Transformations

CS425: Computer Graphics I

Fabio Miranda

https://fmiranda.me

UIC COMPUTER SCIENCE

Overview

- Matrices
- Homogeneous coordinates
- Transformations (2D)

Matrices

- A matrix can be used as a tool for manipulating vectors and points.
- A matrix $\mathbf{A}$ is described by $p \times q$ scalars:

$$\mathbf{A} = \begin{pmatrix} a_{0,0} & a_{0,1} & \cdots & a_{0,q-1} \\ a_{1,0} & a_{1,1} & \cdots & a_{1,q-1} \\ \vdots & \vdots & \ddots & \vdots \\ a_{p-1,0} & a_{p-1,1} & \cdots & a_{p-1,q-1} \end{pmatrix}, \text{with } a_{ij} \in \mathbb{R}, 0 \leq i \leq p-1, 0 \leq j \leq q-1$$

- Unit matrix or identity matrix $\mathbf{I}$: square matrix containing ones in the diagonal and zeros elsewhere.

Operations

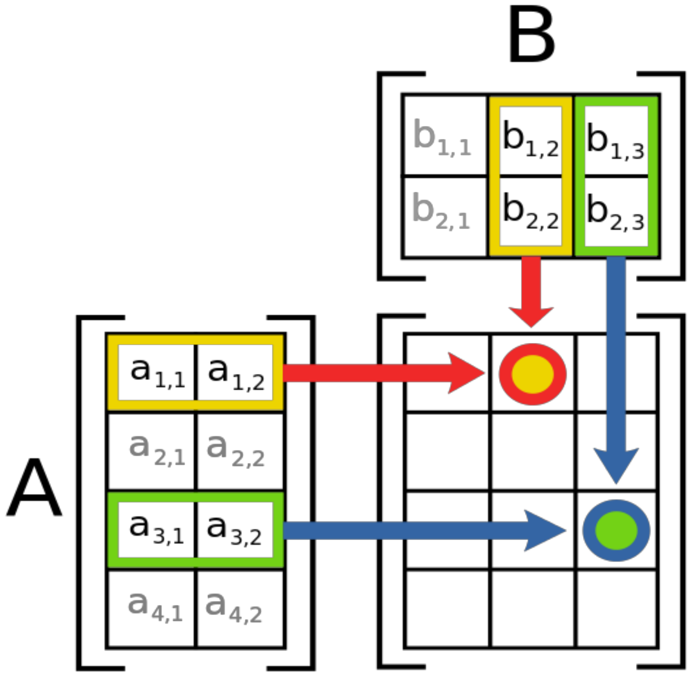
$$\mathbf{A} + \mathbf{B} = \begin{pmatrix} a_{0,0} & a_{0,1} \\ a_{1,0} & a_{1,1} \end{pmatrix} + \begin{pmatrix} b_{0,0} & b_{0,1} \\ b_{1,0} & b_{1,1} \end{pmatrix} = \begin{pmatrix} a_{0,0} + b_{0,0} & a_{0,1} + b_{0,1} \\ a_{1,0} + b_{1,0} & a_{1,1} + b_{1,1} \end{pmatrix}$$ Addition

$$\alpha \mathbf{A} = \alpha \begin{pmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{pmatrix} = \begin{pmatrix} \alpha a_{0,0} & \alpha a_{0,1} \\ \alpha a_{1,0} & \alpha a_{1,1} \end{pmatrix}$$ Multiplication by scalar

Matrix multiplication

- $\mathbf{AB}$: Entry i, j is given by multiplying the entries on the i-th row of $\mathbf{A}$ with the entries of the j-th column of $\mathbf{B}$ and summing the results.
- Product $\mathbf{AB}$ defined iff number of columns in $\mathbf{A}$ equals the number of rows in $\mathbf{B}$.
- It is NOT commutative.

$$\mathbf{AB} \neq \mathbf{BA}$$



Matrix multiplication

- Dot product between rows of $\mathbf{A}$ and columns of $\mathbf{B}$.

