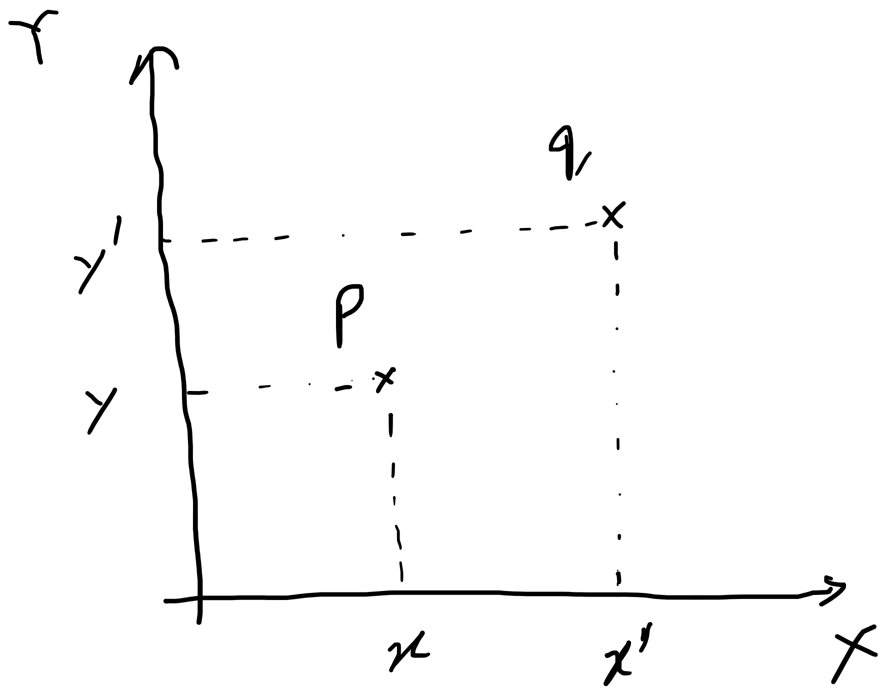
Transformations

2D

Translation



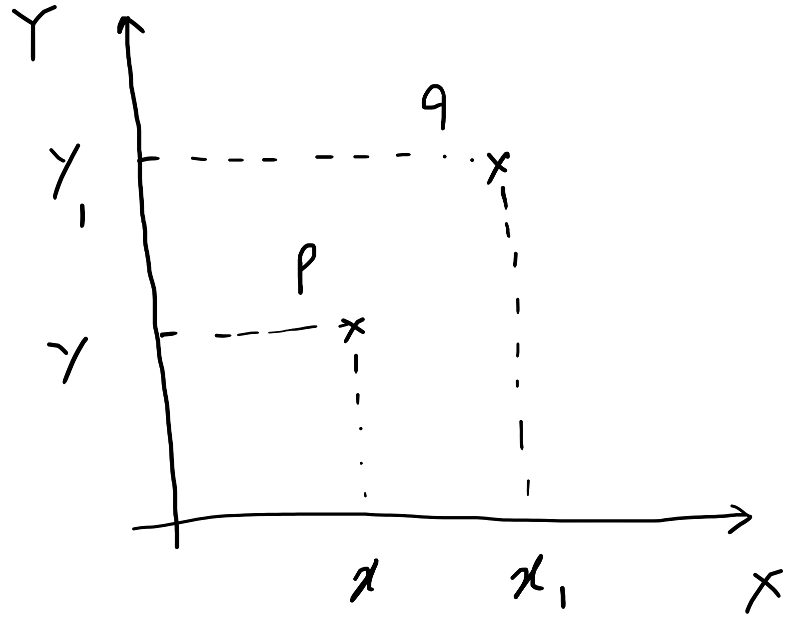
$$x' = x + t_x$$

$$y' = y + t_y$$

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

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

Scaling



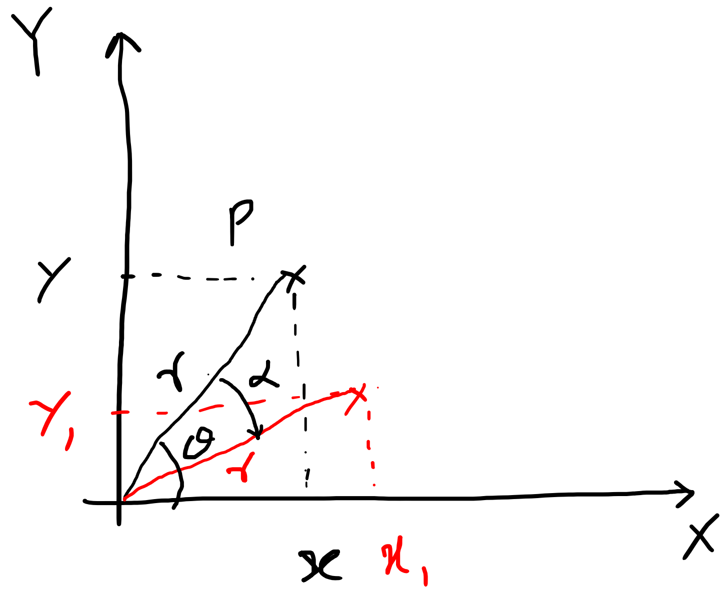
$$x_1 = C_x \, x$$

$$Y_1 = C_y \, Y$$

$$\begin{bmatrix} x_1 \\ Y_1 \end{bmatrix} = \begin{bmatrix} C_x & 0 \\ 0 & C_y \end{bmatrix} \begin{bmatrix} x \\ Y \end{bmatrix}$$

$$\begin{bmatrix} x_1 \\ Y_1 \\ 1 \end{bmatrix} = \begin{bmatrix} C_x & 0 & 0 \\ 0 & C_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ Y \\ 1 \end{bmatrix}$$

