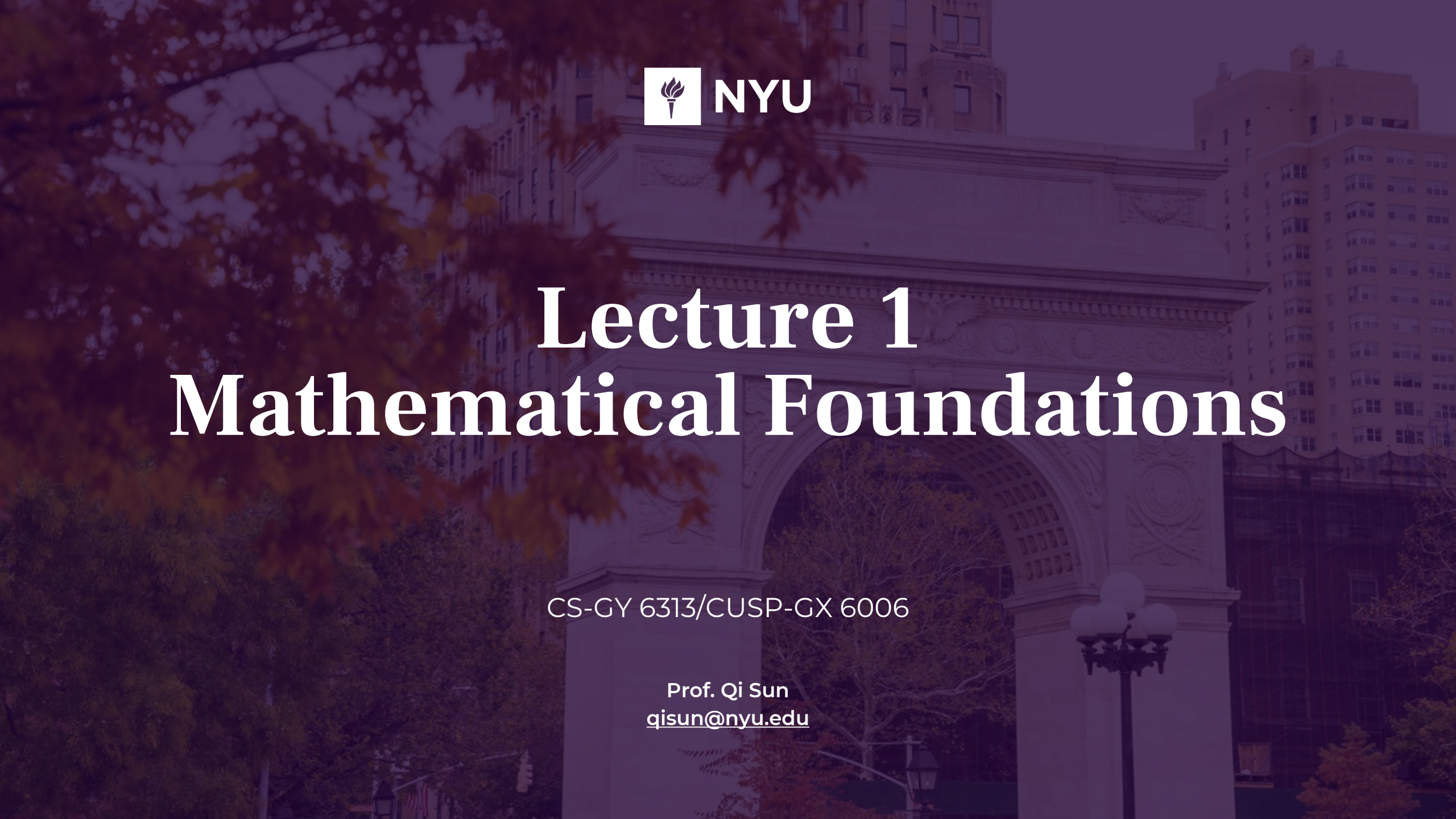


NYU

Lecture 1
Mathematical Foundations

CS-GY 6313/CUSP-GX 6006

Prof. Qi Sun
qisun@nyu.edu

Lecture 1.1 - Physical Foundations --Basic Visual Optics

Basic Visual Optics – What is Light?

Sir Isaac Newton
1643-1727

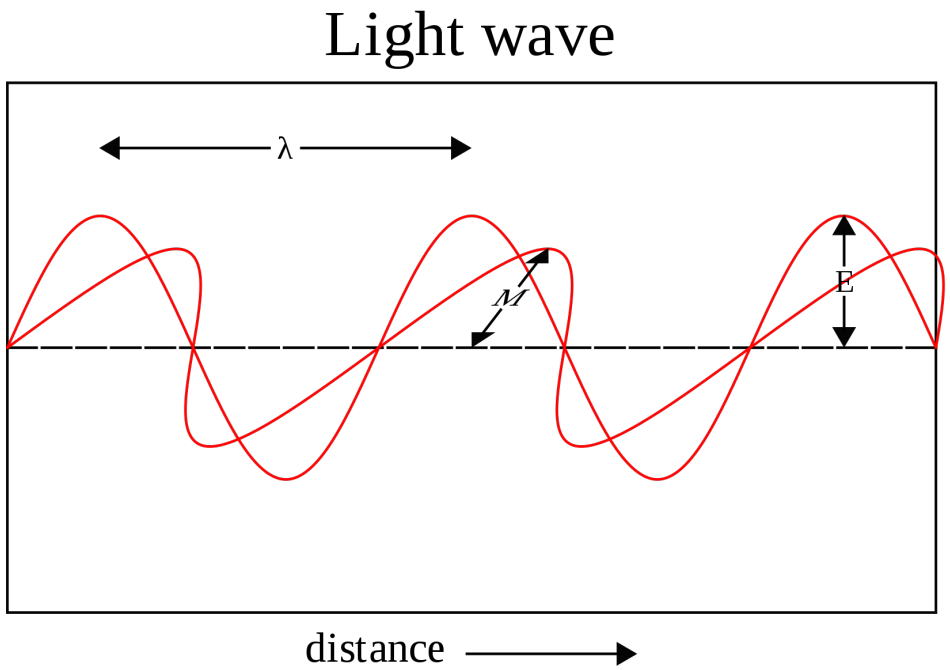
Christiaan Hyugens
1629-1695

"… very small bodies emitted from shining substances"

– Issac Newton

" "A wave motion" spreading out from the course of all directions"

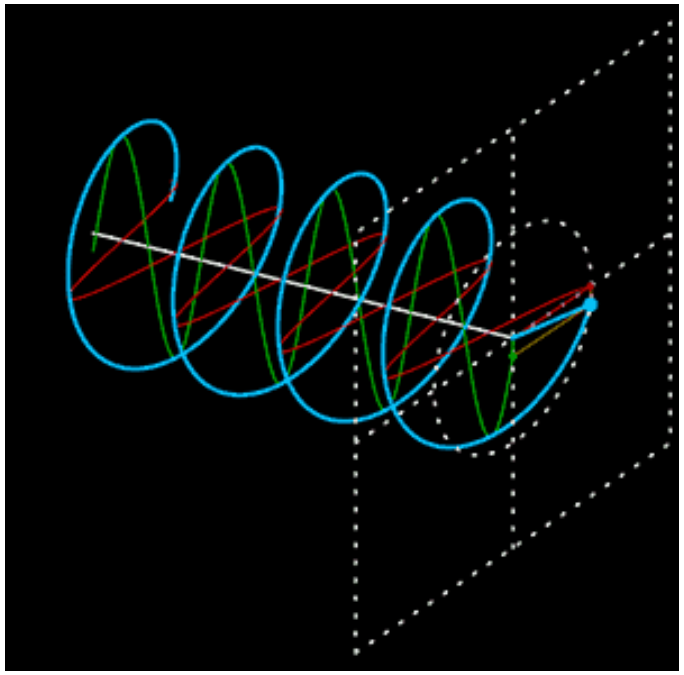
-- Christian Huygens



λ = wave length

E = amplitude of electric field

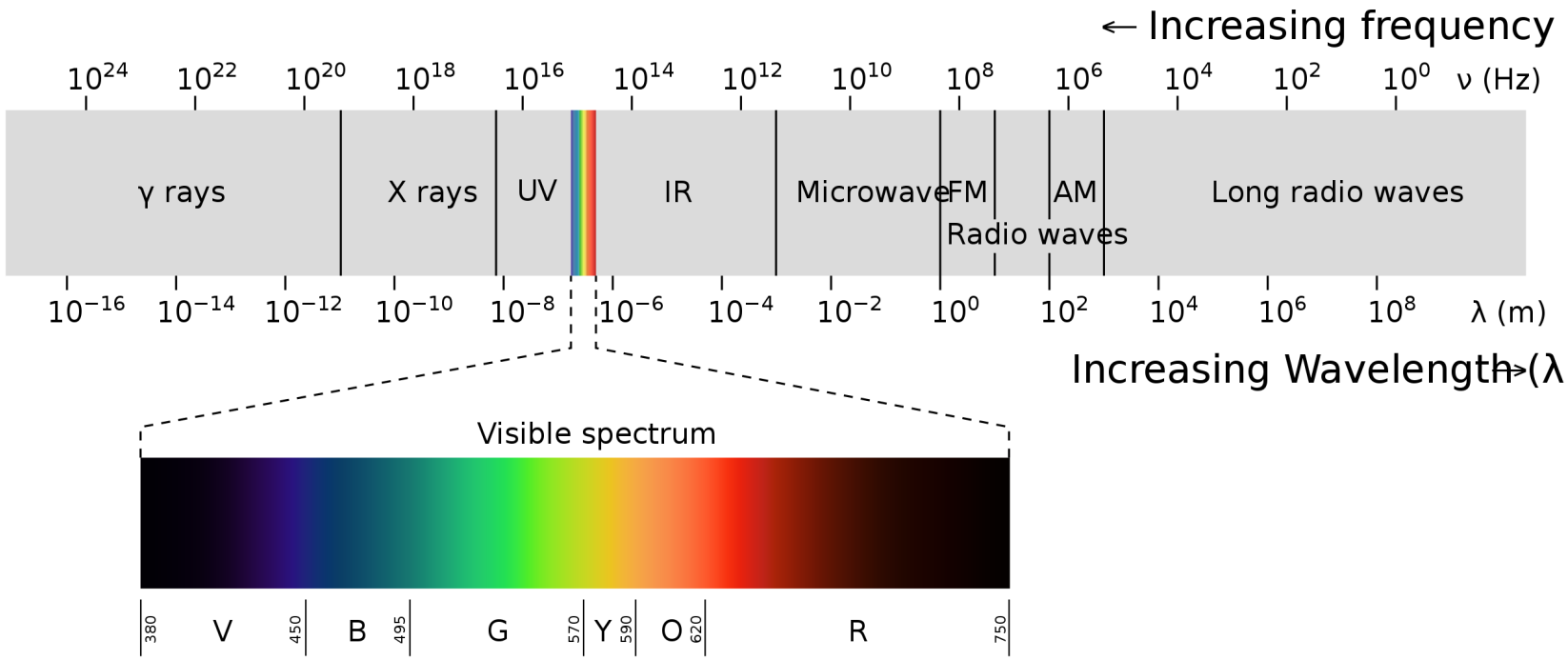
M = amplitude of magnetic field



NYU

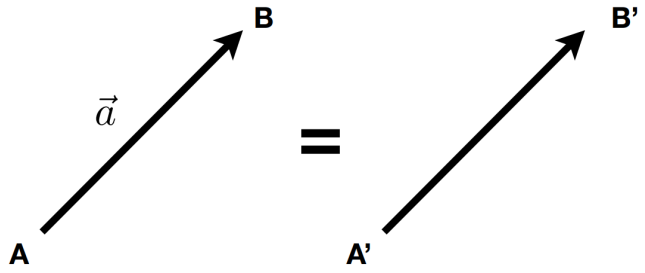
Basic Visual Optics – What is Light?

