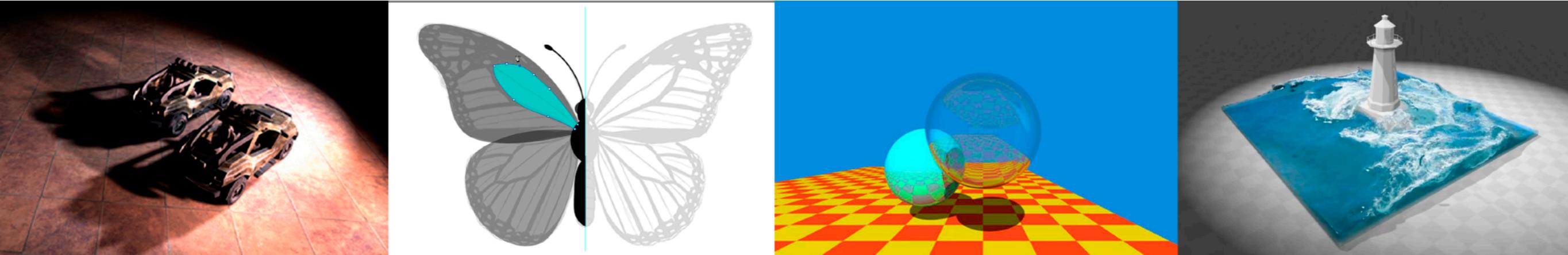


# Introduction to Computer Graphics

AMES101, Lingqi Yan, UC Santa Barbara

## Lecture 3: Transformation



# Announcements

- Assignment 0 will be released soon
- Do not register the homework submission system with QQ email

# Last Lecture

- Vectors
  - Basic operations: addition, multiplication
- Dot Product
  - Forward / backward (dot product positive / negative)
- Cross Product
  - Left / right (cross product outward / inward)
- Matrices

# This Week

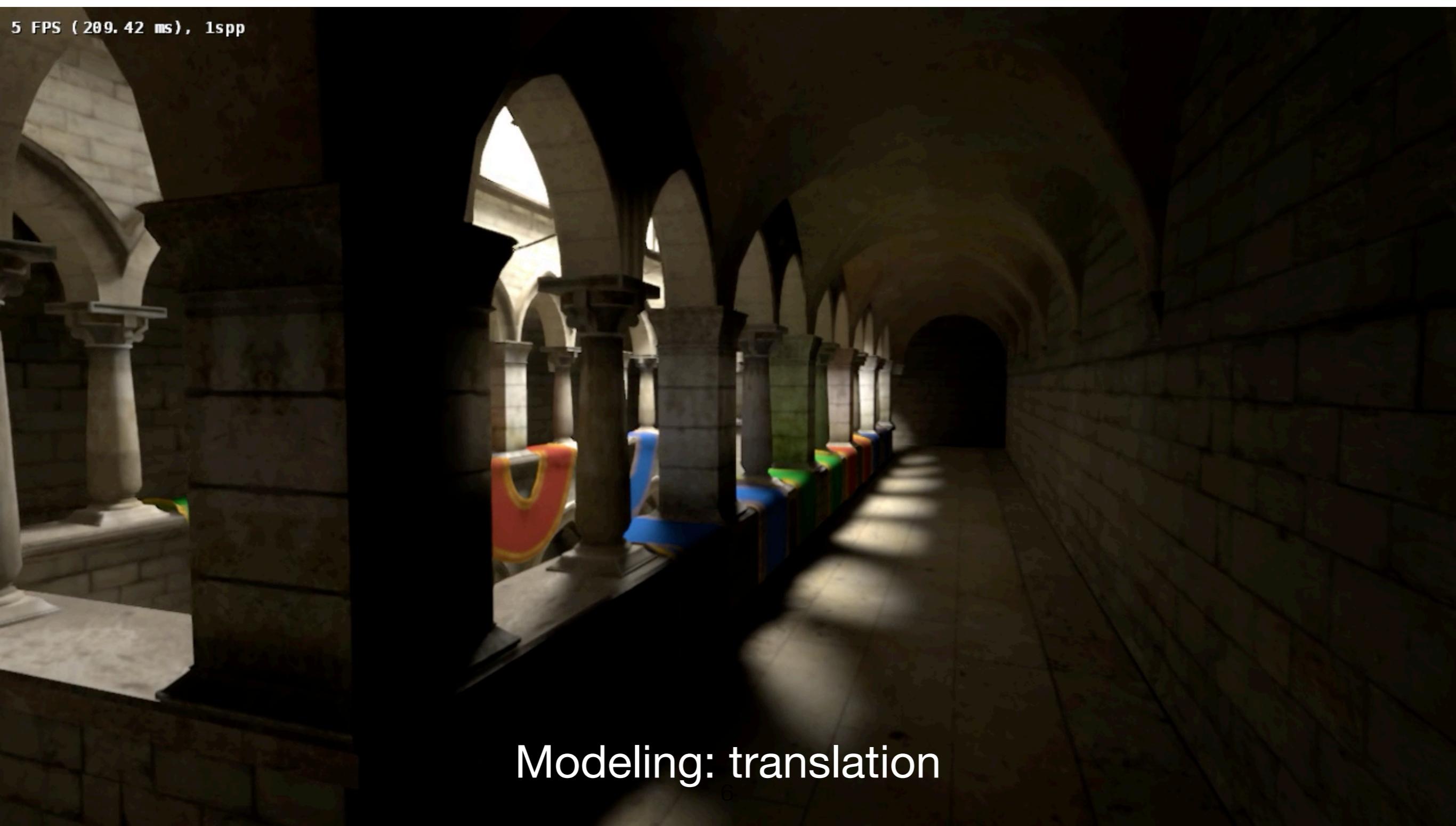
- Transformation!
- Today
  - Why study transformation
  - 2D transformations: rotation, scale, shear
  - Homogeneous coordinates
  - Composing transforms
  - 3D transformations

# Today

- Why study transformation
  - Modeling
  - Viewing
- 2D transformations
- Homogeneous coordinates

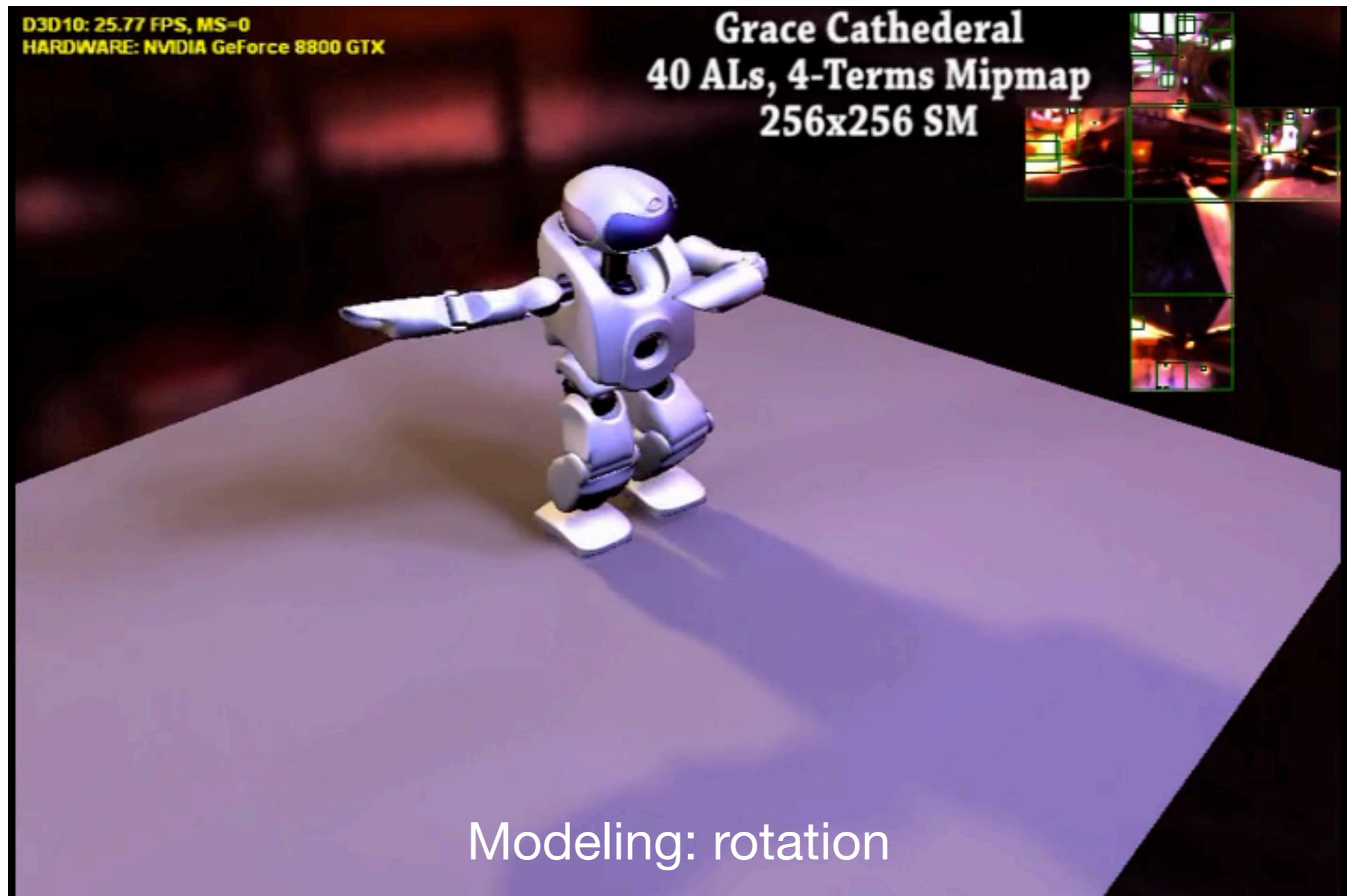
# Why Transformation?

