Computer Vision 2024

群号: 791184228

扫一扫二维码，加入群聊

QQ

2D Metric Transformation

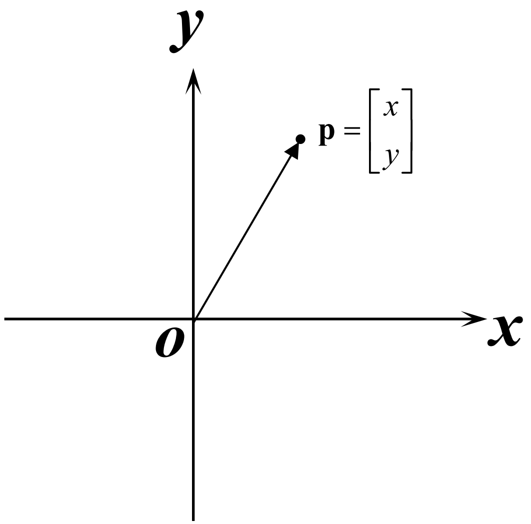
Dr. Yan Li

2D Metric Transformation (2D 度量变换)

- Translation （平移变换）
- Rotation （旋转变换）
- Scaling （比例变换）
- Combining （复合） transformations

Euclidean Plane

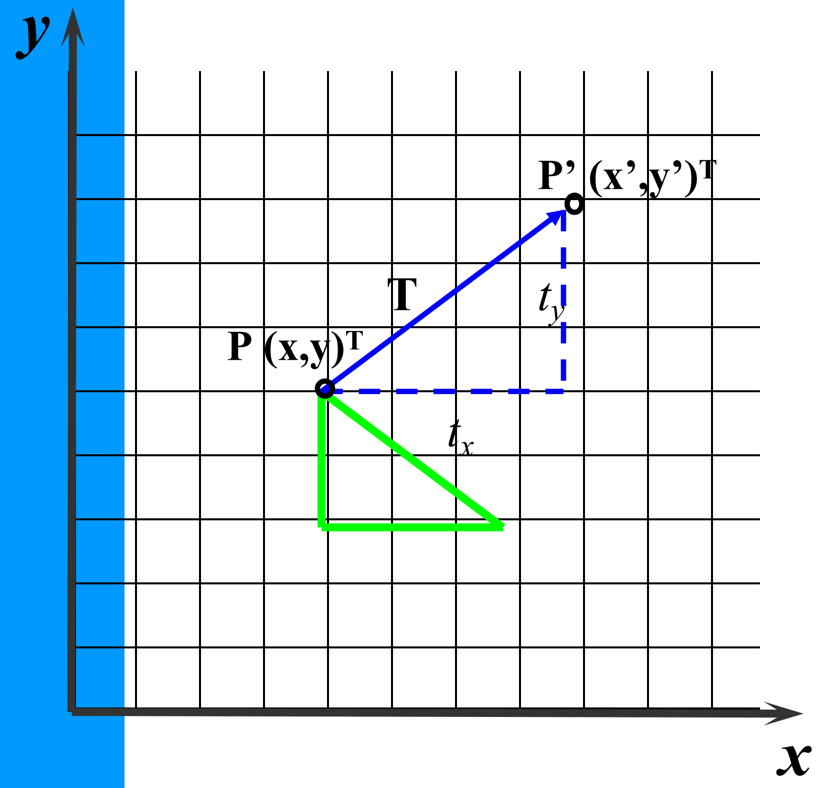
- Euclidean plane: two-dimensional real vector space equipped with an inner product



- A point in 2D space is represented by a column two vector

$$\mathbf{p} = \begin{bmatrix} x \\ y \end{bmatrix}$$

Object Translation



- A translation moves all points in an object along the same straight-line path to new positions.
- The path is represented by a vector $\mathbf{T} = \begin{bmatrix} t_x \\ t_y \end{bmatrix}$, called the translation or shift vector.

Object Translation

We can write the components:

$$x' = x + t_x$$

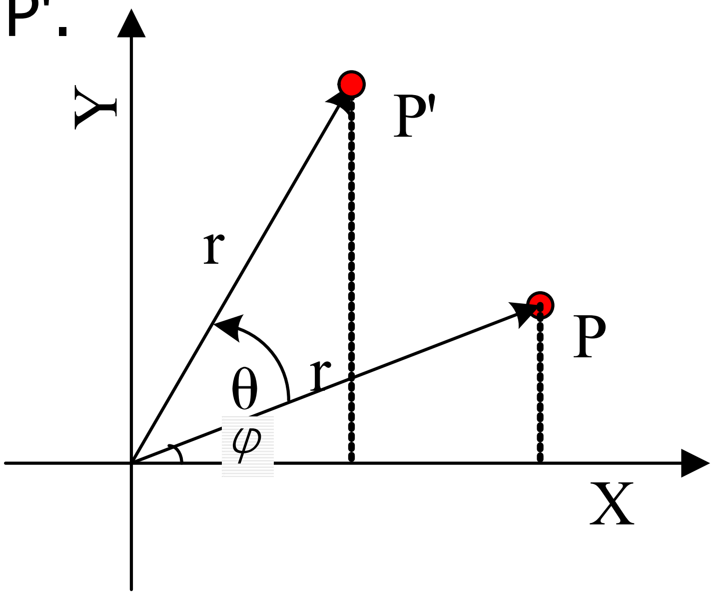
$$y' = y + t_y$$

or in matrix form:

