
Computer Graphics

3 - Transformations

Yoonsang Lee
Hanyang University

Spring 2026

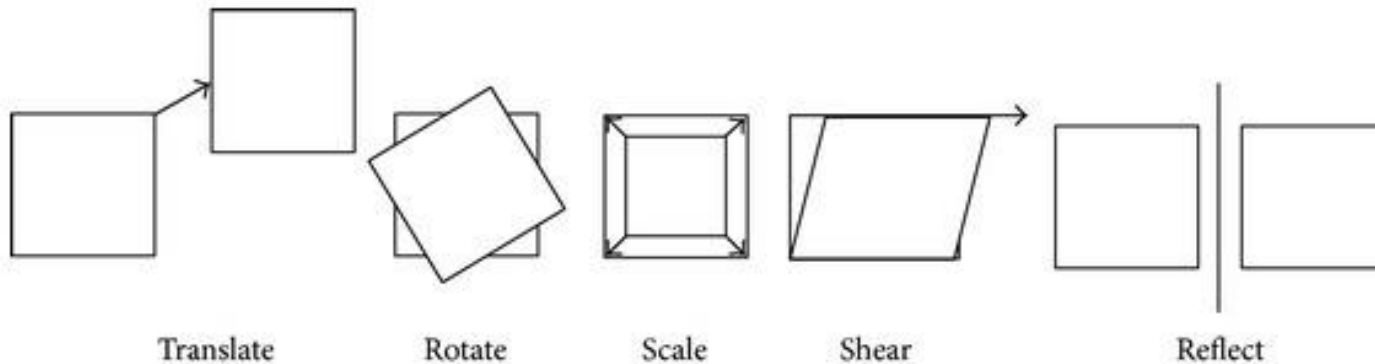
Outline

- 2D Transformations
 - Scaling, Rotation, Shearing, Reflection
 - Translation
- Classes of Transformations
- Composition of Transformations & Homogeneous Coordinates
- Two Types of 3D Cartesian Coordinate System
- 3D Affine Transformations

2D Transformations

What is Transformation?

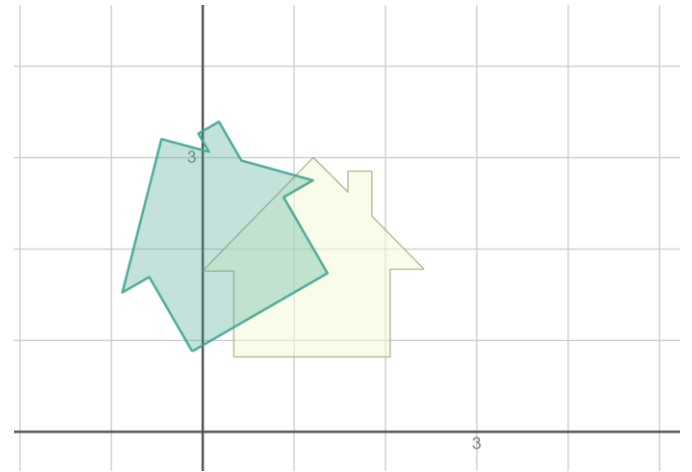
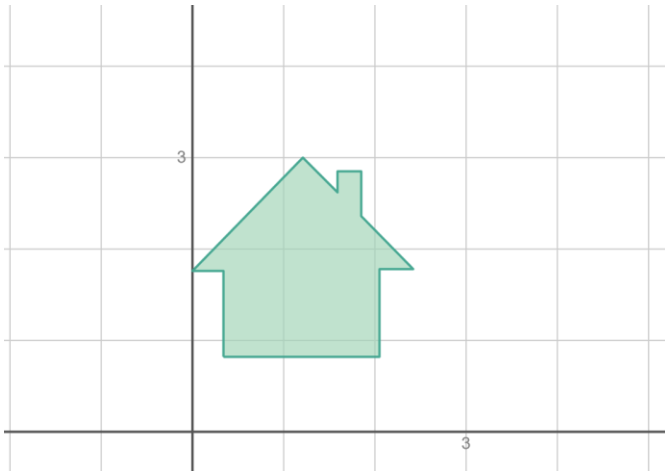
- Geometric **Transformation**
 - The process of changing the position, orientation, size, or shape of a geometric object using mathematical operations.
→ “Moving a set of points”
 - Essential in computer graphics as it allows the creation of complex scenes and animations.
- Examples:



Transformation

- “Moving a set of points”
 - Transformation T maps any input vector $\mathbf{v}$ in the vector space S to $T(\mathbf{v})$.

$$S \rightarrow \{T(\mathbf{v}) \mid \mathbf{v} \in S\}$$



Linear Transformation

- One way to define a transformation is by matrix multiplication:

$$T(\mathbf{v}) = M\mathbf{v}$$

- This is called a **linear transformation** because a *matrix multiplication* represents a *linear mapping*.
 - A linear transformation must satisfy *linearity*:
$$T(\mathbf{v}_1 + \mathbf{v}_2) = T(\mathbf{v}_1) + T(\mathbf{v}_2), \quad T(c\mathbf{v}) = cT(\mathbf{v})$$
 - A matrix multiplication satisfies them:
$$M(\mathbf{v}_1 + \mathbf{v}_2) = M\mathbf{v}_1 + M\mathbf{v}_2, \quad M(c\mathbf{v}) = c(M\mathbf{v})$$

2D Linear Transformations

- 2x2 matrices represent 2D linear transformations such as:
 - uniform scaling
 - non-uniform scaling
 - rotation
 - shearing
 - reflection

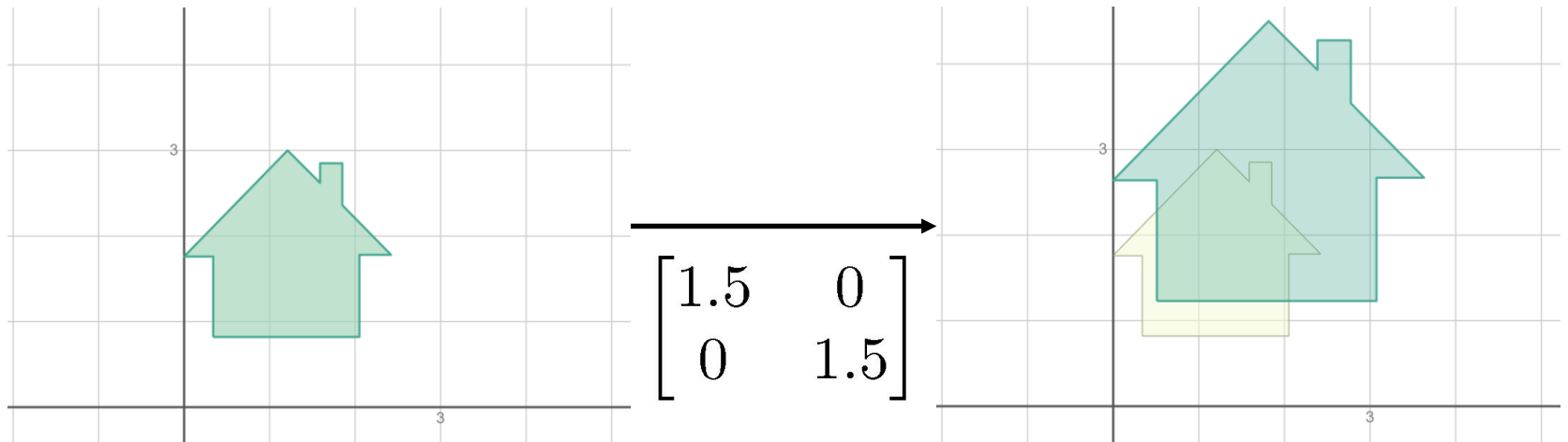
2D Linear Trans. – Uniform Scaling

- Uniformly shrinks or enlarges both in x and y directions.