5 FPS ( 209.42 ms ), 1spp



Modeling: translation

# Why Transformation?



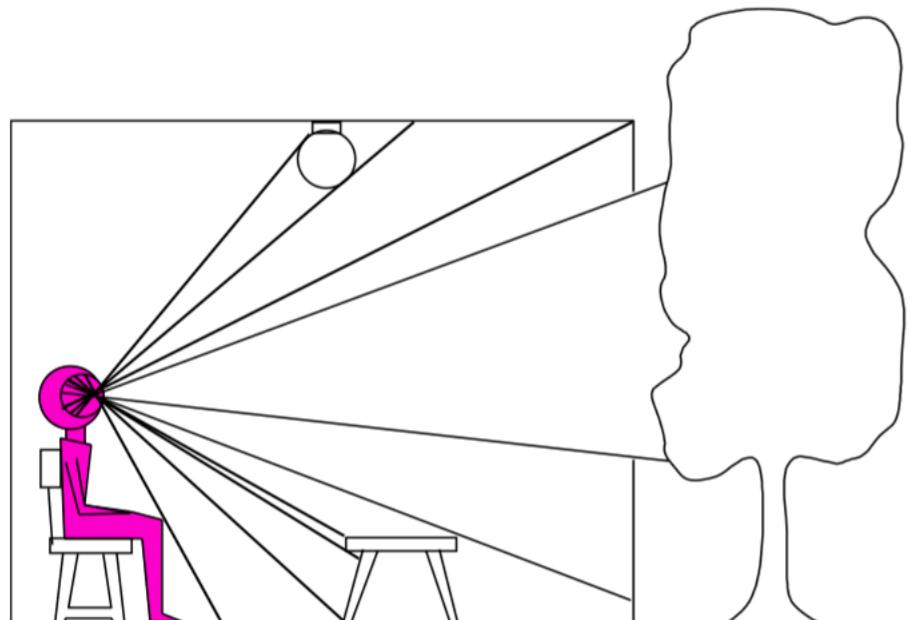
# Why Transformation?

P I X A R

Modeling: scaling

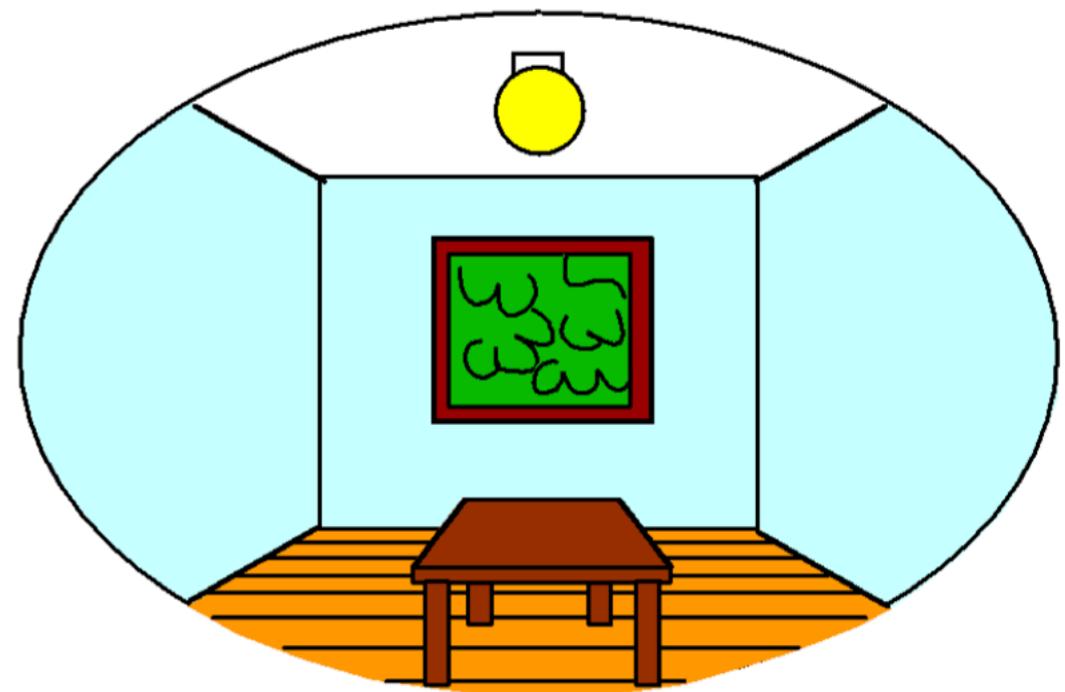
# Why Transformation?