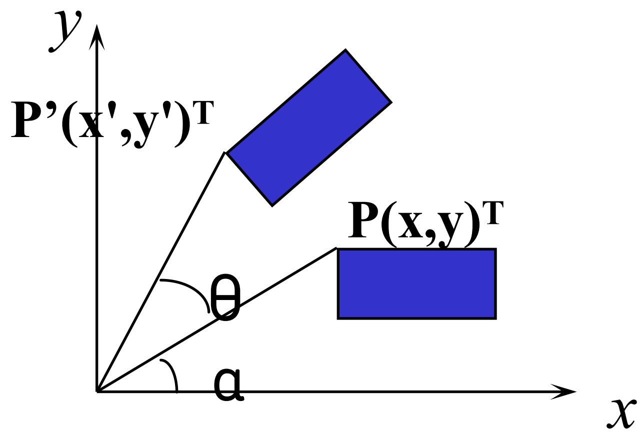
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix} \quad \textbf{or} \quad \mathbf{P}' = \mathbf{P} + \mathbf{T}$$

Object Rotation

- Two-dimensional rotation transformation refers to the repositioning process of rotating point P around the **coordinate origin** by a certain angle (positive counterclockwise, negative clockwise) to obtain a new point P'.



Originally, point **P(x,y)**:



$$\begin{cases} x = r\cos(\alpha) \\ y = r\sin(\alpha) \end{cases}$$

After rotation, point **P->P'(x', y')**:

$$\begin{cases} x' = r\cos(\alpha+\theta) = \underline{r\cos\alpha}\cos\theta - \underline{r\sin\alpha}\sin\theta \\ y' = r\sin(\alpha+\theta) = \underline{r\sin\alpha}\cos\theta + \underline{r\cos\alpha}\sin\theta \end{cases}$$

or in matrix form:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$ or $$\mathbf{P}' = \mathbf{R}_\theta \cdot \mathbf{P}$$

More rotation

- Rotate about the origin

- AntiClockwise rotation, θ is positive
  Clockwise rotation, θ is negative

- **R**, the rotation matrix, looks like: $R = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$

  R is an unity orthogonal matrix

Q: Properties of unity orthogonal matrix?

Scaling (about the axes): $S(S_x, S_y)$

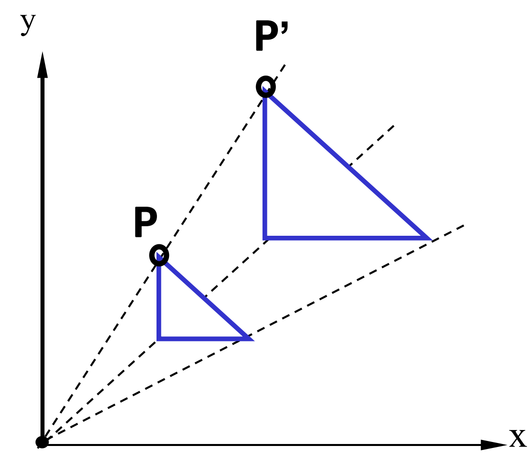
- Scaling alters the size of an object.

s_x: the amount of scaling in the direction parallel to the x-axis

s_y : the amount of scaling in the direction parallel to the y-axis

Note: scaling is a transformation relative to the origin.

Scale transformation also changes the relative position between the object and coordinate origin.