$$\mathbf{A}_{p,q}\mathbf{B}_{q,r} = \begin{pmatrix} \mathbf{r}_0 \\ \mathbf{r}_1 \\ \vdots \\ \mathbf{r}_{p-1} \end{pmatrix}_{p\times q} \begin{pmatrix} \mathbf{c}_0 & \mathbf{c}_1 & \dots & \mathbf{c}_{q-1} \end{pmatrix}_{q\times r}$$

$$= \begin{pmatrix} \mathbf{r}_0 \cdot \mathbf{c}_1 & \mathbf{r}_0 \cdot \mathbf{c}_1 & \dots & \mathbf{r}_0 \cdot \mathbf{c}_{q-1} \\ \mathbf{r}_1 \cdot \mathbf{c}_1 & \mathbf{r}_1 \cdot \mathbf{c}_1 & \dots & \mathbf{r}_1 \cdot \mathbf{c}_{q-1} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{r}_{p-1} \cdot \mathbf{c}_1 & \mathbf{r}_{p-1} \cdot \mathbf{c}_1 & \dots & \mathbf{r}_{p-1} \cdot \mathbf{c}_{q-1} \end{pmatrix}_{p\times r}$$

Transpose

- The transpose of a matrix is a new matrix whose entities are reflected over the diagonal.

$$\begin{pmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{pmatrix}^T = \begin{pmatrix} a_{00} & a_{10} \\ a_{01} & a_{11} \end{pmatrix}$$

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

- The transpose of a product is the product of the transposed, in reverse order.

$$(\mathbf{AB})^T = \mathbf{B}^T \mathbf{A}^T$$

Inverse matrices

- The inverse of a matrix $\mathbf{A}$ is the matrix $\mathbf{A}^{-1}$ such that $\mathbf{A}\mathbf{A}^{-1} = \mathbf{I}$, where I is the identity matrix.

$$\mathbf{I} = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & 0 & 0 \\ \vdots & 0 & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}$$

- The inverse of a product is the product of the inverse in opposite order:

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

Diagonal matrices

- Matrix (usually a $n \times n$ square one) where all entries outside the diagonal are all zero: $\forall i, j \in \{0,1,\dots,n\}, i \neq j \Rightarrow d_{i,j} = 0$

$$\mathbf{D} = \begin{pmatrix} d_{0,0} & 0 & \cdots & 0 \\ 0 & d_{1,1} & 0 & 0 \\ \vdots & 0 & \ddots & \vdots \\ 0 & 0 & \cdots & d_{n-1,n-1} \end{pmatrix}$$

- Useful properties:

$$\mathbf{D} = \mathbf{D}^T \qquad \mathbf{D}^{-1} = \begin{pmatrix} d_{0,0}^{-1} & 0 & \cdots & 0 \\ 0 & d_{1,1}^{-1} & 0 & 0 \\ \vdots & 0 & \ddots & \vdots \\ 0 & 0 & \cdots & d_{n-1,n-1}^{-1} \end{pmatrix}$$

Orthogonal matrices

- Matrix where:
  1. Each column is a vector of length 1.
  2. Each column is orthogonal to all the other columns.
- Useful property: their inverse corresponds to their transpose

$$(\mathbf{R}^T\mathbf{R}) = I = (\mathbf{R}\mathbf{R}^T)$$

Transformation

- In graphics, a transformation is an operation that takes entities such as points, vectors, or colors, and converts them in some way.
- With transformations, we can position, reshape, and animate objects, lights, and cameras.
  - Rotate object
  - Translate object
  - Scale object
  - Change camera position
  - RGB to CMYK

Linear transformation

- A *linear* transformation is one that preserves vector addition and scalar multiplication:

$$f(\mathbf{x}) + f(\mathbf{y}) = f(\mathbf{x} + \mathbf{y})$$
$$kf(\mathbf{x}) = f(k\mathbf{x})$$

- Examples:

$f(\mathbf{x}) = 5\mathbf{x}$

Linear:
$5\mathbf{a} + 5\mathbf{b} = 5(\mathbf{a} + \mathbf{b})$ (scaling)

$f(\mathbf{x}) = \mathbf{x} + (7,3,2)$

Not linear: transforming two vectors add $(7,3,2)$ twice to form result
$\left(\mathbf{a} + (7,3,2)\right) + \left(\mathbf{b} + (7,3,2)\right) \neq (\mathbf{a} + \mathbf{b}) + (7,3,2)$ (translation)

