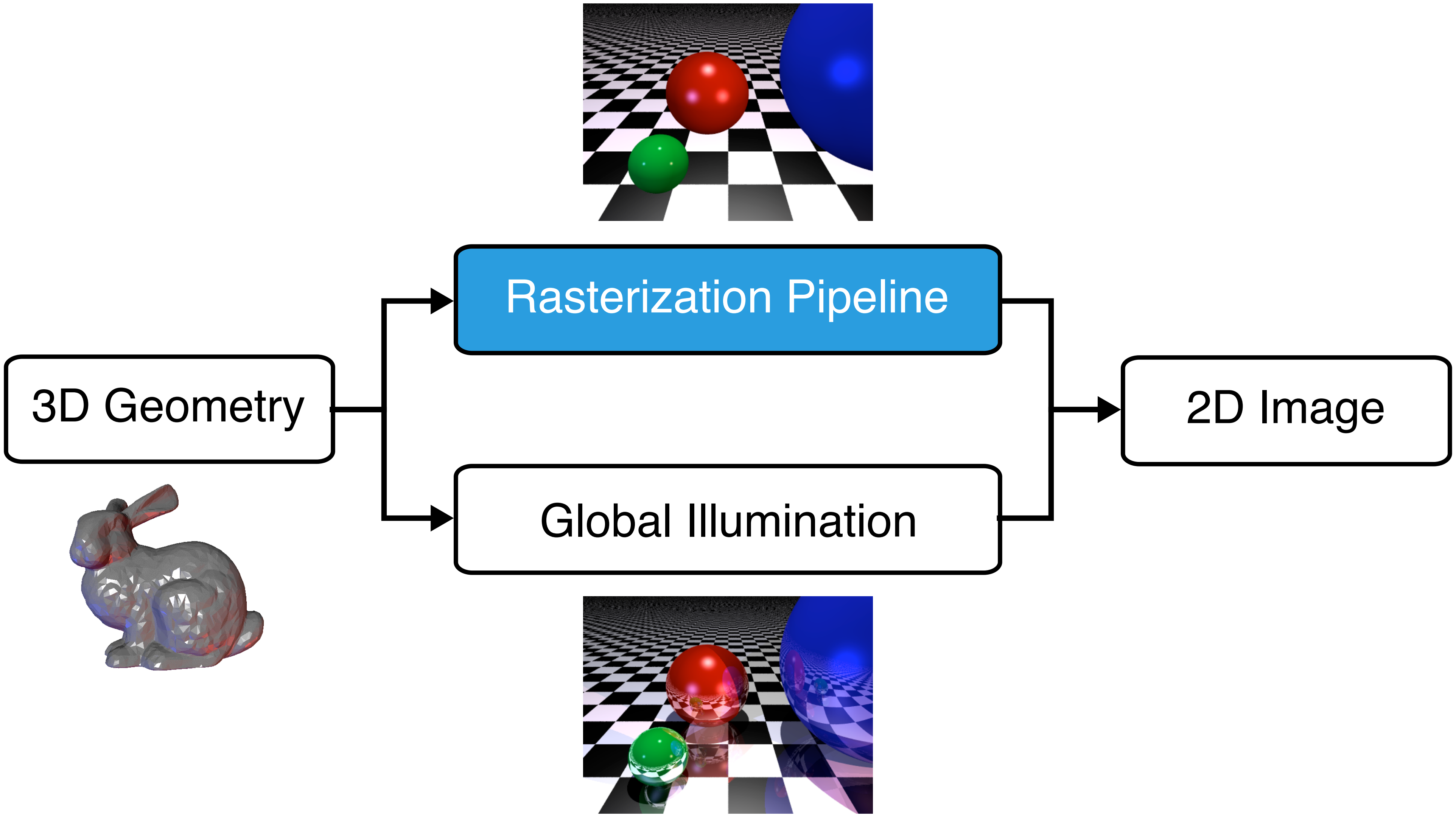
Computer Graphics

Transformations in 2D / 3D

Mark Pauly

Geometric Computing Laboratory

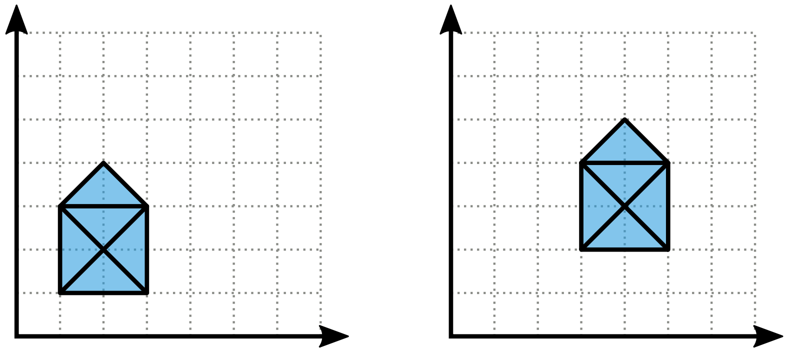
Rasterization Pipeline



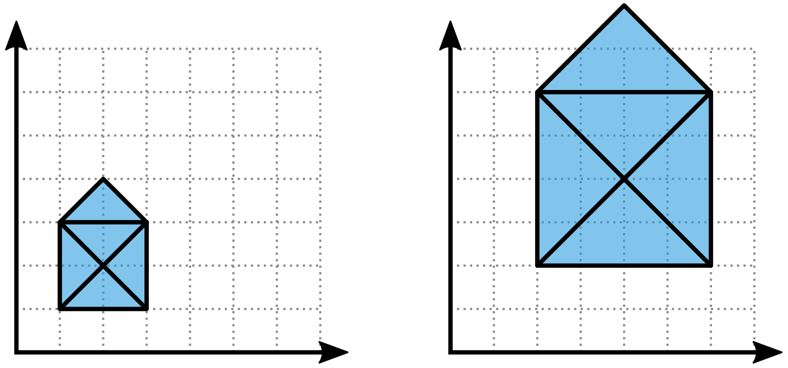
Which transformations do we need?

Our solar system

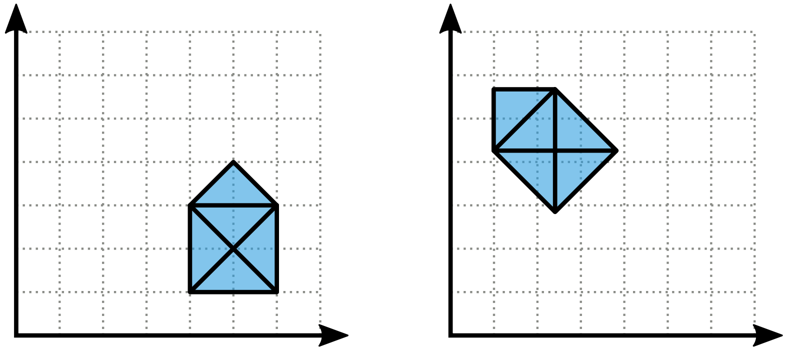
2D Transformations



translation



scaling



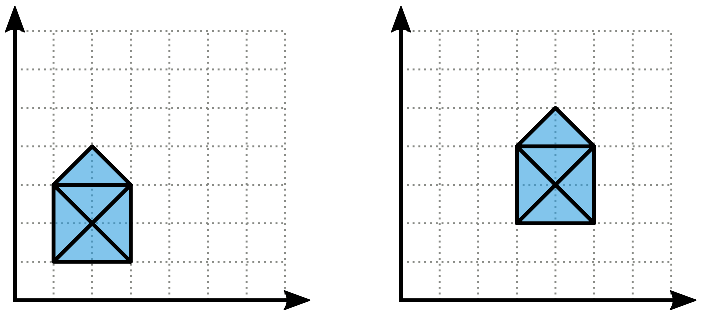
rotation

Some videos on Linear Algebra

2D Translation