Visible light is electromagnetic radiation within the portion of the electromagnetic spectrum that is perceived by the human eye.



Lecture 1.2
Mathematical Foundations
-- Revisiting Basic Algebra

Vectors

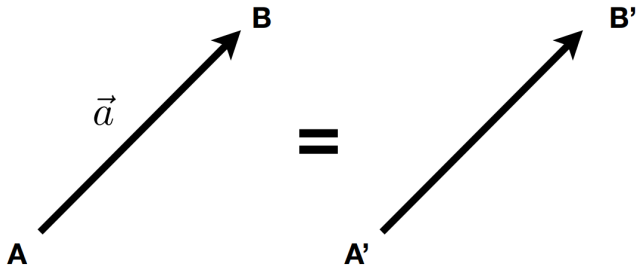


- A vector $\overrightarrow{(x, y, z, \ldots)}$ describes a **direction** and a **length** **without** a starting point
- A vector is NOT a pair {x,y}, NOR a position (x,y)

$$\vec{a} \qquad \boldsymbol{a} \qquad \overrightarrow{AB} = B - A$$

Vector Operation

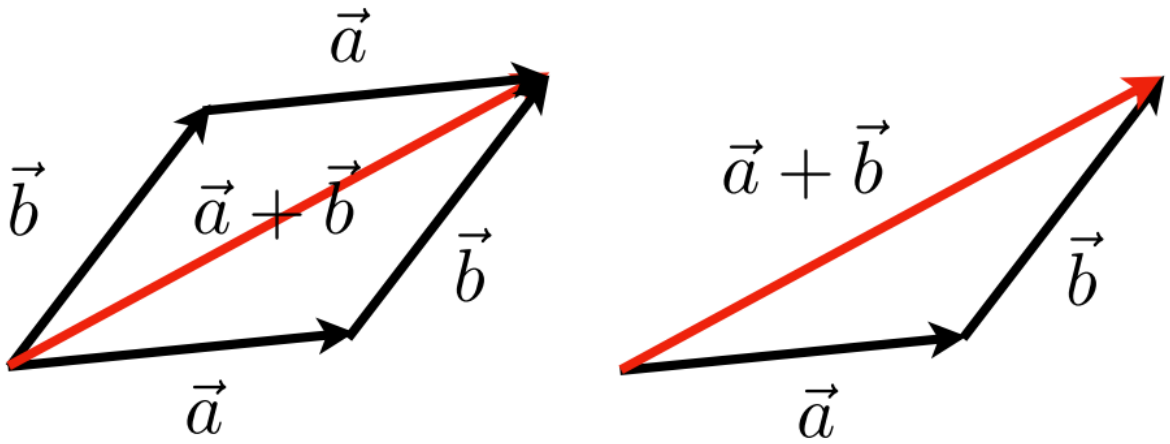
unit & normalization



- A unit vector is a vector with length = 1 (direction-only)
- Normalization - $\hat{a} = \vec{a} / \left\| \vec{a} \right\|$

Vector Operation

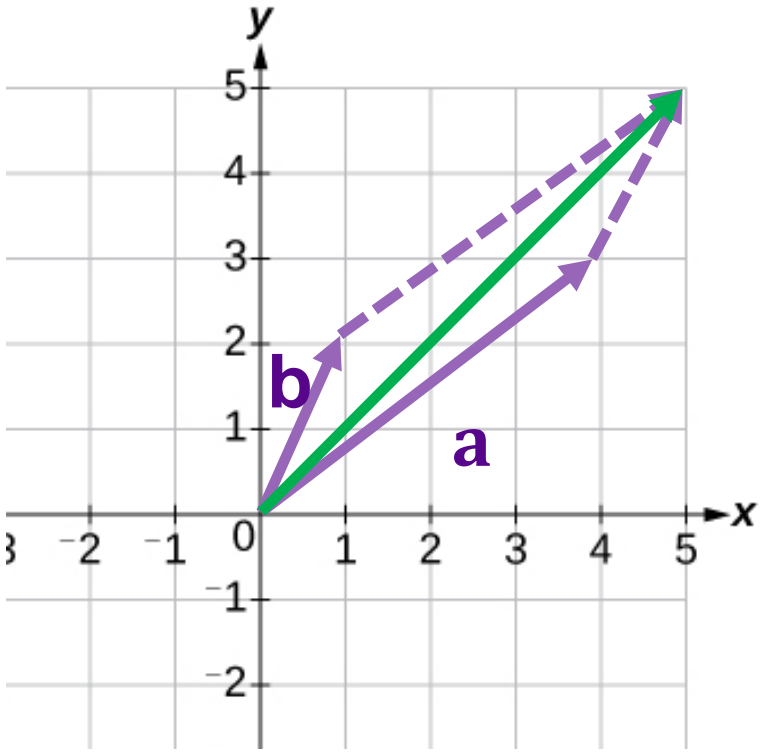
(positive/negative) add



$$\vec{a} + \vec{b} = \vec{b} + \vec{a}$$

Vector Operation

Cartesian Coordinate



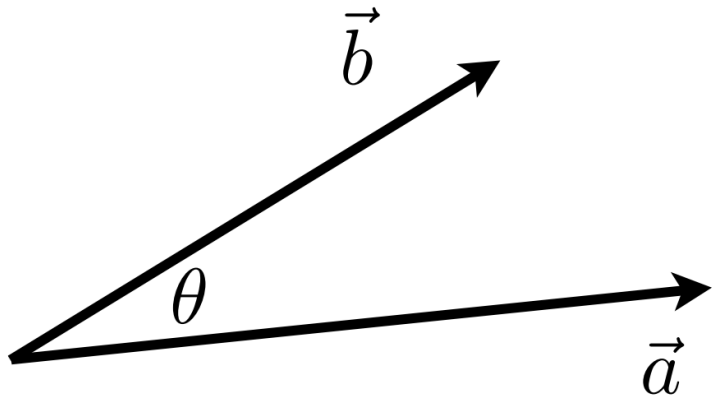
$$\mathbf{a} = \begin{pmatrix} 4 \\ 3 \end{pmatrix} = 4\mathbf{x} + 3\mathbf{y}$$

$$\mathbf{a} + \mathbf{b} = (\mathbf{4} + \mathbf{1}, \mathbf{3} + \mathbf{2})$$

$$|\mathbf{a}| = \sqrt{\mathbf{4^2} + \mathbf{3^2}}$$

Vector Operation

Dot Product



$$\vec{a} \cdot \vec{b} = \|\vec{a}\| \|\vec{b}\| \cos\theta$$

$$\cos\theta = \frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\| \|\vec{b}\|}$$

Unit Vector:

$$\cos\theta = \hat{a} \cdot \hat{b}$$

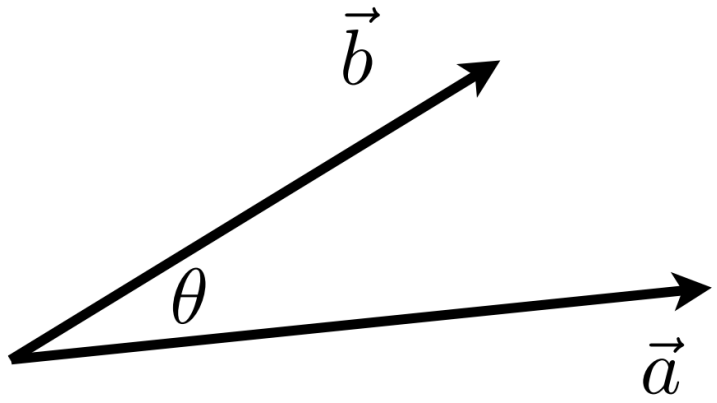
$$\vec{a} \cdot \vec{b} = \vec{b} \cdot \vec{a}$$

$$\vec{a} \cdot (\vec{b} + \vec{c}) = \vec{a} \cdot \vec{b} + \vec{a} \cdot \vec{c}$$

$$(k\vec{a}) \cdot \vec{b} = \vec{a} \cdot (k\vec{b}) = k(\vec{a} \cdot \vec{b})$$

Vector Operation

Dot Product in Cartesian Coordinate

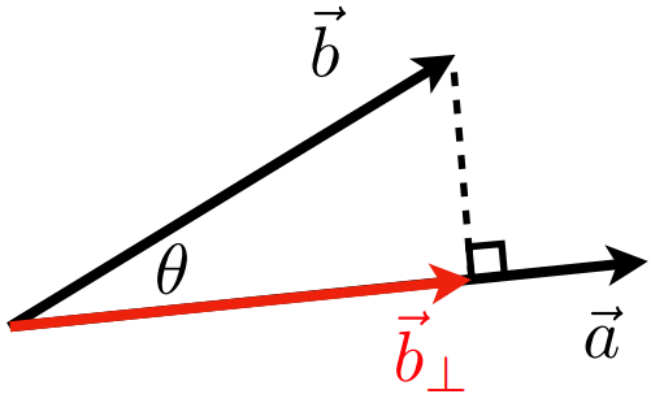


$$\vec{a} \cdot \vec{b} = \begin{pmatrix} x_a \\ y_a \end{pmatrix} \cdot \begin{pmatrix} x_b \\ y_b \end{pmatrix} = x_a x_b + y_a y_b$$

$$\vec{a} \cdot \vec{b} = \begin{pmatrix} x_a \\ y_a \\ z_a \end{pmatrix} \cdot \begin{pmatrix} x_b \\ y_b \\ z_b \end{pmatrix} = x_a x_b + y_a y_b + z_a z_b$$

Vector Operation

Dot Product in Visualization



Find projected length

$\vec{b}_\perp$: projection of $\vec{b}$ onto $\vec{a}$

- $\vec{b}_\perp$ must be along $\vec{a}$ (or along $\hat{a}$)
  - $\vec{b}_\perp = k\hat{a}$
- What's its magnitude k?
  - $k = \|\vec{b}_\perp\| = \|\vec{b}\| \cos\theta$

Find angles between vectors

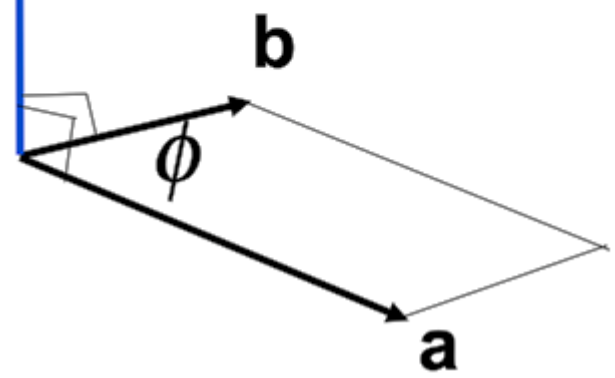
$$\theta = \arccos\left(\frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\| \cdot \|\vec{b}\|}\right)$$

Vector Operation

Cross Product

$$a \times b = -b \times a$$

$$||a \times b|| = ||a|| ||b|| \sin\phi$$



- Orthogonal to two initial vectors
- Direction determined by right-hand rule
- Useful in constructing coordinate systems (later)

Cross product: Properties

$$\vec{x} \times \vec{y} = +\vec{z}$$

$$\vec{y} \times \vec{x} = -\vec{z}$$

$$\vec{y} \times \vec{z} = +\vec{x}$$

$$\vec{z} \times \vec{y} = -\vec{x}$$

$$\vec{z} \times \vec{x} = +\vec{y}$$

$$\vec{x} \times \vec{z} = -\vec{y}$$

$$\vec{a} \times \vec{b} = -\vec{b} \times \vec{a}$$

$$\vec{a} \times \vec{a} = \vec{0}$$

$$\vec{a} \times (\vec{b} + \vec{c}) = \vec{a} \times \vec{b} + \vec{a} \times \vec{c}$$

$$\vec{a} \times (k\vec{b}) = k(\vec{a} \times \vec{b})$$

NYU

Cross Product in Cartesian Coordinate

$$\vec{a} \times \vec{b} = \begin{vmatrix} \vec{x} & \vec{y} & \vec{z} \\ x_a & y_a & z_a \\ x_b & y_b & z_b \end{vmatrix} = \begin{pmatrix} y_a z_b - y_b z_a \\ z_a x_b - x_a z_b \\ x_a y_b - y_a x_b \end{pmatrix}$$

determinant

$$\vec{a} \times \vec{b} = A^* b = \begin{pmatrix} 0 & -z_a & y_a \\ z_a & 0 & -x_a \\ -y_a & x_a & 0 \end{pmatrix} \begin{pmatrix} x_b \\ y_b \\ z_b \end{pmatrix}$$

dual matrix of vector a

NYU

Basic Vectors/Arrays in Python

Python has a data type known as a list. For our purposes, lists are arrays.