Scaling about the axes

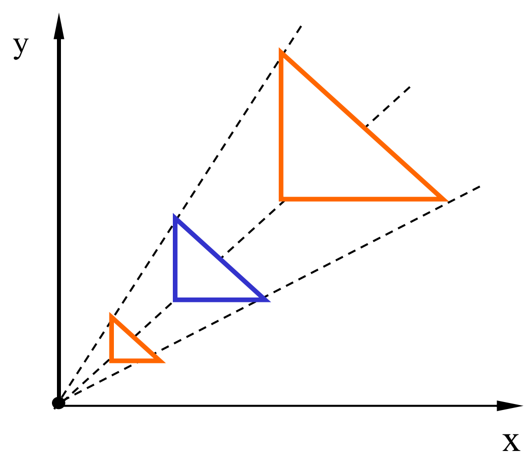
- Properties of scaling

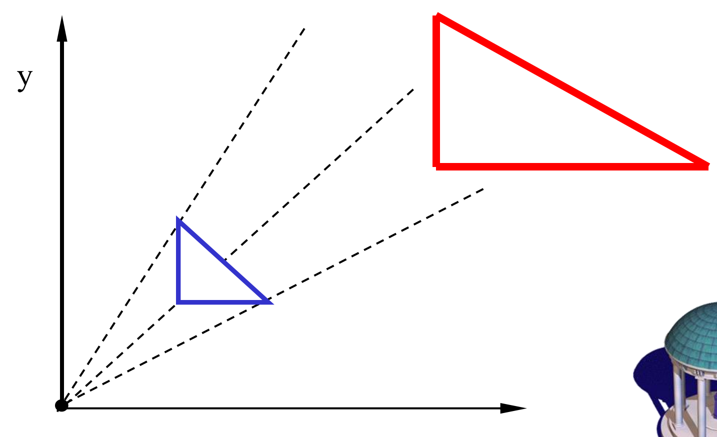
$s_x = s_y$, the figure before transformation is similar to the figure after transformation

$s_x = s_y > 1$, the graph will zoom in and move away from the coordinate origin

$s_x = s_y < 1$, the graph will zoom out and get closer to the origin of the coordinates

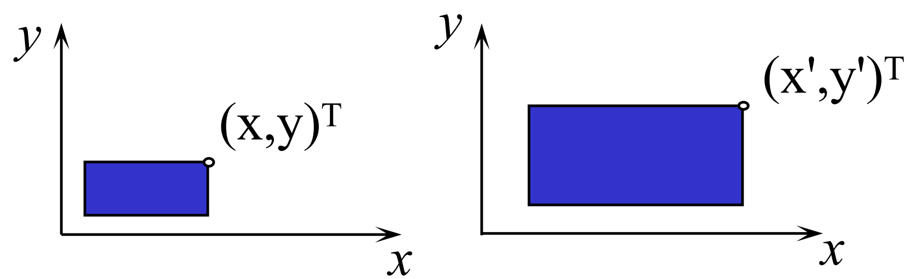
$s_x \neq s_y$





$s_x > s_y > 1$

Scaling: $S(S_x, S_y)$



- We can write the components:

$$x' = s_x \bullet x$$
$$y' = s_y \bullet y$$

or in matrix form:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

or **P' = S • P**

Summary

- Translate transform

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix} \quad \text{or} \quad \mathrm{P'} = \mathrm{P} + \mathrm{T}$$

- Scaling transform

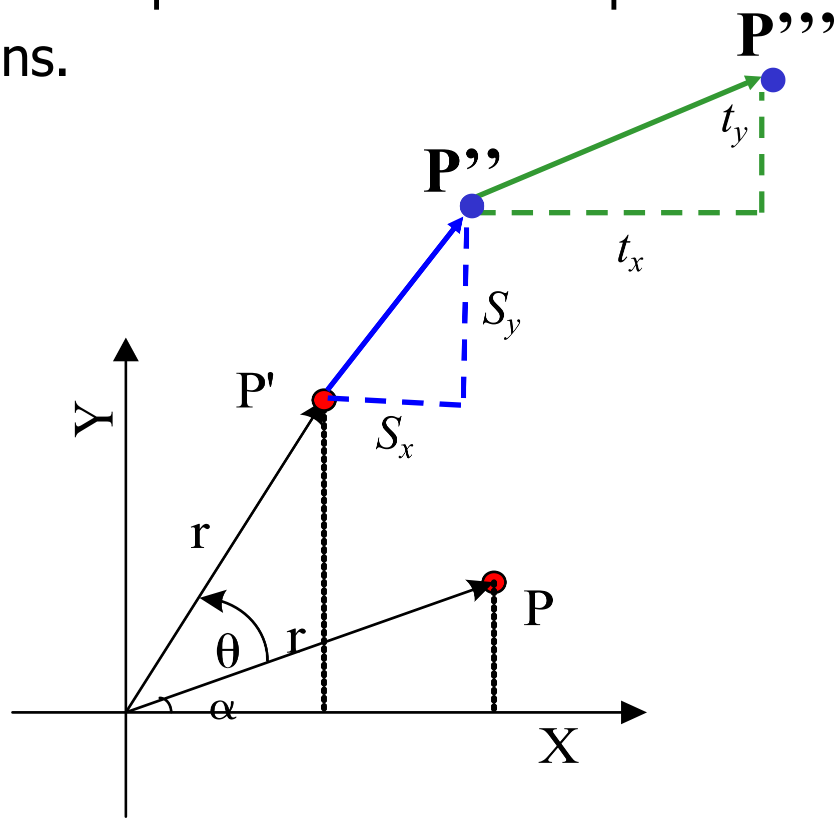
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \quad \text{or} \quad \mathrm{P'} = \mathrm{S} \cdot \mathrm{P}$$

- Rotation transform

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \quad \text{or} \quad \mathbf{P}' = \mathbf{R}_\theta \cdot \mathbf{P}$$

Ex. Combined transformation

Point $\mathbf{P}(x, y)$ first rotate around the origin counterclockwisely With angle θ , then along the x and y directions Scale Sx and Sy times respectively, and finally translate $(t_x\ t_y)^T$. Calculate the final position $\mathbf{P'''}$ of the point $\mathbf{P}$ after all these transformtions.