2x2 scale matrix S

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

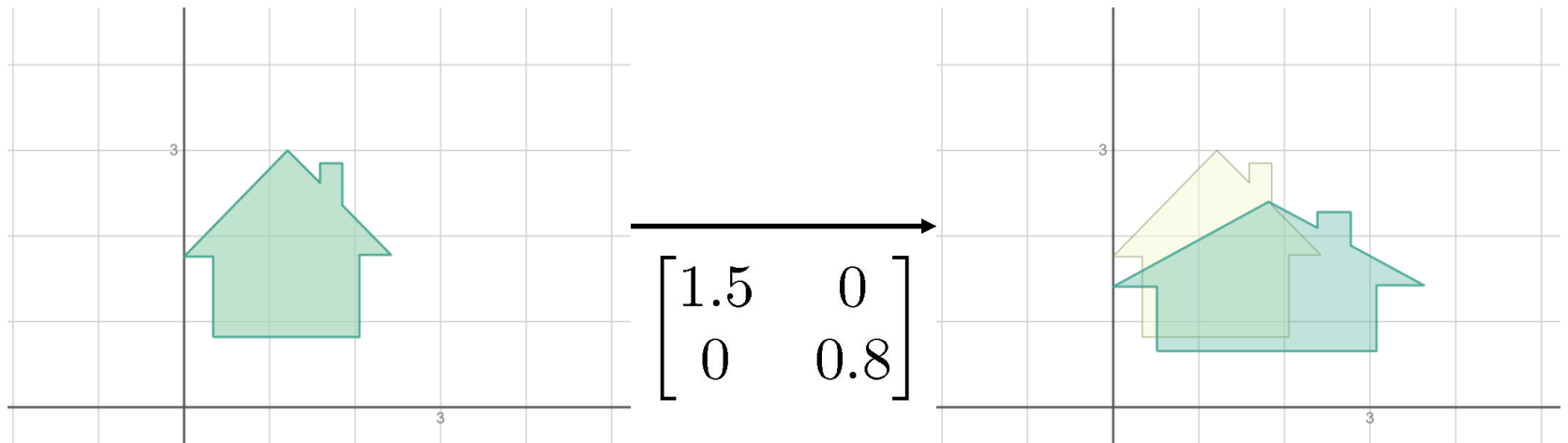
$p = p'$



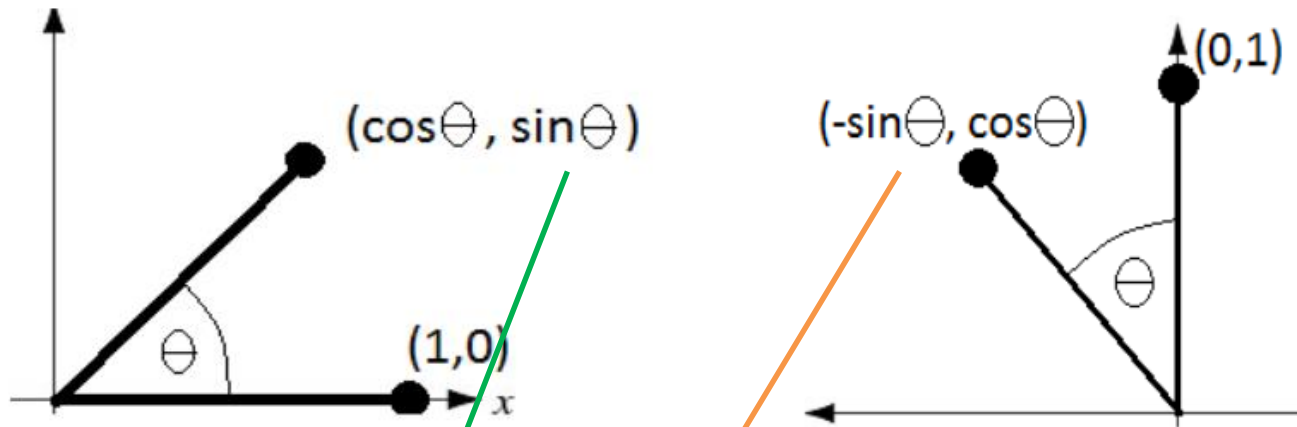
2D Linear Trans. – Nonuniform Scaling

- Non-uniformly shrinks or enlarges in x and y directions.

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



2D Linear Trans. – Rotation



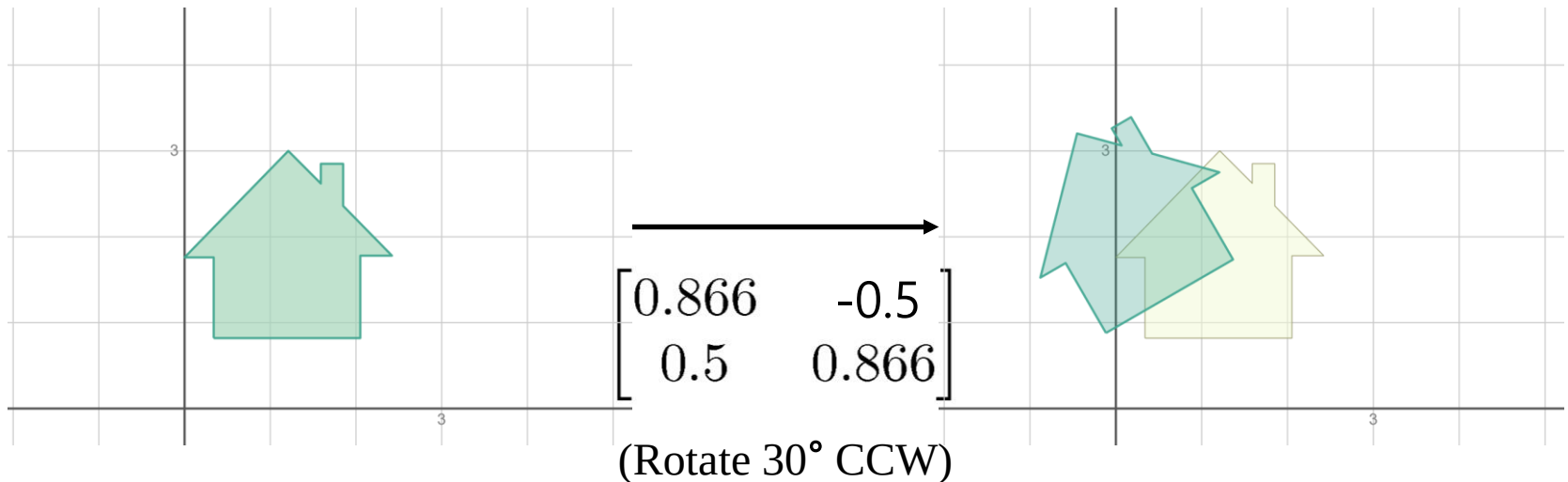
$$\Rightarrow R_{\theta} = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix}$$

: Rotation matrix

2D Linear Trans. – Rotation

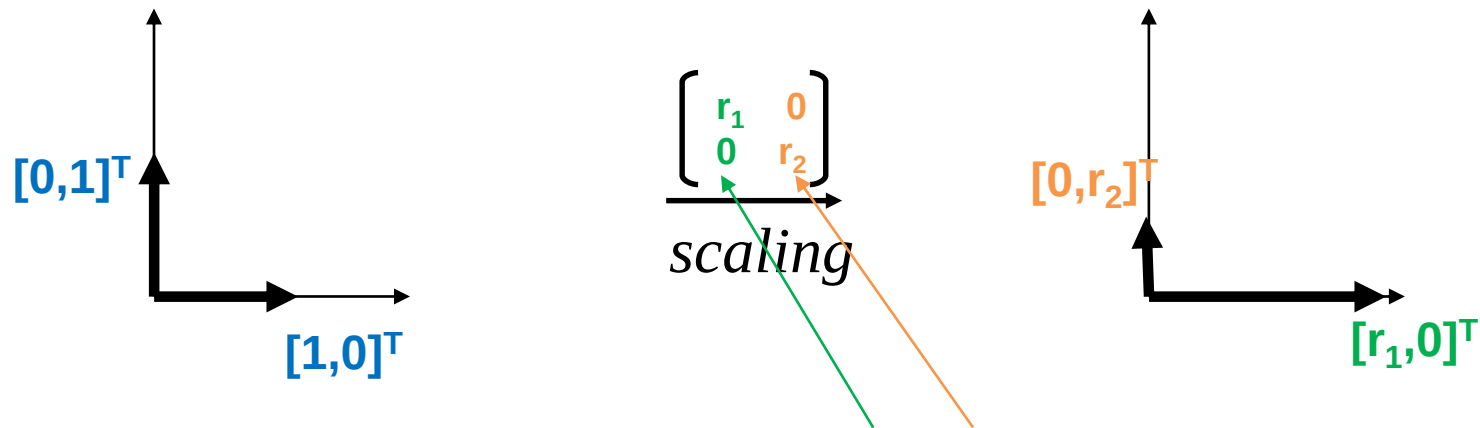
- Rotation can be written in matrix multiplication, so it's also a linear transformation.
 - Note that positive angle means CCW rotation.

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



Numbers in Matrices: Scaling, Rotation

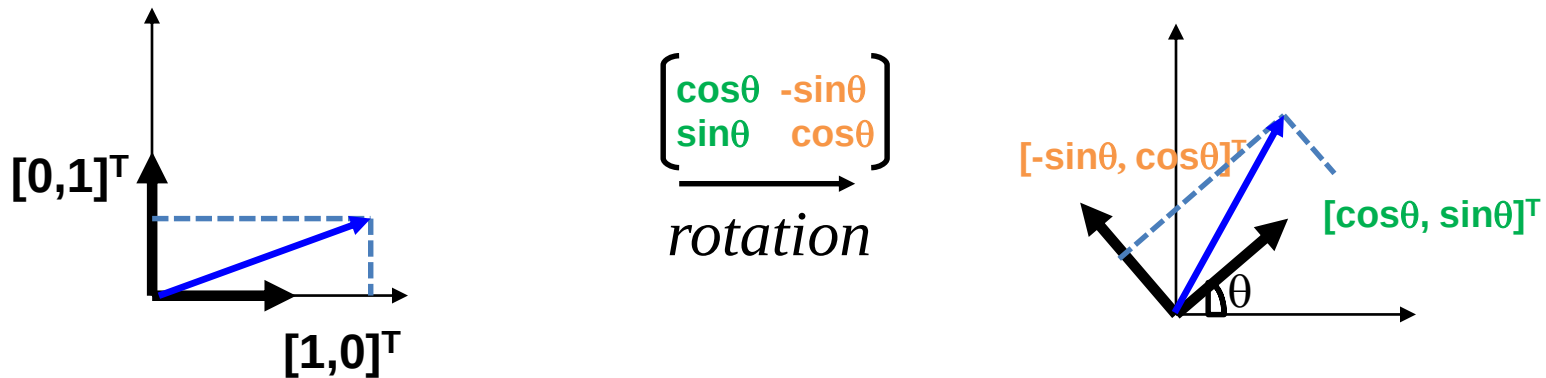
- Let's think about what the numbers in the matrix means.



Standard basis vectors: unit vectors pointing in the direction of the axes of a Cartesian coordinate system.

1^{st} & 2^{nd} basis vectors of the transformed coordinates
→ Each column of the matrix is where the basis vectors go.