- Translate object by t_x in x and t_y in y

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} x + t_x \\ y + t_y \end{pmatrix}$$

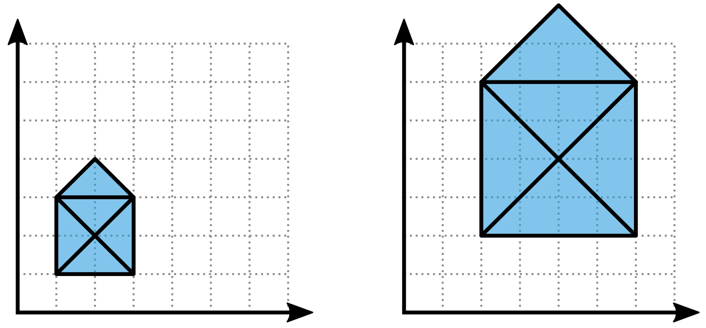


translation by $(2, 1)$

2D Scaling

- Scale object by s_x in x and s_y in y (around the origin!)

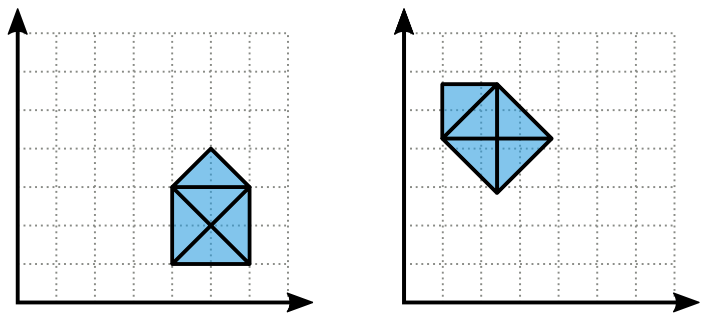
$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} s_x \cdot x \\ s_y \cdot y \end{pmatrix}$$



scaling by $(2, 2)$

2D Rotation

- Rotate object by θ degrees (around the origin!)



