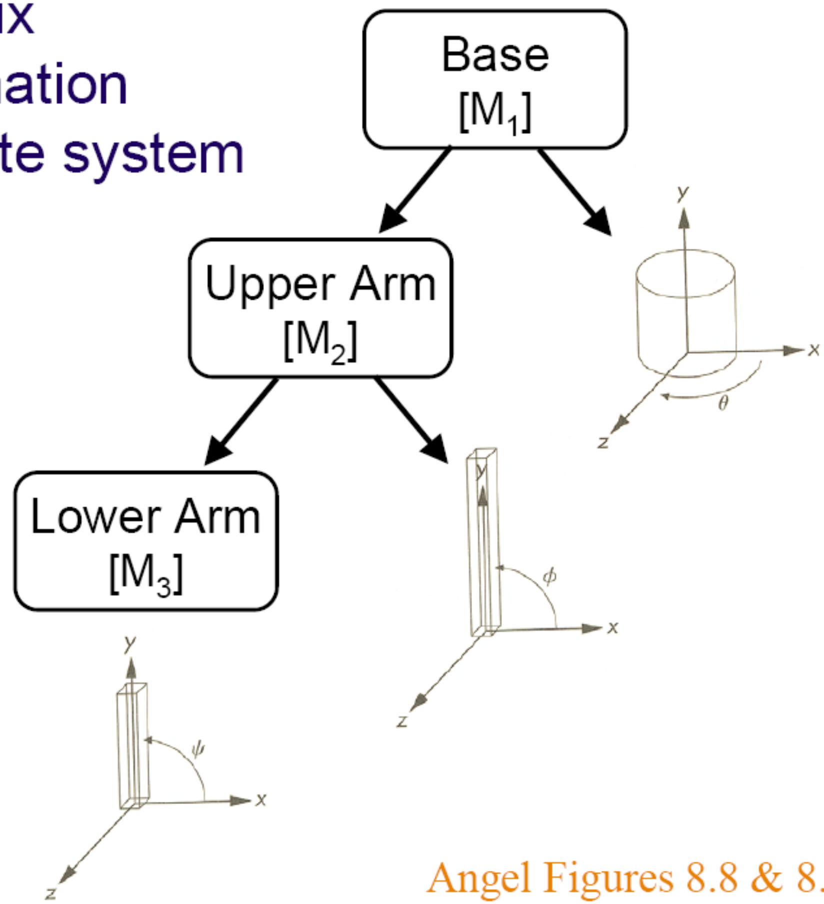
Therefore… $R^{-1}(\theta) = R(-\theta) = R^T(\theta)$

Transformation Hierarchies

- Scene may have hierarchy of coordinate systems
  - Each level stores matrix representing transformation from parent's coordinate system