*3D world*



Point of observation

*2D image*



Figures © Stephen E. Palmer, 2002

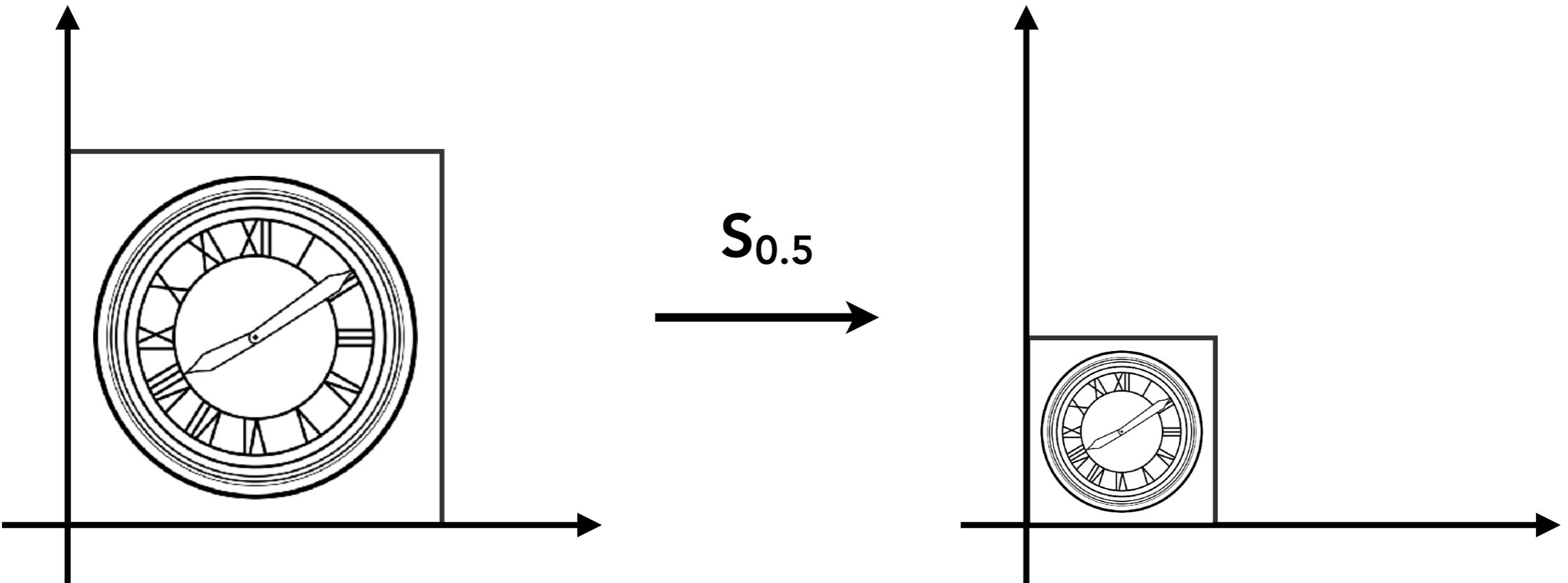
Viewing: (3D to 2D) projection

投影变换

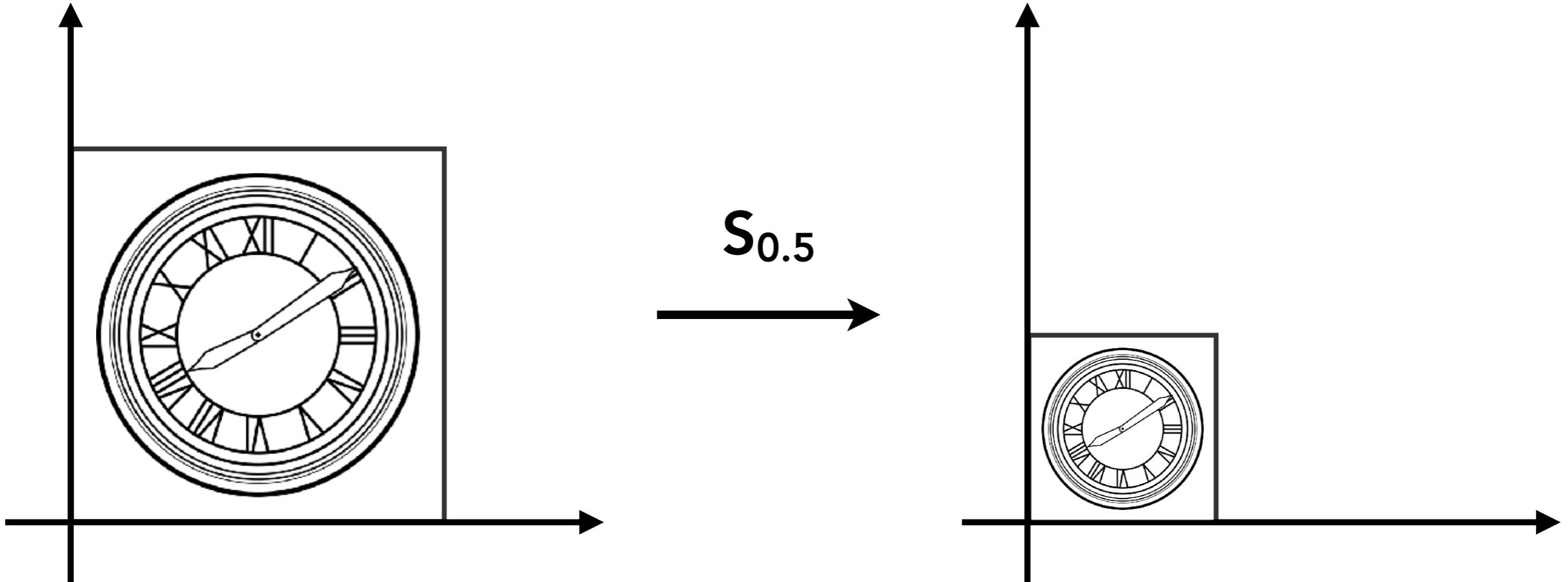
# Today

- Why study transformation
- 2D transformations
  - Representing transformations using matrices
  - Rotation, scale, shear
- Homogeneous coordinates