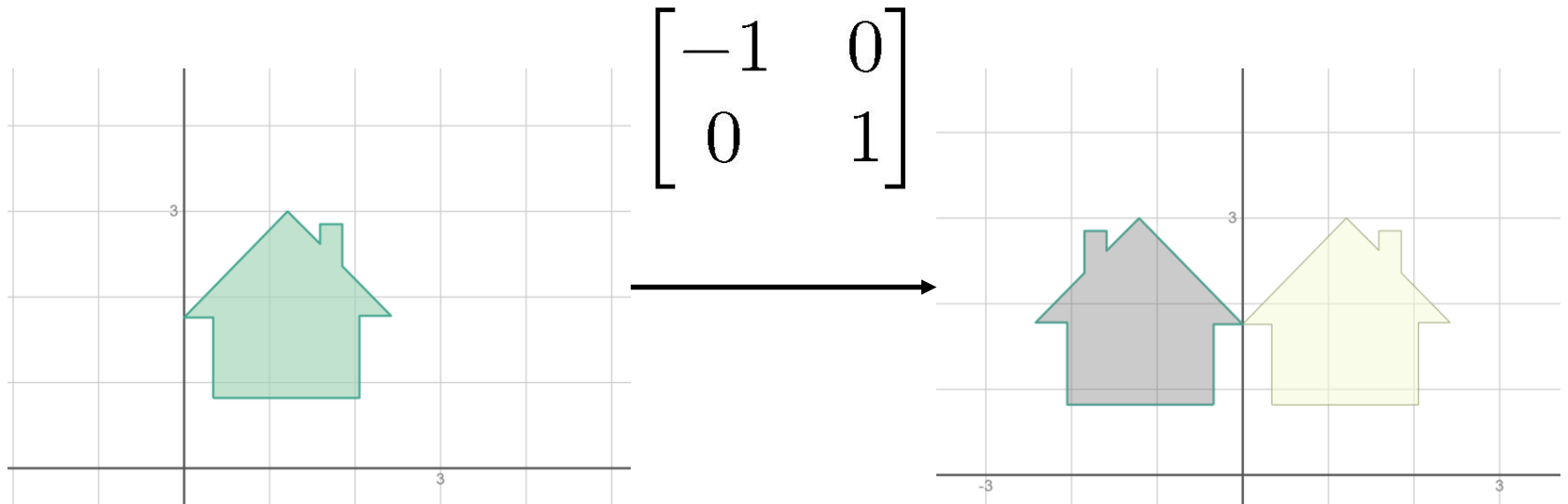
Numbers in Matrices: Scaling, Rotation



- Each column tells us **where each basis vector goes**.
- So all transformed vectors are **linear combinations** of these columns.
 - Linear combination: multiplying vectors by scalars and adding them together.
- This is called the *column space* — the span (a set of all possible linear combinations) of the columns.

2D Linear Trans. – Reflection

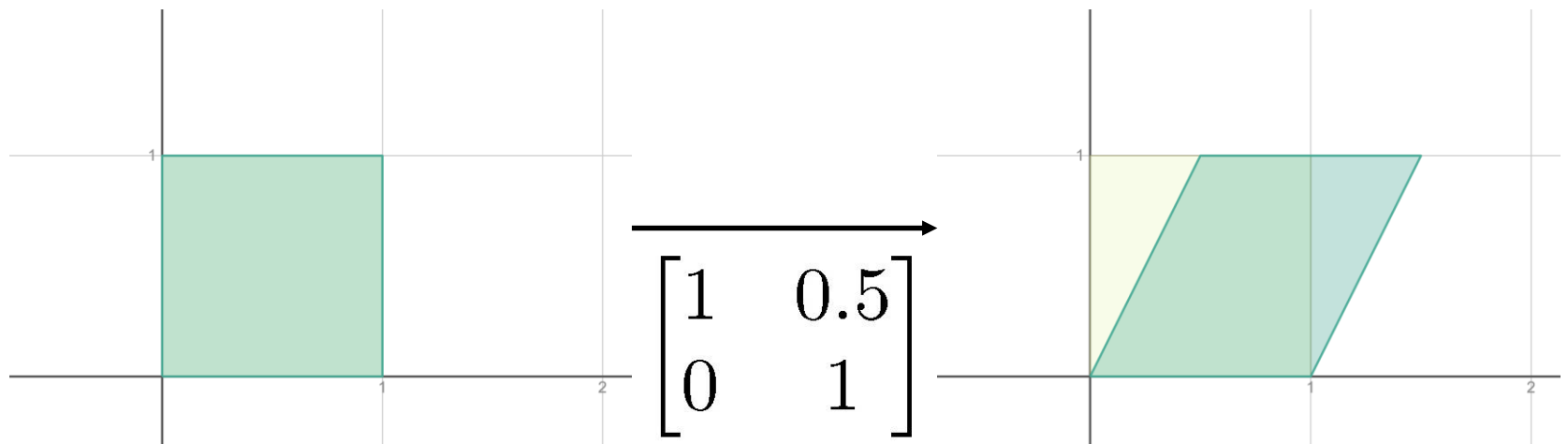
- Reflection can be considered as a special case of non-uniform scale.



2D Linear Trans. – Shearing

- "Push things sideways"

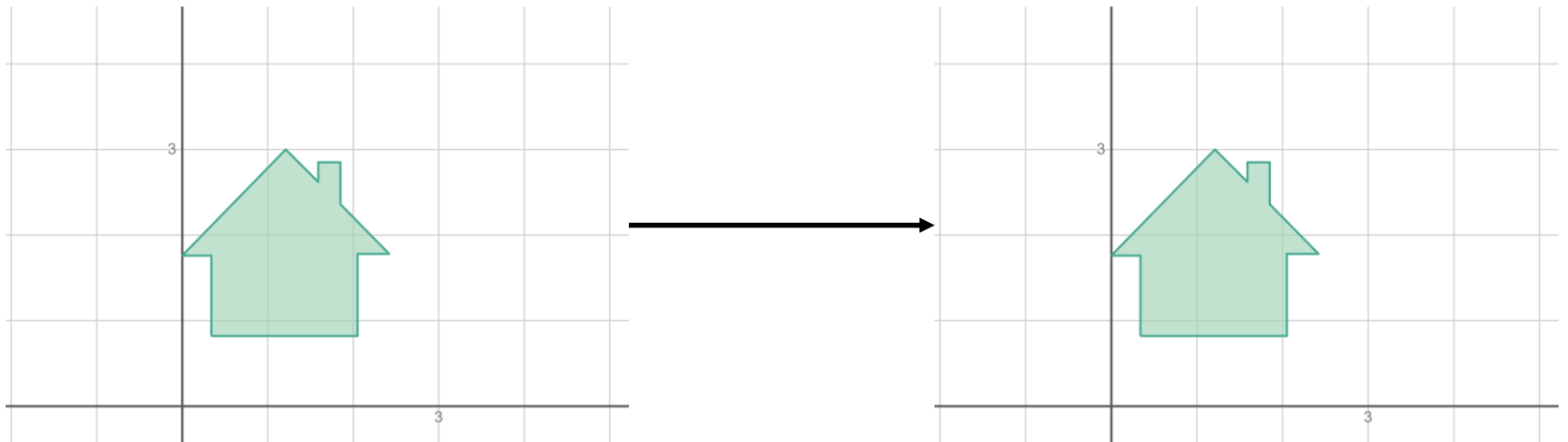
$$\begin{bmatrix} 1 & a \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x + ay \\ y \end{bmatrix}$$



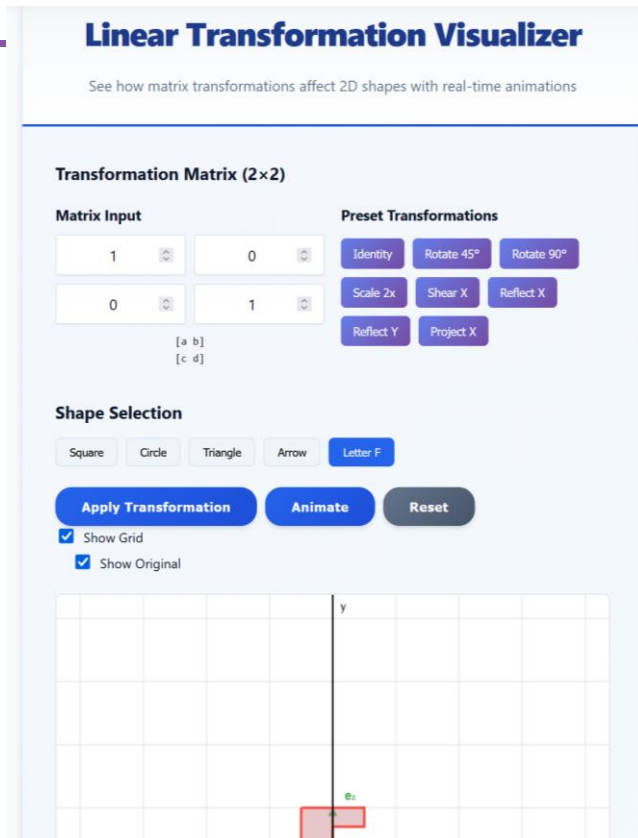
Identity Matrix

- "Doing nothing"

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



[Demo] 2D Linear Transformations



<https://web-apps.thecoatlessprofessor.com/linear-algebra/linear-transformation-visualizer.html>

- Select a shape (e.g., Letter F)
- Try different transformations or modify the matrix
- Click “Apply” or “Animate” and observe how the shape changes

Quiz 1

- Go to <https://www.slido.com/>
- Join #cg-ys
- Click "Polls"
- Submit your answer in the following format:
 - **Student ID: Your answer**
 - e.g. **2024123456: 4.0**
- Note that your quiz answer must be submitted **in the above format** to be counted for attendance.