Declaration syntax: <name> = [<value>, <value>, <value>, …, <value>]
<name> = [<default value>] * <initial array size>

Example:
numbers = [12, 49, -2, 26, 5, 17, -6]
zeros = [0] * 10

Indexing: Lists have zero based indexing from front

Negative Indexing: You can also refer to an element by a negative index representing how far it is from the end.

Example:
index from front *0 1 2 3 4 5*
numbers = [13, 25, 39, 46, 54, 68]
index from back *-6 -5 -4 -3 -2 -1*

NYU

Basic Vectors/Arrays in Python

Basic Methods – directly modify the lists

- list.append(item) – appends the item to the end of the list
- list.insert(index, item) – inserts the item at the specified index
- list.remove(item) – removes the first occurrence of item from the list
- list.extend(second_list) – appends second_list to the list

Mathematical Operators – behave as you would expect them to

- (+) Returns a new list by adding two lists. Appends the right-hand list to the left-hand list.
- (+=) Appends the right-hand list to the left-hand list. Modifies left list. Acts like extend()
- (*) Multiplies a list and an integer "n". Returns a new list that has n-1 versions of original list appended to it

Examples:

```
list = [34, 21, 29, 86, 29]

list.append(3)  => [34, 21, 29, 86, 29, 3]
list.insert(2, 3) => [34, 21, 3, 29, 86]
```

```
list2 = [1, 2, 3, 4]

list.remove(29)  => [34, 21, 86, 29]
list.extend(list2) => [34, 21, 86, 29, 1, 2, 3, 4]
[0] * 5            => [0, 0, 0, 0, 0]
```

NYU

Basic Vectors/Arrays in Python

More Methods

- list.count(element) – returns number of times element occurs in the list
- list.sort – sorts the element in place
- list.reverse – reverses the element in place

Slicing – can get a sub list of a list

<name>[<first index inclusive> : <second index not-inclusive>]

list = [4, 23, 16, 7, 29, 56, 81]
list[3:6] => [16, 7, 29]

Length of lists

len(list) => 7

Split – returns a list

“lets try some splitting here”.split(“ “) => ['lets', 'try', 'some', 'splitting', 'here']

NYU

Basic Vectors/Arrays Operations in Numpy

```
import numpy as np


arr1 = np.array([3, 2, 1])
arr2 = np.array([1, 2, 3])
print ("1st array : ", arr1)
print ("2nd array : ", arr2)
out_arr = np.add(arr1, arr2) print ("added array : ", out_arr)
```

Coordinates

Orthonormal Bases / Coordinate Frames

- Important for representing points, positions, locations
- Often, many sets of coordinate systems
  - Global, local, world, model, parts of model (head, hands, …)
- Critical issue is transforming between these systems/bases

Orthonormal Coordinate Frames

Any set of 3 vectors (in 3D) that:

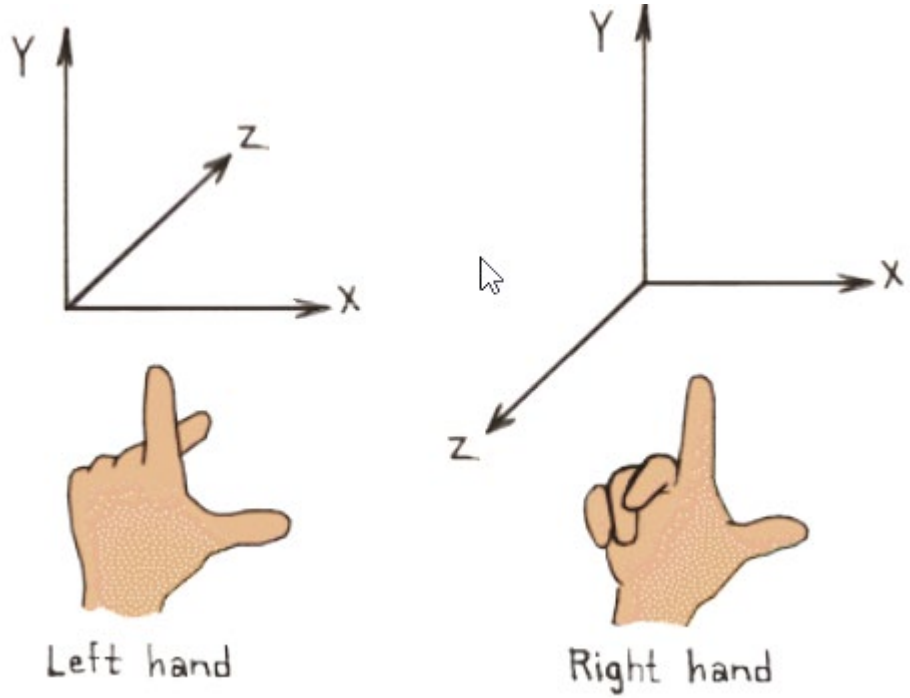
$$\|\vec{u}\| = \|\vec{v}\| = \|\vec{w}\| = 1$$

$$\vec{u} \cdot \vec{v} = \vec{v} \cdot \vec{w} = \vec{u} \cdot \vec{w} = 0$$

$$\vec{w} = \vec{u} \times \vec{v}$$ (right-handed)

$$\vec{p} = (\vec{p} \cdot \vec{u})\vec{u} + (\vec{p} \cdot \vec{v})\vec{v} + (\vec{p} \cdot \vec{w})\vec{w}$$

(projection)



Matrix

Matrices (array of numbers)
are essential for data representation and transformation

$$\begin{pmatrix} 1 & 3 \\ 5 & 2 \\ 0 & 4 \end{pmatrix}$$

NYU

Matrix Operations

multiplication

- # (number of) columns in A must = # rows in B (M x N) (N x P) = (M x P)
- Element (i, j) in the product is the dot product of row i from A and column j from B

$$\begin{pmatrix} 1 & 3 \\ 5 & 2 \\ 0 & 4 \end{pmatrix} \begin{pmatrix} 3 & 6 & 9 & 4 \\ 2 & 7 & 8 & 3 \end{pmatrix} = \begin{pmatrix} 9 & ? & 33 & 13 \\ 19 & 44 & 61 & 26 \\ 8 & 28 & 32 & ? \end{pmatrix}$$

Matrix Operations

Multiplication properties

- Non-commutative
  (AB and BA are different in general)

- Associative and distributive
  - A(B+C) = AB + AC
  - (A+B)C = AC + BC

Matrix-Vector Multiplication

- Treat vector as a column matrix (m×1)
- Key for transforming points (next lecture)
- Official spoiler: 2D reflection about y-axis

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

NYU

Transpose of a Matrix

- Switch rows and columns (ij -> ji)

$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{pmatrix}^T = \begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{pmatrix}$$

- Property

$$(AB)^T = B^T A^T$$

Identity Matrix and Inverses

$$I_{3\times 3} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$AA^{-1} = A^{-1}A = I$$

$$(AB)^{-1} = B^{-1}A^{-1}$$

NYU

Vector Multiplication in Matrix Form

- Dot product?

$$\vec{a} \cdot \vec{b} = \vec{a}^T \vec{b}$$

$$= \begin{pmatrix} x_a & y_a & z_a \end{pmatrix} \begin{pmatrix} x_b \\ y_b \\ z_b \end{pmatrix} = \begin{pmatrix} x_a x_b + y_a y_b + z_a z_b \end{pmatrix}$$

- Cross product?

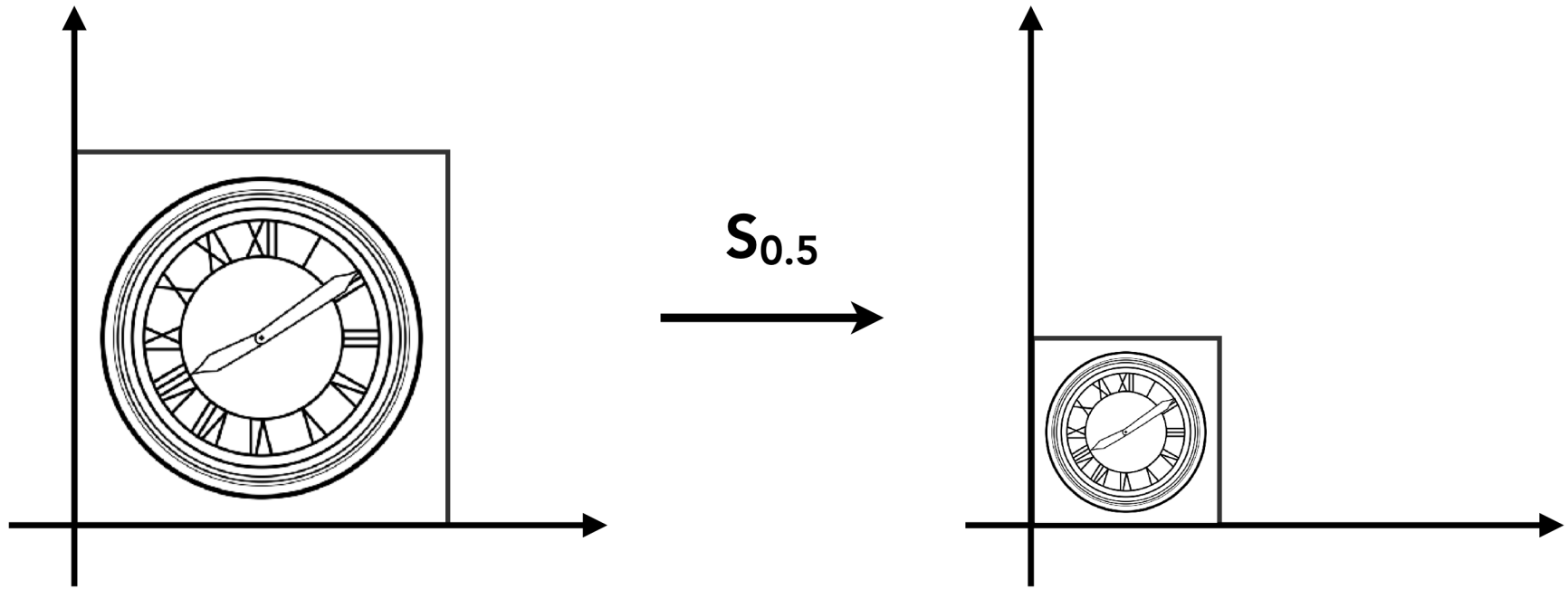
$$\vec{a} \times \vec{b} = A^* b = \begin{pmatrix} 0 & -z_a & y_a \\ z_a & 0 & -x_a \\ -y_a & x_a & 0 \end{pmatrix} \begin{pmatrix} x_b \\ y_b \\ z_b \end{pmatrix}$$

Why Matrix?