# Scale



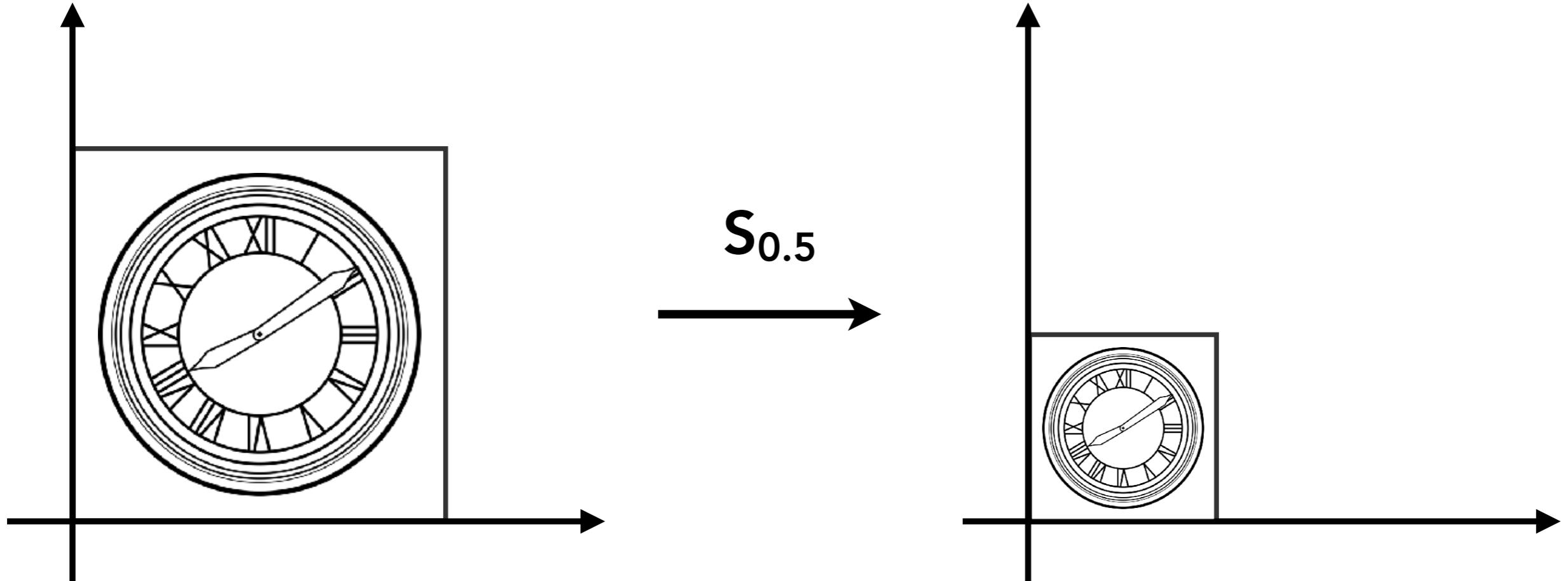
# Scale Transform



$$x' = sx$$

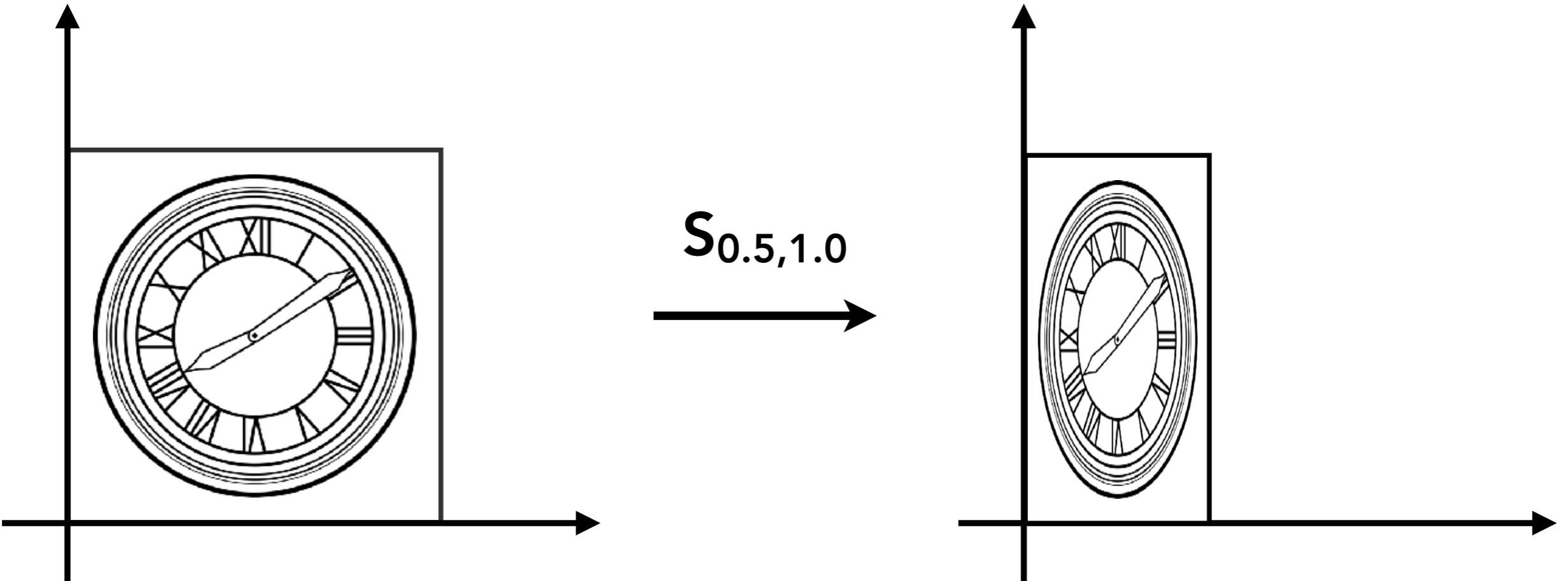
$$y' = sy$$

# Scale Matrix



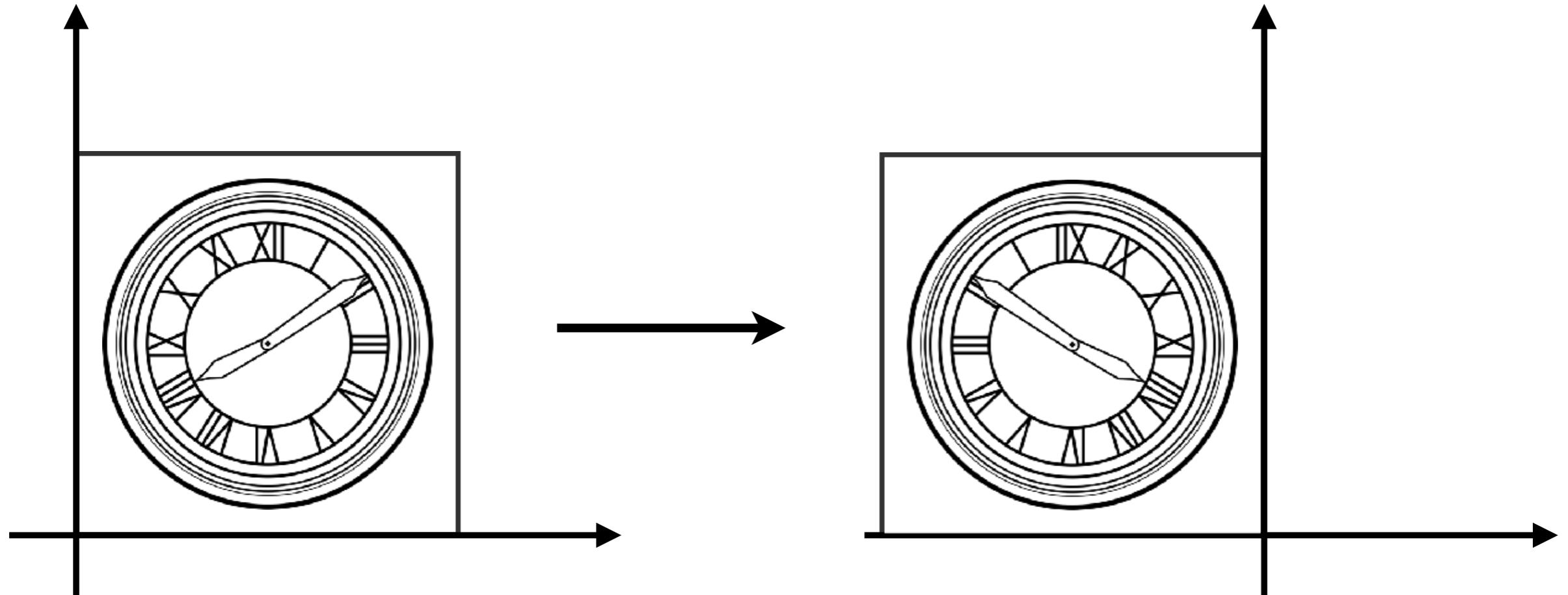
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} s & 0 \\ 0 & s \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

# Scale (Non-Uniform)



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

# Reflection Matrix



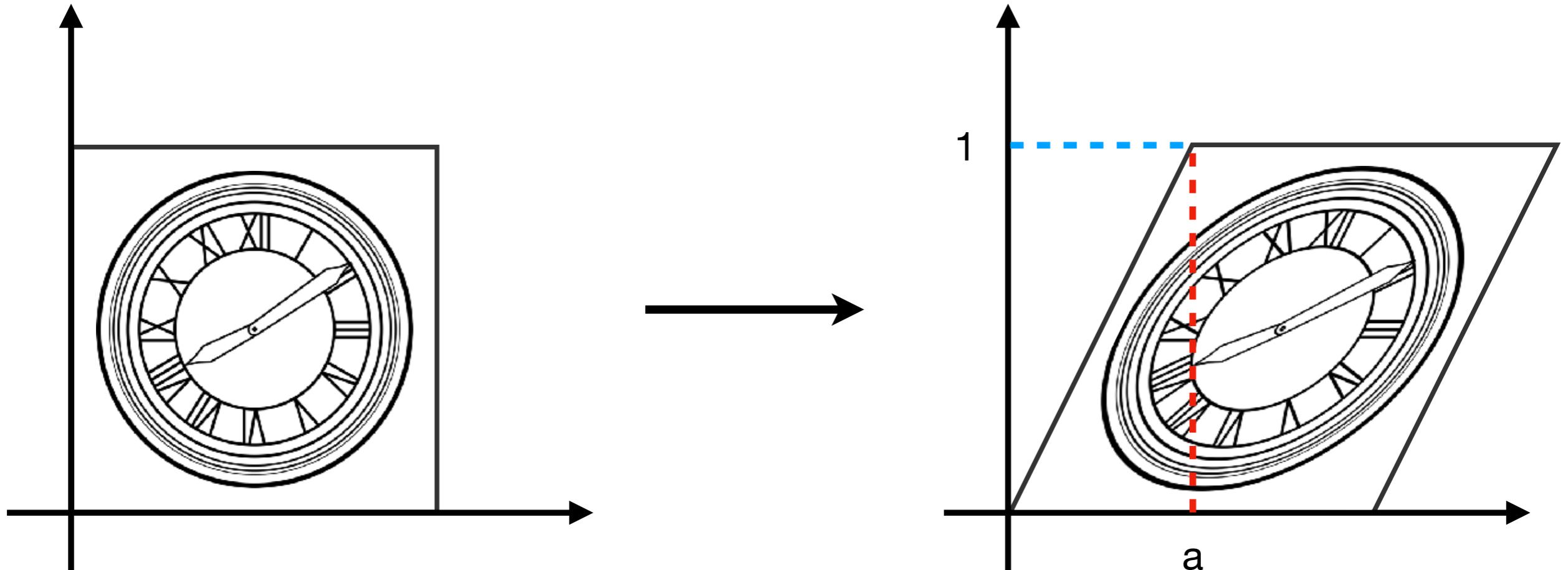
Horizontal reflection:

$$x' = -x$$

$$y' = y$$

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

# Shear Matrix



Hints:

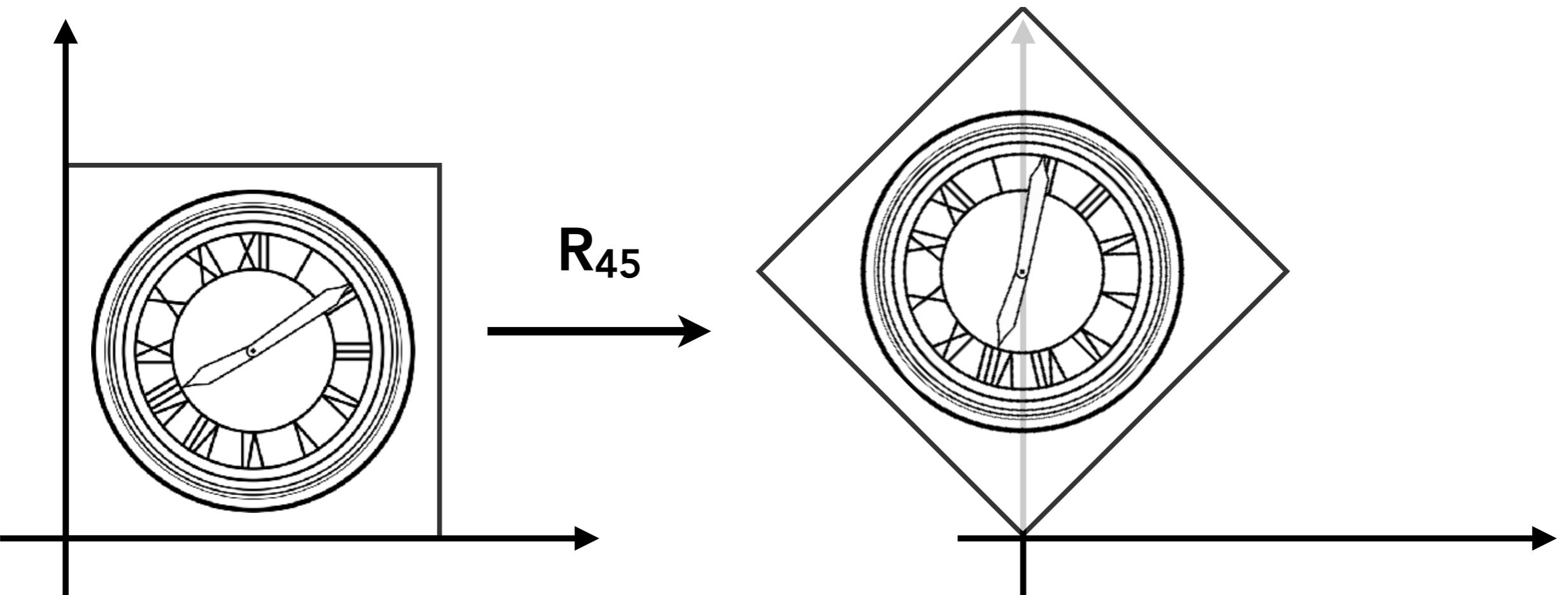
Horizontal shift is 0 at  $y=0$

Horizontal shift is  $a$  at  $y=1$

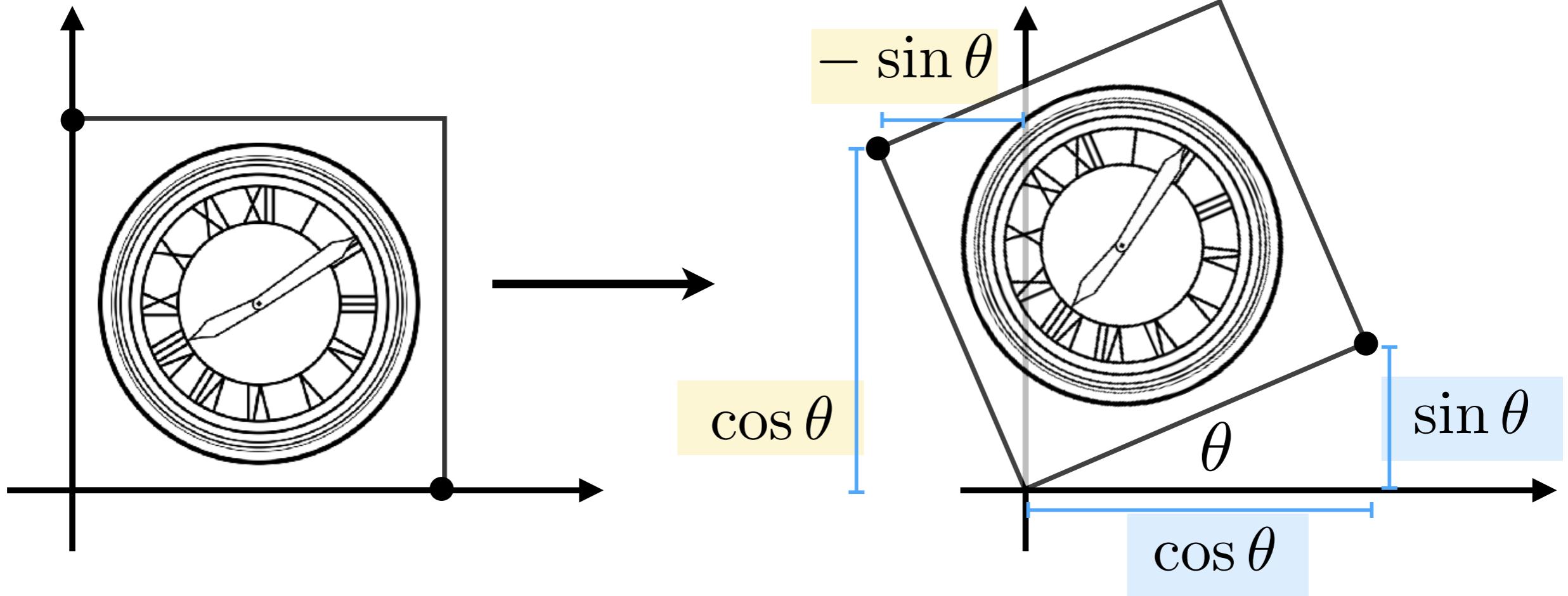
Vertical shift is always 0

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

# Rotate (about the origin (0, 0), CCW by default)



# Rotation Matrix



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

# Linear Transforms = Matrices

(of the same dimension)

