19CSE433 Computer Graphics & Visualization

Professional Elective 1

5^{th} Semester,2021-22 Odd

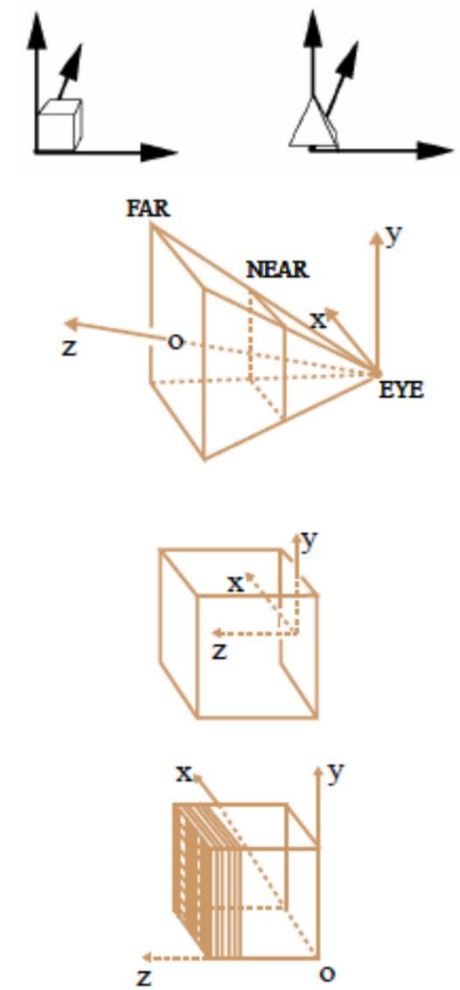
2019-22 Batch, BTech CSE

Dr.S.Padmavathi,

Department of Computer Science and Engineering,

Amrita School of Engineering, Coimbatore

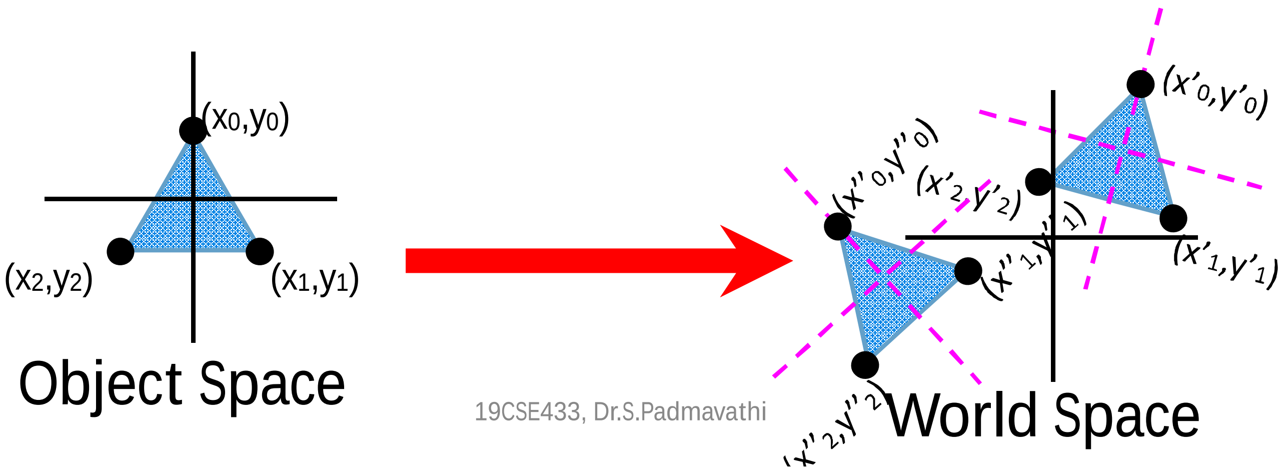
Common Coordinate Systems



- Object space
  - local to each object
- World space
  - common to all objects
- Eye space / Camera space
  - derived from view frustum
- Screen space
  - indexed according to hardware attributes

Object vs. World Space

- Makes building large models easier
- Example:



Transformations

What is a Transformation?

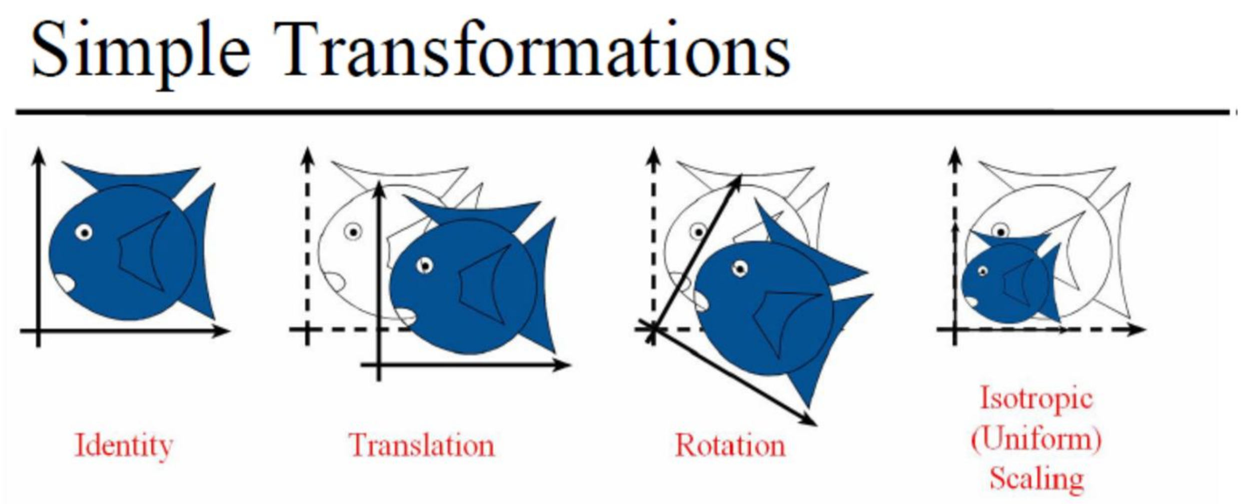
- Maps points (*x*, *y*) in one coordinate system to points (*x′*, *y′*) in another coordinate system

$x' = ax + by + c$

$y' = dx + ey + f$

Transformations are used:

- Position objects in a scene (modeling)
- Change the shape of objects
- Create multiple copies of objects
- Projection for virtual cameras
- Animations



Classes of Transformations

- Rigid Body / Euclidean Transforms: Preserves distances, Preserves angles

  Translation, Rotation
- Similitudes/ Similarity Transforms :Preserves angles
  - Translation, Rotation***, uniform scaling***
- Linear: Rotation, Scaling, Shear, Reflection
- Affine: preserves parallel lines

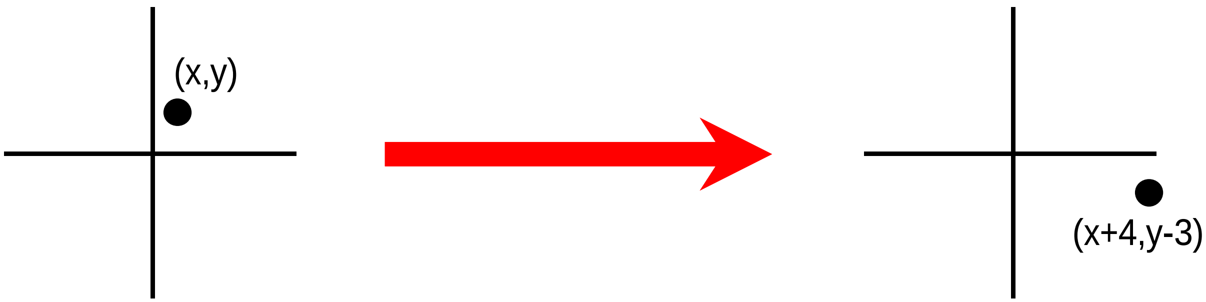
  Translation, Rotation, Scaling, Shear, Reflection
- Projective: preserves lines—Perspective transformation

Types of transformations.