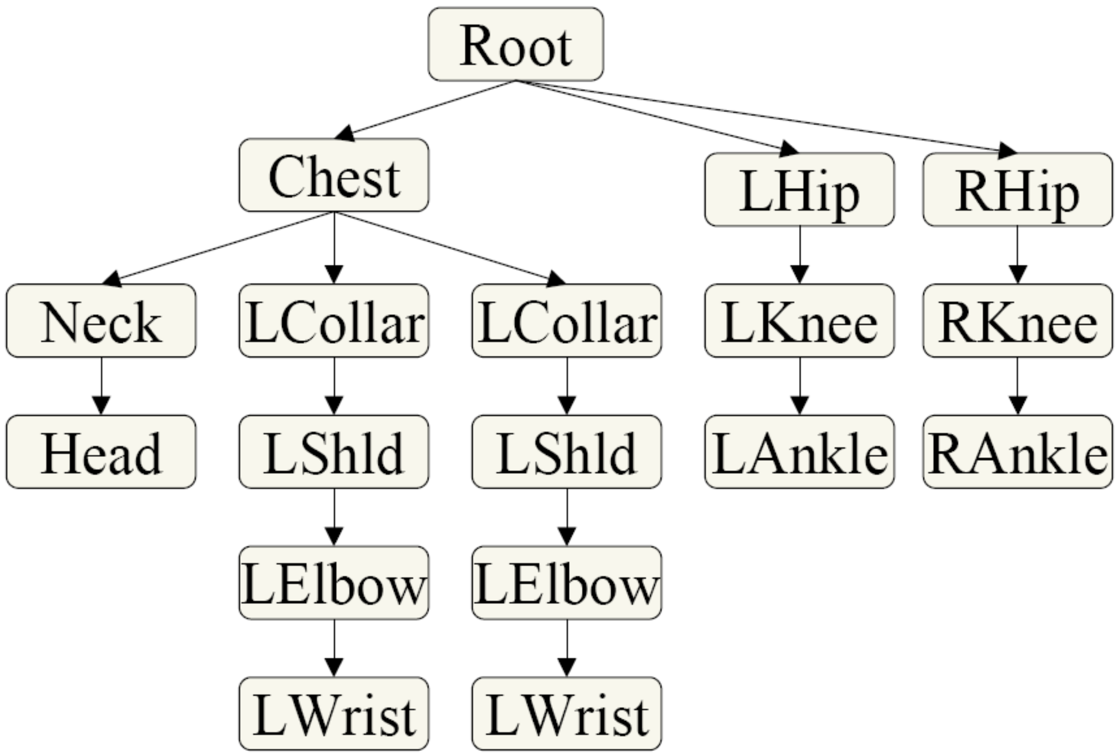
Robot Arm

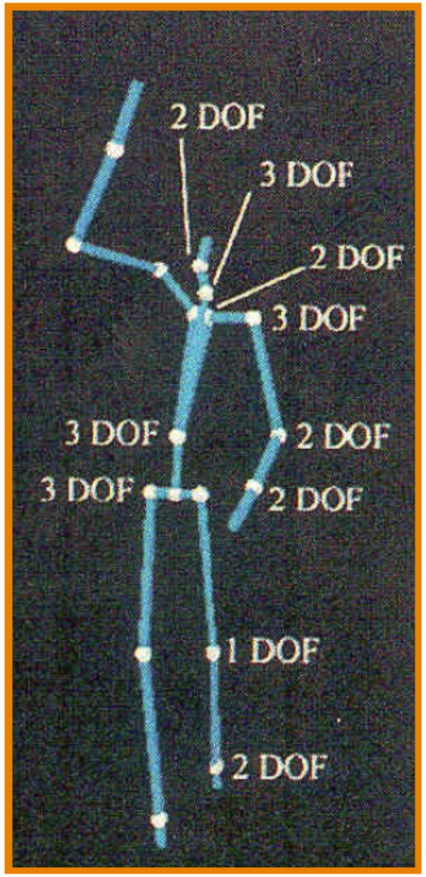


Angel Figures 8.8 & 8.9

Hierarchy Example 1: Articulated Bodies

- Well-suited for humanoid characters

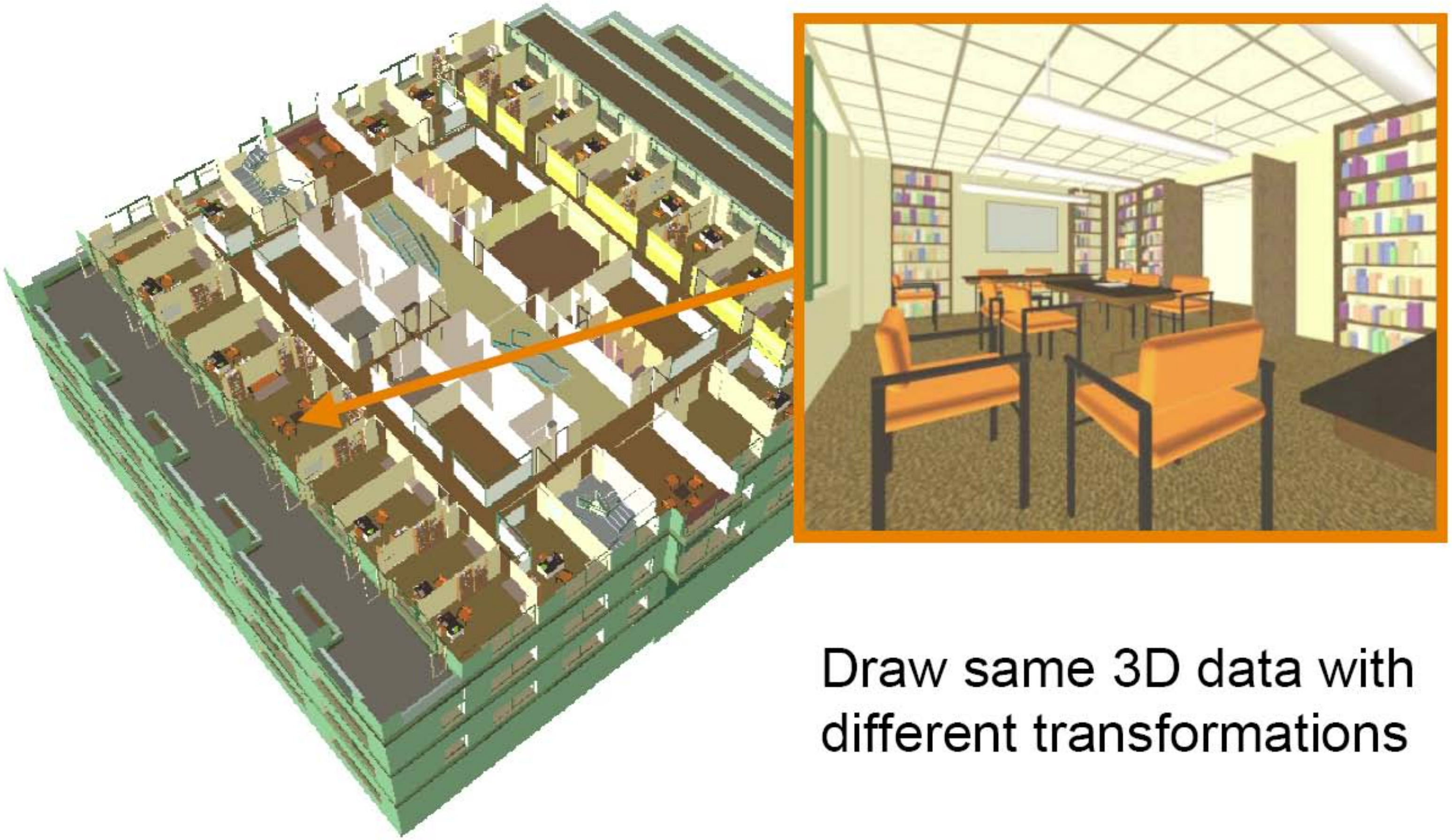




Rose et al. `96

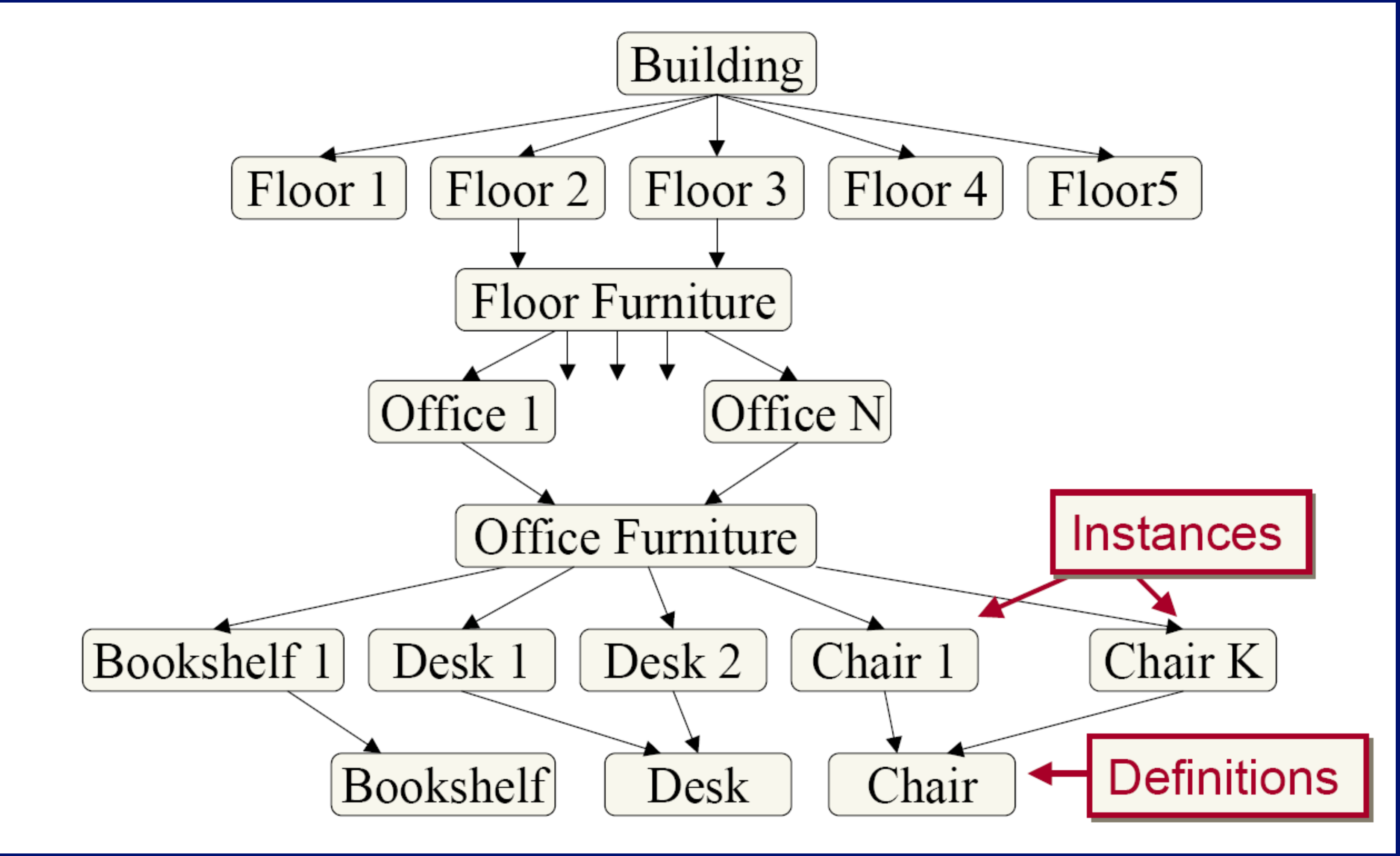
Hierarchy Example 2: Architecture

- An object may appear in a scene multiple times



Draw same 3D data with different transformations

Hierarchy Example 2: Architecture



Summary

Coordinate systems

- World vs. modeling coordinates

2-D and 3-D transformations

- Trigonometry and geometry
- Matrix representations
- Linear vs. affine transformations

Matrix operations

- Matrix composition