Transform your data!

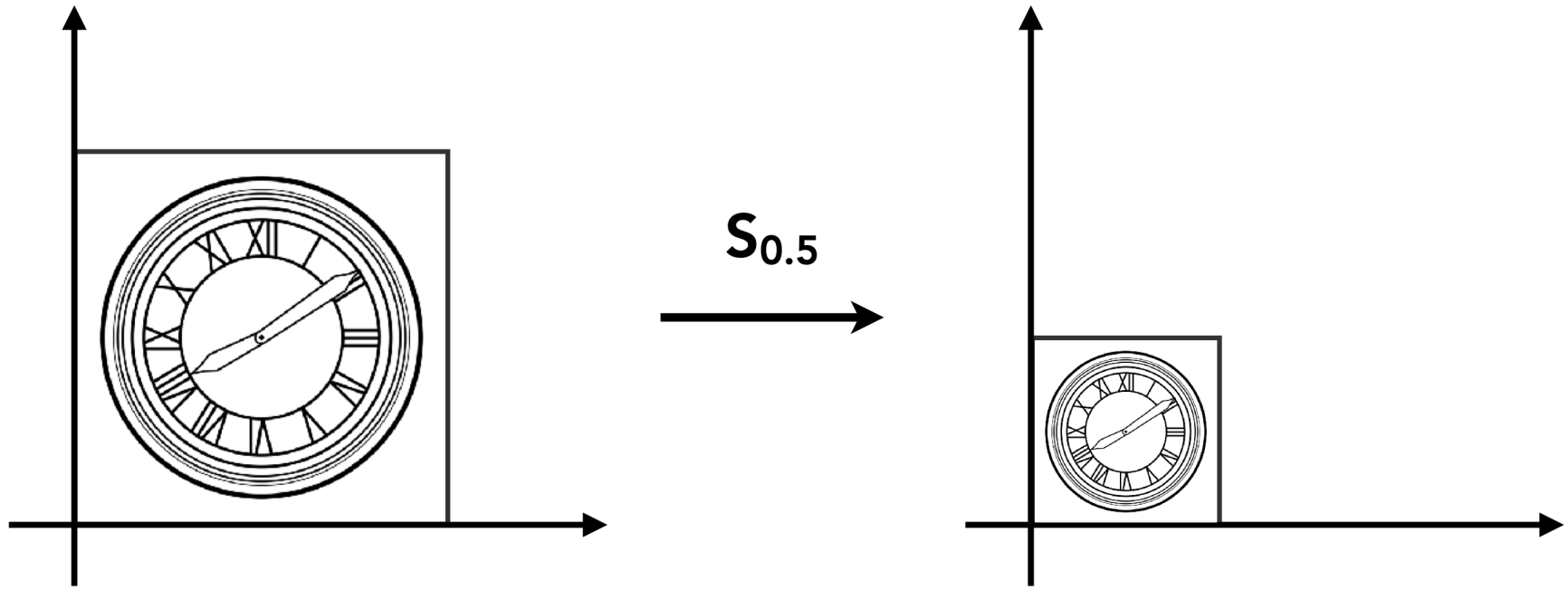
https://developers.google.com/maps/documentation/javascript/webgl/tilt-rotation

Scale



slide courtesy of Ren Ng

Scale Transform

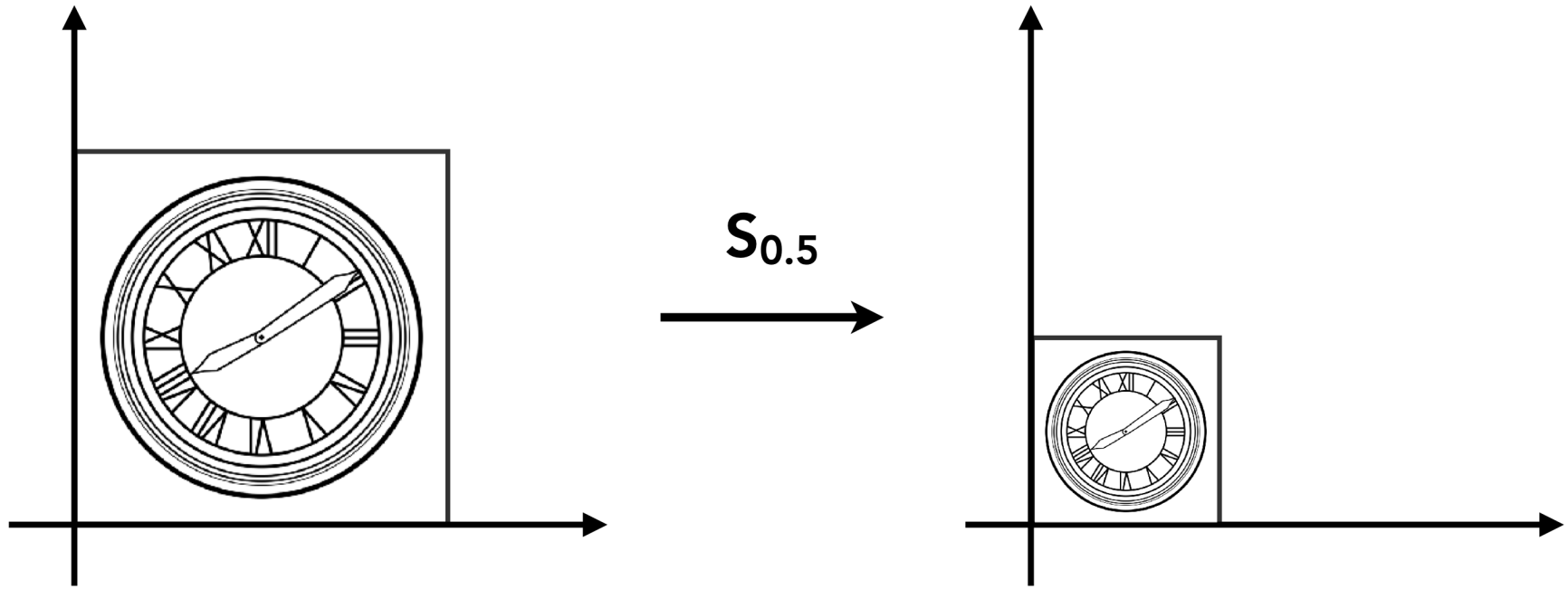


$$x' = sx$$

$$y' = sy$$

slide courtesy of Ren Ng

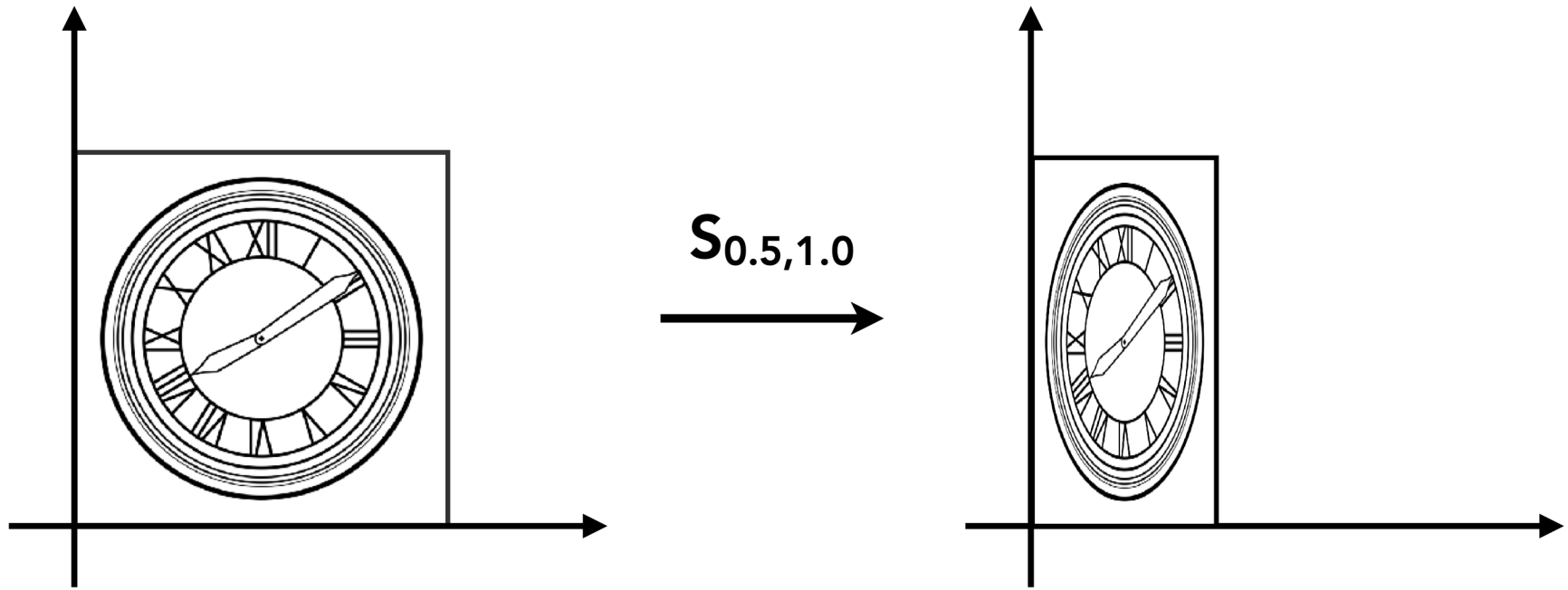
Scale Matrix



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

slide courtesy of Ren Ng

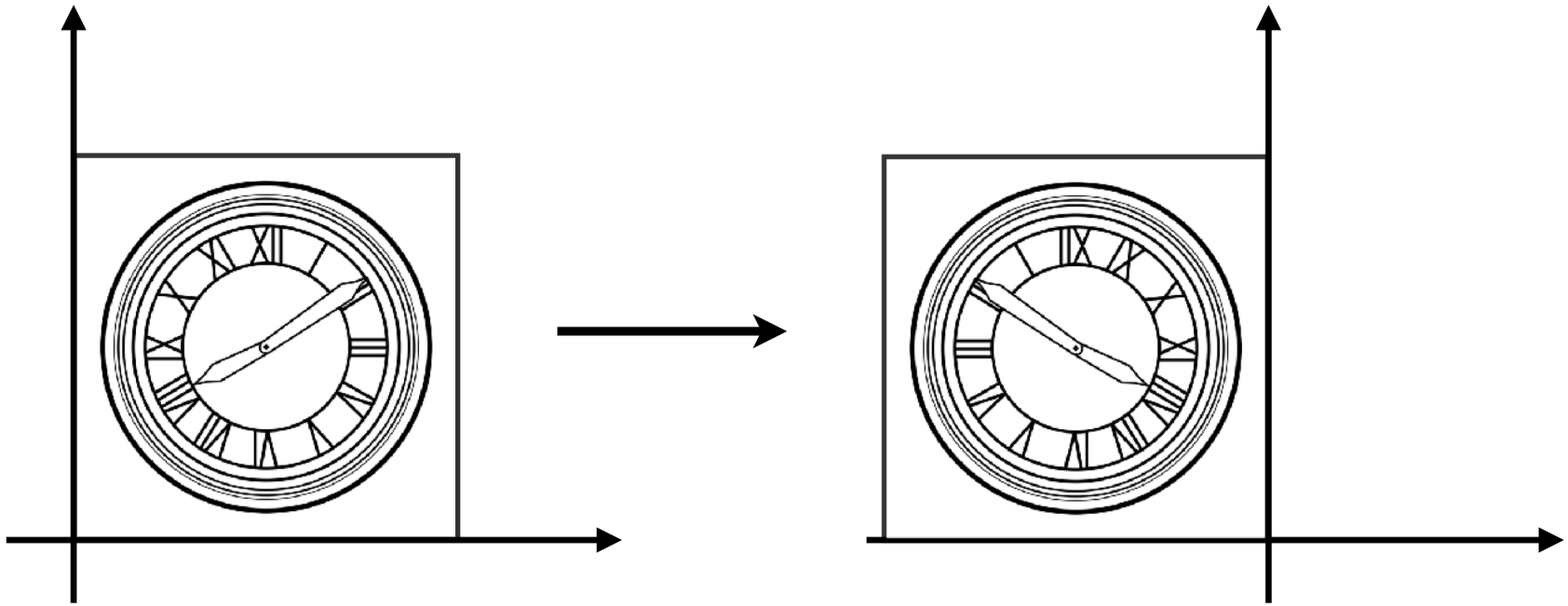
Scale (Non-Uniform)