- Rotation and translation
  - Angles and distances are preserved
  - Unit cube is always unit cube
  - *Rigid-Body* transformations.
- Rotation, translation and scale.
  - Angles & distances not preserved.
  - But parallel lines are.

Translation

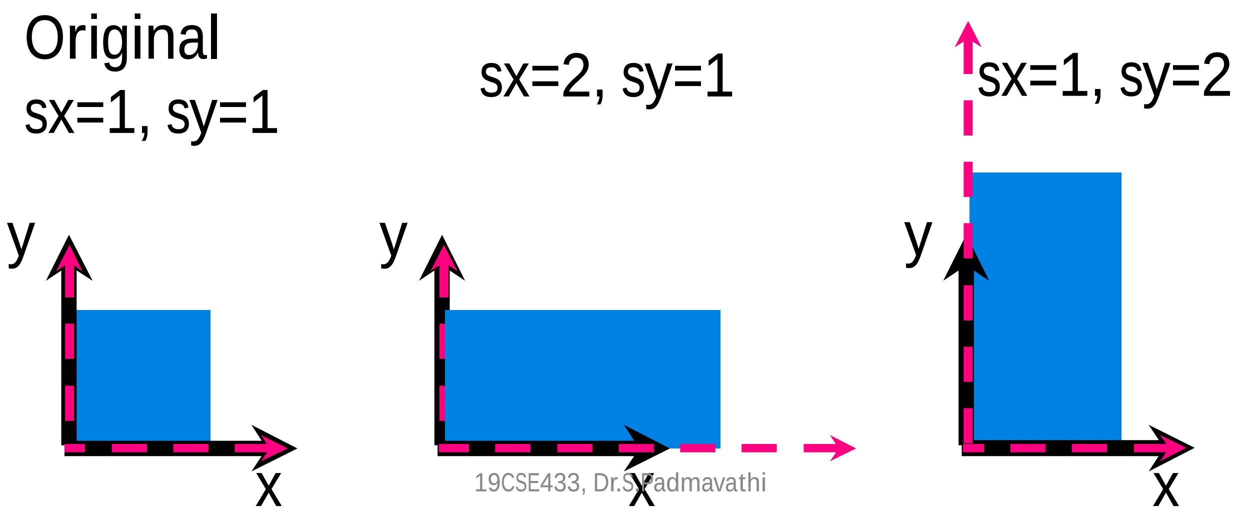
- Basically, just moving points
  - In 2D, up, down, left, or right
  - All points move in the same way
- For example, we may want to move all points 4 pixels to the right and 3 down: $x' = x+4$, $y' = y-3$
- In general new coordinates are $x' = x + tx$, $y' = y + ty$
- Up: positive ty, down: negative ty, left: negative tx, or right: positive tx



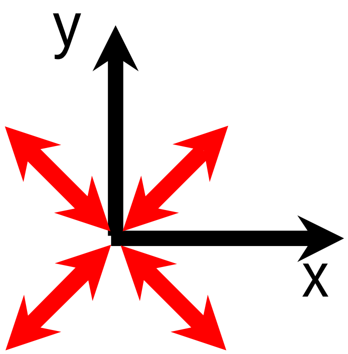
Object Space World Space

Scaling

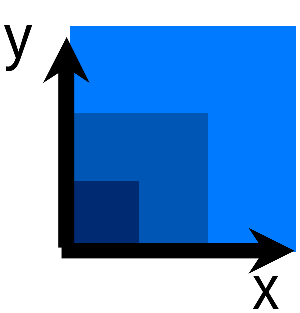
- Want to stretch or shrink the entire space in one or more dimensions.
- Scaling factors: sx = stretch in x, sy = stretch in y
- Integer values result in stretch and fractional values result in shrink
- New coordinates after scaling: x'= x . sx,   y' = y . sy

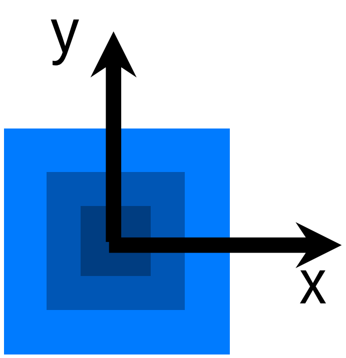


Scaling



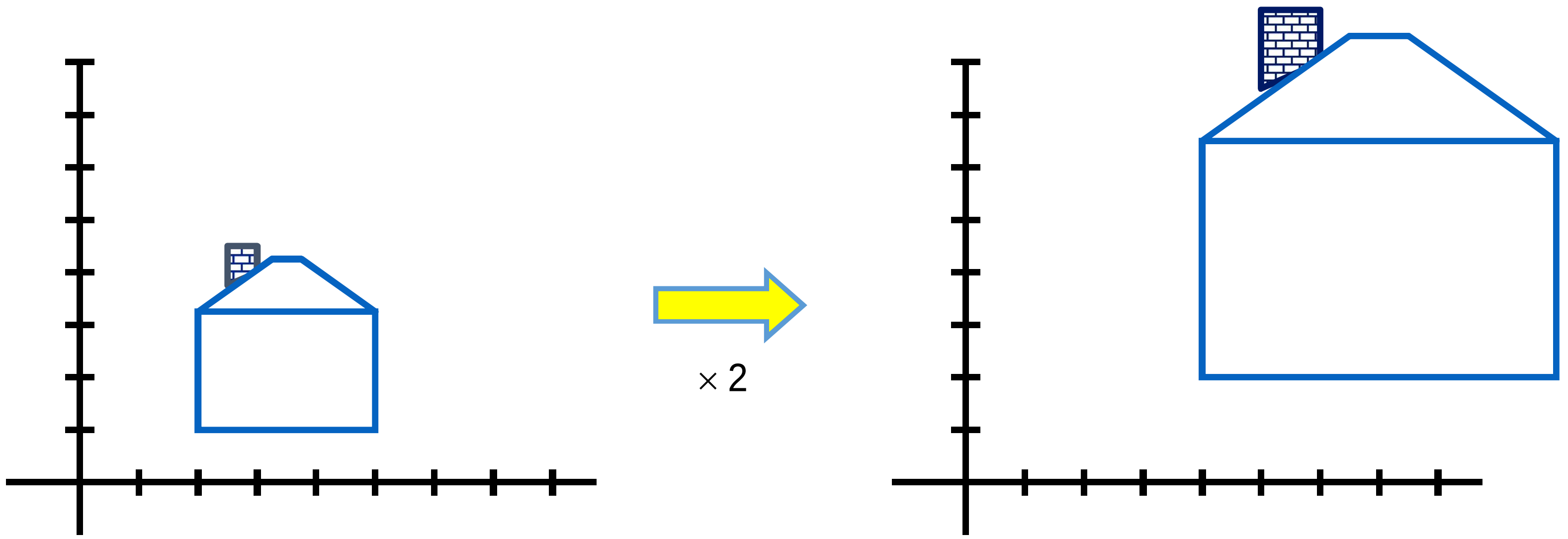
- Scaling is centered around the origin
  - Points either get pulled toward the origin or pushed away from it