$$x' = a x + b y$$

$$y' = c x + d y$$

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

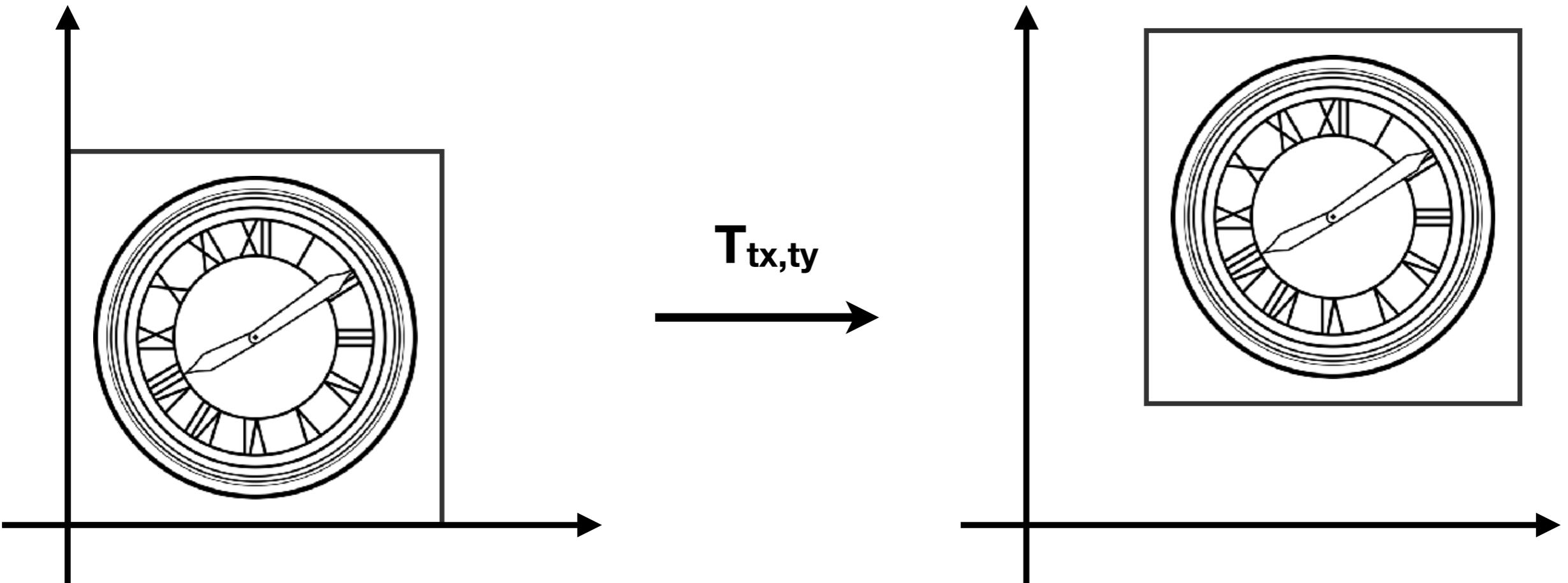
$$\mathbf{x}' = \mathbf{M} \mathbf{x}$$

# Questions?

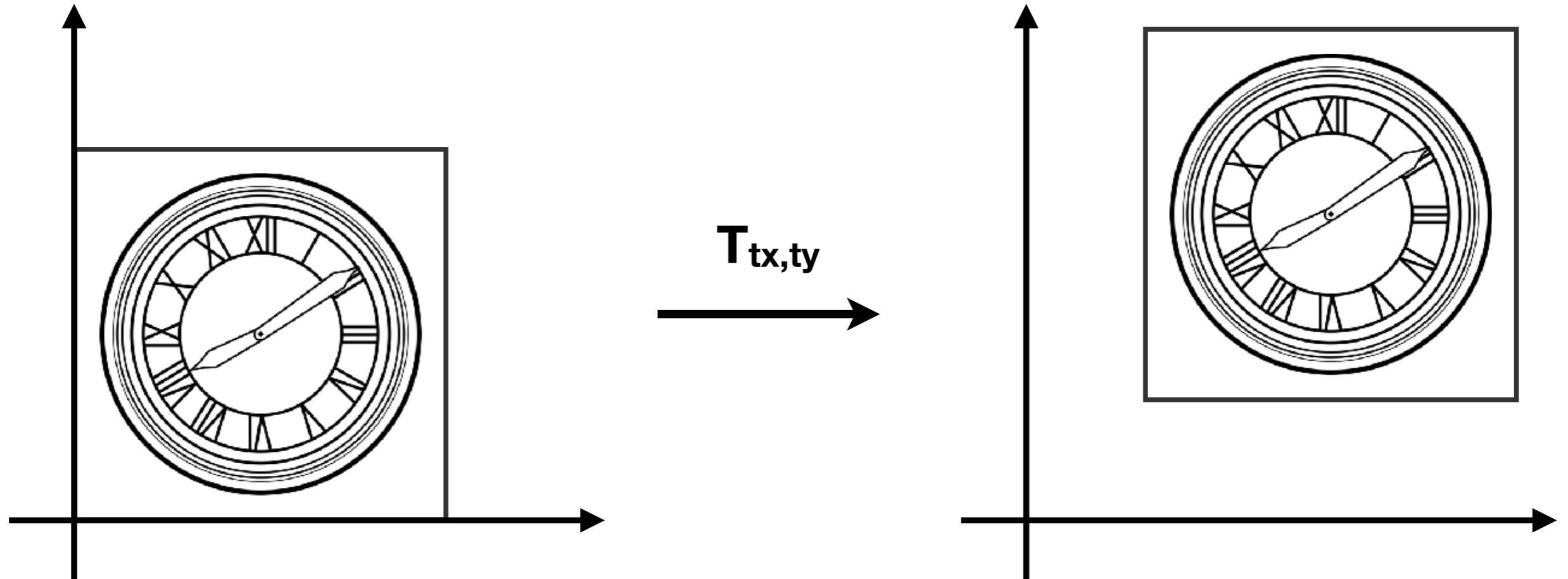
# Today

- Why study transformation
- 2D transformations
- **Homogeneous coordinates**
  - Why homogeneous coordinates
  - Affine transformation

# Translation



# Translation??



$$x' = x + t_x$$

$$y' = y + t_y$$

# Why Homogeneous Coordinates

引入齐次坐标的原因是无法将平移纳入矩阵表示形式

- 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 坐标转为 3D 坐标，在后面添加 1

- 2D vector =  $(x, y, 0)^T$

2D 向量转为 3D 向量，在后面添加 0，保证平移不变性

## Matrix representation of translations

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

What if you translate a vector?

# Homogenous Coordinates

Valid operation if w-coordinate of result is 1 or 0

- **vector + vector = vector** 向量加上向量，表明向量平移
- **point - point = vector** 点与点相减表明两点之间的向量
- **point + vector = point** 点 + 向量 = 点按向量方向平移向量模长的单位
- **point + point = ??** 点与点相加，表示两点中点

In homogeneous coordinates,

$$\begin{pmatrix} x \\ y \\ w \end{pmatrix} \text{ is the 2D point } \begin{pmatrix} x/w \\ y/w \\ 1 \end{pmatrix}, w \neq 0$$

# 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:**

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

# 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

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

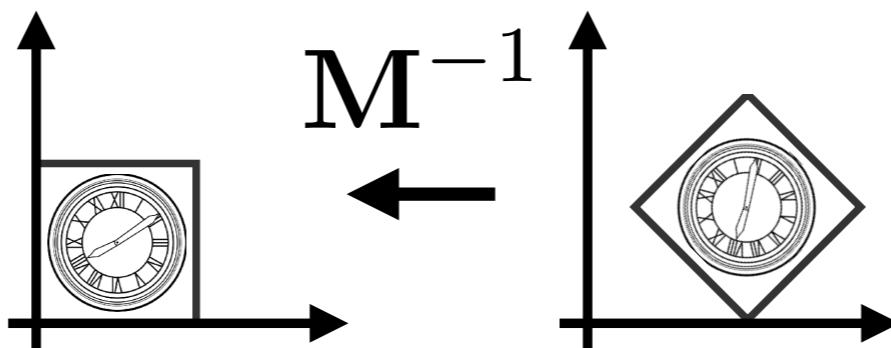
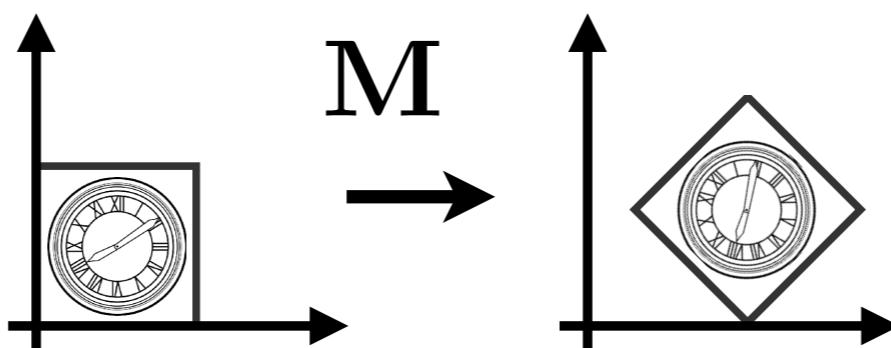
## Translation