Linear transformation

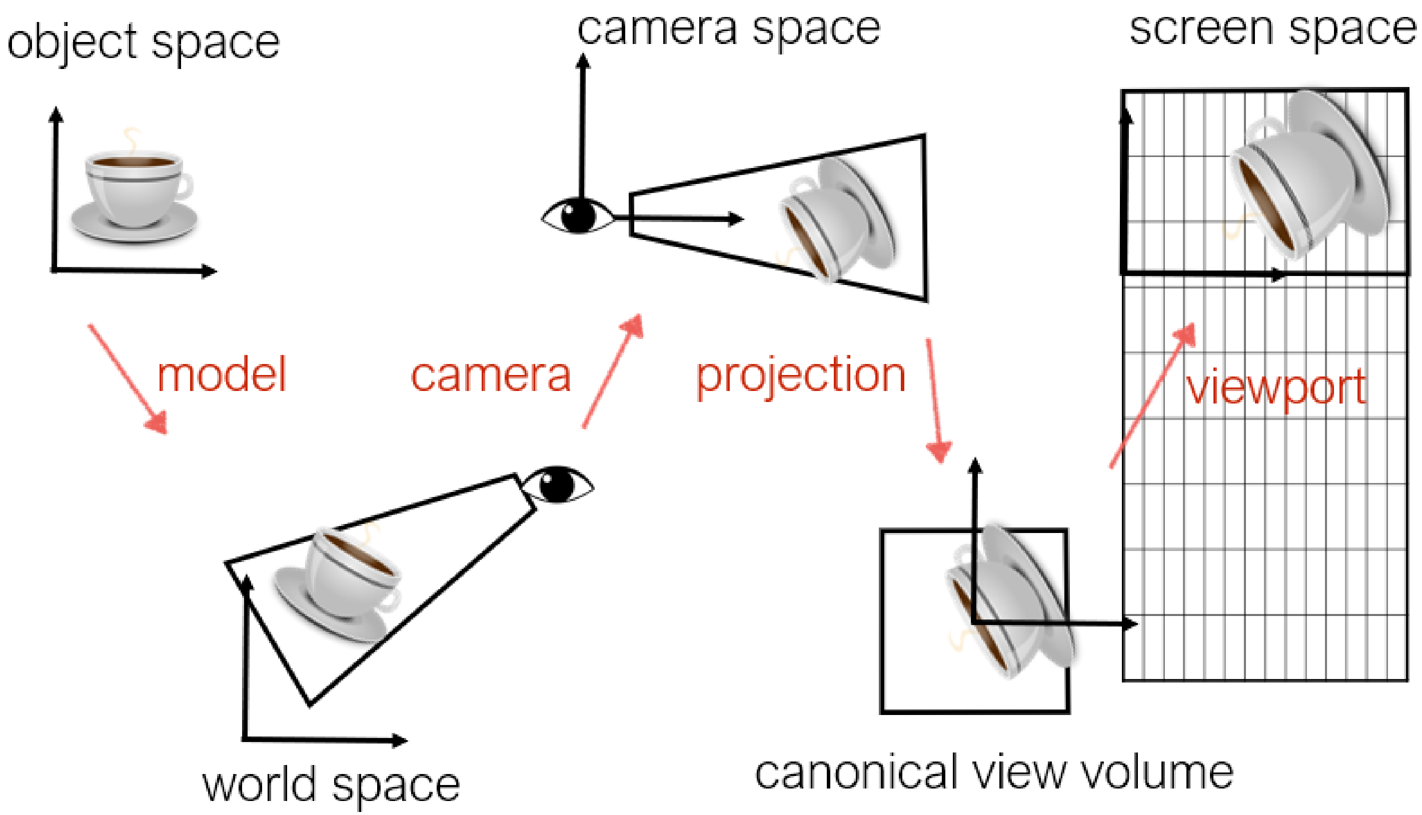
- A 2D **linear** map can be represented by a unique 2×2 matrix.

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

- Concatenation of mappings corresponds to multiplication of matrices.