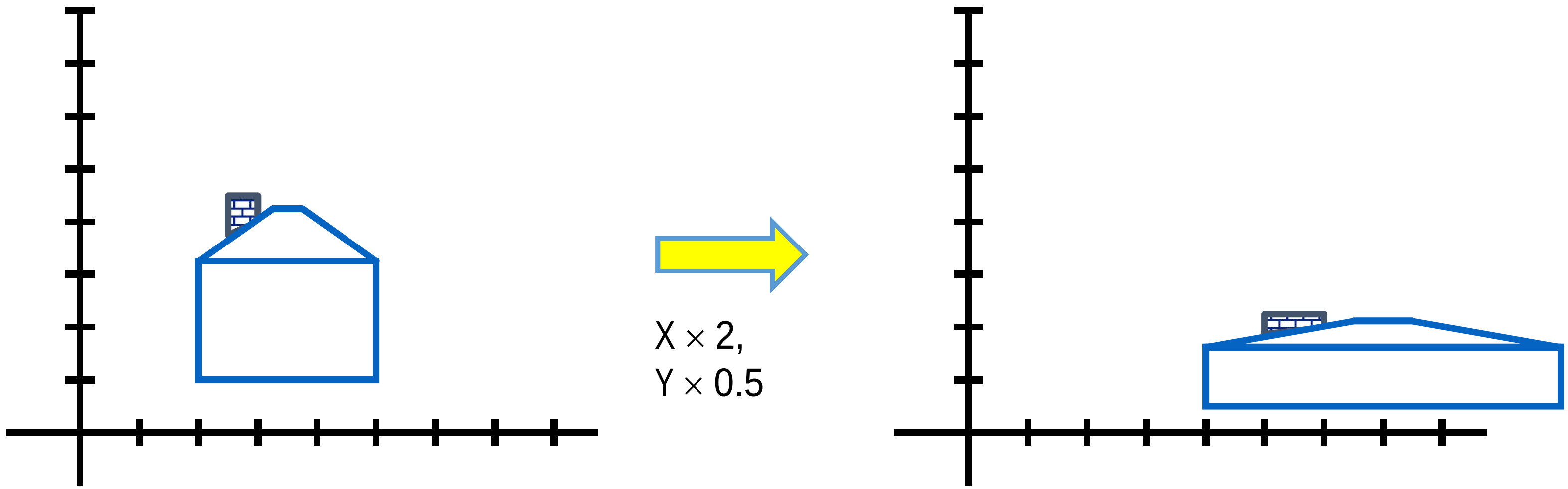
Scaling

- Scaling a coordinate means multiplying each of its components by a scalar
- Uniform scaling means this scalar is the same for all components:



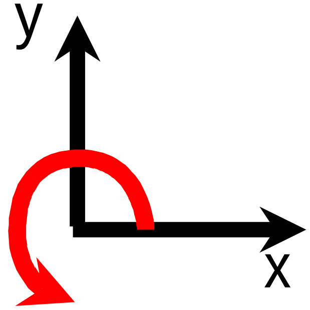
Scaling

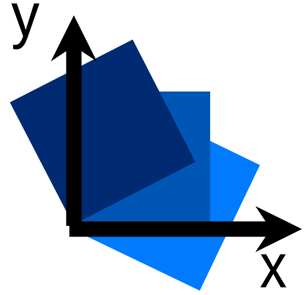
- Non-uniform scaling: different scalars per component:

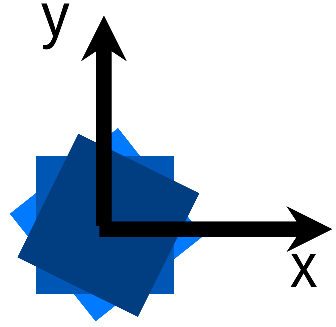


Rotation

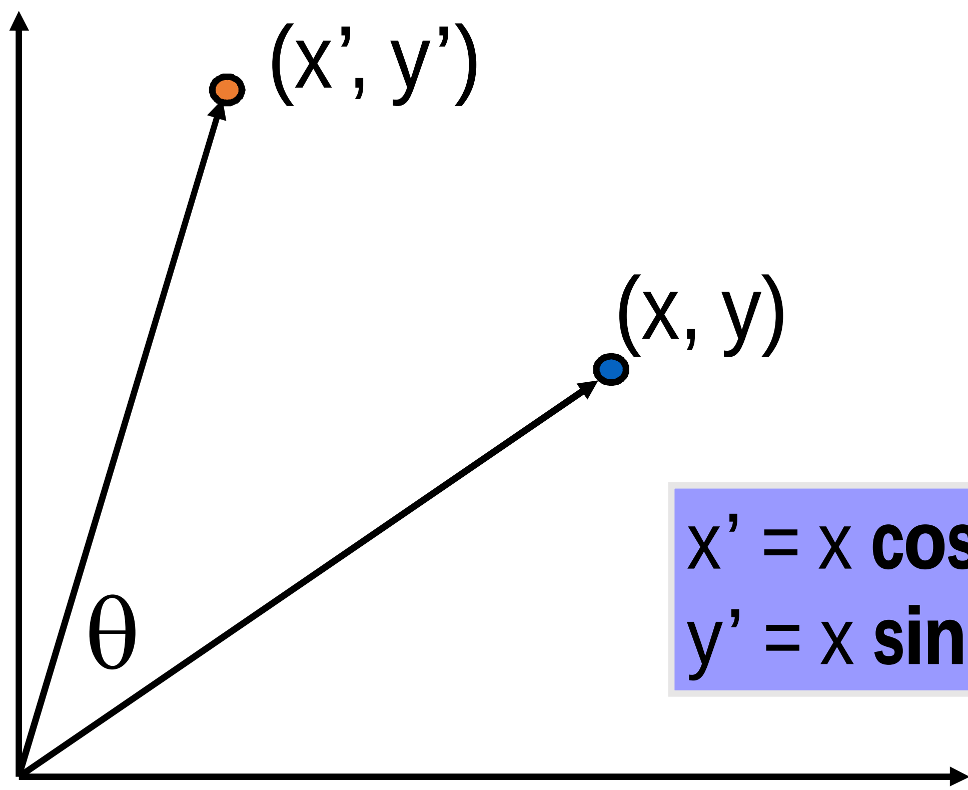
- Like scaling, rotations are centered about the origin