Combining transformations

We have a general expression of transformations of a point:

$$\mathbf{P'} = \mathbf{M} \bullet \mathbf{P} + \mathbf{A}$$

When we scale and rotate, we set **M**.

When we translate, we set **A**.

To combine multiple transformations, we must explicitly compute each transformed point.

It'd be nicer if the two-dimensional transformation can be uniformly expressed as the multiplication of matrices.

Homogeneous Coordinates

Homogeneous coordinate （齐次坐标） representation is to represent an n-dimensional vector with an n+1-dimensional vector.

Take a point (x, y) in 2D space as an example. It can be represented by point $(x, y, 1)$ in the new space:

$$\mathbf{x} = \begin{pmatrix} x \\ y \end{pmatrix} \longleftrightarrow \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \sim \begin{pmatrix} 2x \\ 2y \\ 2 \end{pmatrix} \sim \cdots \sim \begin{pmatrix} kx \\ ky \\ k \end{pmatrix}$$

Given homogeneous coordinate $(x_1, x_2, x_3)^{\mathrm{T}}$, We can always map back to the original 2D point by dividing by the last coordinate as $(x_1/x_3, x_2/x_3)^{\mathrm{T}}$.

Normalized Homogeneous Coordinates

The ***normalized homogeneous coordinate*** representation is the homogeneous coordinate representation of x_3=1.

$$(x, y) \Leftarrow (x, y, 1)$$

So what?

With homogeneous coordinate, we can express the translate as the multiplication of matrices.

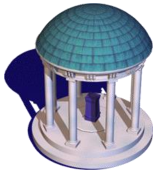
Our point now has three coordinates. So our matrix is needs to be 3x3.

$$x' = x + t_x$$
$$y' = y + t_y$$
$$1 = 1$$

We want a matrix which gives us:

$$x' = 1 \cdot x + 0 \cdot y + t_x \cdot 1$$
$$y' = 0 \cdot x + 1 \cdot y + t_y \cdot 1$$
$$1 = 0 \cdot x + 0 \cdot y + 1 \cdot 1$$

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



And?

Rotations:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Scales:

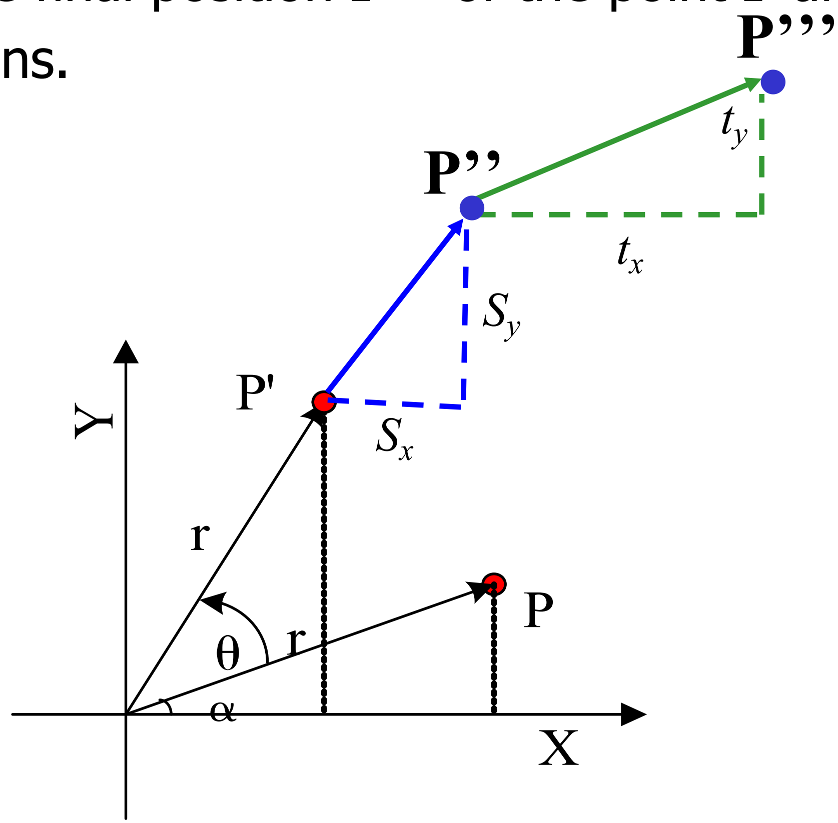
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Translate

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

Ex. Combined transformation

Point **P**(x, y) first rotate around the origin counterclockwisely With angle θ , then along the x and y directions Scale Sx and Sy times respectively, and finally translate $(t_x\ t_y)^T$. Calculate the final position **P'''** of the point **P** after all these transformtions.