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

# Inverse Transform

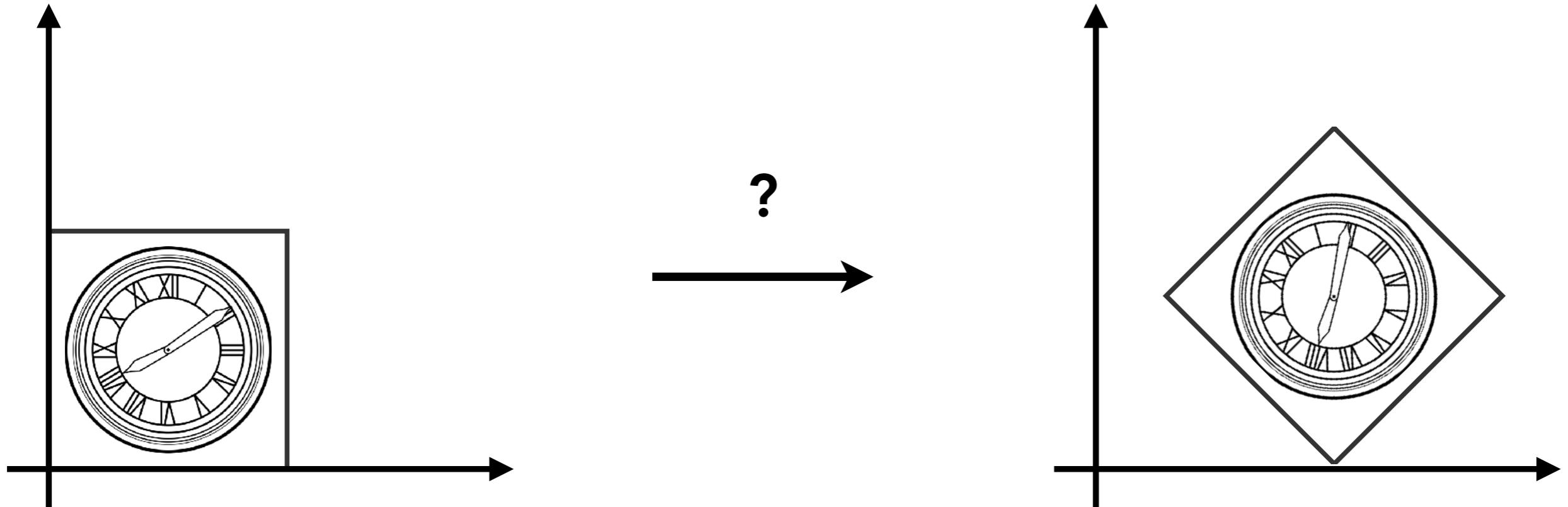
$$M^{-1}$$

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

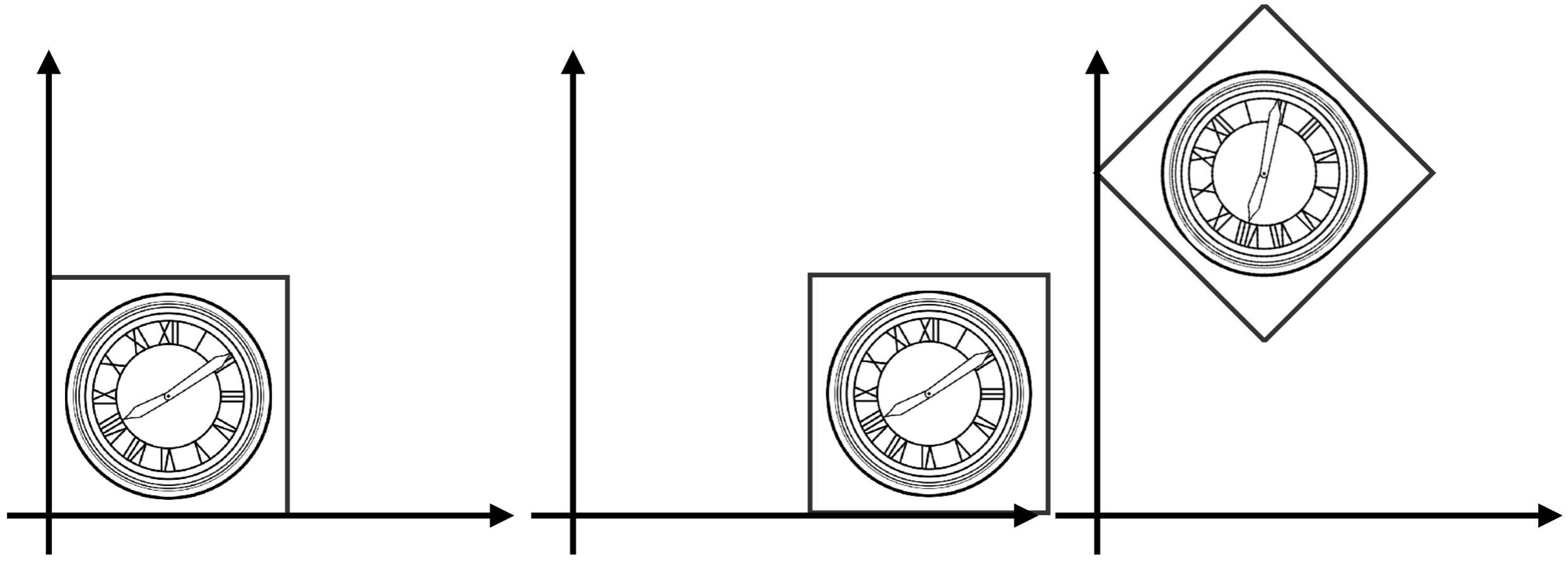


# **Composing Transforms**

# Composite Transform



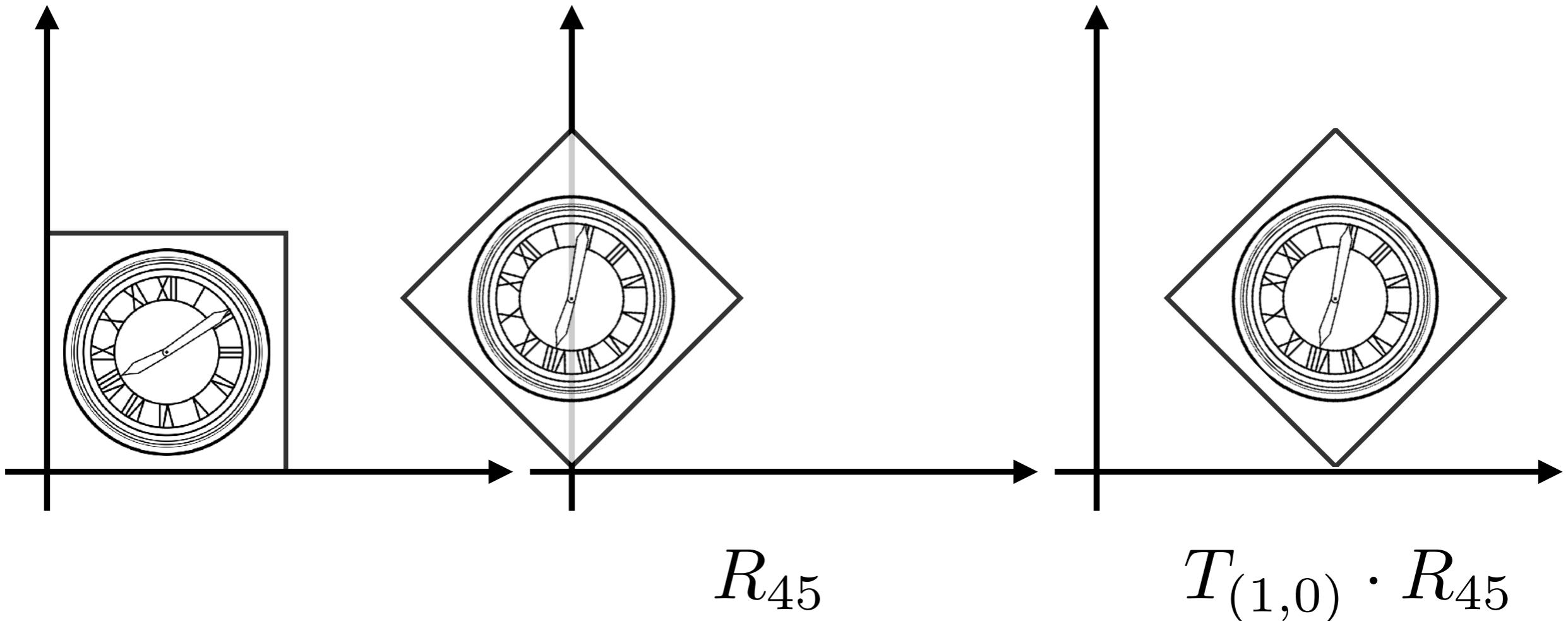
# Translate Then Rotate?