$$L_2(L_1(\mathbf{x})) = \mathbf{L}_2 \mathbf{L}_1 \mathbf{x}$$

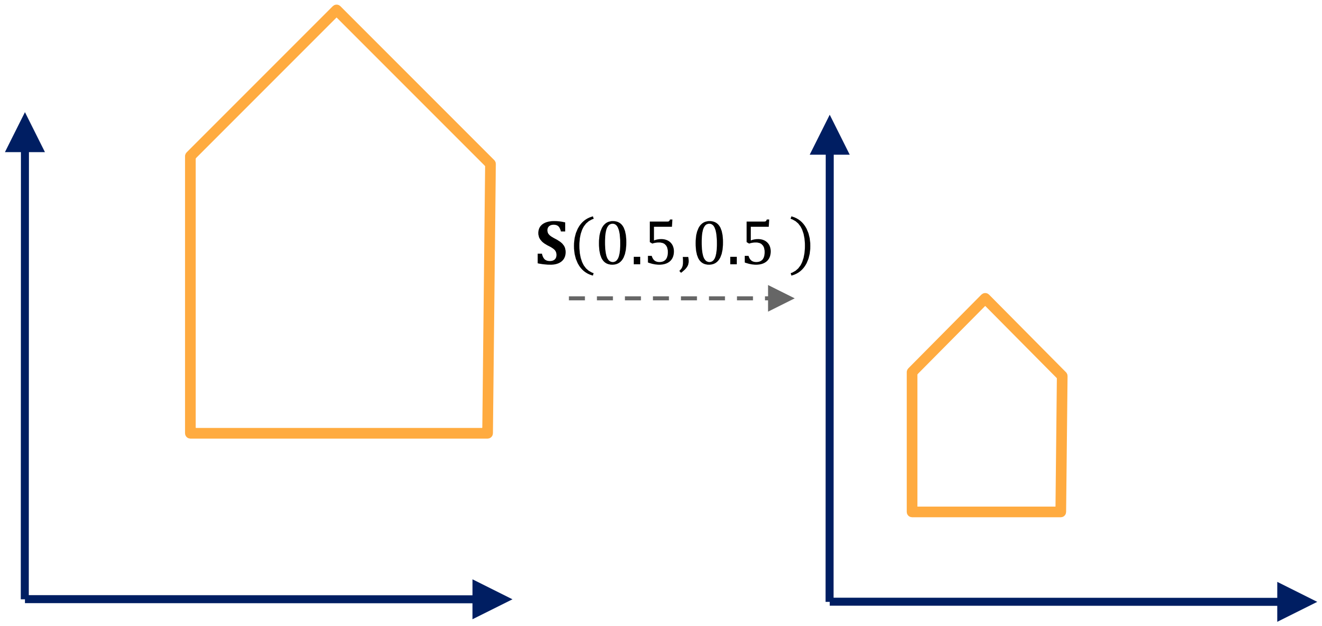
Linear transformation



Scaling in 2D

- A scaling matrix $\mathbf{S}(\mathbf{s}) = \mathbf{S}(s_x, s_y)$ scales an entity with factors s_x, and s_y along the x, and y directions.

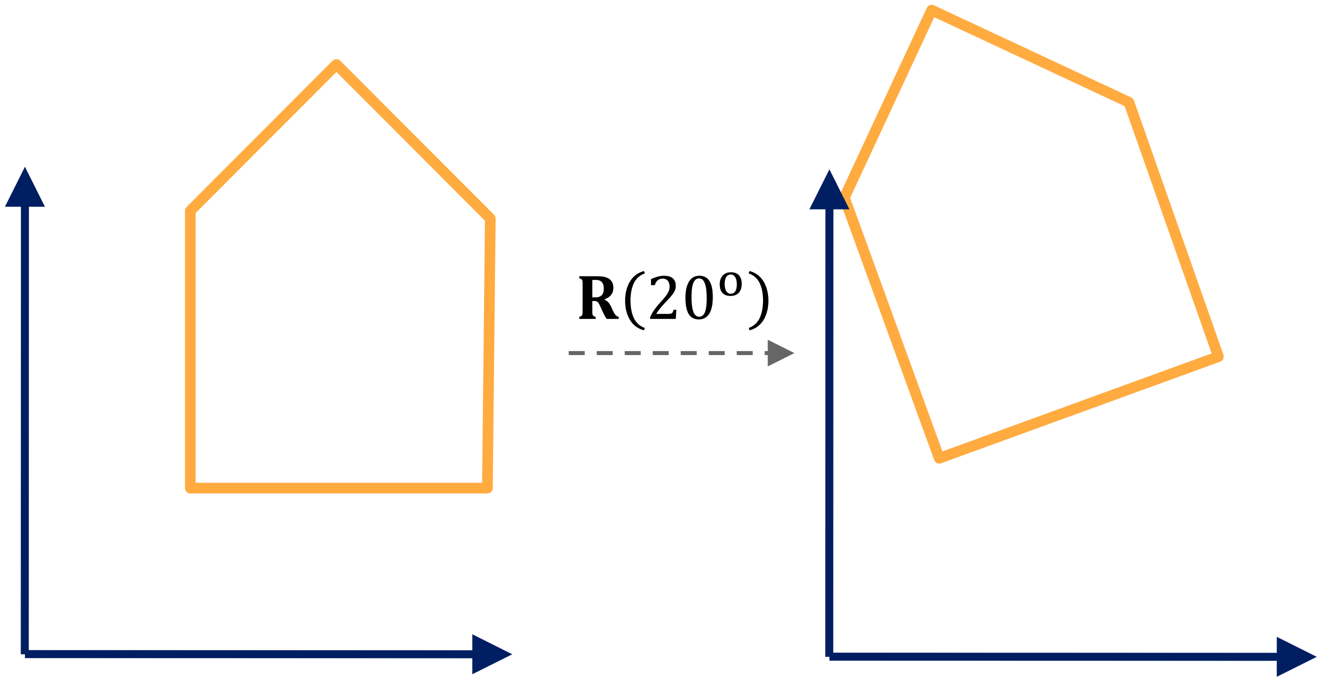
$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \underbrace{\begin{pmatrix} s_x & 0 \\ 0 & s_y \end{pmatrix}}_{\mathbf{S}(s_x, s_y)} \begin{pmatrix} x \\ y \end{pmatrix}$$