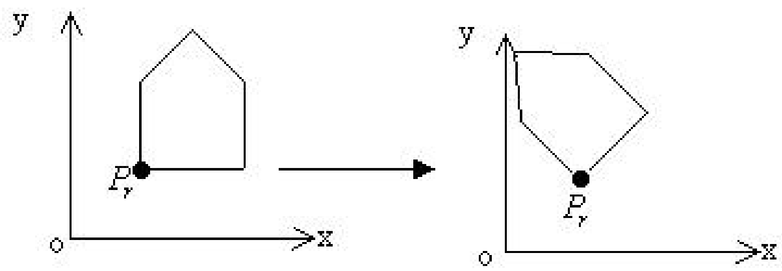
With homogeneous coordinates

- We can express **any of transformations** as a **single matrix**.
- No special cases when transforming a point – $\mathbf{P'} = \mathbf{H} \bullet \mathbf{P}$.
- Compound transformations – $\mathbf{H} = \mathbf{H_n} \bullet \ldots \bullet \mathbf{H_2} \bullet \mathbf{H_1}$.

Compound Transformation

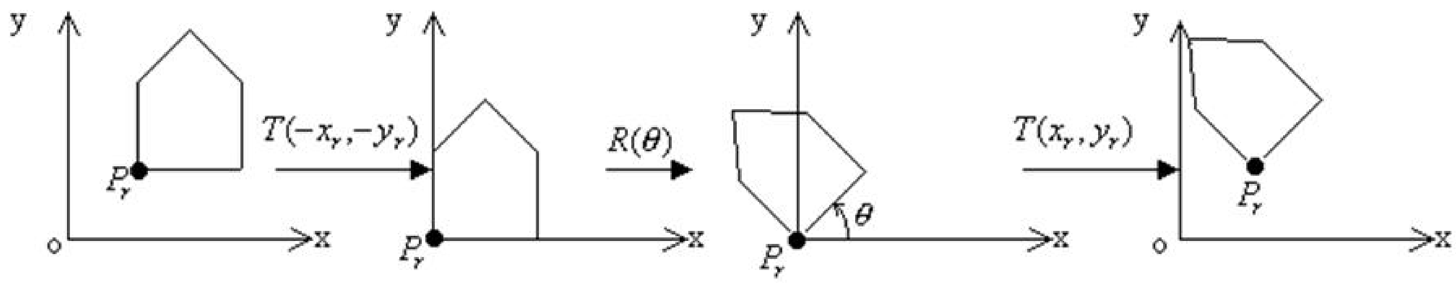
- Rotate about arbitrary point

Example 1: rotate about the point $\mathbf{P}_{\mathrm{r}}\,(x_{\mathrm{r}}, y_{\mathrm{r}})$



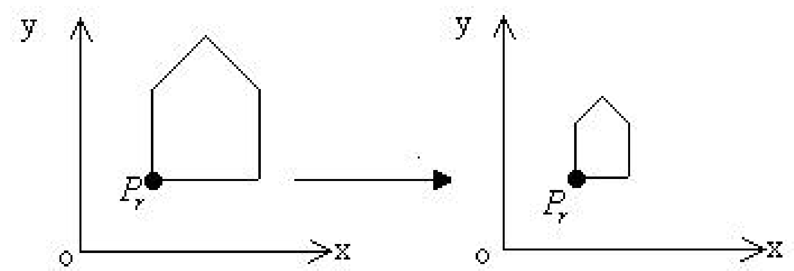
Compound Transformation

- Rotate about a pivot point



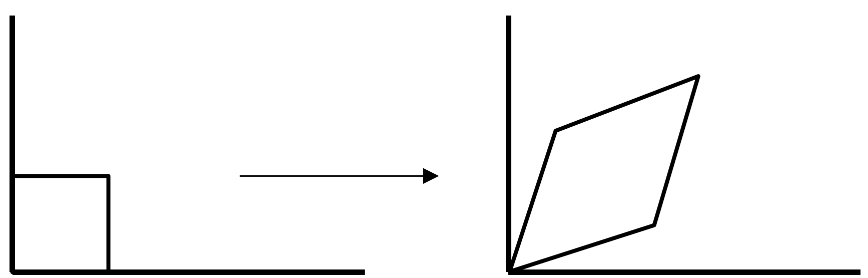
$$T_{RF} = \begin{bmatrix} 1 & 0 & x_r \\ 0 & 1 & y_r \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & -x_r \\ 0 & 1 & -y_r \\ 0 & 0 & 1 \end{bmatrix}$$