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

slide courtesy of Ren Ng

Reflection Matrix



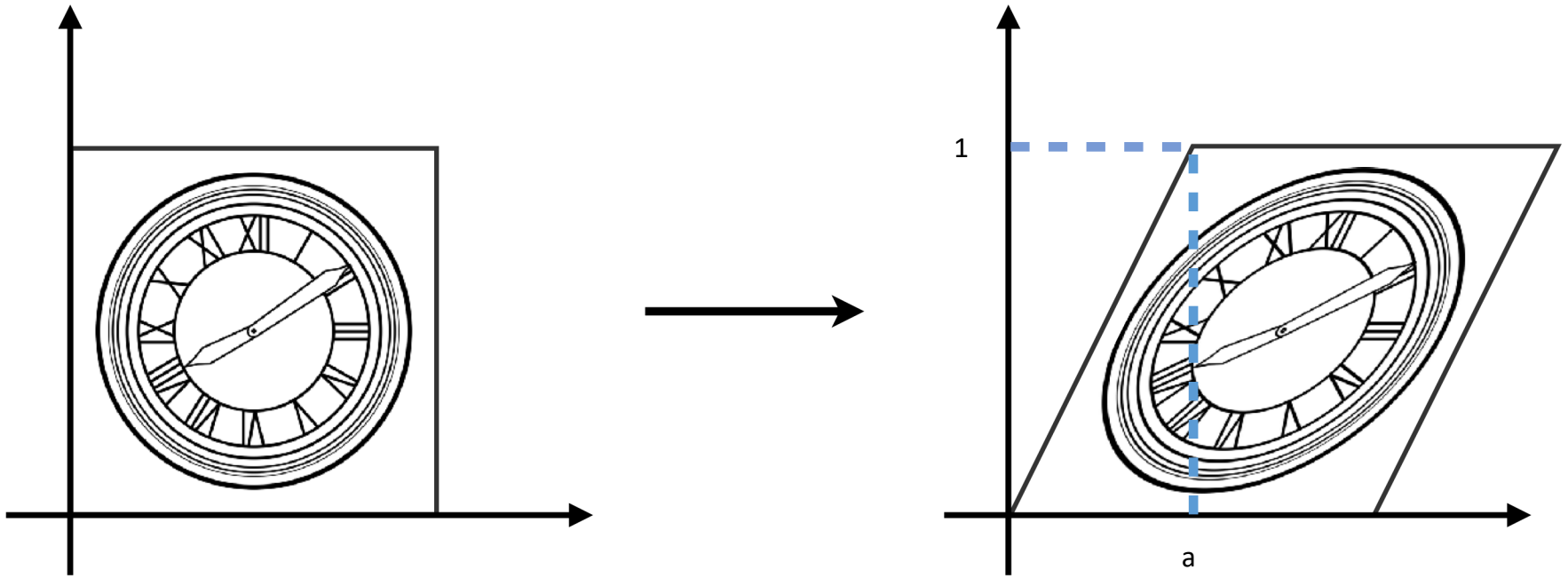
Horizontal reflection:

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

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

slide courtesy of Ren Ng

Shear Matrix



Hints:

Horizontal shift is 0 at y=0

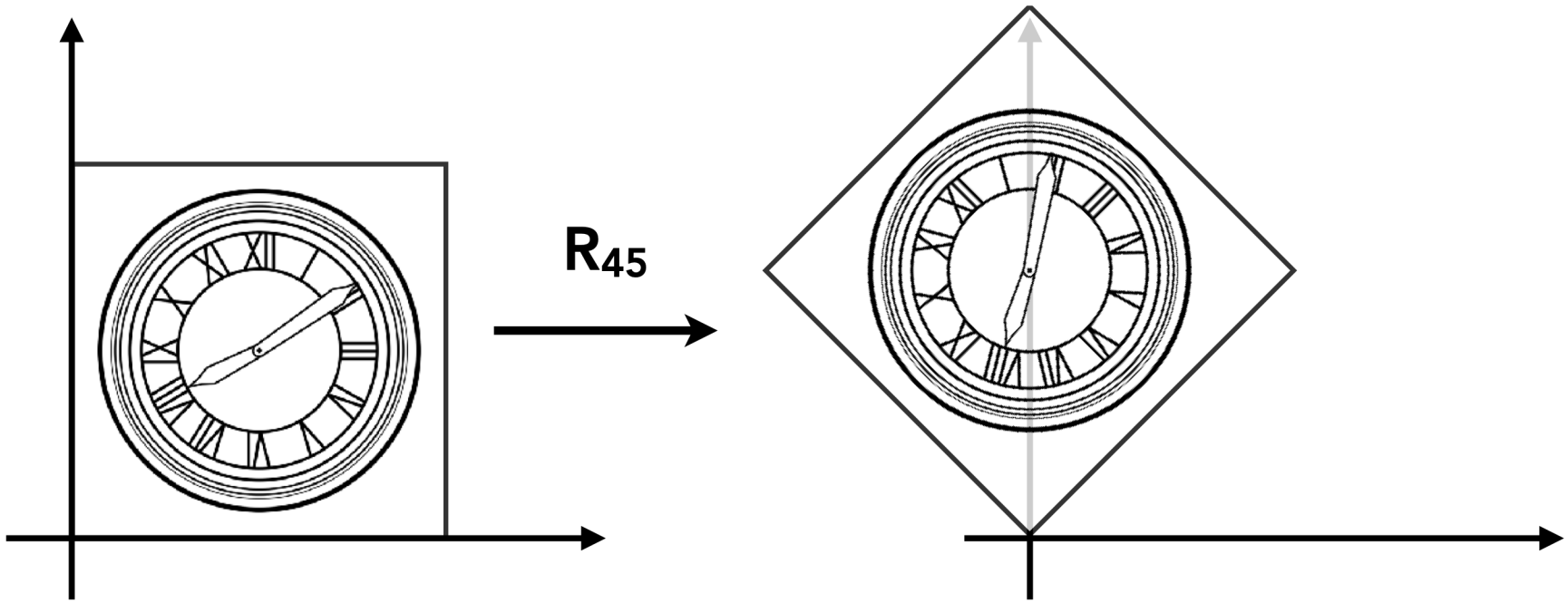
Horizontal shift is a at y=1

Vertical shift is always 0

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

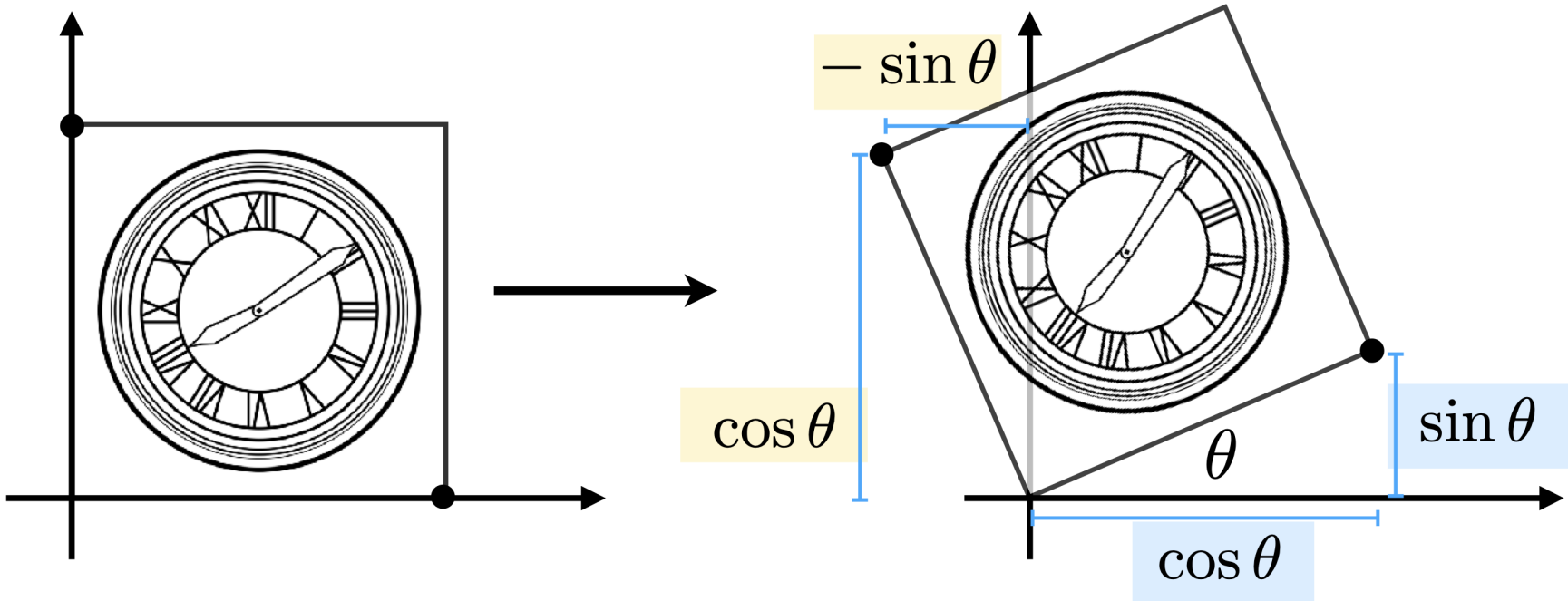
slide courtesy of Ren Ng

Rotate



slide courtesy of Ren Ng

Rotation Matrix



$$\mathbf{R}_\theta = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

slide courtesy of Ren Ng

Linear Transforms = Matrices

(of the same dimension)