rotation by 45 degrees

2D Rotation

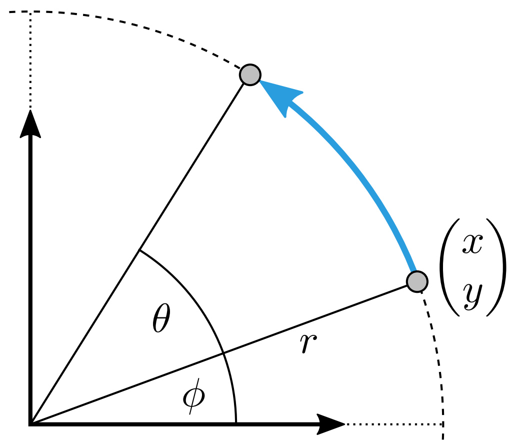
- Rotate point $(x, y) = (r\cos\phi, r\sin\phi)$ by θ degrees around the origin

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} r\cos(\phi+\theta) \\ r\sin(\phi+\theta) \end{pmatrix}$$

$$= \begin{pmatrix} r\cos\phi\cos\theta - r\sin\phi\sin\theta \\ r\cos\phi\sin\theta + r\sin\phi\cos\theta \end{pmatrix}$$

$$= \begin{pmatrix} \cos\theta\cdot x - \sin\theta\cdot y \\ \cos\theta\cdot y + \sin\theta\cdot x \end{pmatrix}$$

$$= \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix}$$

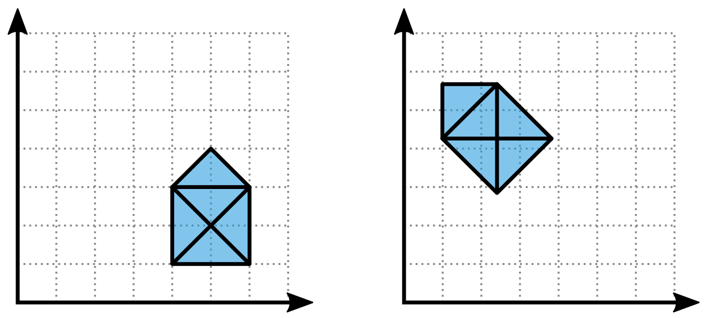


rotation by θ degrees

2D Rotation

- Rotate object by θ degrees (around the origin!)

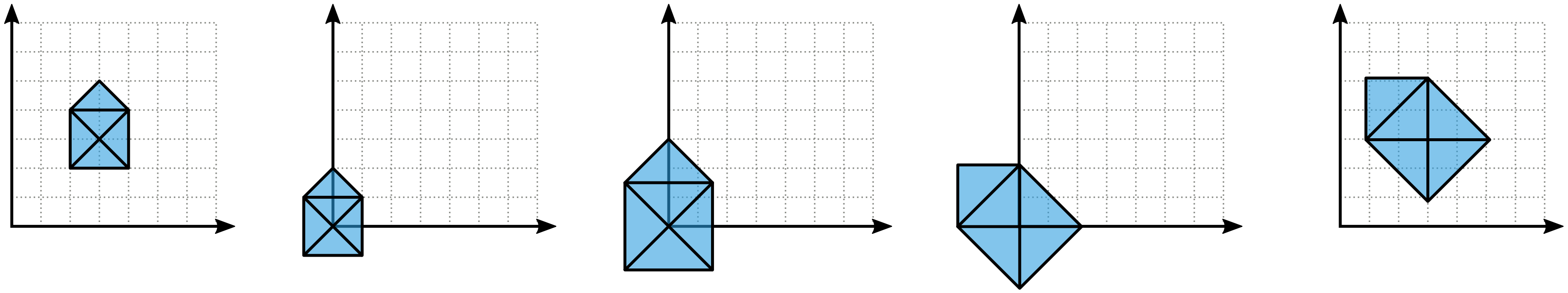
$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix}$$



rotation by 45 degrees

How to rotate/scale around object center?

1. Translate center to origin
2. Scale object
3. Rotate object
4. Translate center back



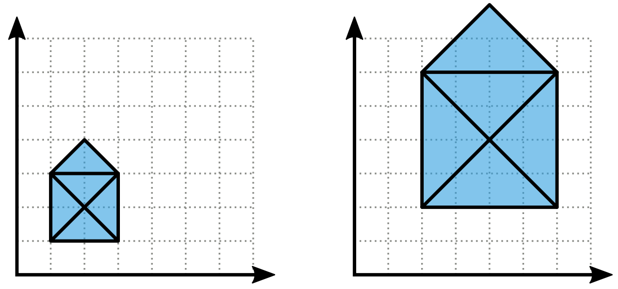
This can get quite messy!

Important Questions

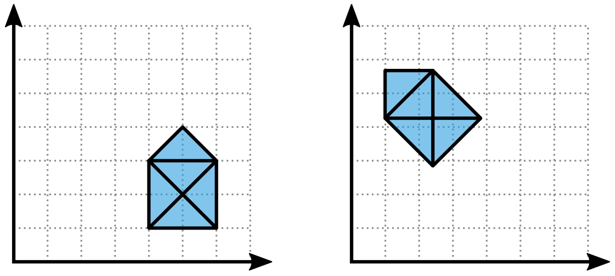
- How to efficiently combine several transformations?