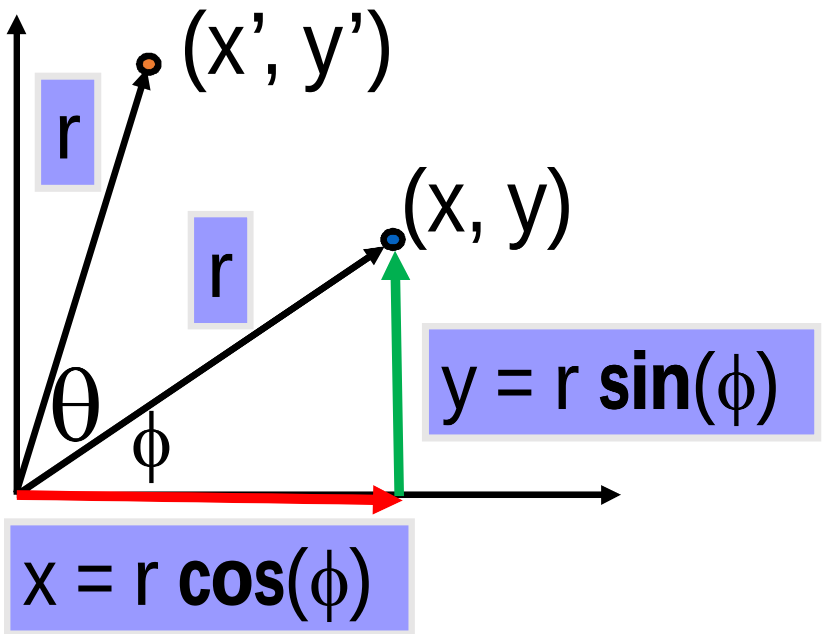
2-D Rotation



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

2-D Rotation

r-distance does not change



$x = r\cos(\phi)$

$y = r\sin(\phi)$

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

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

Trig Identity…

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

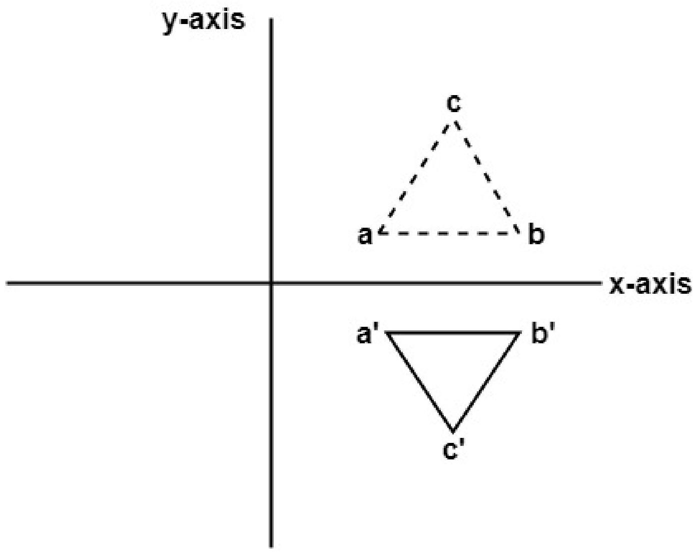
$y' = r\cos(\phi)\sin(\theta) + r\sin(\phi)\cos(\theta)$

Substitute…

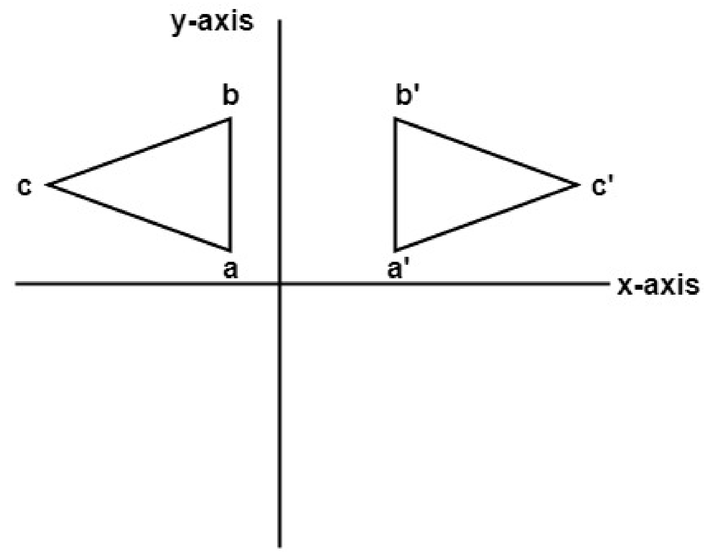
$x' = x\,\mathbf{cos}(\theta) - y\,\mathbf{sin}(\theta)$

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

Reflection about X and Y axis



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

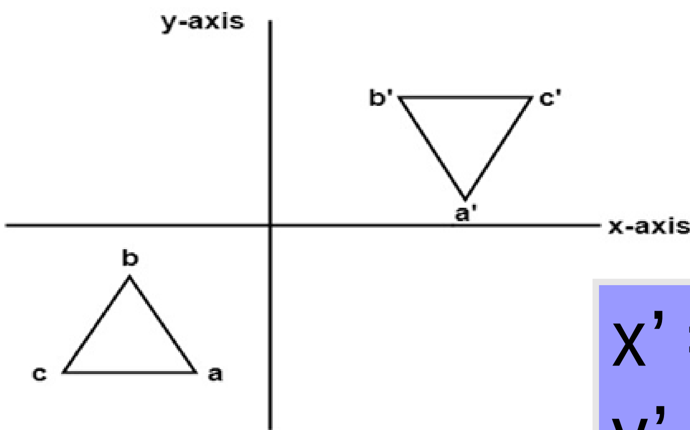


$$x' = -x$$
$$y' = y$$

Reflection about origin

$$x' = -x$$

$$y' = -y$$



x’ = x **cos**(θ) - y **sin**(θ)
y’ = x **sin**(θ) + y **cos**(θ)
θ=180

Shearing

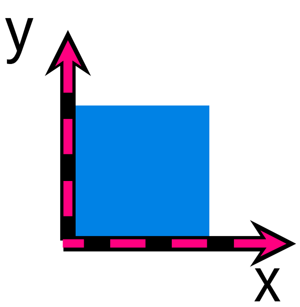
- One side is fixed and other layer are moved
- Horizontal shear: bottom layer fixed, force applied to above layers, layers pushed to right by an amount proportional to y value
- Y value unchanged
- New coordinates:

$$x' = x + shx.y,$$

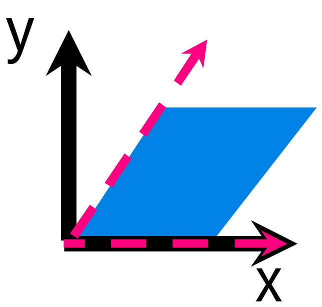
$$y' = y$$

Where shx is the shearing force/factor

Original



Horizontal Shear



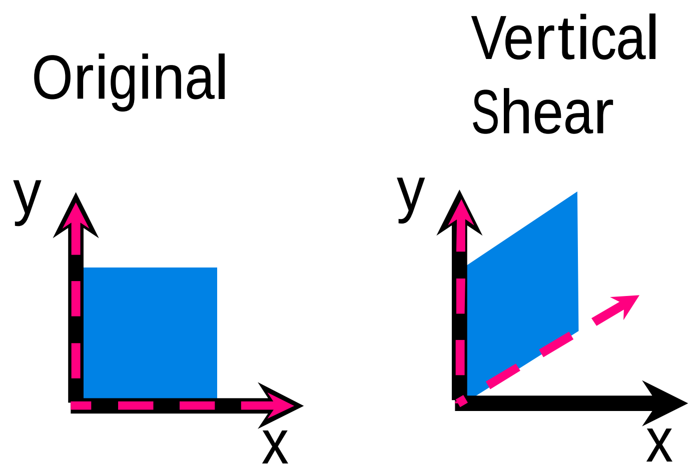
Shearing

- One side is fixed and other layer are moved
- Vertical shear: left layer fixed, force applied to right side layers, layers pushed to up by an amount proportional to x value
- x value unchanged
- New coordinates:

$$x' = x$$

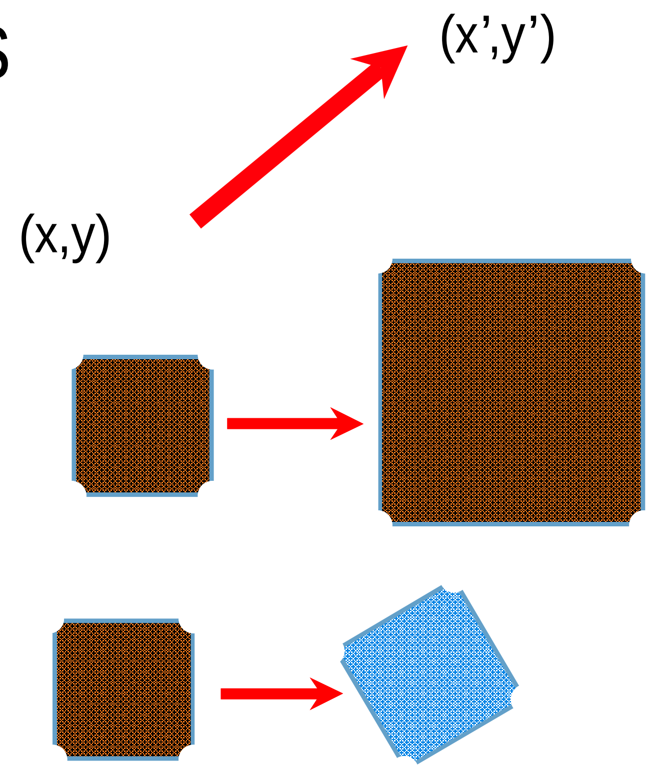
$$y' = y + shy \cdot x$$

Where shy is the shearing force/factor



Basic 2D Transformations

- **Translation**
$$x' = x + tx$$
$$y' = y + ty$$
- **Scale**
$$x' = x \times sx$$
$$y' = y \times sy$$
- **Rotation**
$$x' = x \times \cos\theta - y \times \sin\theta$$
$$y' = y \times \sin\theta + y \times \cos\theta$$
- **Shear**
$$x' = x + Shx \times y$$
$$y' = y + Shy \times x$$



Matrix Representation

- **Represent a 2D Transformation by a Matrix**

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

- **Apply the Transformation to a Point**

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

$$\Rightarrow \quad \begin{bmatrix} x' \\ y' \end{bmatrix} = \underbrace{\begin{bmatrix} a & b \\ c & d \end{bmatrix}}_{\text{Transformation Matrix}} \underbrace{\begin{bmatrix} x \\ y \end{bmatrix}}_{\text{Point}}$$

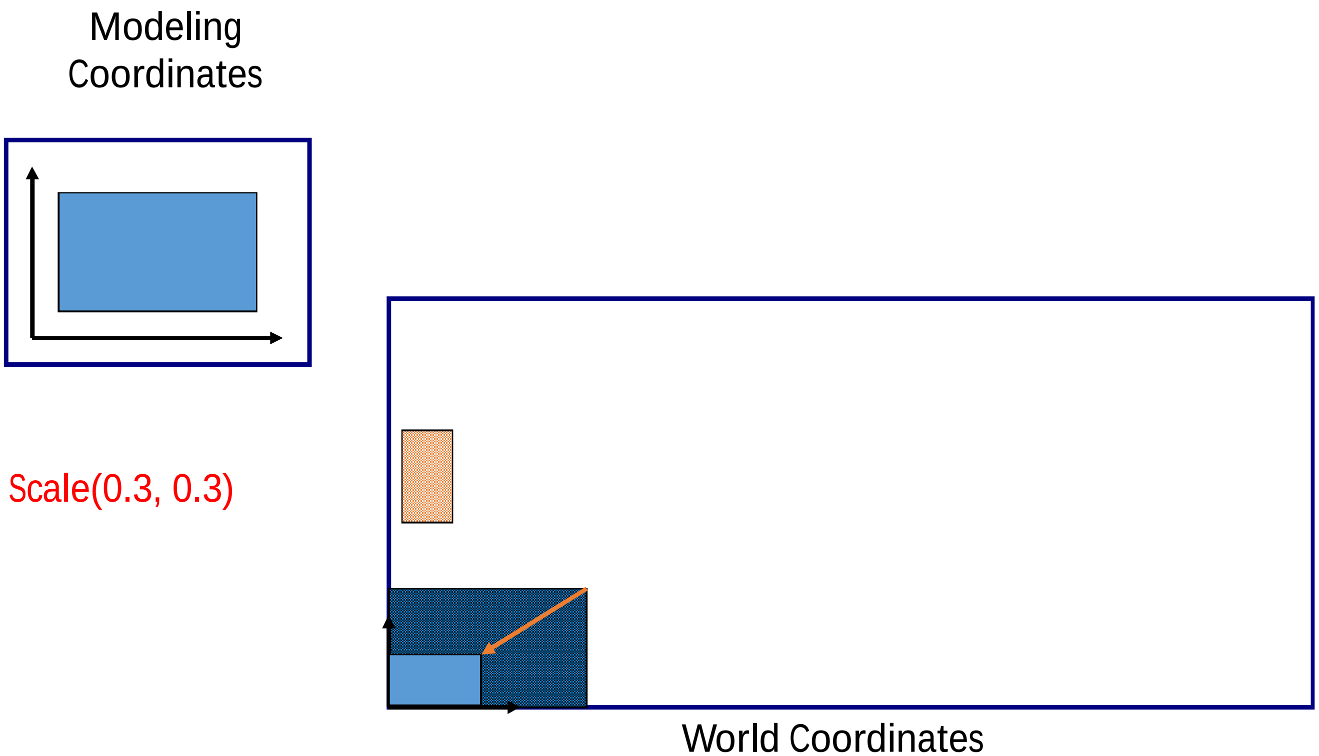
Matrix Representation

- Transformations combined by multiplication

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

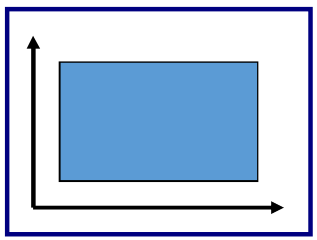
Matrices are a convenient and efficient way to represent a sequence of transformations!