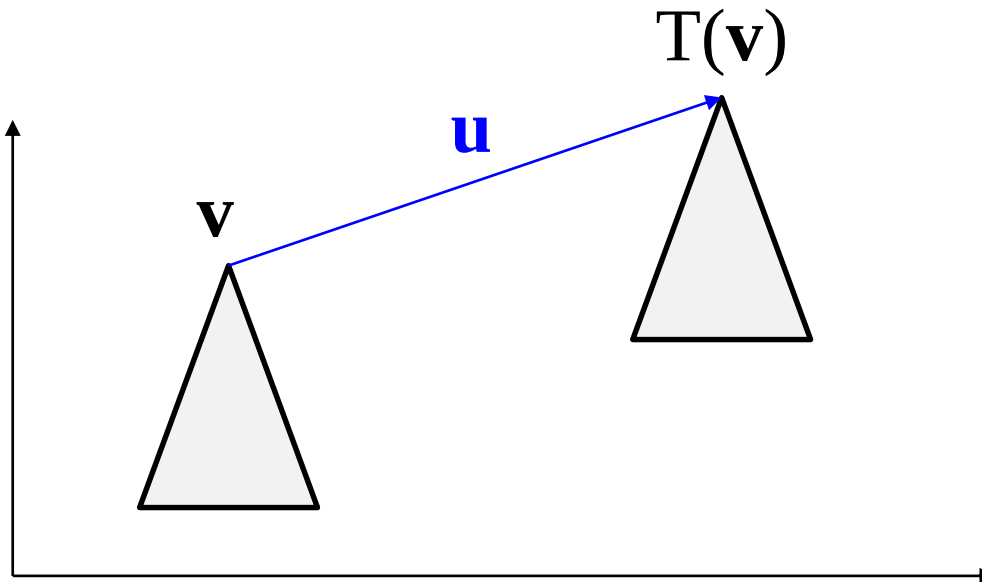
2D Translation

- Translation is the simplest transformation:

$$T(\mathbf{v}) = \mathbf{v} + \mathbf{u}$$

- Inverse:

$$T^{-1}(\mathbf{v}) = \mathbf{v} - \mathbf{u}$$



Is translation linear transformation?

- No. It does not satisfy *linearity*.

$$T(\mathbf{v}_1 + \mathbf{v}_2) = T(\mathbf{v}_1) + T(\mathbf{v}_2) \quad T(c\mathbf{v}) = cT(\mathbf{v})$$

$$\text{e.g. } cT(\mathbf{v}) = c(\mathbf{v} + \mathbf{b}) = c\mathbf{v} + c\mathbf{b} \neq T(c\mathbf{v}) = c\mathbf{v} + \mathbf{b}$$

- We can express translation using vector addition:

$$T(\mathbf{v}) = \mathbf{v} + \mathbf{u}$$

- Combining with linear transformation:

$$T(\mathbf{v}) = M\mathbf{v} + \mathbf{u}$$

- **→ Affine transformation**

Let's check again

- Linear transformation
 - Scaling, rotation, reflection, shearing
 - Represented as matrix multiplications

$$T(\mathbf{v}) = M\mathbf{v}$$

- Translation
 - Not a linear transformation
 - Can be expressed using vector addition

$$T(\mathbf{v}) = \mathbf{v} + \mathbf{u}$$

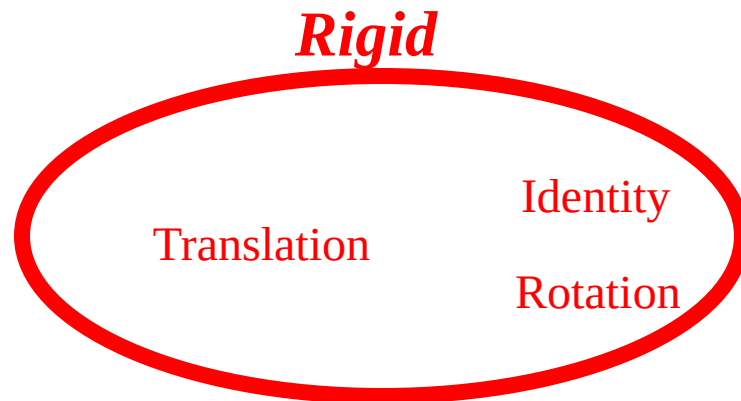
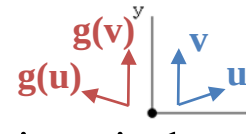
- Affine transformation
 - Combination of linear transformation and translation

$$T(\mathbf{v}) = M\mathbf{v} + \mathbf{u}$$

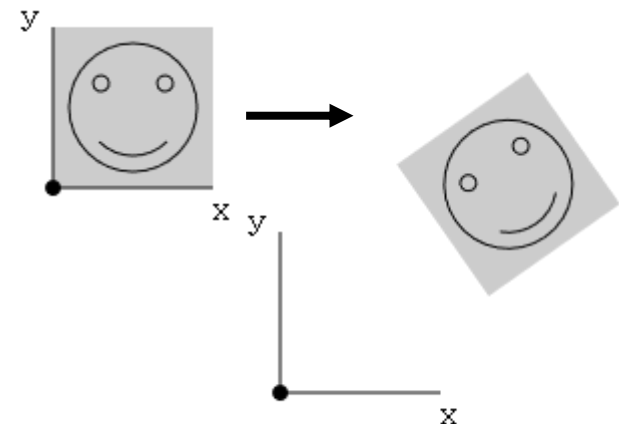
Classes of Transformations

Rigid Transformations

- Preserve distances between all points.
 - $\|g(\mathbf{u}) - g(\mathbf{v})\| = \|\mathbf{u} - \mathbf{v}\|, \forall \mathbf{u}, \mathbf{v} \in \mathbb{R}^3$ (g : rigid transform map)
- Preserve "handedness".
 - Condition for handedness-preserving linear transformation (Rotation):
 $g(\mathbf{u}) \times g(\mathbf{v}) = g(\mathbf{u} \times \mathbf{v}), \forall \mathbf{u}, \mathbf{v} \in \mathbb{R}^3$.
 - Reflections do not satisfy this property.
 - Translation doesn't affect handedness, since it doesn't change orientation.



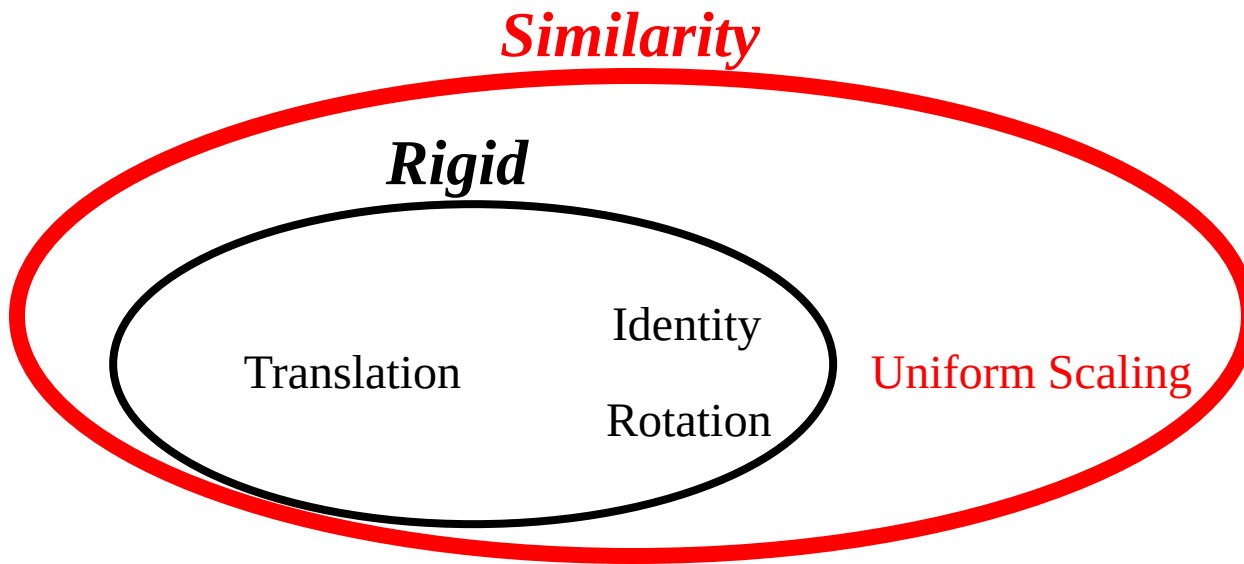
(Some people categorize reflection as a rigid transformation. We don't do that in this lecture.)



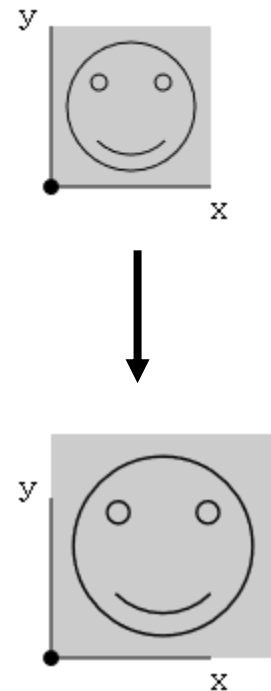
* The diagram is from the slides of Prof. Frédo Durand and Prof. Barbara Cutler (MIT):

Similarity Transformations

- Preserve angles.
- (This diagram indicates rigid transforms also preserve angles.)