Represent transformations as matrices!

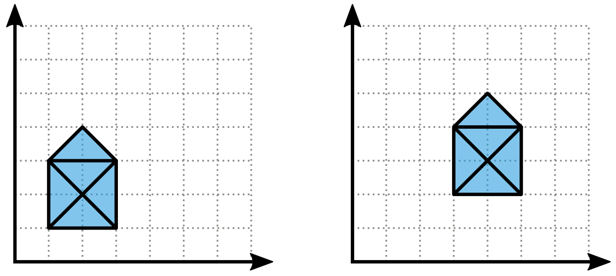
Matrix Representation



scaling



rotation



translation

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

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

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

Which transformations can be written as matrices?

Linear Maps & Matrices

- Assume a *linear* transformation $L\colon \mathbb{R}^n \to \mathbb{R}^n$
  - $L(\mathbf{a} + \mathbf{b}) = L(\mathbf{a}) + L(\mathbf{b})$
  - $L(\alpha\, \mathbf{a}) = \alpha\, L(\mathbf{a})$
- Point $\mathbf{x} = (x_1, \ldots, x_n)$ can be written as

$$\mathbf{x} = x_1\mathbf{e}_1 + x_2\mathbf{e}_2 + \ldots + x_n\mathbf{e}_n$$

Linear Maps & Matrices

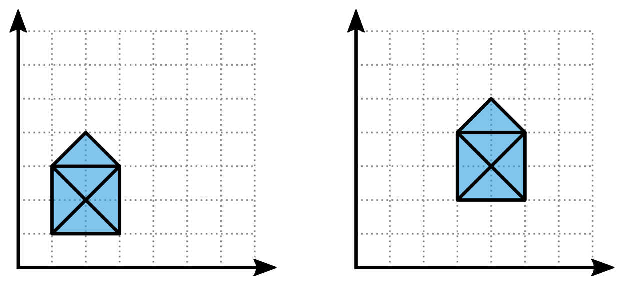
- Exploit linearity of $L\colon \mathbb{R}^n \to \mathbb{R}^n$

$$
\begin{aligned}
L(\mathbf{x}) &= L(x_1\mathbf{e}_1 + x_2\mathbf{e}_2 + \ldots + x_n\mathbf{e}_n) \\
&= x_1\, L(\mathbf{e}_1) + x_2\, L(\mathbf{e}_2) + \ldots + x_n\, L(\mathbf{e}_n) \\
&= \underbrace{\left(L(\mathbf{e}_1),\; L(\mathbf{e}_2),\; \ldots,\; L(\mathbf{e}_n)\right)}_{=:\mathbf{L}} \cdot \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} \;=\; \mathbf{L}\,\mathbf{x}
\end{aligned}
$$

- Every linear transformation $L\colon \mathbb{R}^n \to \mathbb{R}^n$ can be written as a *unique* $(n \times n)$ matrix $\mathbf{L}$ whose columns are the images of the basis vectors $\{\mathbf{e}_1, \ldots, \mathbf{e}_n\}$.

VERY useful fact! VERY-VERY useful!

Linear vs. Affine Transformations



- Every linear transformation has to preserve the origin
  - $L(\mathbf{0}) = \mathbf{L} \cdot \mathbf{0} = \mathbf{0}$
- Translation is not a linear mapping
  - $T(0,0) = (t_x, t_y)$
- Translation is an **affine** transformation
  - affine mapping = linear mapping + translation
  - $\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} a & b \\ c & d \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix} = \mathbf{Lx} + \mathbf{t}$

But we REALLY want to represent translations as matrices!

Homogeneous Coordinates

- Extend cartesian coordiantes (x, y) to *homogeneous* coordinates (x, y, w)
  - Points are represented by $(x, y, 1)^\mathsf{T}$
  - Vectors are represented by $(x, y, 0)^\mathsf{T}$
- Only homogeneous coordinates with $w \in \{0, 1\}$ make sense
  - vector + vector = vector
  - point - point = vector
  - point + vector = point
  - point + point = ??
- Only affine combinations of points $\mathbf{x}_i$ result in a point
  - $\sum_i \alpha_i \mathbf{x}_i$ with $\sum_i \alpha_i = 1$

Quiz: Transformations

Which matrix represents the 2D translation $\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} x+a \\ y+b \end{pmatrix}$?