Example: 2D Scaling

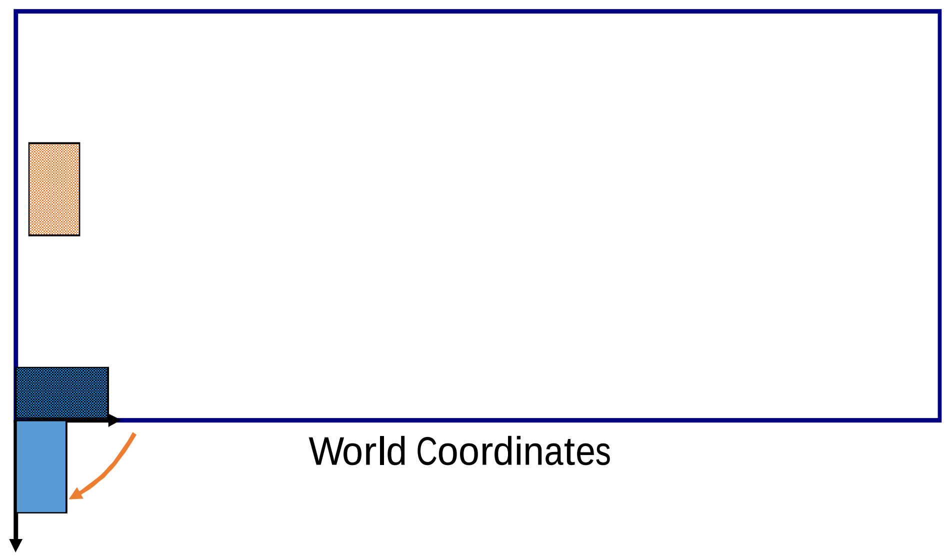


Example: 2D Rotation

Modeling
Coordinates

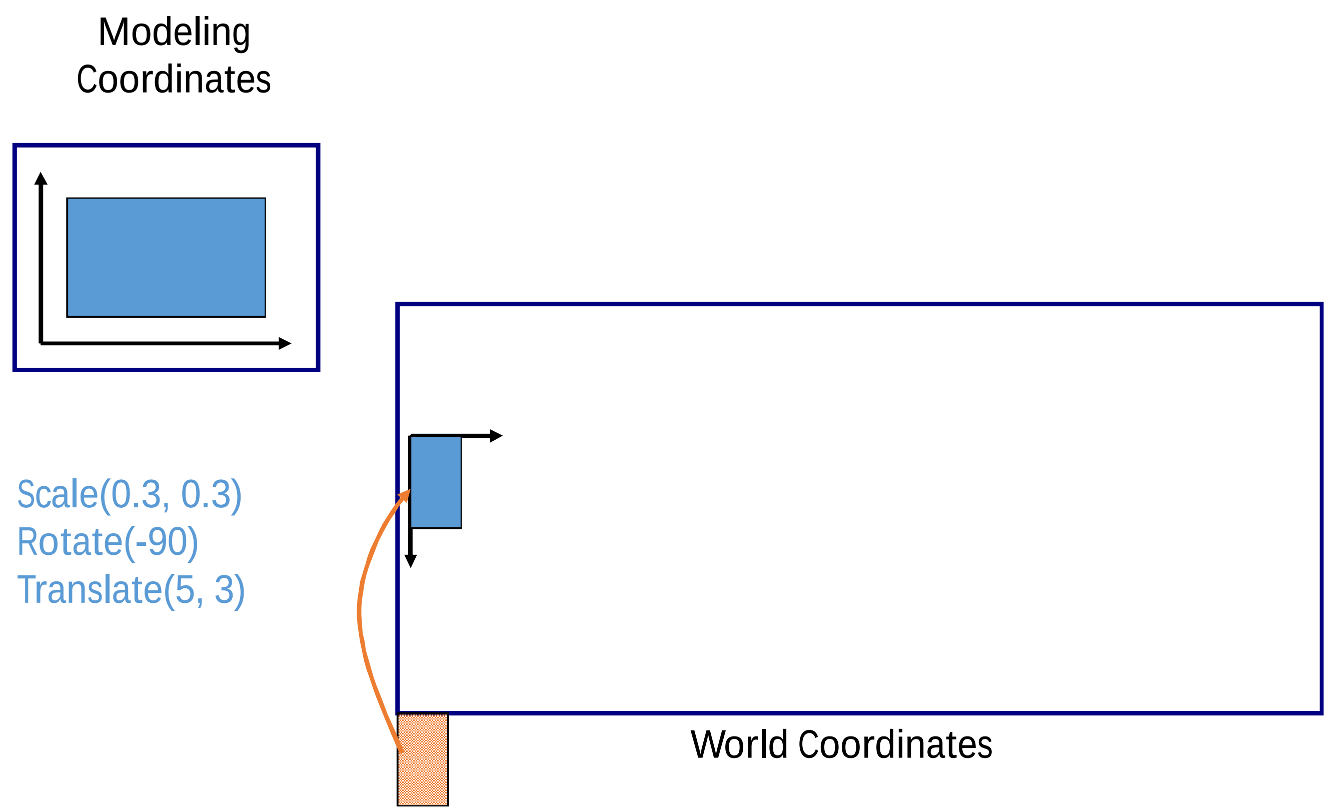


Scale(0.3, 0.3)
Rotate(-90)



World Coordinates

Example: 2D Translation



Matrix Composition

- Matrices are a convenient and efficient way to represent a sequence of transformations
  - General purpose representation
  - Hardware matrix multiply

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

Matrix Composition

- Be aware: order of transformations matters
  - Matrix multiplication is not commutative

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

“Global” “Local”

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)?

$$\boldsymbol{x}' = \boldsymbol{s}_x * \boldsymbol{x}$$
$$\boldsymbol{y}' = \boldsymbol{s}_y * \boldsymbol{y}$$

$$\begin{bmatrix} \boldsymbol{x}' \\ \boldsymbol{y}' \end{bmatrix} = \begin{bmatrix} \boldsymbol{s}_x & 0 \\ 0 & \boldsymbol{s}_y \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix}$$

Scaling

- Scaling operation: $$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} ax \\ by \end{bmatrix}$$

- Or, in matrix form: $$\begin{bmatrix} x' \\ y' \end{bmatrix} = \underbrace{\begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}}_{\textit{scaling matrix}} \begin{bmatrix} x \\ y \end{bmatrix}$$

2-D Rotation

- matrix form:

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

- Even though sin(θ) and cos(θ) are nonlinear functions of θ,
  - ***x’ is a linear combination of x and y***
  - ***y’ is a linear combination of x and y***

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

2x2 Matrices

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

  2D Rotate around (0,0)?

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

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

  2D Shear?

$$\boldsymbol{x}' = \boldsymbol{x} + \boldsymbol{sh}_x * \boldsymbol{y}$$
$$\boldsymbol{y}' = \boldsymbol{sh}_y * \boldsymbol{x} + \boldsymbol{y}$$

$$\begin{bmatrix} \boldsymbol{x}' \\ \boldsymbol{y}' \end{bmatrix} = \begin{bmatrix} 1 & \boldsymbol{sh}_x \\ \boldsymbol{sh}_y & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix}$$

2x2 Matrices

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

2D Mirror about 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 over (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

So How Do We Do It?

- What transformation matrix will add 4 to x and subtract 3 from y?
  - That is, what are the values of a, b, c, and d needed for this transformation?

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

Transformation Matrix

- Actually, this is impossible to do with a 2x2 matrix and 2-vectors

How Do We Do It?

- Option 1: Implement translation as a 2-step process

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

e is the x-offset
f is the y-offset

- What are the values for our example?

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 4 \\ -3 \end{bmatrix}$$

So How Do We Do It?

- Option 2: Use bigger matrices

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

- If we set w = 1, then

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

c is the x-offset
f is the y-offset

How We Do It

- This is the way we'll normally do it
- However, in computer science, we really like square matrices, so it'll be written as:

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

So what does this w stand for?

Homogeneous Coordinates

- We refer to this as a homogeneous coordinate:

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

- This mathematical construct allows us to
- Represent affine transforms with a single matrix
- Do calculations in projective space (vectors are unique only up to scaling)

Homogeneous Coordinates

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

- For points, w must be non-zero
  - If w=1, the point is "normalized"
  - If w!=1, can normalize by

$$\begin{bmatrix} \frac{x}{w} \\ \frac{y}{w} \\ \frac{w}{w} \end{bmatrix}$$

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 graphics operations much easier

Homogeneous Coordinates

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

$$\boldsymbol{x}' = \boldsymbol{x} + \boldsymbol{t}_x$$

$$\boldsymbol{y}' = \boldsymbol{y} + \boldsymbol{t}_y$$

- A: Using the rightmost column:

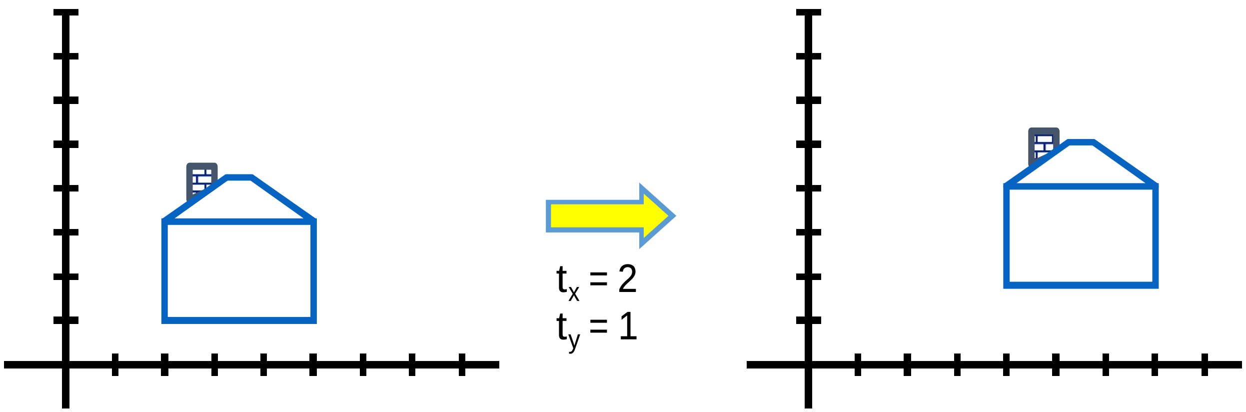
$$\mathbf{T}\boldsymbol{ranslation} = \begin{bmatrix} 1 & 0 & \boldsymbol{t}_x \\ 0 & 1 & \boldsymbol{t}_y \\ 0 & 0 & 1 \end{bmatrix}$$

Translation

- Example of translation

Homogeneous Coordinates

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



Translation as a Transformation matrix

- We will represent translation with a matrix of the following form:

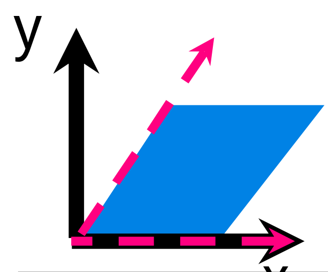
$$\mathbf{M} = \begin{bmatrix} 1 & 0 & u \\ 0 & 1 & v \\ 0 & 0 & 1 \end{bmatrix}$$

u is the x-offset
v is the y-offset

Shearing

Horizontal Shear

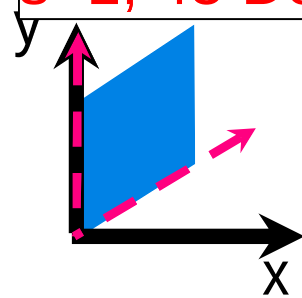
$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} 1 & s & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$



s=0, No Shear
s=1, 45 Degree Shear

Vertical Shear

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



s is -tan(θ), where θ is the desired shear angle

Basic 2D Transformations

- **Basic 2D transformations as 3x3 Matrices**

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

Translate

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

Scale

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

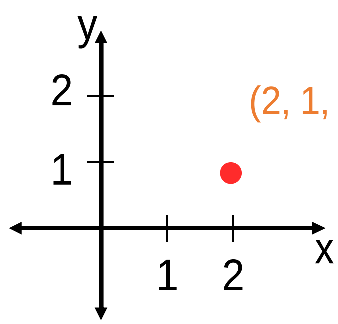
Rotate

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

Shear

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



(2, 1, 1) or (4, 2, 2) or (6, 3, 3)

Convenient Coordinate System to Represent Many Useful Transformations

Linear Transformations

- **Linear transformations are combinations of …**
  - Scale
  - Rotation
  - Shear, and
  - Mirror

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

- **Properties of linear transformations**
  - Satisfies:
  - Origin maps to origin
  - Lines map to lines
  - Parallel lines remain parallel
  - Ratios are preserved
  - Closed under composition

$$T(s_1 p_1 + s_2 p_2) = s_1 T(p_1) + s_2 T(p_2)$$

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 map to origin
  - Lines map to lines
  - Parallel lines remain parallel
  - Ratios are preserved
  - Closed under composition

Projective Transformations

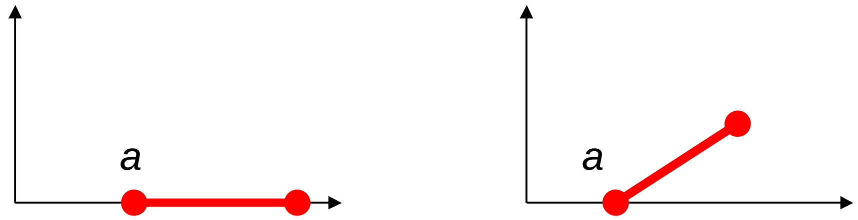
- **Projective transformations…**
  - Affine transformations, and
  - Projective warps

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

- **Properties of projective transformations**
  - Origin does not map to origin
  - Lines map to lines
  - Parallel lines do not necessarily remain parallel
  - Ratios are not preserved
  - Closed under composition

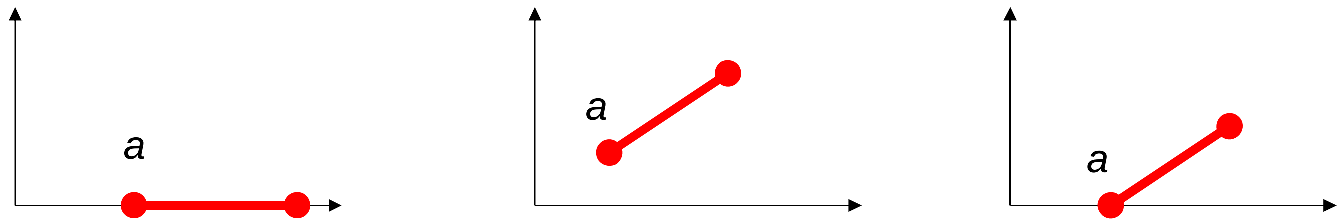
Matrix Composition

- What if we want to rotate and translate?
  - Ex: Rotate line segment by 45 degrees about endpoint *a and lengthen*



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?

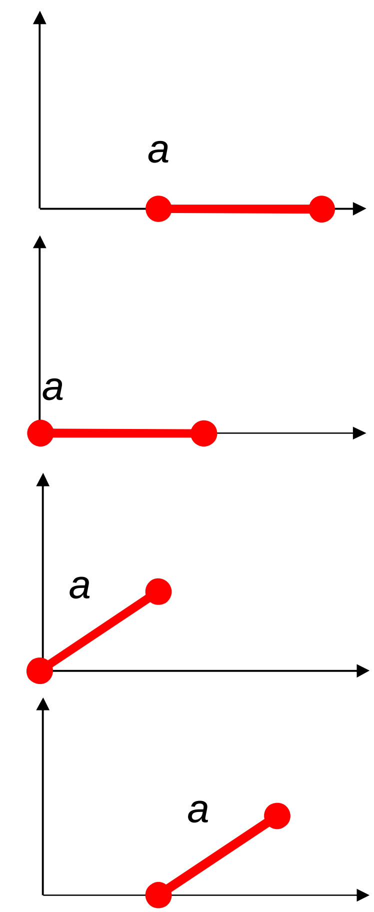


Wrong
R(45)

Correct
T(-3) R(45) T(3)

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?