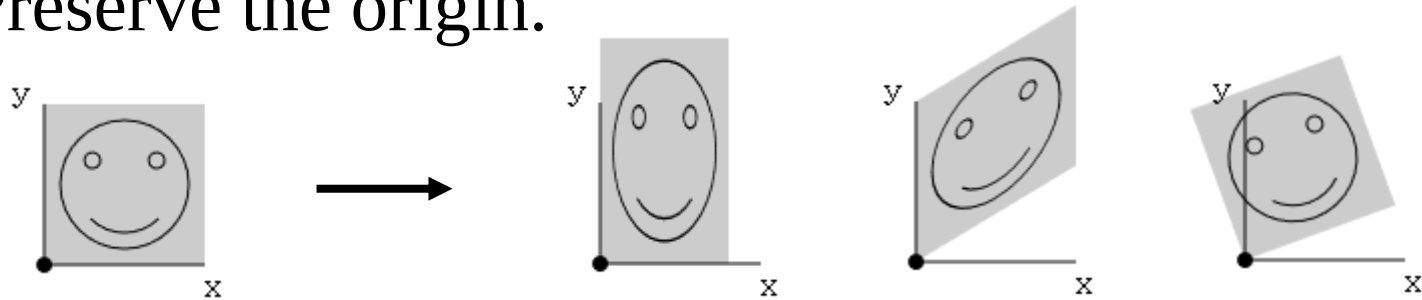


(Some people categorize reflection as a similarity transformation. We don't do that in this lecture.)

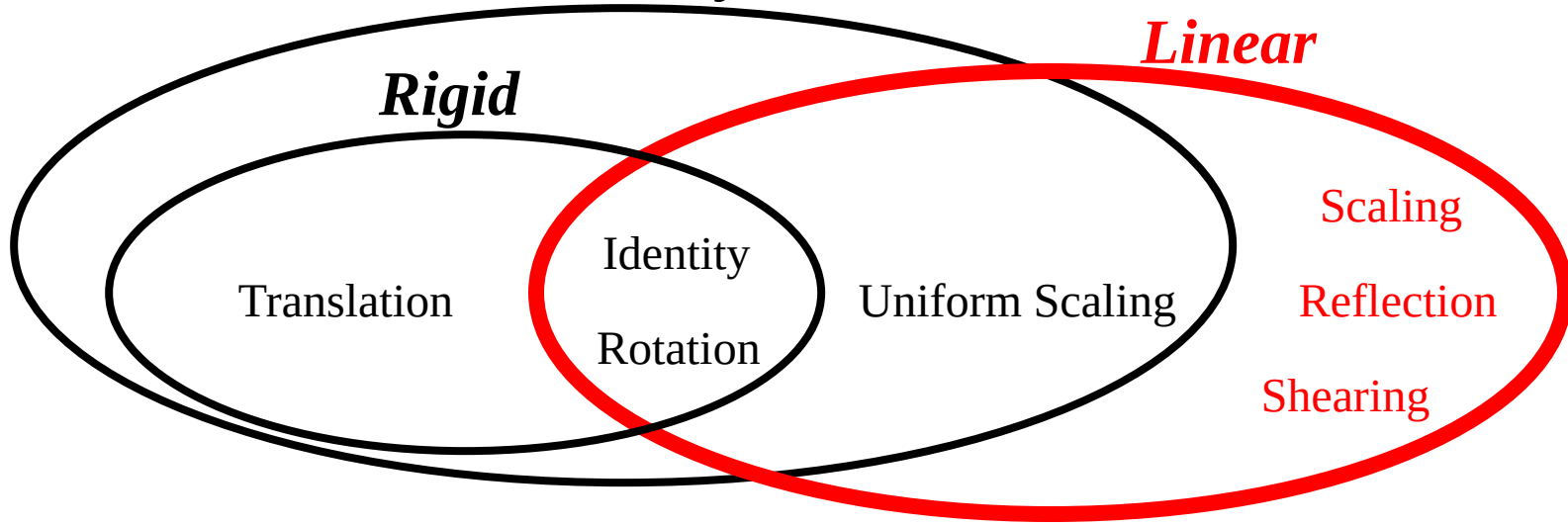


Linear Transformations

- Preserve the origin.

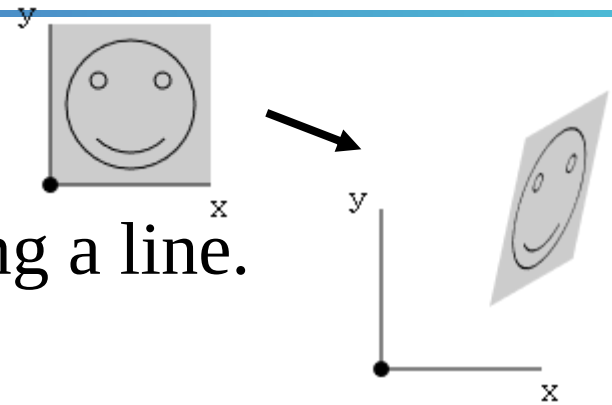


Similarity



Affine Transformations

- Preserve parallel lines.
- Preserve ratios of distance along a line.



Affine

Similarity

Linear

Rigid

Translation

Identity

Rotation

Uniform Scaling

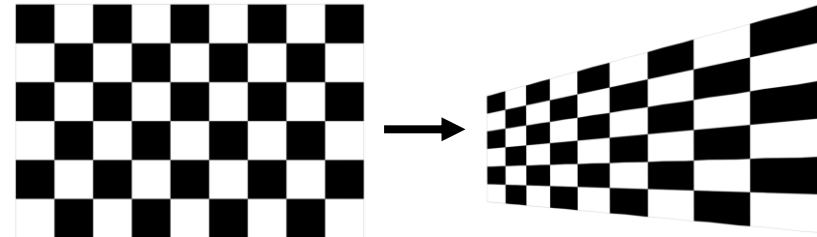
Scaling

Reflection

Shearing

Projective Transformations

- Preserve lines.



Projective

Affine

Similarity

Linear

Rigid

Translation

Identity

Rotation

Uniform Scaling

Scaling

Reflection

Shearing

Perspective

Composition of Transformations & Homogeneous Coordinates

Composition of Transformations

- Move an object by T , then move it more by S :

$$\mathbf{p} \rightarrow T(\mathbf{p}) \rightarrow S(T(\mathbf{p})) = (S \circ T)(\mathbf{p})$$

- **Composing 2D linear transformations** just works by **2x2 matrix multiplication**:

$$T(\mathbf{p}) = M_T \mathbf{p}; S(\mathbf{p}) = M_S \mathbf{p}$$

$$(S \circ T)(\mathbf{p}) = S(T(\mathbf{p})) = M_S(M_T \mathbf{p}) = (M_S M_T) \mathbf{p}$$

* The equation images are from the slides of Prof. Steve Marschner (Cornell University):

<https://www.cs.cornell.edu/courses/cs4620/2018fa/>

Composition of Transformations: Order Matters!