A: $\begin{pmatrix} 1 & 0 & a \\ 0 & 1 & b \\ 0 & 0 & 1 \end{pmatrix}$

B: $\begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & 1 \end{pmatrix}$

C: $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ a & b & 1 \end{pmatrix}$

D: $\begin{pmatrix} 0 & 0 & a \\ 0 & 0 & b \\ 0 & 0 & 1 \end{pmatrix}$

Homogeneous Coordinates

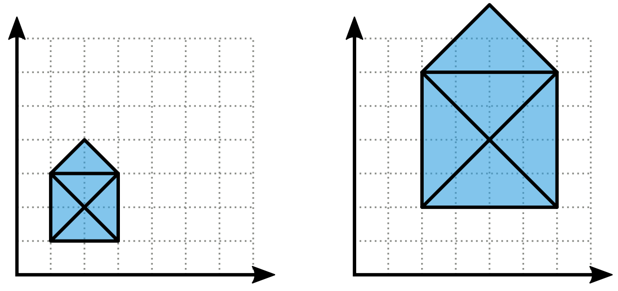
- Matrix representation of translations

$$\begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix} \quad \longleftrightarrow \quad \begin{pmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

- Matrix representation of arbitrary affine transformation

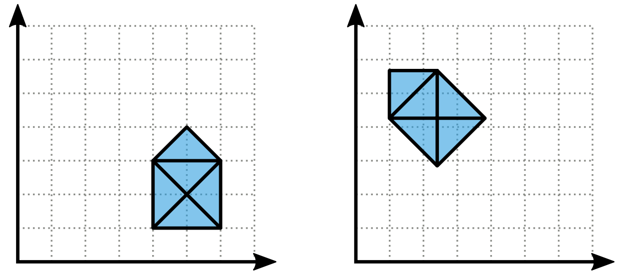
$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix} \quad \longleftrightarrow \quad \begin{pmatrix} a & b & t_x \\ c & d & t_y \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Matrix Representation



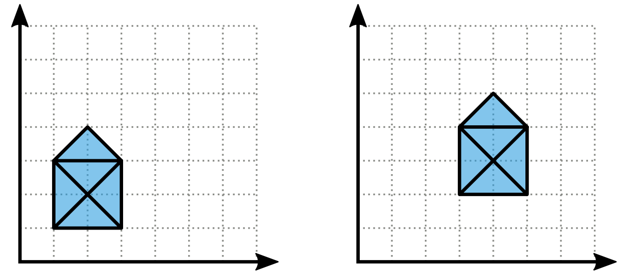
scaling

$$\begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{pmatrix}$$



rotation

$$\begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}$$



translation

$$\begin{pmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{pmatrix}$$

Columns of matrix are images of basis vectors!

Concatenation of Transformations

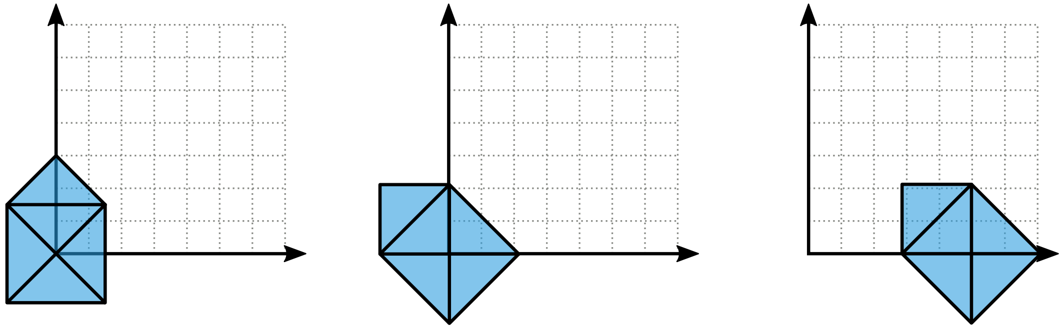
- Apply sequence of affine transformations $\mathbf{A}_1, \ldots, \mathbf{A}_k$
- Concatenate transformations by matrix multiplication

$$A_k(\ldots A_2(A_1(\mathbf{x}))) \;=\; \underbrace{\mathbf{A}_k \cdots \mathbf{A}_2 \cdot \mathbf{A}_1}_{\mathbf{M}} \cdot \mathbf{x}$$

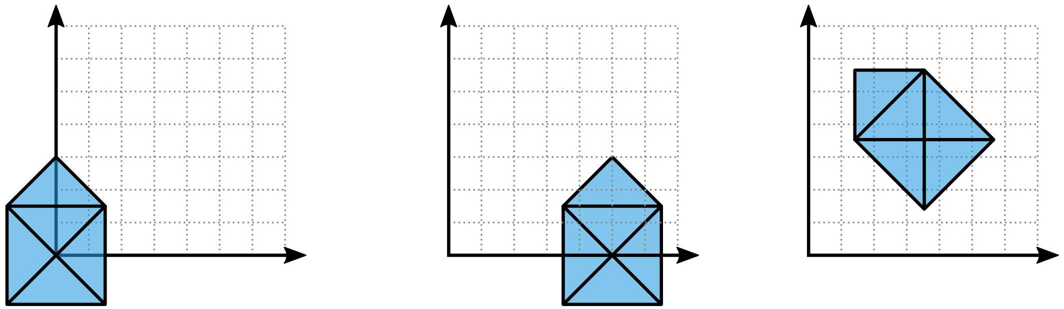
- Precompute matrix $\mathbf{M}$ and apply it to all (=many!) object vertices.
  Very important for performance!