Rotation



$$x = r\cos\theta$$

$$y = r\sin\theta$$

$$x_1 = r\cos(\theta-\alpha)$$

$$y_1 = r\sin(\theta-\alpha)$$

$$x_1 = r\cos\theta\cos\alpha + r\sin\theta\sin\alpha$$

$$\left.\begin{aligned} x_1 &= x\cos\alpha + y\sin\alpha \\ y_1 &= y\cos\alpha - x\sin\alpha \end{aligned}\right\}$$

$$\begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

R + T (Euclidean)

$$x' = Rx + t$$

S + R (Similarity)

$$x' = SRx + t$$

Affine

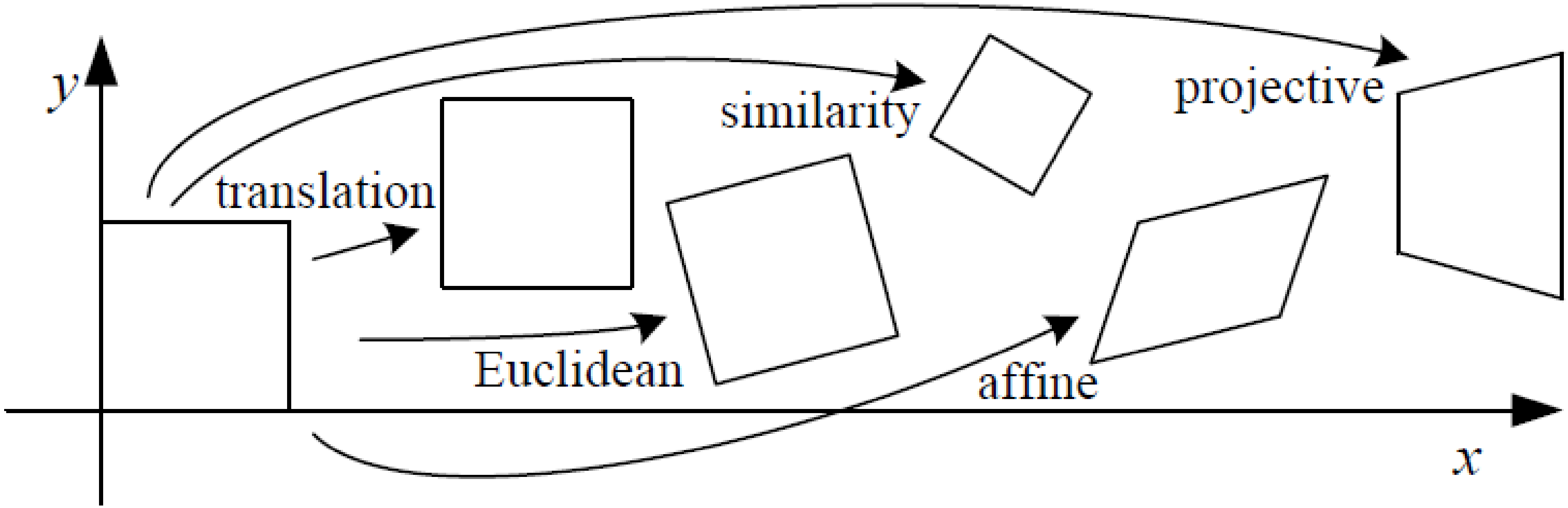
$$x' = Ax$$

Projective (Homography)

$$\tilde{x}' = \tilde{H}\tilde{x}$$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & h_{22} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