- Matrix multiplication:
associative,

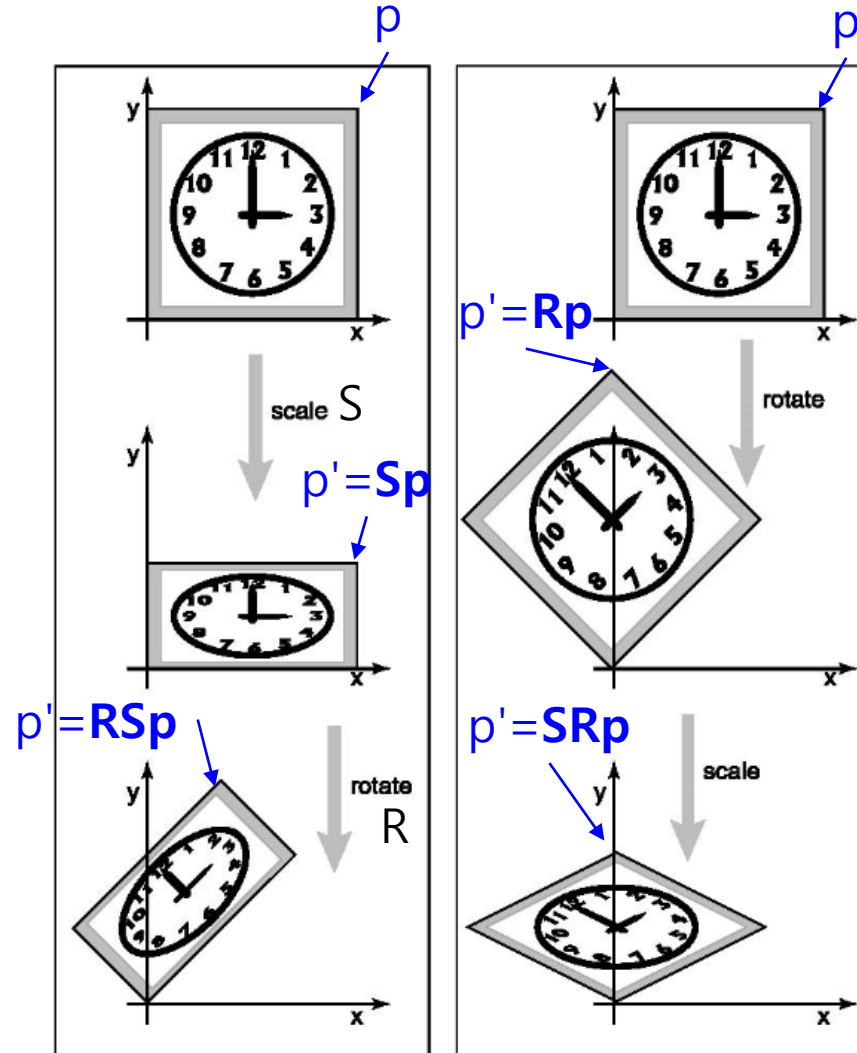
$$(AB)C = A(BC)$$

NOT commutative

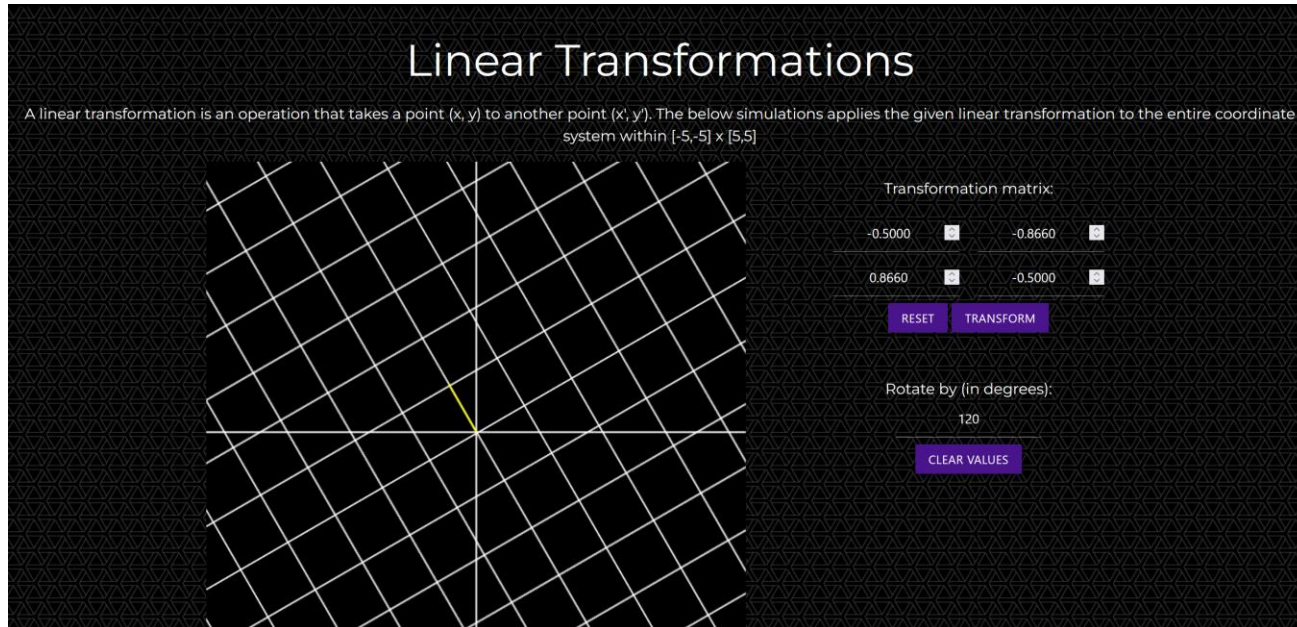
$$AB \neq BA$$

→ This means order matters

- Same transforms, different order**
→ **different result**



[Demo] Composition of Linear Transformations



https://visualize-it.github.io/linear_transformations/simulation.html

- Reset the scene (“Reset” / “Clear Values”)
- Enter a transformation: *scale x by 2* $\rightarrow [2 \ 0 ; 0 \ 1]$, then click “Transform”
- Enter another transformation: *shear along x* $\rightarrow [1 \ 1 ; 0 \ 1]$, then click “Transform”
- Try a different order
- Compare the results

Problems with Composing Affine Transformations

- Linear transformations and translations cannot be handled in the same way.
 - Because translation cannot be represented as a matrix multiplication.
- This makes composing affine transformations complicated:

$$\begin{aligned} T(\mathbf{p}) &= M_T \mathbf{p} + \mathbf{u}_T & (S \circ T)(\mathbf{p}) &= M_S(M_T \mathbf{p} + \mathbf{u}_T) + \mathbf{u}_S \\ S(\mathbf{p}) &= M_S \mathbf{p} + \mathbf{u}_S & &= (M_S M_T) \mathbf{p} + (M_S \mathbf{u}_T + \mathbf{u}_S) \end{aligned}$$

- We need a cleaner way!
- → Represent everything as matrix multiplication
- → **Homogeneous coordinates**

Homogeneous Coordinates

- Key idea: Represent 2D points in 3D coordinate space.
- Introduce an extra component w :
 - For points, use $w = 1$
 - 2D point $[x, y]^T \rightarrow [x, y, 1]^T$
- Use 3×3 matrices to represent 2D transformations.
- 2D linear transformations are represented as:

$$\begin{bmatrix} a & b & 0 \\ c & d & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} ax + by \\ cx + dy \\ 1 \end{bmatrix}$$

Homogeneous Coordinates

- 2D translations are represented as:

$$\begin{bmatrix} 1 & 0 & t \\ 0 & 1 & s \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x + t \\ y + s \\ 1 \end{bmatrix}$$

- 2D affine transformations are represented as:

linear part  $\begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \\ 0 & 0 & 1 \end{bmatrix}$  translational part $\begin{bmatrix} u_x \\ u_y \end{bmatrix}$

→ Now we can include translation as matrix multiplication

Homogeneous Coordinates

- **Composing affine transformations reduces to 3x3 matrix multiplication.**

$$T(\mathbf{p}) = M_T \mathbf{p} + \mathbf{u}_T$$

$$S(\mathbf{p}) = M_S \mathbf{p} + \mathbf{u}_S$$



$$T(\mathbf{p}) = \begin{bmatrix} M_T^{2 \times 2} & \mathbf{u}_T^{2 \times 1} \\ 0 & 1 \end{bmatrix}$$

$$S(\mathbf{p}) = \begin{bmatrix} M_S^{2 \times 2} & \mathbf{u}_S^{2 \times 1} \\ 0 & 1 \end{bmatrix}$$

(in block matrix representation)

Homogeneous Coordinates

- **Composing affine transformations reduces to 3x3 matrix multiplication.**

$$(S \circ T)(\mathbf{p}) = \begin{bmatrix} M_S^{2 \times 2} & \mathbf{u}_S^{2 \times 1} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} M_T^{2 \times 2} & \mathbf{u}_T^{2 \times 1} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{p}^{2 \times 1} \\ 1 \end{bmatrix}$$
$$= \begin{bmatrix} \underline{(M_S M_T) \mathbf{p} + (M_S \mathbf{u}_T + \mathbf{u}_S)} \\ 1 \end{bmatrix}$$

- The result is the same, but much cleaner.
 - cf. $(S \circ T)(\mathbf{p}) = M_S(M_T \mathbf{p} + \mathbf{u}_T) + \mathbf{u}_S$
 $= \underline{(M_S M_T) \mathbf{p} + (M_S \mathbf{u}_T + \mathbf{u}_S)}$

* The equation images are from the slides of Prof. Steve Marschner (Cornell University):

<https://www.cs.cornell.edu/courses/cs4620/2018fa/>

[Demo] Composition of Affine Transformations in Homogeneous Coordinates

Transformation demo

An interactive demo for experimenting with 2D transformation matrix composition.

$$T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



<https://observablehq.com/@esperanc/transformation-demo>

- Add translation and linear transforms in various orders with '+' buttons.
- Drag the slider to see the matrix value change and the shape transform.
- **Note that the last transform added is the first applied transform.**

Summary: Homogeneous Coordinates in 2D

- Use $(x,y,1)^T$ instead of $(x,y)^T$ for **2D points**
- Use **3x3 matrices** instead of 2x2 matrices **for 2D linear transformations**
- Use **3x3 matrices** instead of vector addition **for 2D translations.**

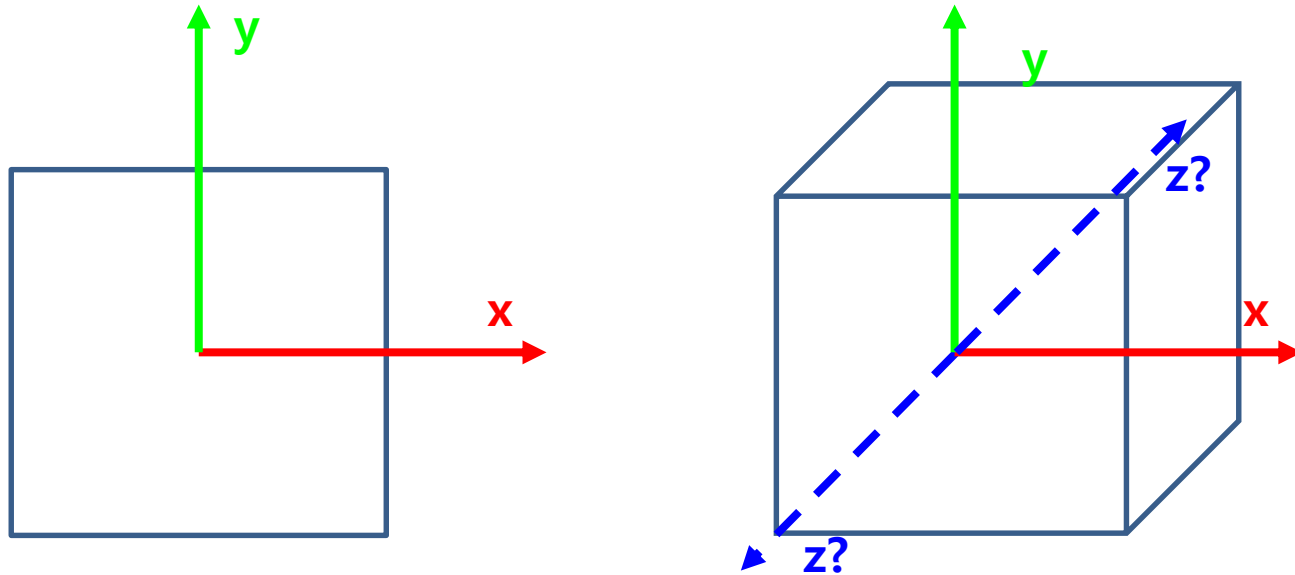
- $\rightarrow$ We can treat linear transformations and translations **in a consistent manner!**