Ordering of Matrix Multiplication

- First rotation, then translation

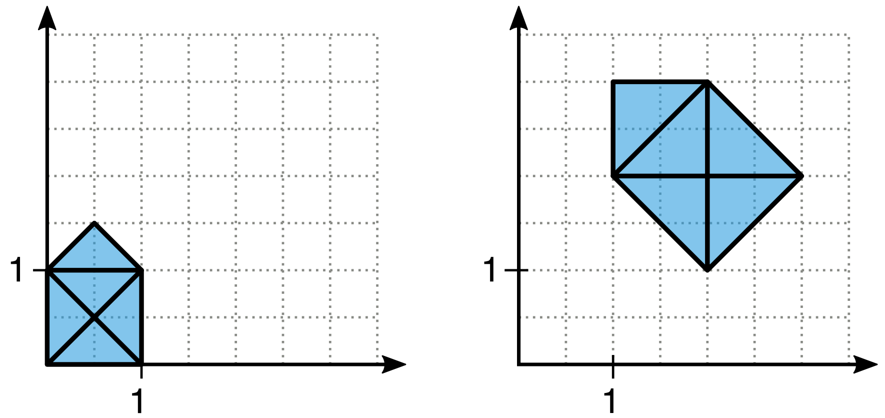


- First translation, then rotation



Quiz: Transformations

Which matrix computes the transformation on the right?



A: $\mathbf{T}(2,1) \cdot \mathbf{S} \cdot \mathbf{R}$

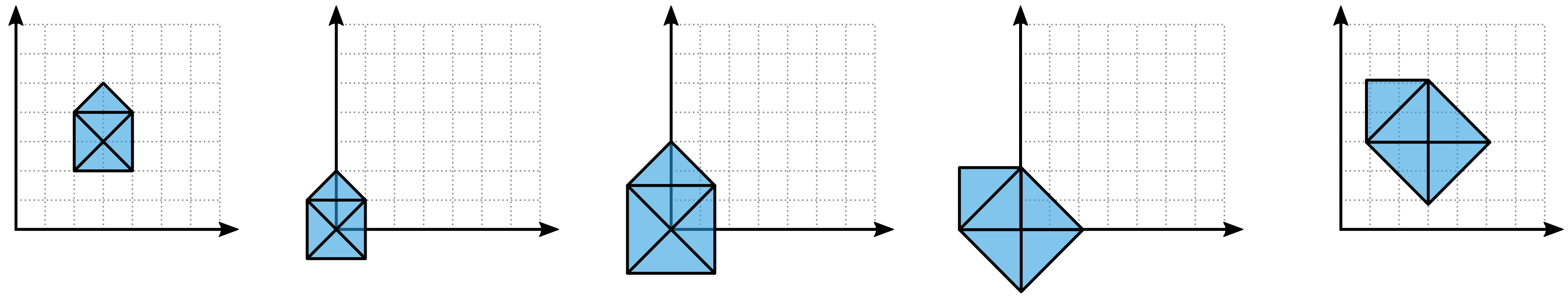
B: $\mathbf{T}(2,2) \cdot \mathbf{S} \cdot \mathbf{R} \cdot \mathbf{T}\left(-\frac{1}{2}, -\frac{1}{2}\right)$

C: $\mathbf{R} \cdot \mathbf{S} \cdot \mathbf{T}\left(\frac{3}{2}, \frac{3}{2}\right)$

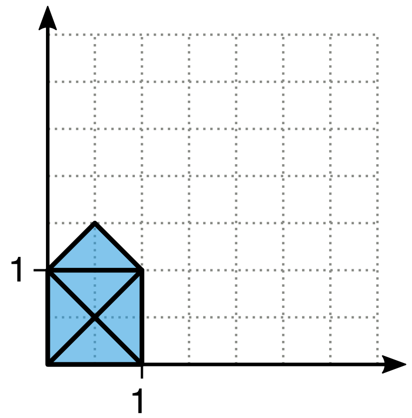
D: $\mathbf{T}\left(-\frac{1}{2}, -\frac{1}{2}\right) \cdot \mathbf{R} \cdot \mathbf{S} \cdot \mathbf{T}(2,2)$

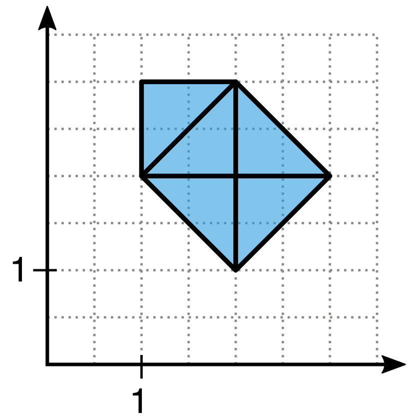
How to rotate/scale around object center?

1. Translate center to origin
2. Scale object
3. Rotate object
4. Translate center back



Matrix Representation?





What are the images of basis vectors?

Important Questions

- What is preserved by affine transformations?
- What is preserved by orthogonal transformations?