$$x' = a\,x + b\,y$$
$$y' = c\,x + d\,y$$

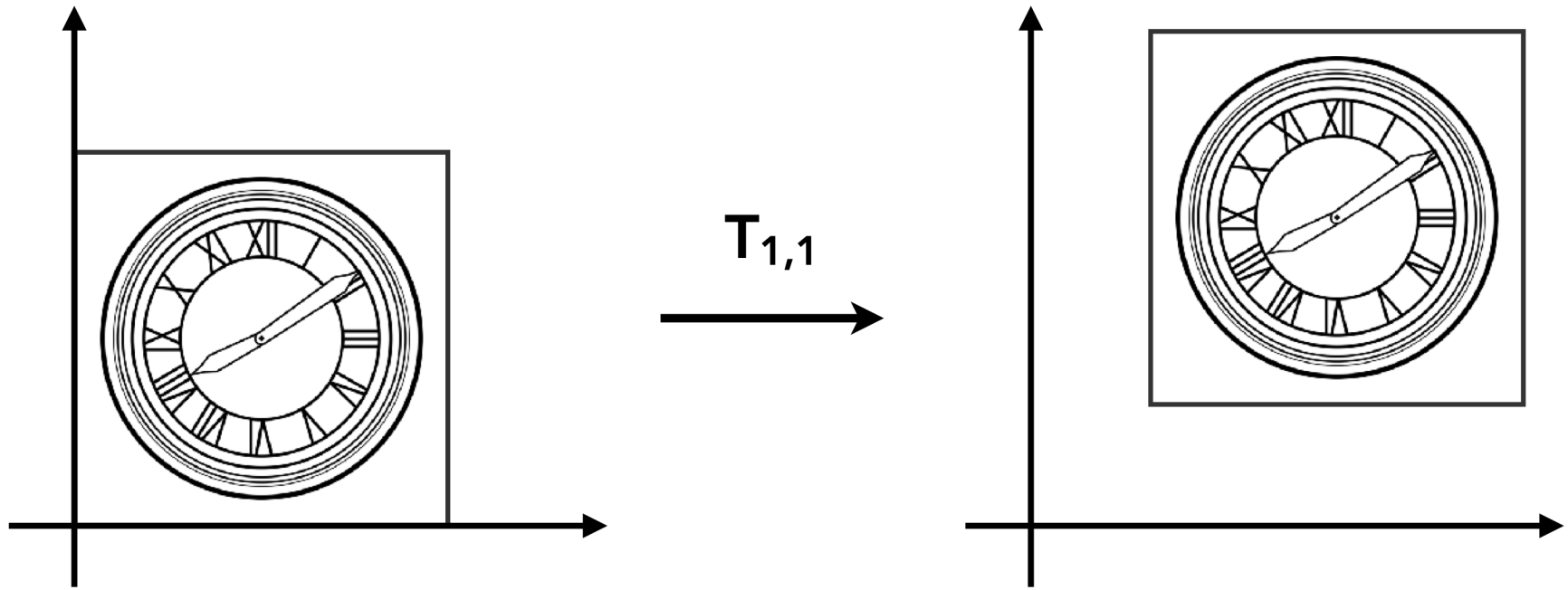
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\mathbf{x}' = \mathbf{M}\,\mathbf{x}$$

slide courtesy of Ren Ng

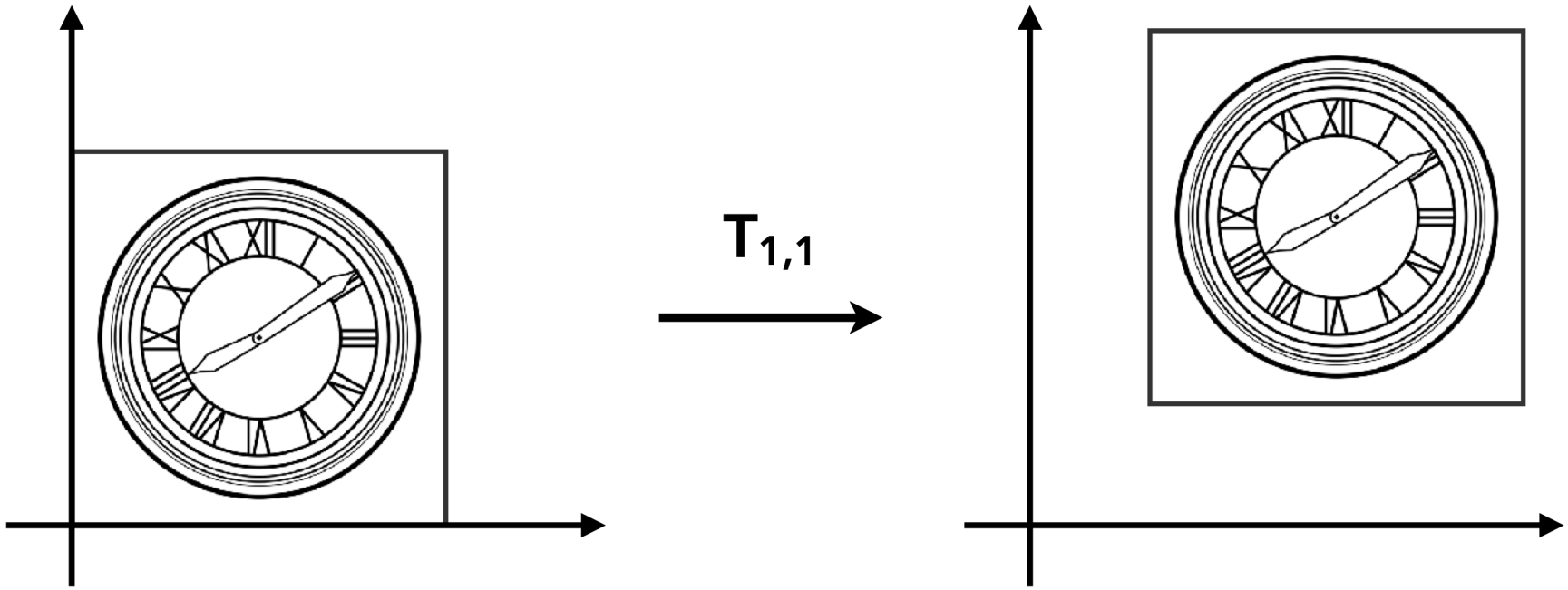
- Homogeneous Coordinates
  - Why homogeneous coordinates
  - Affine transformation

Translate



slide courtesy of Ren Ng

Translation??



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

slide courtesy of Ren Ng

Why Homogeneous Coordinates

- Translation cannot be represented in matrix form

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

(So, translation is NOT linear transform)

- But we don't want translation to be a special case
- Is there a unified way to represent all transformations?
  (and what's the cost?)

Solution: Homogenous Coordinates

Add a third coordinate (*w-coordinate*)

- **2D point = (x, y, 1)^T**
- **2D vector = (x, y, 0)^T**

Matrix representation of translations

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

slide courtesy of Ren Ng

Affine Transformations

Affine map = linear map + translation

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}$$

Using homogenous coordinates:

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} a & b & t_x \\ c & d & t_y \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

slide courtesy of Ren Ng

2D Transformations

Scale

$$\mathbf{S}(s_x, s_y) = \begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Rotation

$$\mathbf{R}(\alpha) = \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Translation

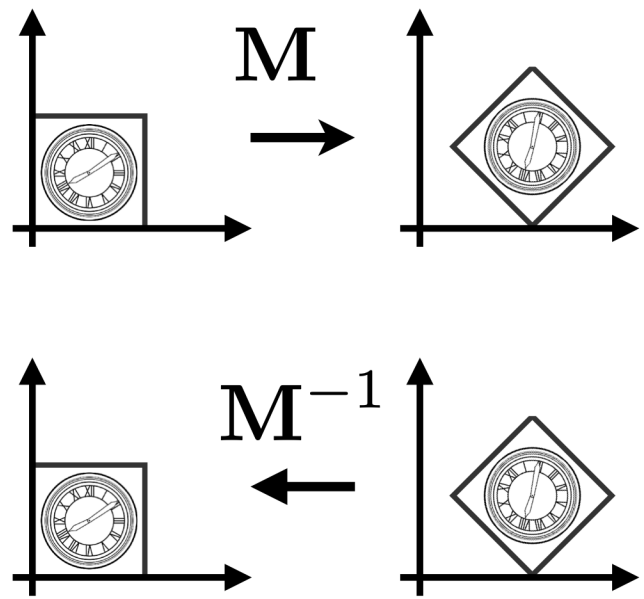
$$\mathbf{T}(t_x, t_y) = \begin{pmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{pmatrix}$$

slide courtesy of Ren Ng

Inverse Transform

$$\mathbf{M}^{-1}$$

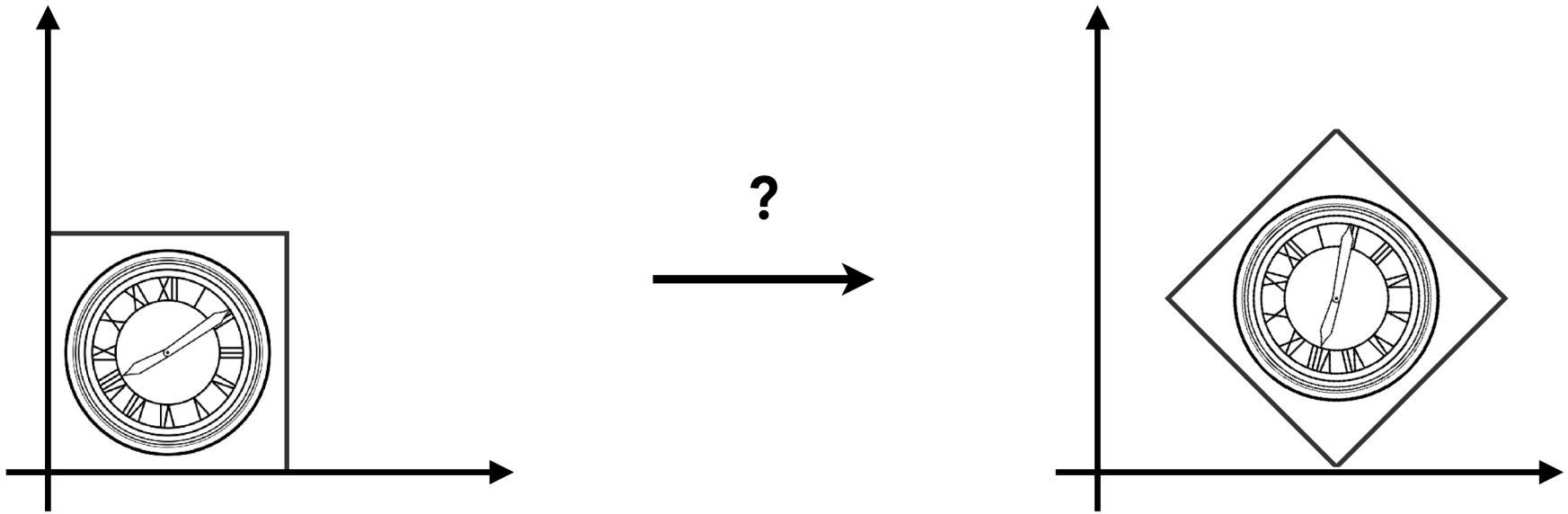
$\mathbf{M}^{-1}$ **is the inverse of transform** $\mathbf{M}$ **in both a matrix and geometric sense**