Quiz 2

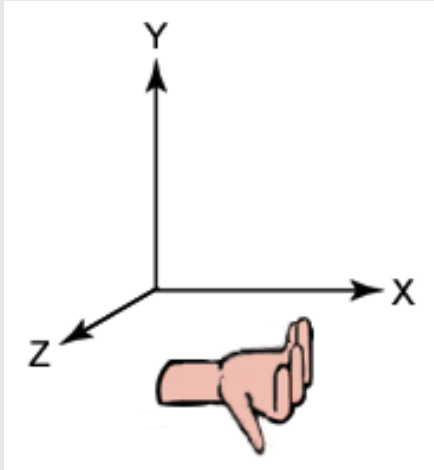
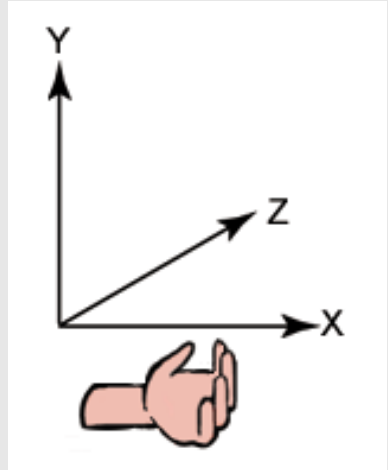
- Go to <https://www.slido.com/>
- Join #cg-ys
- Click "Polls"
- Submit your answer in the following format:
 - **Student ID: Your answer**
 - e.g. **2024123456: 4.0**
- Note that your quiz answer must be submitted **in the above format** to be counted for attendance.

Two Types of 3D Cartesian Coordinate System

Now, Let's go to the 3D world!



- Coordinate system (좌표계)
 - Cartesian coordinate system (직교좌표계)

	Right-handed system	Left-handed system
Hand rule (axes & rotation):	Right hand	Left hand
Coordinates axes: <i>fingers</i> $\rightarrow +x$ <i>up</i> $\rightarrow +y$ <i>thumb</i> $\rightarrow +z$		
Positive rotation: <i>thumb</i> $\rightarrow$ <i>axis</i> <i>fingers</i> $\rightarrow +$ <i>rotation</i>	counterclockwise (CCW) about the rotation axis 	clockwise (CW) about the rotation axis 
Used in:	OpenGL , Maya, Houdini, AutoCAD, ... Standard for Physics & Math $\rightarrow$ Used in this course	DirectX, Unity, Unreal, ...

3D Affine Transformations

Point Representations in Cartesian & Homogeneous Coordinate System

	Cartesian coordinate system	Homogeneous coordinate system
A 2D point is represented as...	$\begin{bmatrix} p_x \\ p_y \end{bmatrix}$	$\begin{bmatrix} p_x \\ p_y \\ \boxed{1} \end{bmatrix}$ for points
A 3D point is represented as...	$\begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix}$	$\begin{bmatrix} p_x \\ p_y \\ p_z \\ \boxed{1} \end{bmatrix}$ for points

Review of Linear Transformations in 2D

- Linear transformations in **2D** can be represented as matrix multiplication of ...

2x2 matrix
(in Cartesian coordinates)

$$\begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix} \begin{bmatrix} p_x \\ p_y \end{bmatrix}$$

or

3x3 matrix
(in homogeneous coordinates)

$$\begin{bmatrix} m_{11} & m_{12} & 0 \\ m_{21} & m_{22} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ 1 \end{bmatrix}$$

Linear Transformations in 3D

- Linear transformations in **3D** can be represented as matrix multiplication of ...

3x3 matrix
(in Cartesian coordinates)

or

4x4 matrix
(in homogeneous coordinates)

$$\begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix}$$

$$\begin{bmatrix} m_{11} & m_{12} & m_{13} & 0 \\ m_{21} & m_{22} & m_{23} & 0 \\ m_{31} & m_{32} & m_{33} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix}$$

Linear Transformations in 3D

Scale:

$$\begin{array}{cc} \mathbf{3D} & \mathbf{3D-H} \\ \mathbf{S_s} = \begin{bmatrix} \mathbf{S_x} & 0 & 0 \\ 0 & \mathbf{S_y} & 0 \\ 0 & 0 & \mathbf{S_z} \end{bmatrix} & \mathbf{S_s} = \begin{bmatrix} \mathbf{S_x} & 0 & 0 & 0 \\ 0 & \mathbf{S_y} & 0 & 0 \\ 0 & 0 & \mathbf{S_z} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{array}$$

Shear (in x, based on y,z position):

$$\mathbf{H_{x,d}} = \begin{bmatrix} 1 & \mathbf{d_y} & \mathbf{d_z} \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \mathbf{H_{x,d}} = \begin{bmatrix} 1 & \mathbf{d_y} & \mathbf{d_z} & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

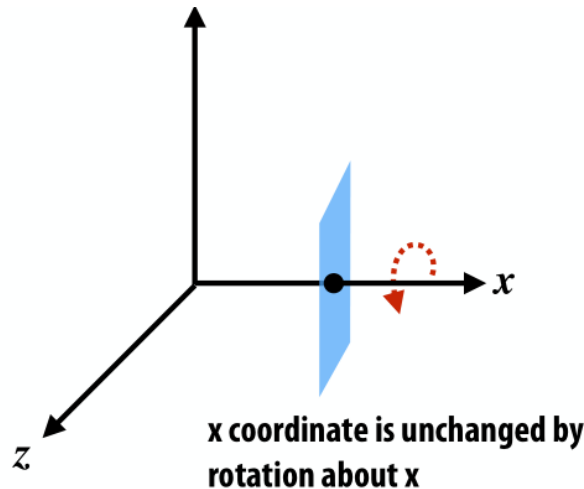
* This slide is from the slides of Prof. Kayvon Fatahalian and Prof. Keenan Crane (CMU):

<http://15462.courses.cs.cmu.edu/fall2015/>

Linear Transformations in 3D

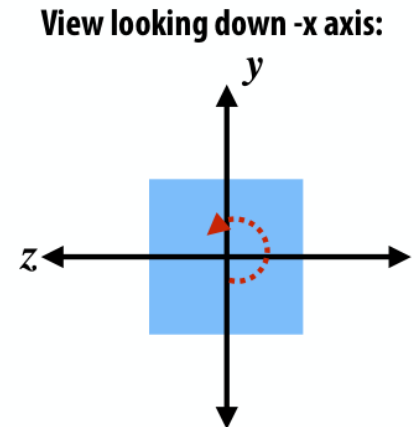
Rotation about x axis:

$$\mathbf{R}_{x,\theta} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \theta & -\sin \theta \\ 0 & \sin \theta & \cos \theta \end{bmatrix}$$

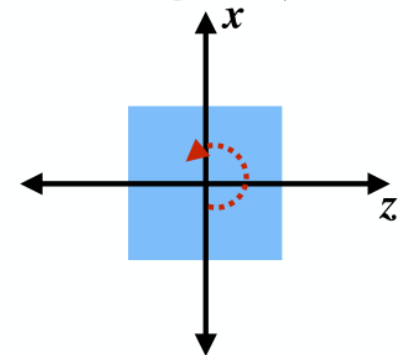


Rotation about y axis:

$$\mathbf{R}_{y,\theta} = \begin{bmatrix} \cos \theta & 0 & \sin \theta \\ 0 & 1 & 0 \\ -\sin \theta & 0 & \cos \theta \end{bmatrix}$$

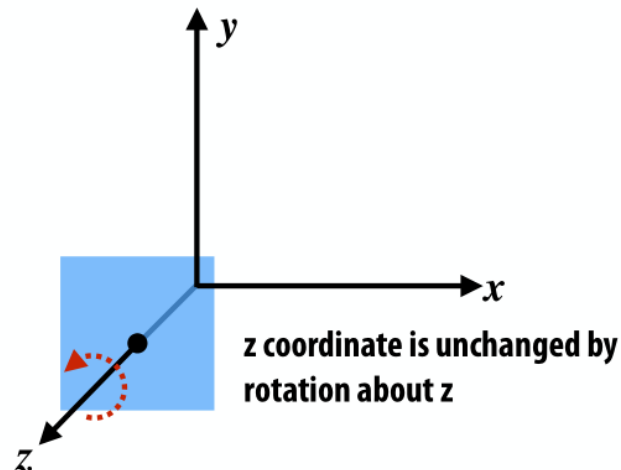


View looking down -y axis:



Rotation about z axis:

$$\mathbf{R}_{z,\theta} = \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



Review of Translations in 2D

- Translations in **2D** can be represented as ...