Affine Transformations

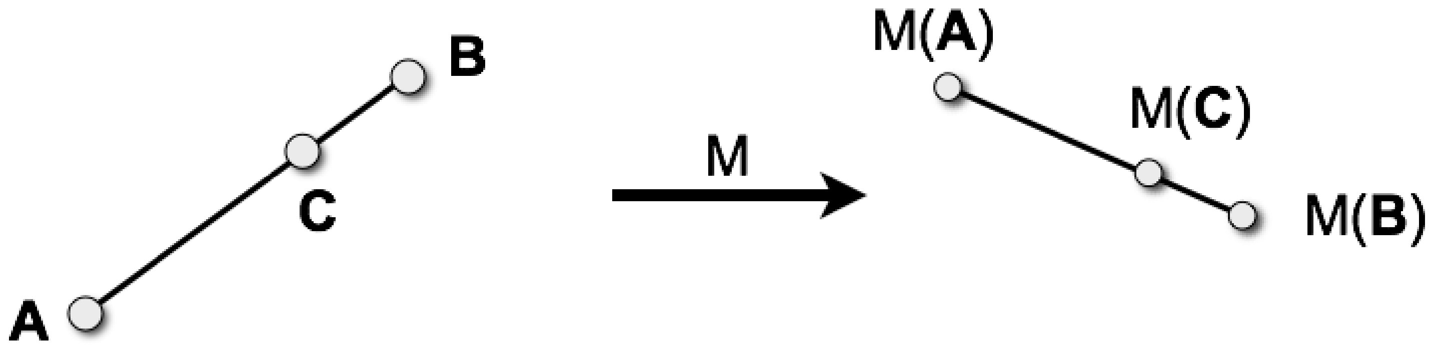
- Any point $\mathbf{C}$ on a line is an affine combination of its endpoints $\mathbf{A}$ and $\mathbf{B}$:

$$(1-\alpha)\mathbf{A} + \alpha\mathbf{B}$$

- Affine transformation $\mathbf{M}$ preserves affine combinations

$$\mathbf{M}((1-\alpha)\mathbf{A} + \alpha\mathbf{B}) \ = \ (1-\alpha)\mathbf{M}(\mathbf{A}) + \alpha\mathbf{M}(\mathbf{B})$$

- Straight lines stay straight lines



Orthogonal Transformations

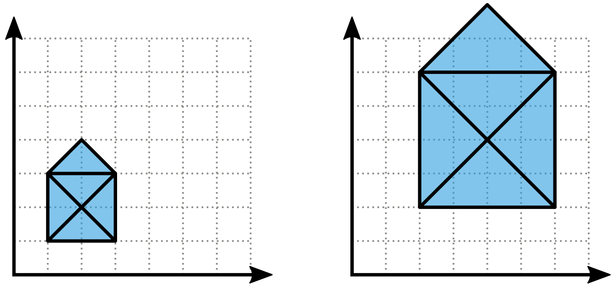
- A matrix $\mathbf{M}$ is *orthogonal* iff…
  - …its columns are orthonormal vectors
  - …its rows are orthonormal vectors
  - …its inverse is its transposed: $\mathbf{M}^{-1} = \mathbf{M}^{\mathsf{T}}$
- Orthogonal matrices / mappings…
  - …preserve angles and lengths
  - …can only be rotations or reflections
  - …have determinant +1 or -1

3D Transformations

- Use homogeneous coordinates (x, y, z, w)!
  - Points are represented by $(x, y, z, 1)^\mathsf{T}$
  - Vectors are represented by $(x, y, z, 0)^\mathsf{T}$
- Represent affine transformation by (4×4) matrices

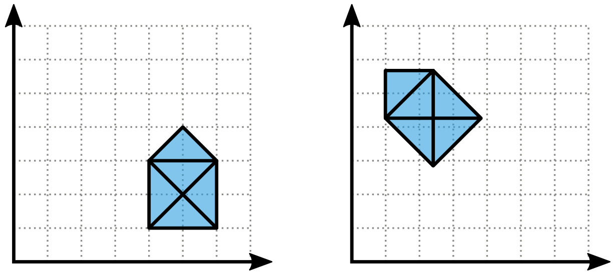
$$\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ z \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \\ t_z \end{pmatrix} \quad \longleftrightarrow \quad \begin{pmatrix} a & b & c & t_x \\ d & e & f & t_y \\ g & h & i & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix}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

3D Transformations



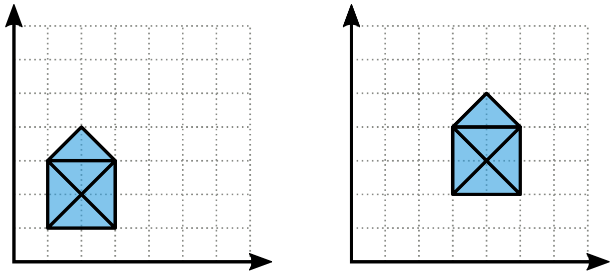
scaling

$$\begin{pmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$



rotation

$$\begin{pmatrix} ? & ? & ? & ? \\ ? & ? & ? & ? \\ ? & ? & ? & ? \\ ? & ? & ? & ? \end{pmatrix}$$



translation

$$\begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Rotation around x/y/z axes