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

Composing Transforms

- Composing 2 transforms is just multiplying the 2 transform matrices together

WARNING: The order in which matrix multiplications
are performed may (and usually does) change the result!
(i.e. they are not commutative)

Basic 2D Transformations

- **Translation**

$$x' = x + tx$$
$$y' = y + ty$$

- **Scale**

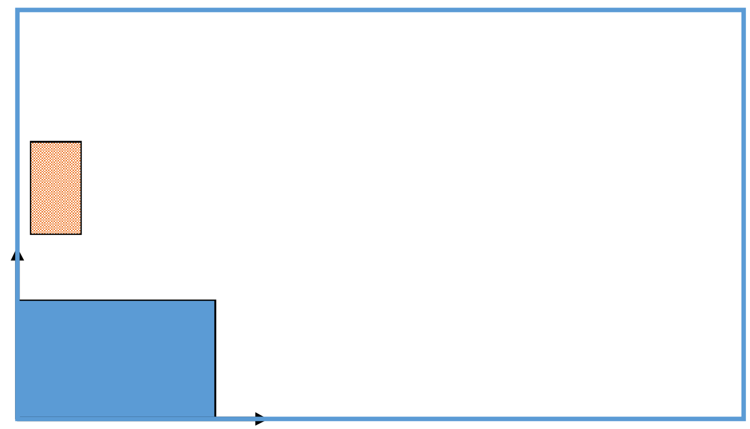
$$x' = x \times sx$$
$$y' = y \times sy$$

- **Rotation**

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

- **Shear**

$$x' = x + hx \times y$$
$$y' = y + hy \times x$$



Transformations
can be combined
(with simple algebra)

Basic 2D Transformations

- **Translation**

$$x' = x + tx$$
$$y' = y + ty$$

- **Scale**

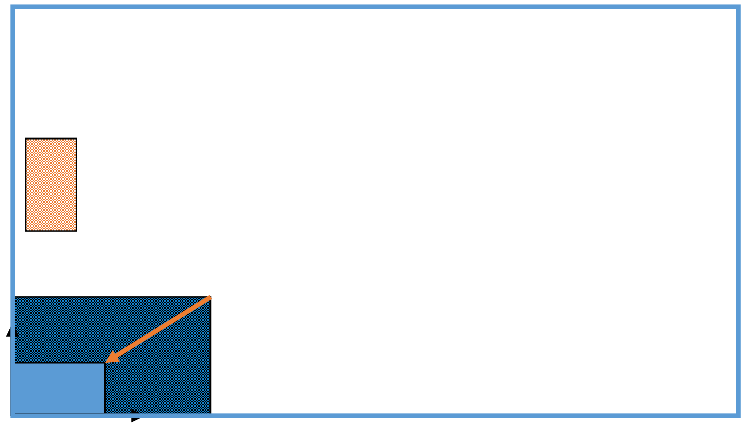
$$x' = x \times sx$$
$$y' = y \times sy$$

- **Rotation**

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

- **Shear**

$$x' = x + hx \times y$$
$$y' = y + hy \times x$$



$$x' = x \times \underline{sx}$$
$$y' = y \times \underline{sy}$$

Basic 2D Transformations

- **Translation**

$$x' = x + tx$$
$$y' = y + ty$$

- **Scale**

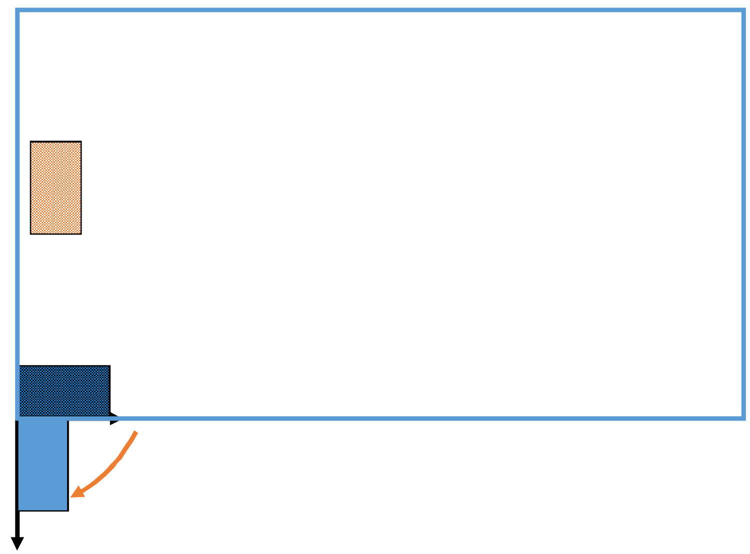
$$x' = x \times sx$$
$$y' = y \times sy$$

- **Rotation**

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

- **Shear**

$$x' = x + hx \times y$$
$$y' = y + hy \times x$$



$$\boxed{\begin{aligned} x' &= ((x \times sx) \times \underline{\cos\theta} - (y \times sy) \times \underline{\sin\theta}) \\ y' &= ((x \times sx) \times \underline{\sin\theta} + (y \times sy) \times \underline{\cos\theta}) \end{aligned}}$$

Basic 2D Transformations

- **Translation**

$$x' = x + tx$$
$$y' = y + ty$$

- **Scale**

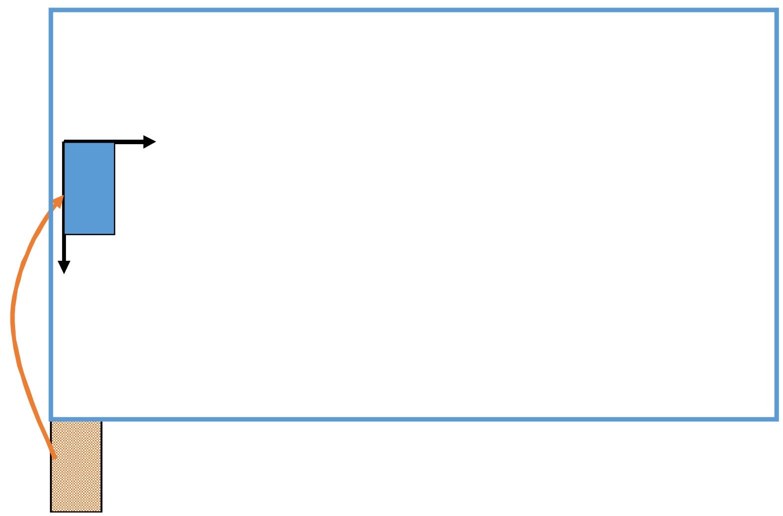
$$x' = x \times sx$$
$$y' = y \times sy$$

- **Rotation**

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

- **Shear**

$$x' = x + hx \times y$$
$$y' = y + hy \times x$$



$$\boxed{\begin{aligned} x' &= ((x \times sx) \times \cos\theta - (y \times sy) \times \sin\theta) + \underline{tx} \\ y' &= ((x \times sx) \times \sin\theta + (y \times sy) \times \cos\theta) + \underline{ty} \end{aligned}}$$

Basic 2D Transformations

- **Translation**

$$x' = x + tx$$
$$y' = y + ty$$

- **Scale**

$$x' = x \times sx$$
$$y' = y \times sy$$

- **Rotation**

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

- **Shear**

$$x' = x + hx \times y$$
$$y' = y + hy \times x$$

$$\boxed{\begin{aligned} x' &= ((x \times sx) \times \cos\theta - (y \times sy) \times \sin\theta) + tx \\ y' &= ((x \times sx) \times \sin\theta + (y \times sy) \times \cos\theta) + ty \end{aligned}}$$

Matrix Representation

- **Transformations can be combined by matrix multiplication**

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

Transformation Matrix

Matrices are a <u>convenient</u> and <u>efficient</u> way to represent a sequence of transformations

Matrix Composition

- **Transformations can be combined by matrix multiplication**

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

$$p' \quad = \quad \mathrm{T(tx,\ ty)} \qquad \mathrm{R}(\theta) \qquad \mathrm{S(sx,\ sy)} \qquad p$$

- **Efficiency with premultiplication**
  - Matrix multiplication is associative

$$\mathrm{p}' = (\mathrm{T} \times (\mathrm{R} \times (\mathrm{S} \times \mathrm{p}))) \Rightarrow \mathrm{p}' = (\mathrm{T} \times \mathrm{R} \times \mathrm{S}) \times \mathrm{p}$$

Matrix Composition

- After correctly ordering the matrices
- Multiply matrices together
- This results is one matrix
- Multiply this matrix by the vector of each vertex
- All vertices easily transformed with one matrix multiplication