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


y
translation
Euclidean
similarity
affine
projective
x

Concatenation of several transforms

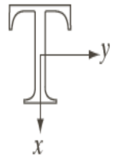
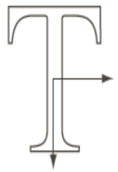
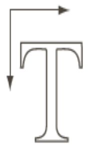
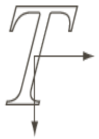
$$P(x, y) \longrightarrow q(x_1, y_1)$$

$$q = R_\theta(S(T(P)))$$

$$T \rightarrow S \rightarrow R$$

Affine transformations

TABLE 2.2
Affine transformations based on Eq. (2.6.–23).

Transformation Name	Affine Matrix, T	Coordinate Equations	Example
Identity	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x = v$ $y = w$	
Scaling	$\begin{bmatrix} c_x & 0 & 0 \\ 0 & c_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x = c_x v$ $y = c_y w$	
Rotation	$\begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x = v\cos\theta - w\sin\theta$ $y = v\cos\theta + w\sin\theta$	
Translation	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ t_x & t_y & 1 \end{bmatrix}$	$x = v + t_x$ $y = w + t_y$	
Shear (vertical)	$\begin{bmatrix} 1 & 0 & 0 \\ s_v & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x = v + s_v w$ $y = w$	
Shear (horizontal)	$\begin{bmatrix} 1 & s_h & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x = v$ $y = s_h v + w$	

Food for thought!

1. What is an affine transformation in image processing?
2. What is image scaling? How does scaling affect image size?
3. What is image translation? Write the basic transformation equation.
4. What happens to an image when a rotation transformation is applied?
5. What is shearing in image transformation? Mention one effect of shearing on an image.

Programming assignment

- **Implement and analyze basic affine transformations on a digital image to understand how geometric transformations affect pixel coordinates and image appearance.**
- **Concepts Used:**
  - Affine transformation
  - Image scaling
  - Image translation
  - Image rotation
  - Image shearing
- **Tasks:**
  - Read a grayscale image.
  - Implement image scaling using given scaling factors.
  - Implement image translation using given translation parameters.
  - Implement image rotation about the image center by a specified angle.
  - Implement image shearing along the x- or y-direction.
  - Display the original image and all transformed images for comparison.

AI supported self-learning on Image Interpolation (Prompts compatible with ChatGPT)

Active Learners (Learning by Doing)

1. Give me a small image matrix and ask me to apply scaling, translation, rotation, or shearing step by step. Let me try first, then explain the solution.
2. Create a coordinate-based example where I compute the new pixel location after an affine transformation and then explain the correct method.

Reflective Learners (Learning by Thinking)

1. Explain affine transformations in image processing step by step and summarize how scaling, translation, rotation, and shearing differ.
2. Explain why affine transformations are represented using matrices and how homogeneous coordinates are used..

Sensing Learners (Concrete & Practical)

1. Explain image scaling, translation, rotation, and shearing using numerical examples and actual pixel coordinates.
2. Show practical examples where affine transformations change image size, orientation, or shape.

Intuitive Learners (Concepts & Patterns)

1. Explain the conceptual idea behind affine transformations and how they preserve straight lines and parallelism.
2. Explain how different affine transformations can be combined and interpreted geometrically.

Visual Learners (Diagrams & Structure)

1. Explain affine transformations using diagrams or coordinate grids showing original and transformed images.
2. Visually compare scaling, translation, rotation, and shearing on the same image or shape.

Verbal Learners (Words & Explanation)

1. Explain affine transformations in simple language using everyday examples like zooming, shifting, or tilting an image.
2. Explain the difference between scaling, translation, rotation, and shearing as if teaching it to a beginner.

Sequential Learners (Step-by-Step Logic)

1. Explain the steps involved in applying an affine transformation using matrix multiplication.
2. Explain step by step how rotation is performed about the origin and about an arbitrary point.

Global Learners (Big Picture First)

1. First explain the overall role of affine transformations in digital image processing, then explain each type.
2. Explain where affine transformations are used in real-world applications before explaining the mathematical details.