2.1.1: 2D Transformations

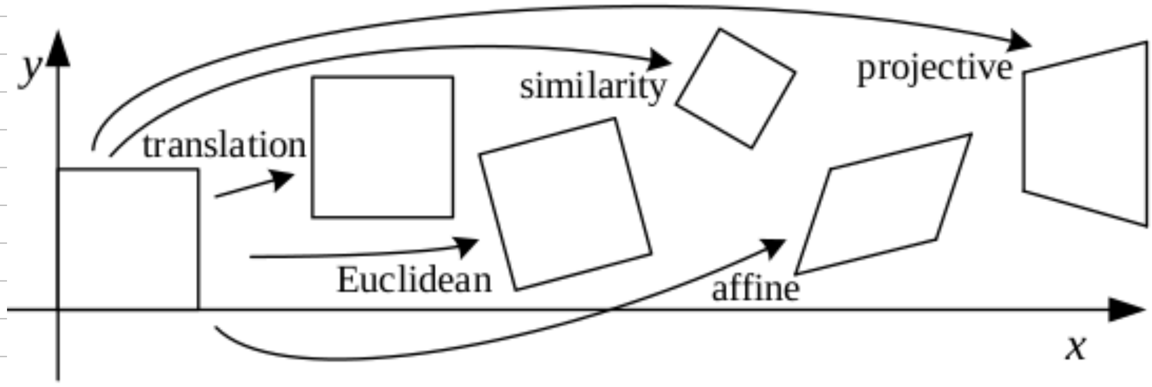

Figure 2.4 *Basic set of 2D planar transformations.*

Translation: $x' = x + t$, or $x' = [I \quad t]\bar{x}$ — this is a 2×3 matrix

I: 2×2 identity matrix

$$\bar{x}' = \begin{bmatrix} I & t \\ 0^T & 1 \end{bmatrix} \bar{x}$$

3×3 matrix ($[0^T \; 1]$ row appended)
can chain transformations together

Rotation + translation: • Euclidean distances are preserved $x' = Rx + t$

$x' = [R \quad t]\bar{x}$

R = orthonormal rotation matrix

$$R = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \qquad RR^T = I, \quad |R| = 1$$

Scaled rotation: • angles between lines are preserved $x' = sRx + t$ (s = arbitrary scale factor)

$$x' = [sR \quad t]\bar{x} = \begin{bmatrix} a & -b & t_x \\ b & a & t_y \end{bmatrix} \bar{x}$$

We <u>don't</u> require that $a^2 + b^2 = 1$

Affine: Preserves parallel lines $x' = A\bar{x} = \begin{bmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \end{bmatrix} \bar{x}$

Projective: aka perspective transform $\tilde{x}' = \tilde{H}\tilde{x}$
preserves straight lines

$\tilde{H}$ is an arbitrary 3×3 matrix

x' must be normalized to get an inhomogeneous result

$$x' = \frac{h_{00}x + h_{01}y + h_{02}}{h_{20}x + h_{21}y + h_{22}} \qquad y' = \frac{h_{10}x + h_{11}y + h_{12}}{h_{20}x + h_{21}y + h_{22}}$$

heirarchy of 2D transformations

- these transforms form a nested set of groups → Lie groups

Transformation	Matrix	# DoF	Preserves	Icon
translation	$\begin{bmatrix} \mathbf{I} & \mathbf{t} \end{bmatrix}_{2\times 3}$	2	orientation	
rigid (Euclidean)	$\begin{bmatrix} \mathbf{R} & \mathbf{t} \end{bmatrix}_{2\times 3}$	3	lengths	
similarity	$\begin{bmatrix} s\mathbf{R} & \mathbf{t} \end{bmatrix}_{2\times 3}$	4	angles	
affine	$\begin{bmatrix} \mathbf{A} \end{bmatrix}_{2\times 3}$	6	parallelism	
projective	$\begin{bmatrix} \tilde{\mathbf{H}} \end{bmatrix}_{3\times 3}$	8	straight lines	

Table 2.1 *Hierarchy of 2D coordinate transformations, listing the transformation name, its matrix form, the number of degrees of freedom, what geometric properties it preserves, and a mnemonic icon. Each transformation also preserves the properties listed in the rows below it, i.e., similarity preserves not only angles but also parallelism and straight lines. The 2×3 matrices are extended with a third $[\mathbf{0}^T \ 1]$ row to form a full 3×3 matrix for homogeneous coordinate transformations.*

- 3D transformations are mostly the same
- also talked about 3D rotations, bit too complicated to worry about now
- these kinds of rotations/transformations have many references to group theory
- Unit quaternions (x, y, z, w) → very popular for pose & pose interpolation

Spherical linear interpolation (slerp) → determine a rotation that is partway between 2 given rotations (described in algorithm 2.1 → should try implementing this)

2.1.4: 3D to 2D Projections

P = 3D points
x = 2D points

Orthography & Para-perspective (simplest model)

- drop z component. $x = [I_{2\times2} | 0] p$, $\tilde{x} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tilde{p}$
  - good for long-focal lengths (telephoto lenses) & objects whose depth is shallow relative to distance to camera

Scaled orthography: $x = [sI_{2\times2} | 0] p$ → Scaling can vary from frame-to-frame when estimating structure from motion

- good for reconstructing 3D shape of objects far from camera
- Pose estimation
- Structure & motion estimated from factoring (SVD)
- Para-Perspective

Perspective - Points are projected by dividing them by z component:

inhomogeneous

$$\bar{x} = P_z(p) = \begin{bmatrix} x/z \\ y/z \\ 1 \end{bmatrix}$$

homogeneous

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

- can first project into normalized device coordinates
- can aid in mapping back to 3D w/ info from range sensors

Camera Intrinsics