Rotation in 2D

- A rotation matrix $\mathbf{R}(\alpha)$ rotates an entity **around the origin** by α.

$$\begin{pmatrix}x'\\ y'\end{pmatrix} = \underbrace{\begin{pmatrix}\cos\alpha & -\sin\alpha\\ \sin\alpha & \cos\alpha\end{pmatrix}}_{\mathbf{R}(\alpha)}\begin{pmatrix}x\\ y\end{pmatrix}$$

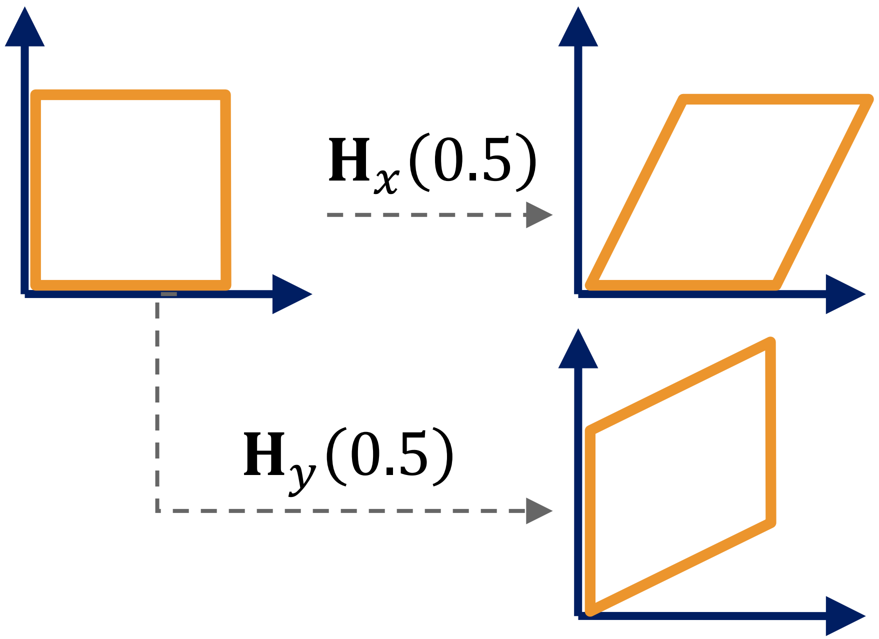


Shearing in 2D

- A shearing matrix distorts an entity along the x and y axis.

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & a \\ 0 & 1 \end{pmatrix}}_{\mathbf{H}_x(a)} \begin{pmatrix} x \\ y \end{pmatrix}$$