- Scale about arbitrary point



$$\mathbf{H} = \mathbf{H}_T(\text{fixed}) \bullet \mathbf{H}_S(\text{scale}) \bullet \mathbf{H}_T(-\text{fixed})$$

- General scaling directions



$$\mathbf{H}=\mathbf{H}_R(-\theta) \bullet \mathbf{H}_S(\text{scale}) \bullet \mathbf{H}_R(\theta)$$

Transformation about any fixed point （相对任意点的变换）

Transformation about a fixed point (x_r, y_r), the transformation process is：

(1) translate；

(2) 2D geometric transformation with respect to the origin；

(3) translate back

Transformation along any direction （相对任意轴的变换）

Transformation along any direction, the transformation process is：

(1) rotation；

(2) 2D geometric transformation with respect to the x or y direction

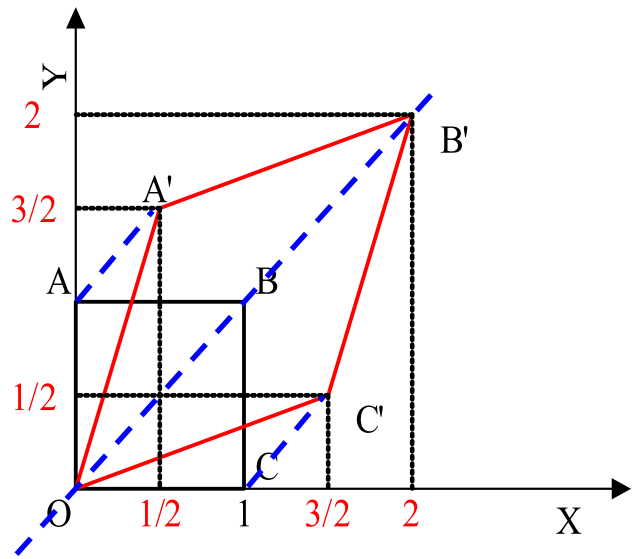
(3) rotate back

Q1: Transformation matrix for symmetrical transformation relative to straight line $y=4x-1$.

（写出相对于直线$y=4x-1$做对称变换的变换矩阵）

Q2. Stretch each point of the square ABCO along the (0,0)→(1,1) direction shown in the figure below, and the result is as shown in the figure. Write its transformation matrix and transformation process.

(将正方形ABCO各点沿下图所示的(0,0)→(1,1)方向进行拉伸，结果为如图所示的，写出其变换矩阵和变换过程。)

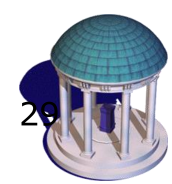


Summary (复合变换小结)

- The graph is subjected to more than one geometric transformation, and the overall transformation is the product of each transformation matrix. (图形作一次以上的几何变换，变换结果是每次变换矩阵的乘积)
- A compound transform has the form (复合变换具有形式):

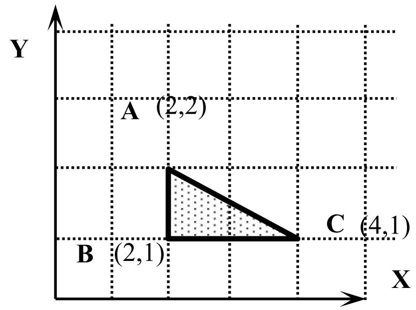
$$\mathbf{P'} = \mathbf{H} \cdot \mathbf{P} = (\mathbf{H_n} \cdot \mathbf{H_{n-1}} \cdot \mathbf{H_{n-2}} \cdots \cdot \mathbf{H_1}) \cdot \mathbf{P} \quad (n > 1)$$

- Any complex geometric transformation can be regarded as a combination of basic geometric transformations.(任何一复杂的几何变换都可以看作基本几何变换的组合形式)



Example

Given the triangle ABC shown in the figure below, calculate the coordinates of the three vertices after performing following transformations:

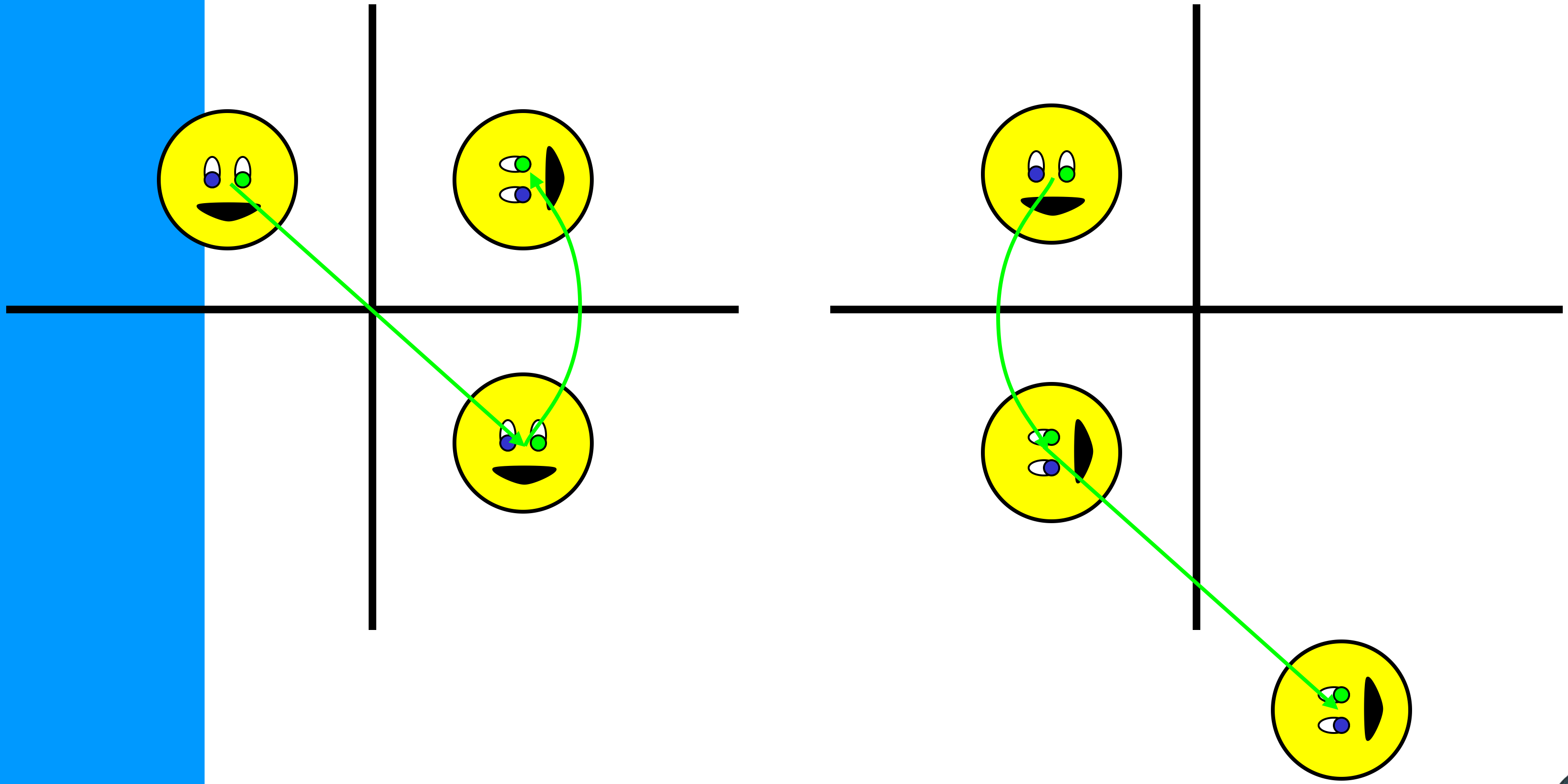


1. Rotate ABC around the original point anticlockwisely by 90 ° and then translate by (2,1).
2. Translate ABC by (2,1) and then rotate around the original point anticlockwisely by 90 ° .

Does the order of operations matter?

Order of operations

So, it does matter. Let’s look at an example:



Continuous translation transformation

First translate $\mathbf{T}_1 = (t_{x1}, t_{y1})^T$, then translate $\mathbf{T}_2 = (t_{x2}, t_{y2})^T$, the transformation matrix is:

$$\mathbf{H}_T = \mathbf{H}_{T2} \cdot \mathbf{H}_{T1} = \begin{bmatrix} 1 & 0 & t_{x2} \\ 0 & 1 & t_{y2} \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & t_{x1} \\ 0 & 1 & t_{y1} \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & t_{x1} + t_{x2} \\ 0 & 1 & t_{y1} + t_{y2} \\ 0 & 0 & 1 \end{bmatrix}$$

Two consecutive translations are additive and the order can be interchanged.

Continuous scale transformation

$$\mathbf{H}_s = \mathbf{H}_{s2} \cdot \mathbf{H}_{s1} = \begin{bmatrix} S_{x2} & 0 & 0 \\ 0 & S_{y2} & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} S_{x1} & 0 & 0 \\ 0 & S_{y1} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} S_{x1} \cdot S_{x2} & 0 & 0 \\ 0 & S_{y1} \cdot S_{y2} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Two consecutive scalings are multiplied and the order can be interchanged.