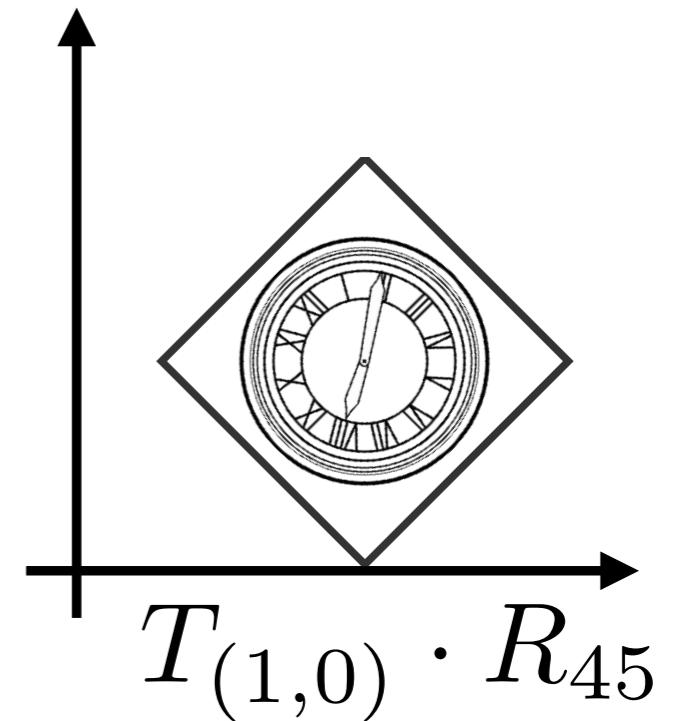
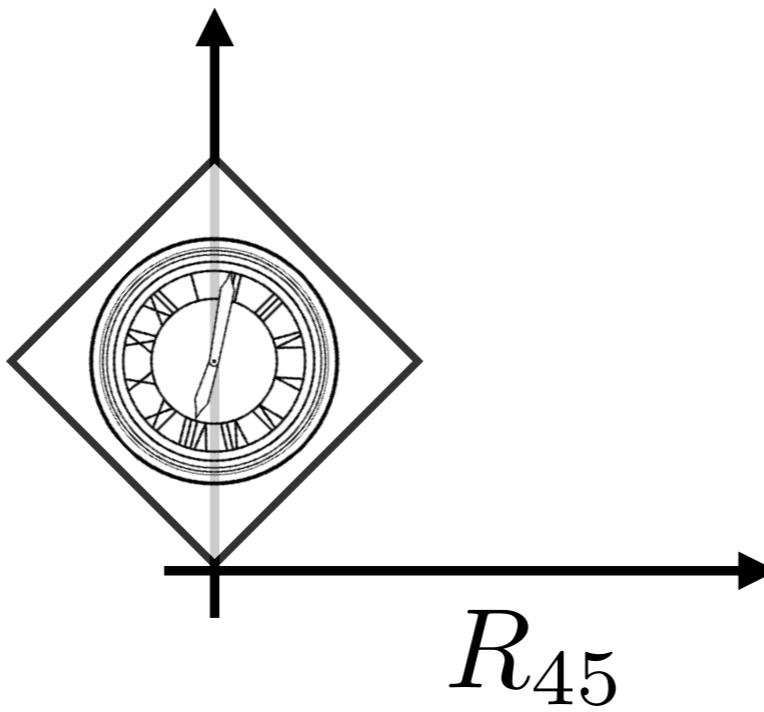
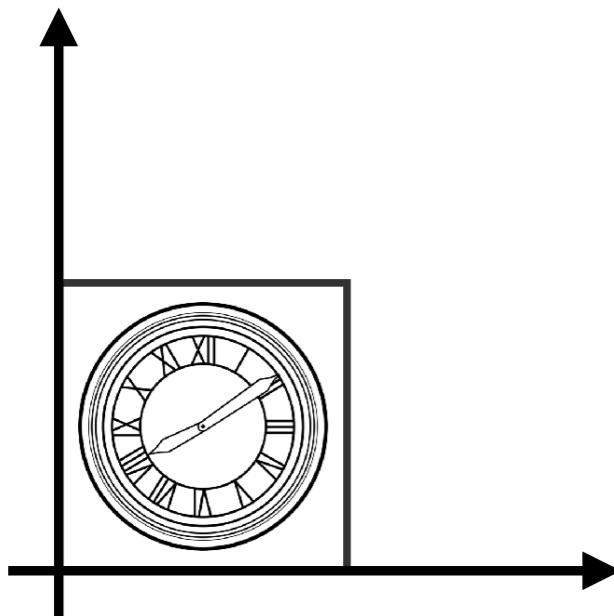
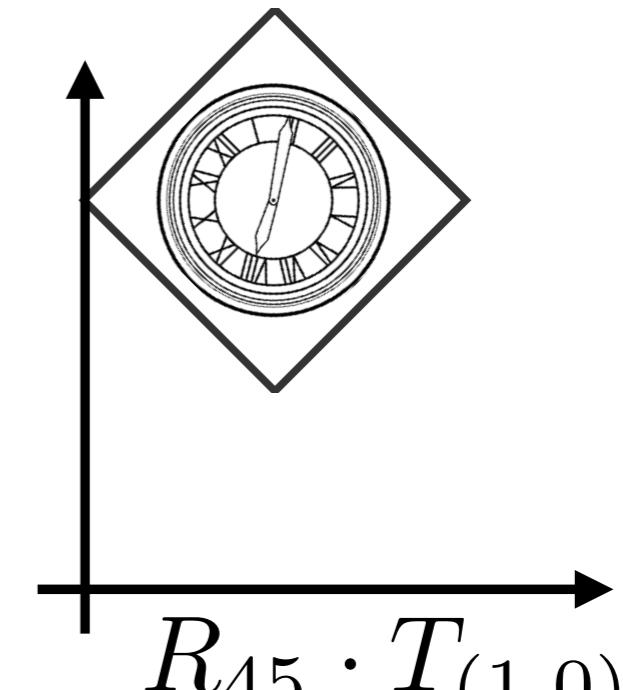
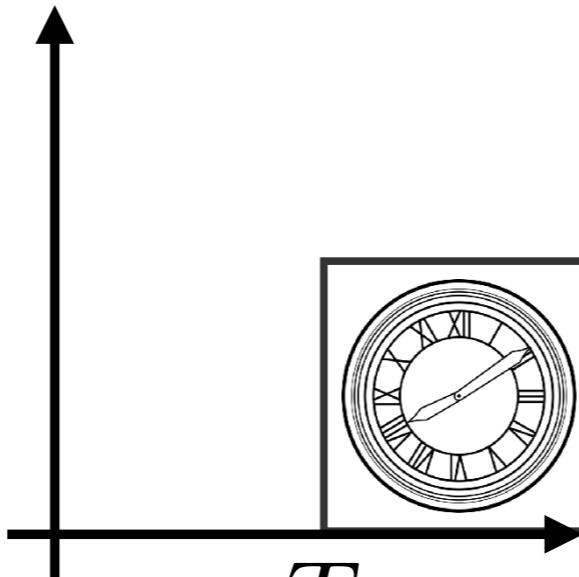
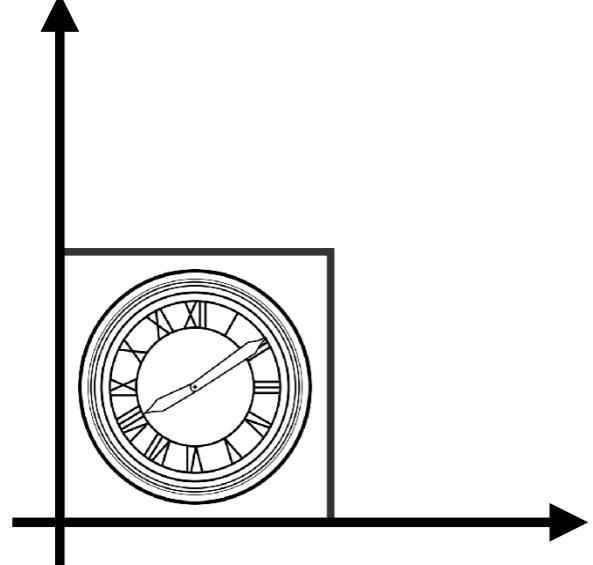
$$T_{(1,0)}$$

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

# Rotate Then Translate



# Transform Ordering Matters!



# Transform Ordering Matters!

Matrix multiplication is not commutative

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

Note that matrices are applied right to left:

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

先旋转，后平移，也是其次坐标矩阵的变换次序

# Composing Transforms

Sequence of affine transforms  $A_1, A_2, A_3, \dots$

- Compose by matrix multiplication
  - Very important for performance!

$$A_n(\dots A_2(A_1(\mathbf{x}))) = \mathbf{A}_n \cdots \mathbf{A}_2 \cdot \mathbf{A}_1 \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$


Pre-multiply  $n$  matrices to obtain a single matrix representing combined transform

# Decomposing Complex Transforms

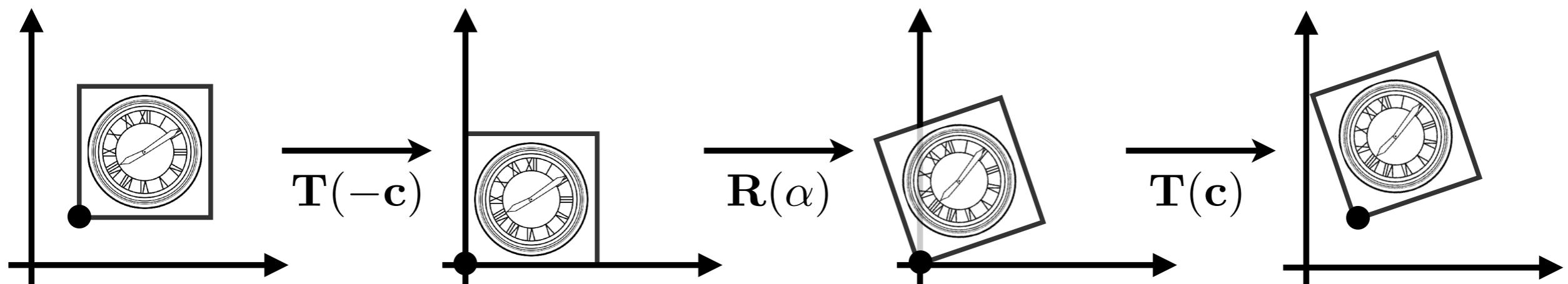
How to rotate around a given point  $c$ ?

1. Translate center to origin

2. Rotate

3. Translate back

复杂的变换可以使用简单的变换【旋转, 平移】进行分解



Matrix representation?

$$T(c) \cdot R(\alpha) \cdot T(-c)$$

# 3D Transforms

# 3D Transformations

Use homogeneous coordinates again:

- 3D point =  $(x, y, z, 1)^T$
- 3D vector =  $(x, y, z, 0)^T$

In general,  $(x, y, z, w)$  ( $w \neq 0$ ) is the 3D point:

$$(x/w, y/w, z/w)$$

# 3D Transformations

Use  $4 \times 4$  matrices for affine transformations

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} a & b & c & t_x \\ d & e & f & t_y \\ g & h & i & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

**What's the order?**

**Linear Transform first or Translation first?**

# Thank you!

(And thank Prof. Ravi Ramamoorthi and Prof. Ren Ng for many of the slides!)