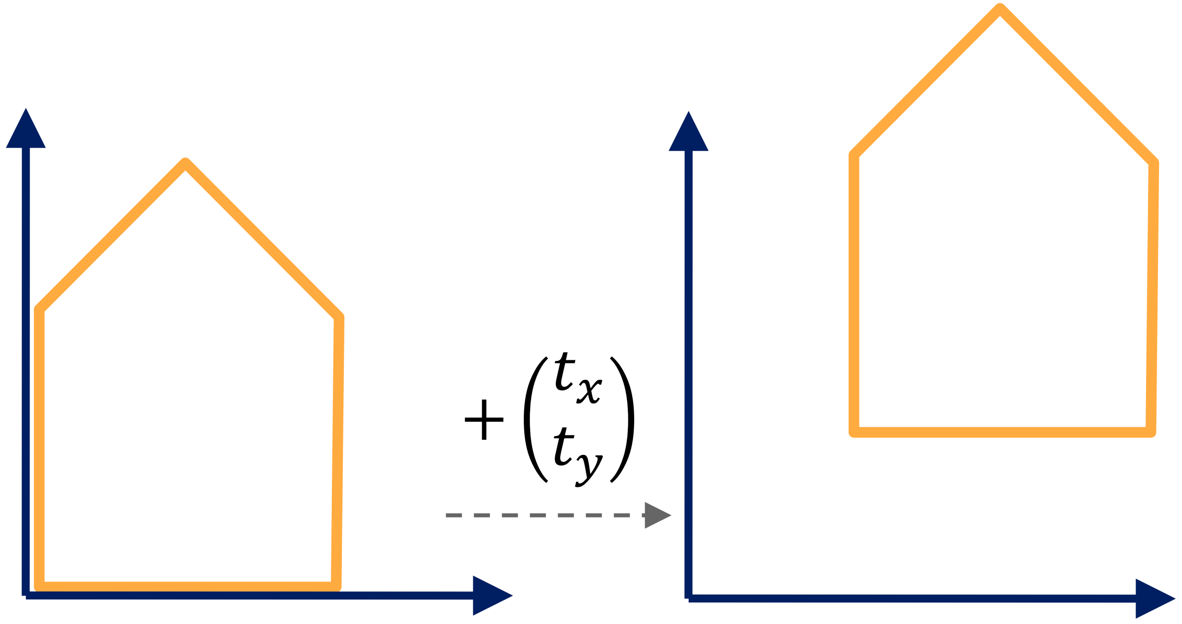
$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 0 \\ b & 1 \end{pmatrix}}_{\mathbf{H}_y(b)} \begin{pmatrix} x \\ y \end{pmatrix}$$



Translation

- Translating an entity?

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



- Matrix representation?

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \mathbf{T}(t_x, t_y) \begin{pmatrix} t_x \\ t_y \end{pmatrix}$$

Affine transformation

- Translation is not linear, but it is **affine**.
  - Affine transformation: preserves lines and parallelism.
- Affine map: linear map + translation

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix} = \mathbf{Lx} + \mathbf{t}$$

- How can we represent affine transformations with matrices?
  - We would like to handle all transformations in a unified framework.
  - Simpler code and easier to optimize.

Homogeneous coordinates

- Add an extra component:
  - 2D point: $(x, y, 1)^T$
  - 2D vector: $(x, y, 0)^T$
- Matrix representation of translations:

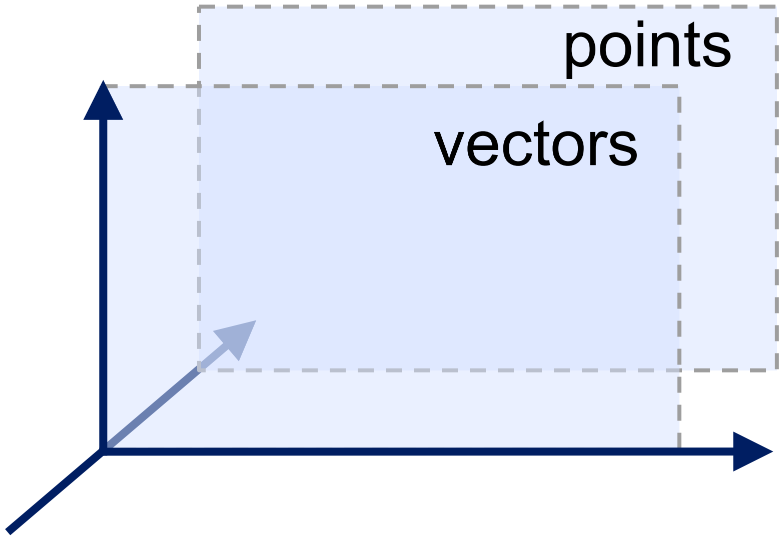
$$\begin{pmatrix} x' \\ y' \\ w' \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{pmatrix}}_{\mathbf{T}(t_x, t_y)} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} x + t_x \\ y + t_y \\ 1 \end{pmatrix}$$

Homogeneous coordinates

- Valid operation if the resulting w-coordinate is 1 or 0:
  - vector + vector = vector
  - point – point = vector
  - point + vector = point
  - point + point = ?