Vector addition

(in Cartesian coordinates)

$$\begin{bmatrix} p_x \\ p_y \end{bmatrix} + \begin{bmatrix} u_x \\ u_y \end{bmatrix}$$

Matrix multiplication of

3x3 matrix

(in homogeneous coordinates)

$$\begin{bmatrix} 1 & 0 & u_x \\ 0 & 1 & u_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ 1 \end{bmatrix}$$

Translations in 3D

- Translations in **3D** can be represented as ...

Vector addition

(in Cartesian coordinates)

$$\begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} + \begin{bmatrix} u_x \\ u_y \\ u_z \end{bmatrix}$$

Matrix multiplication of

4x4 matrix

(in homogeneous coordinates)

$$\begin{bmatrix} 1 & 0 & 0 & u_x \\ 0 & 1 & 0 & u_y \\ 0 & 0 & 1 & u_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix}$$

Review of Affine Transformations in 2D

- In homogeneous coordinates, **2D** affine transformations can be represented as multiplication of **3x3 matrix**:

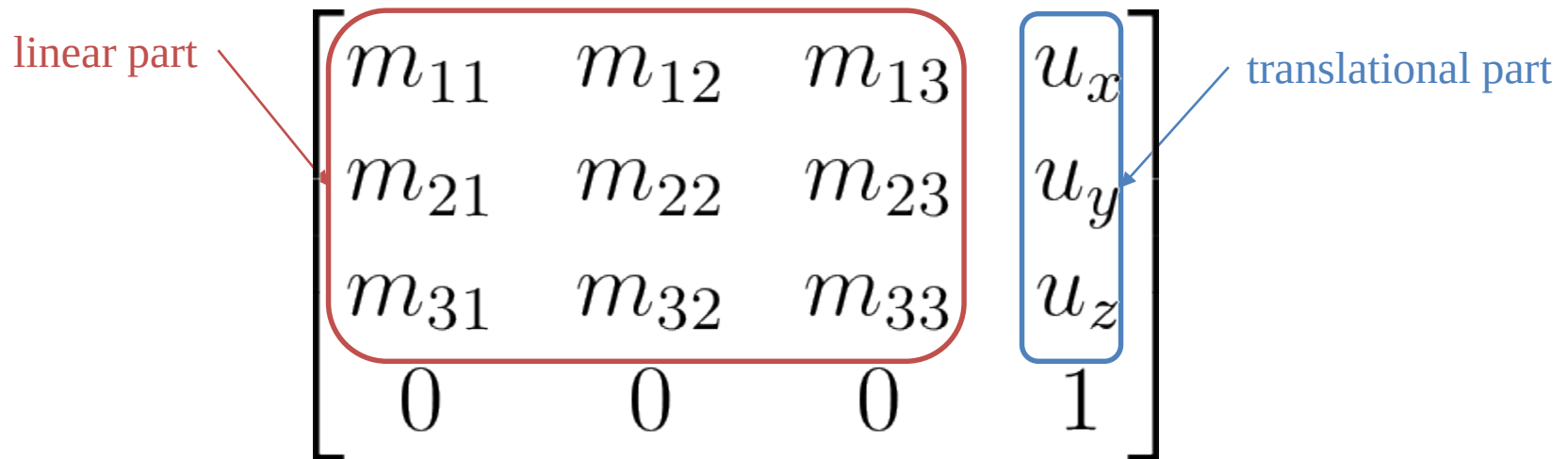
$$\begin{bmatrix} m_{11} & m_{12} & u_x \\ m_{21} & m_{22} & u_y \\ 0 & 0 & 1 \end{bmatrix}$$

linear part

translational part

Affine Transformations in 3D

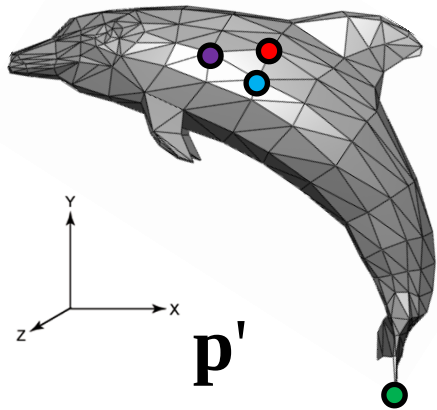
- In homogeneous coordinates, **3D** affine transformations can be represented as multiplication of **4x4 matrix**:



The diagram shows a 4x4 matrix representing a 3D affine transformation in homogeneous coordinates. The matrix is partitioned into two parts: a linear part and a translational part. The linear part, highlighted with a red rounded rectangle, consists of the top three rows and the first three columns, containing elements $m_{11}, m_{12}, m_{13}, m_{21}, m_{22}, m_{23}, m_{31}, m_{32}, m_{33}$. The translational part, highlighted with a blue rounded rectangle, consists of the top three rows and the fourth column, containing elements u_x, u_y, u_z . The bottom row of the matrix contains the values 0, 0, 0, and 1. A red arrow points from the label 'linear part' to the red box, and a blue arrow points from the label 'translational part' to the blue box.

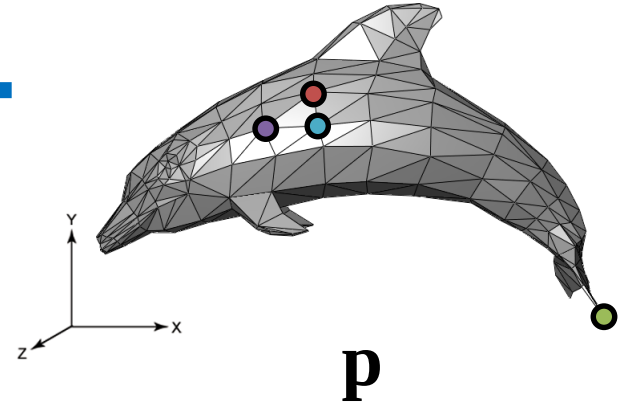
$$\begin{bmatrix} m_{11} & m_{12} & m_{13} & u_x \\ m_{21} & m_{22} & m_{23} & u_y \\ m_{31} & m_{32} & m_{33} & u_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Summary: Affine Transformation



Affine transformation

$$\mathbf{M} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & u_1 \\ m_{21} & m_{22} & m_{23} & u_2 \\ m_{31} & m_{32} & m_{33} & u_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



$$\mathbf{p}_1' \leftarrow \mathbf{M} \mathbf{p}_1$$

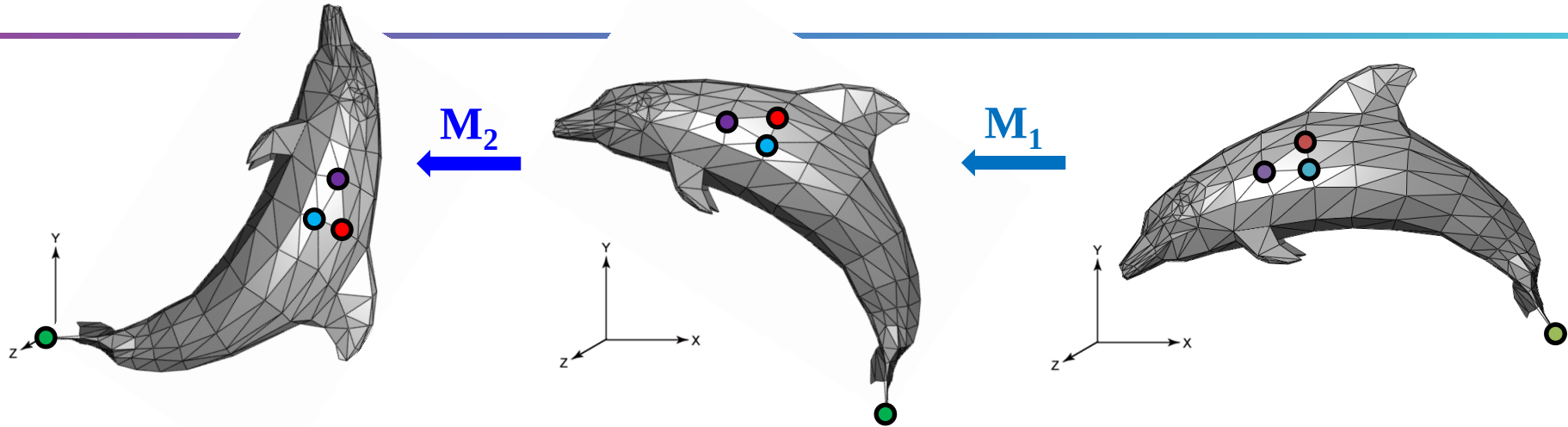
$$\mathbf{p}_2' \leftarrow \mathbf{M} \mathbf{p}_2$$

$$\mathbf{p}_3' \leftarrow \mathbf{M} \mathbf{p}_3$$

$$\begin{matrix} \cdot \\ \cdot \\ \cdot \end{matrix} \leftarrow \begin{matrix} \cdot \\ \cdot \\ \cdot \end{matrix}$$

$$\mathbf{p}_N' \leftarrow \mathbf{M} \mathbf{p}_N$$

Summary: Composition of Affine Transformations



$$\begin{aligned}
 \mathbf{p}_1' &\leftarrow \mathbf{M}_2 \mathbf{M}_1 \mathbf{p}_1 \\
 \mathbf{p}_2' &\leftarrow \mathbf{M}_2 \mathbf{M}_1 \mathbf{p}_2 \\
 \mathbf{p}_3' &\leftarrow \mathbf{M}_2 \mathbf{M}_1 \mathbf{p}_3 \\
 &\vdots \\
 &\vdots \\
 &\vdots \\
 \mathbf{p}_N' &\leftarrow \mathbf{M}_2 \mathbf{M}_1 \mathbf{p}_N
 \end{aligned}$$

Lab Session

- Now, let's start the lab today.