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

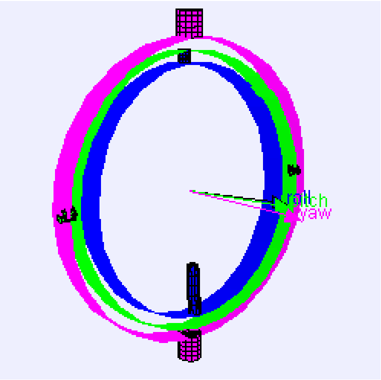
$$\mathbf{R}_y(\theta) = \begin{pmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

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

Rotation around x/y/z axes

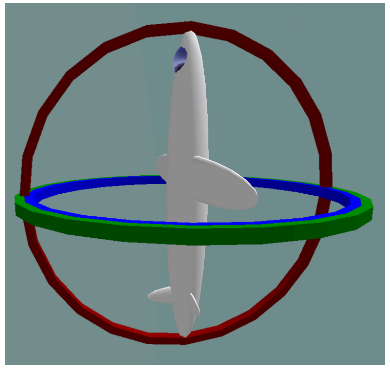
- Can we compose any 3D rotation from rotations around the axes $\mathbf{R}_x$, $\mathbf{R}_y$, $\mathbf{R}_z$?

$$\mathbf{R}(\alpha, \beta, \gamma) \stackrel{?}{=} \mathbf{R}_x(\alpha) \cdot \mathbf{R}_y(\beta) \cdot \mathbf{R}_z(\gamma)$$



Euler angles

- This representation is called *Euler angles*
  - Often used in flight simulators: roll, pitch, yaw
  - Problem: *gimbal lock*!



Gimbal lock

Images from Wikipedia

Gimbal Lock

- Let's represent rotation in Euler angles

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

- What happens if we choose $\beta = 90^\circ$?

$$\mathbf{R}(\alpha,\pi/2,\gamma) = \begin{pmatrix} 0 & 0 & 1 & 0 \\ \sin(\alpha+\gamma) & \cos(\alpha+\gamma) & 0 & 0 \\ -\cos(\alpha+\gamma) & \sin(\alpha+\gamma) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

  - α and γ control the same rotation. Only one degree of freedom left!

3D Rotations: Rodrigues Formula

- Rotation by angle θ around (normalized) axis $\mathbf{a}$

$$\mathbf{R}(\mathbf{a},\theta) \;=\; \mathbf{a}\mathbf{a}^\mathsf{T} + \cos\theta\left(\mathbf{I} - \mathbf{a}\mathbf{a}^\mathsf{T}\right) + \sin\theta \begin{pmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{pmatrix}$$

- Combining the matrices and writing $\mathbf{a} = (x, y, z)$, $c = \cos\theta$, and $s = \sin\theta$ gives

$$\mathbf{R}(\mathbf{a},\theta) \;=\; \begin{pmatrix} x^2(1-c)+c & xy(1-c)-zs & xz(1-c)+ys \\ yx(1-c)+zs & y^2(1-c)+c & yz(1-c)-xs \\ xz(1-c)-ys & yz(1-c)+xs & z^2(1-c)+c \end{pmatrix}$$

3D Rotations: Rodrigues Formula

$$\mathbf{R}(\mathbf{a},\theta) \;=\; \mathbf{a}\mathbf{a}^\mathsf{T} + \cos\theta\left(\mathbf{I} - \mathbf{a}\mathbf{a}^\mathsf{T}\right) + \sin\theta \underbrace{\begin{pmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{pmatrix}}_{=:\,\mathbf{A}}$$

- How to prove this magic formula?
- Matrix $\mathbf{A}$ computes the cross-product $\mathbf{A}\mathbf{x} = \mathbf{a} \times \mathbf{x}$
- Assume orthonormal coordinate system $\mathbf{e}_1, \mathbf{e}_2, \mathbf{a}$

$$\begin{aligned} \mathbf{R}\mathbf{a} &= \mathbf{a} \\ \mathbf{R}\mathbf{e}_1 &= \cos\theta\,\mathbf{e}_1 + \sin\theta\,\mathbf{e}_2 \\ \mathbf{R}\mathbf{e}_2 &= -\sin\theta\,\mathbf{e}_1 + \cos\theta\,\mathbf{e}_2 \end{aligned}$$

Summary

- **Linear transformations**
  - Scaling, rotation
  - Can be represented by 3×3 matrices
- **Affine transformations**
  - Linear transformation + translation
  - Can be represented by 4×4 matrices with homogeneous coordinates
- **Represent all transformations by matrices**
  - Concatenate transformations by matrix multiplication
  - Huge performance benefit!

Literature

- Marschner & Shirley: *Fundamentals of Computer Graphics*, 5th Edition, AK Peters, 2021.
    - Chapter 7