Homogeneous coordinates

- Geometric interpretation: 2 hyperplanes in $\mathbb{R}^3$



Affine transformation

- Affine map = linear map + translation

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix} = \mathbf{Lx} + \mathbf{t}$$

- Using homogeneous coordinates:

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

Transformations in 2D

Scaling

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

Rotation

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

Shearing

$$\mathbf{H}(s, t) = \begin{pmatrix} 1 & 0 & s \\ 0 & 1 & t \\ 0 & 0 & 1 \end{pmatrix}$$

Translation

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

Concatenation of transformations

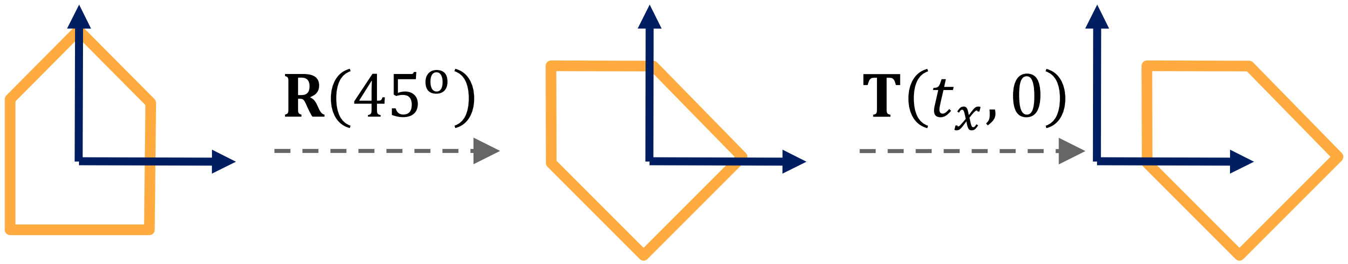
- Sequence of affine maps $\mathbf{A}_1, \mathbf{A}_2, \ldots, \boldsymbol{A_n}$
- Concatenation by matrix multiplication:

$$\mathbf{A}_n\left(\ldots\mathbf{A}_2\left(\mathbf{A}_1(\mathbf{x})\right)\right) = \mathbf{A}_n \ldots \mathbf{A}_2\ \mathbf{A}_1 \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

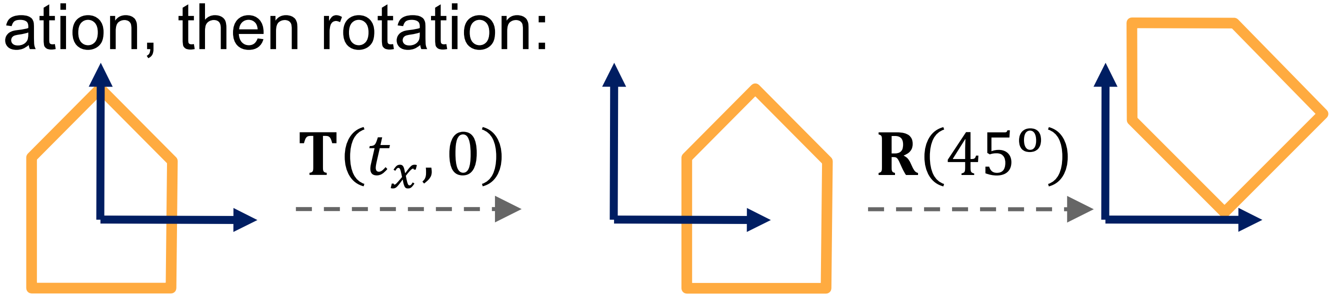
- Very important for performance!
- Matrix multiplication is not commutative, ordering is important!

Rotation and translation

- Matrix multiplication is not commutative!
- First rotation, then translation:

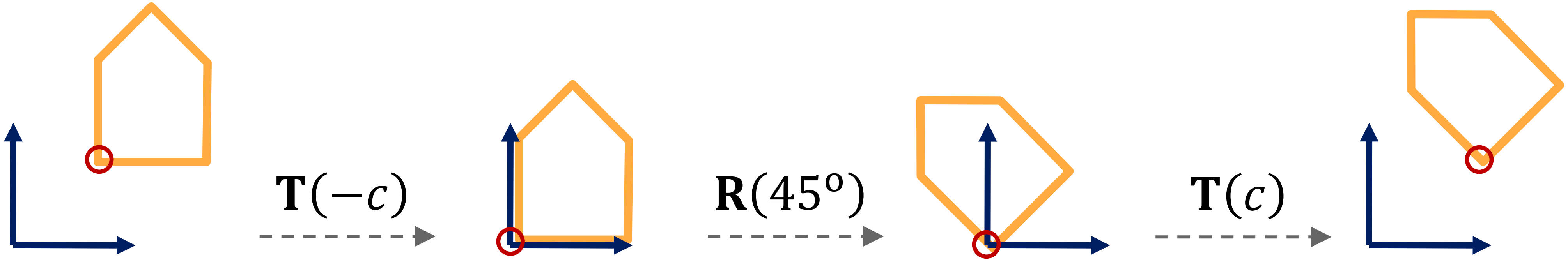


- First translation, then rotation:



Rotation in 2D

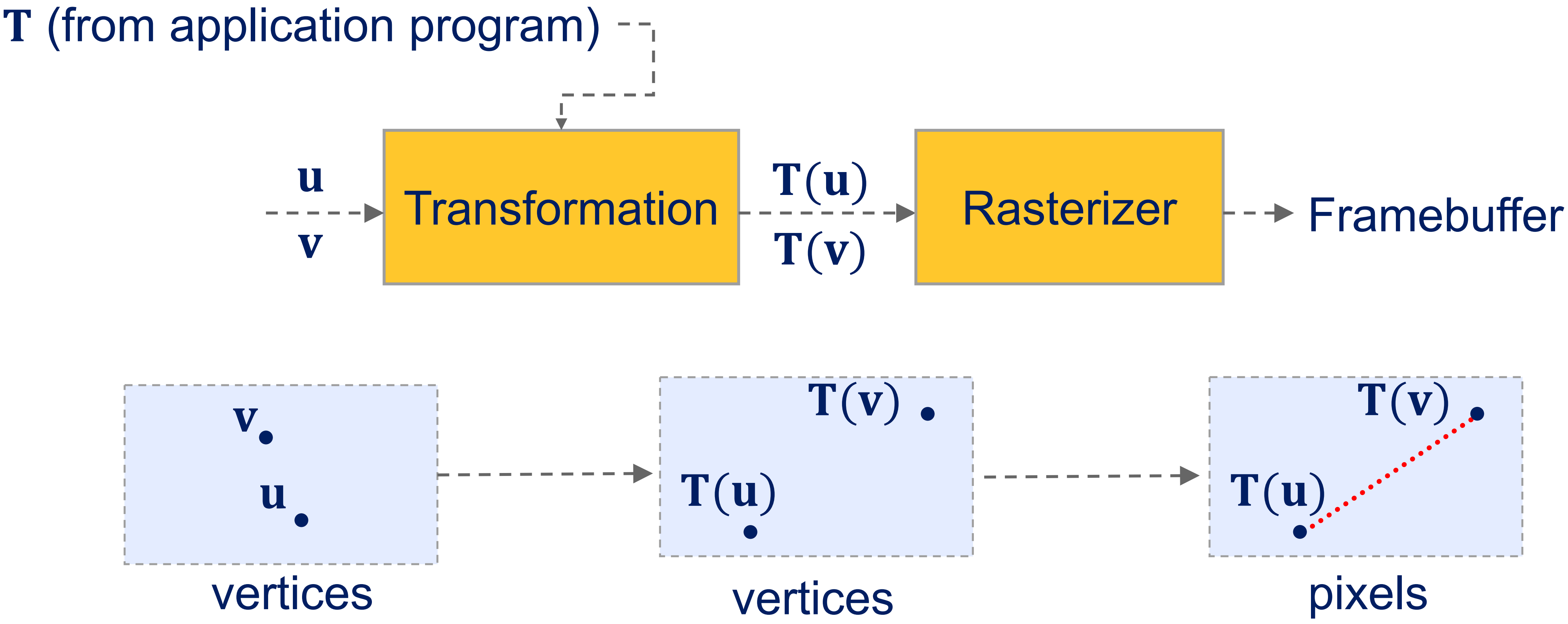
- How to rotate around a given point c?
  1. Translate c to origin
  2. Rotate
  3. Translate back



- Matrix representation:

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

Graphics pipeline



References

- Real-time Rendering, 3rd Ed. by Tomas Akenine-Möller, Eric Haines, and Naty Hoffman (Chapter 4, Appendix A)
- Interactive Computer Graphics 7th Ed. by Ed Angel and Dave Shreiner (Chapter 3)