slide courtesy of Ren Ng

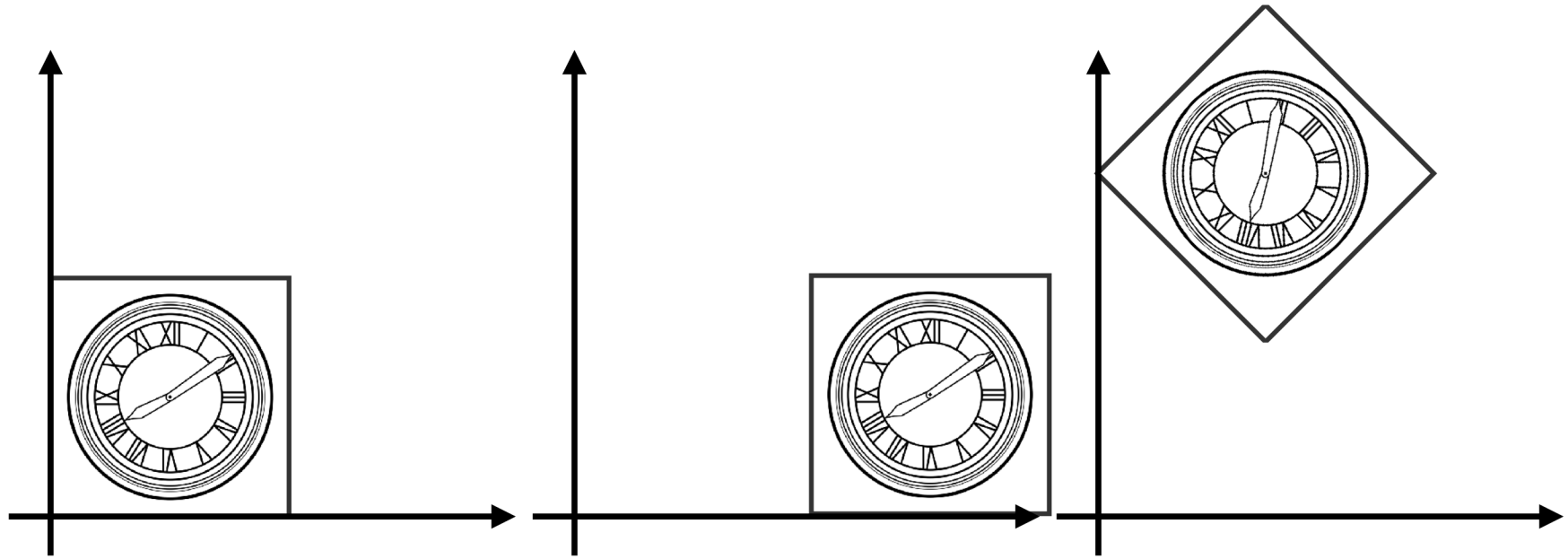
Composing Transforms

Composite Transform



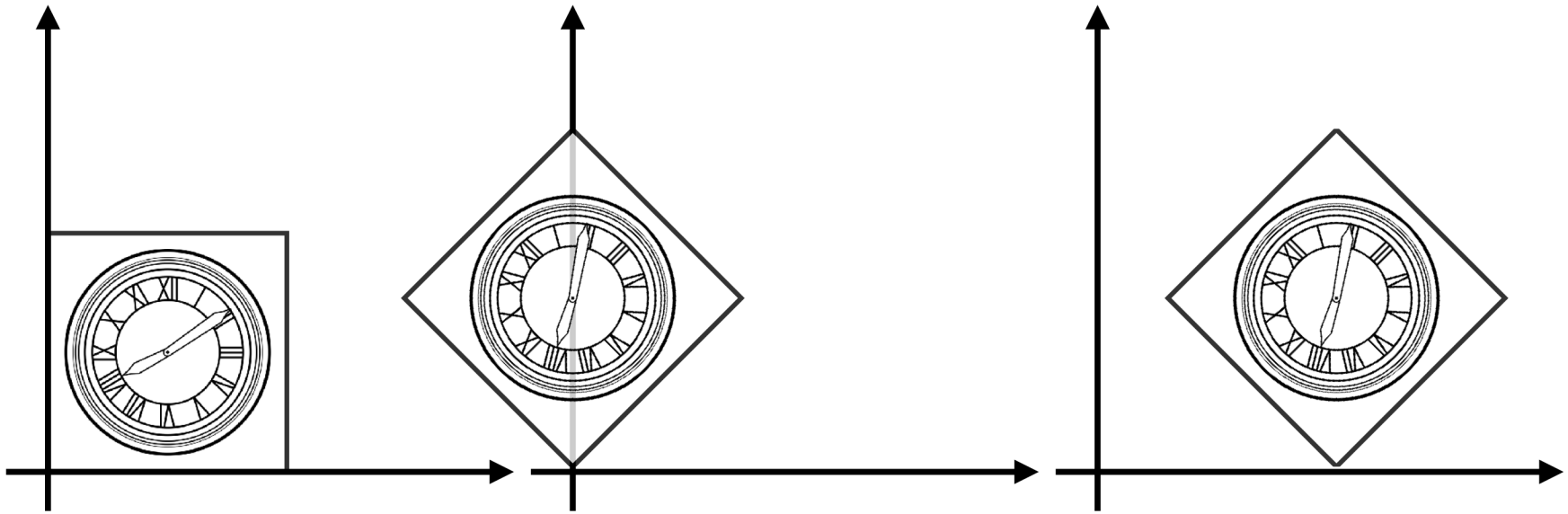
slide courtesy of Ren Ng

Translate Then Rotate?



slide courtesy of Ren Ng

Rotate Then Translate

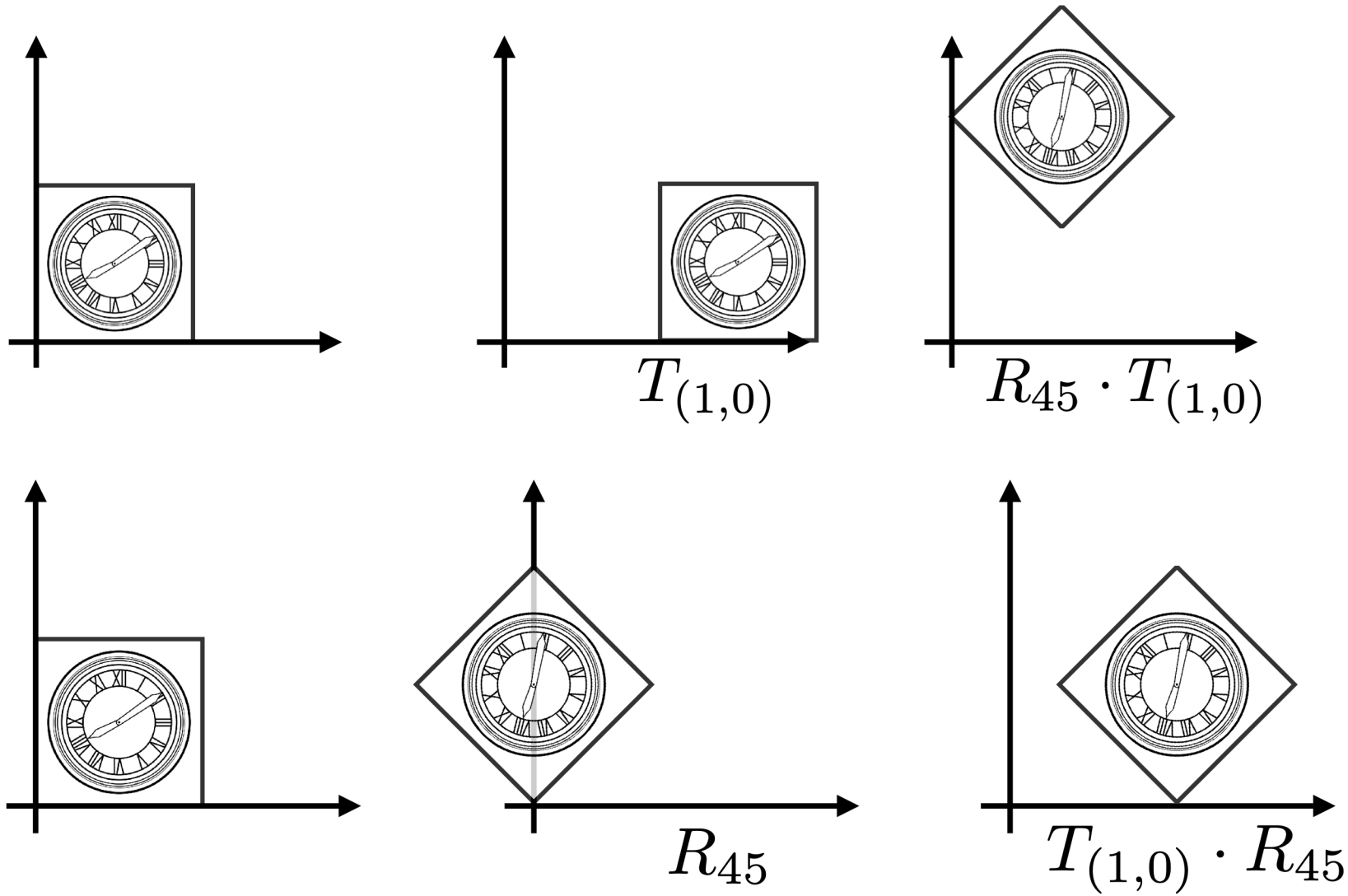


R_{45}

$T_{(1,0)} \cdot R_{45}$

slide courtesy of Ren Ng

Transform Ordering Matters!



slide courtesy of Ren Ng

Transform Ordering Matters!

Matrix multiplication is <u>not</u> commutative

$$R_{45} \cdot T_{(1,0)} \neq T_{(1,0)} \cdot R_{45}$$

Recall the matrix math represented by these symbols:

$$\begin{bmatrix} \cos 45° & -\sin 45° & 0 \\ \sin 45° & \cos 45° & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \neq \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos 45° & -\sin 45° & 0 \\ \sin 45° & \cos 45° & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Note that matrices are applied right to left:

$$T_{(1,0)} \cdot R_{45} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos 45° & -\sin 45° & 0 \\ \sin 45° & \cos 45° & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

slide courtesy of Ren Ng

Composing Transforms

Sequence of affine transforms A_1, A_2, A_3, ...

- **Compose by matrix multiplication**
  - **Very important for performance!**

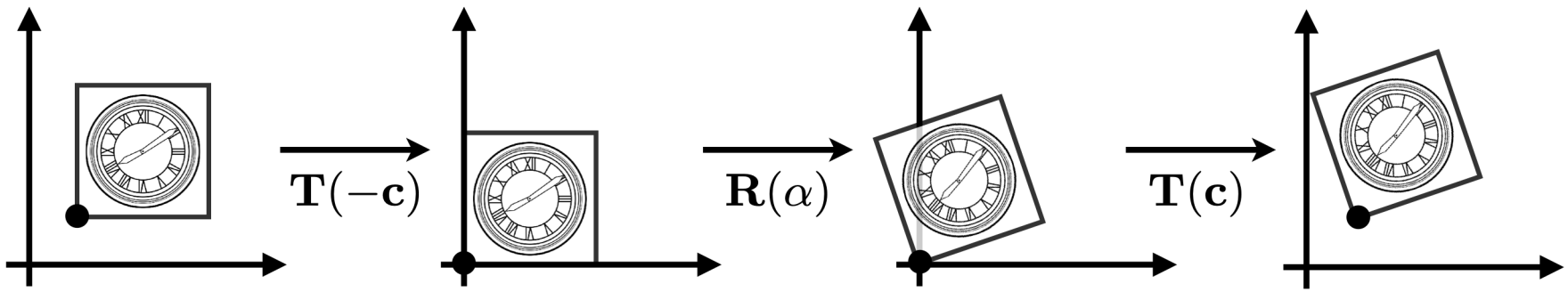
$$A_n(\ldots A_2(A_1(\mathbf{x}))) = \underbrace{\mathbf{A}_n \cdots \mathbf{A}_2 \cdot \mathbf{A}_1}_{\text{Pre-multiply } n \text{ matrices to obtain a single matrix representing combined transform}} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

slide courtesy of Ren Ng

Decomposing Complex Transforms

How to rotate around a given point c?

1. Translate center to origin
2. Rotate
3. Translate back

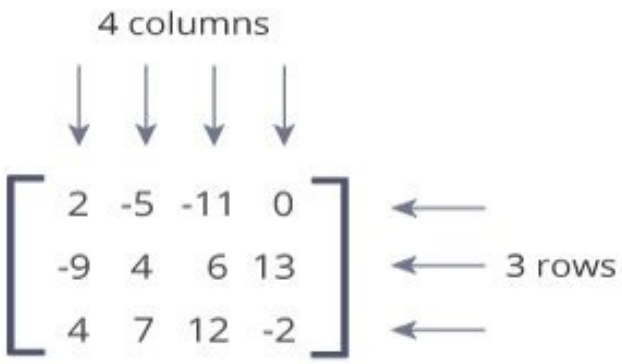


Matrix representation?

$$\mathbf{T}(\mathbf{c}) \cdot \mathbf{R}(\alpha) \cdot \mathbf{T}(-\mathbf{c})$$

slide courtesy of Ren Ng

Basic Matrix in Python



Row-Majored

```
A = [[1, 4, 5],
     [-5, 8, 9]]
```

$$\begin{bmatrix} 1 & 4 & 5 \\ -5 & 8 & 9 \end{bmatrix}$$

NYU

Basic Matrix in Python

```
A = [[1, 4, 5, 12],
     [-5, 8, 9, 0],
     [-6, 7, 11, 19]]

print("A =", A)
print("A[1] =", A[1])      # 2nd row
print("A[1][2] =", A[1][2])   # 3rd element of 2nd row
print("A[0][-1] =", A[0][-1])   # Last element of 1st Row

column = [];        # empty list
for row in A:
  column.append(row[2])

print("3rd column =", column)
```

```
A = [[1, 4, 5, 12], [-5, 8, 9, 0], [-6, 7, 11, 19]]
A[1] = [-5, 8, 9, 0]
A[1][2] = 9
A[0][-1] = 12
3rd column = [5, 9, 11]
```

Basic Matrix Operations in Numpy

```
import numpy as np

A = np.array([[2, 4], [5, -6]])
B = np.array([[9, -3], [3, 6]])
C = A + B      # element wise addition
print(C)

'''
Output:
[[11  1]
 [ 8  0]]
 '''
```

Basic Matrix Operations in Numpy

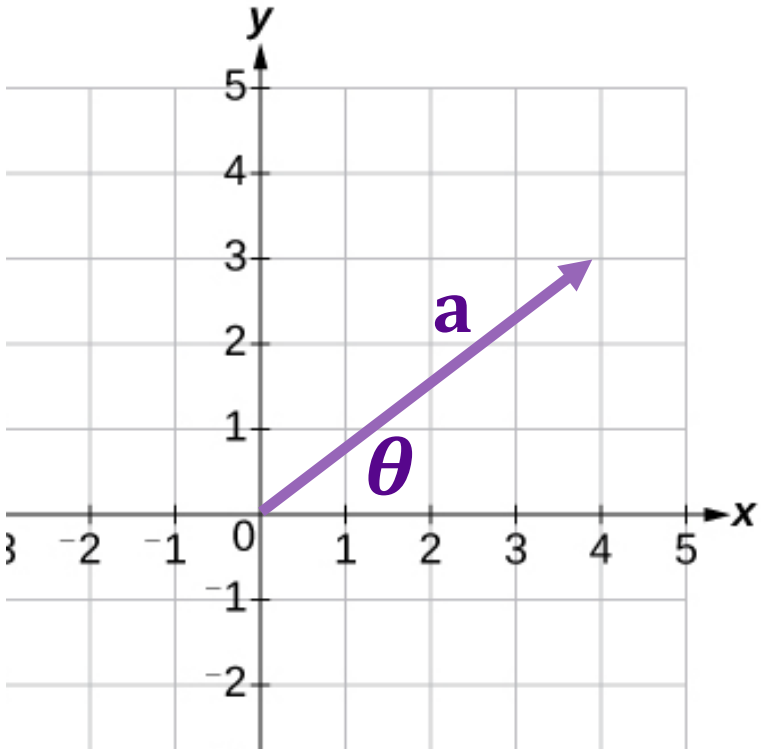
```
import numpy as np

A = np.array([[3, 6, 7], [5, -3, 0]])
B = np.array([[1, 1], [2, 1], [3, -3]])
C = A.dot(B)
print(C)

'''
Output:
[[ 36 -12]
 [ -1   2]]
'''
```

Complex Numbers

$$\mathbf{a} = \begin{pmatrix} 4 \\ 3 \end{pmatrix} = 4\mathbf{x} + 3\mathbf{y} = \mathbf{4 + 3i}$$

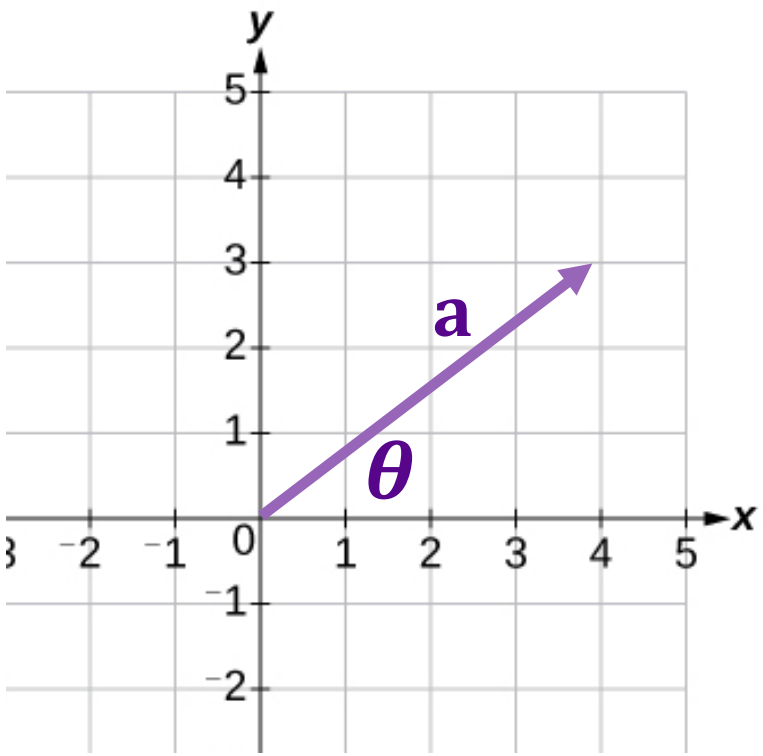


$$\frac{\mathbf{a}}{|\mathbf{a}|} = \cos(\theta) + \mathrm{i}\sin(\theta)$$

$$= e^{i\theta}$$

Euler's Formula

Complex Numbers



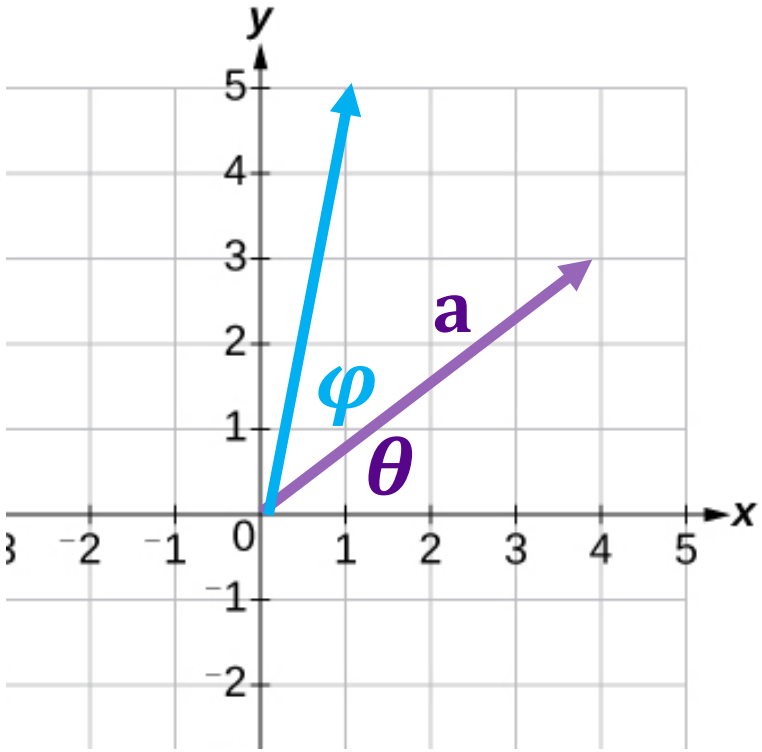
$$(x + yi) + (u + vi) = (x + u) + (y + v)i$$

$$(x + yi)\,(u + vi) = (xu - yv) + (xv + yu)i.$$

$$\frac{1}{z} = \frac{\bar{z}}{z\bar{z}} = \frac{\bar{z}}{|z|^2} = \frac{\bar{z}}{x^2 + y^2} = \frac{x}{x^2 + y^2} - \frac{y}{x^2 + y^2}i$$

Complex Numbers

Representing rotations



$$e^{i\theta}\ e^{i\varphi} = e^{i(\theta+\varphi)}$$

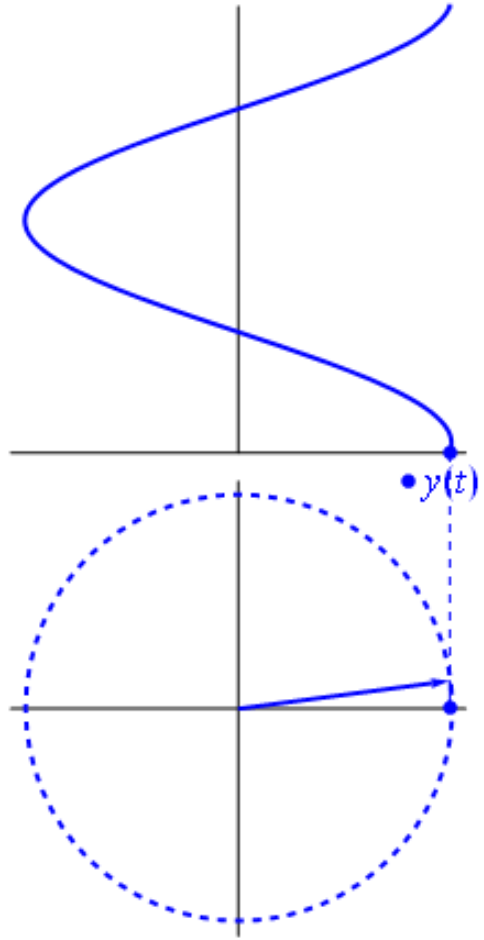
Complex Numbers

Representing light waves

$$A \cdot \cos(\omega t + \theta) = \frac{A \cdot e^{j(\omega t+\theta)}}{2} + \frac{A \cdot e^{-j(\omega t+\theta)}}{2}$$

$$\begin{aligned} A \cdot \cos(\omega t + \theta) &= \mathrm{Re}\left\{ A \cdot e^{j(\omega t+\theta)} \right\} \\ &= \mathrm{Re}\left\{ A e^{j\theta} \cdot e^{j\omega t} \right\} \end{aligned}$$

$$A e^{j\theta}$$



Questions?