Continuous rotation transformation

$$\mathbf{R} = \mathbf{R}_{\theta 2} \cdot \mathbf{R}_{\theta 1} = \begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & 0 \\ \sin\theta_2 & \cos\theta_2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & 0 \\ \sin\theta_1 & \cos\theta_1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \cos(\theta_1+\theta_2) & -\sin(\theta_1+\theta_2) & 0 \\ \sin(\theta_1+\theta_2) & \cos(\theta_1+\theta_2) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{R} = \mathbf{R}_{\theta 2} \cdot \mathbf{R}_{\theta 1} = \mathbf{R}_{(\theta_1+\theta_2)}$$

Two consecutive rotations are multiplied and the order can be interchanged.

Inverse transformation(逆变换)

- translation inverse transform

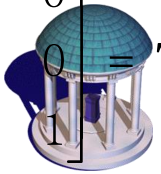
$$\mathrm{T}_T = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \qquad \mathrm{T}_T^{-1} = \begin{bmatrix} 1 & 0 & -t_x \\ 0 & 1 & -t_y \\ 0 & 0 & 1 \end{bmatrix}$$

- scaling inverse transform

$$\mathbf{T}_S = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{T}_S^{-1} = \begin{bmatrix} \frac{1}{s_x} & 0 & 0 \\ 0 & \frac{1}{s_y} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- rotation inverse transform

$$\mathrm{T}_R = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathrm{T}_R^{-1} = \begin{bmatrix} \cos(-\theta) & -\sin(-\theta) & 0 \\ \sin(-\theta) & \cos(-\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} = \mathrm{T}_R^T$$



2D coordinate system transformation

(2D坐标系的变换)

- Coordinate system translation (坐标系平移)

$$\mathbf{P}_2 = \mathbf{T}^{-1}(t_x, t_y)\mathbf{P}_1 \quad \mathbf{P}_1 = \mathbf{T}(t_x, t_y)\mathbf{P}_2$$

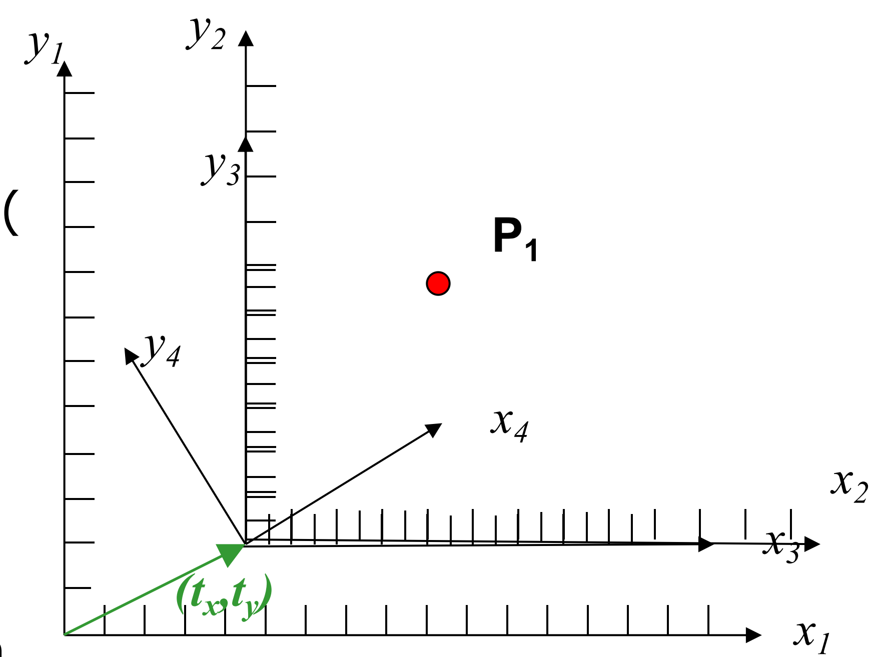
- Coordinate system scaling (坐标系缩放)

$$\mathbf{P}_3 = \mathbf{S}^{-1}(s_x, s_y)\mathbf{P}_2 \qquad \mathbf{P}_2 = \mathbf{S}(s_x, s_y)\mathbf{P}_3$$

- Coordinate system rotation (坐标系旋转)

$$\mathbf{P}_4 = \mathbf{R}^{-1}(\theta)\mathbf{P}_3 \qquad \mathbf{P}_3 = \mathbf{R}(\theta)\mathbf{P}_4$$

- Compound transformation (组合变换)



$$\mathbf{P}_4 = \mathbf{R}^{-1}\mathbf{S}^{-1}\mathbf{T}^{-1}\mathbf{P}_1 \qquad \mathbf{P}_1 = (\mathbf{TSR})\ \mathbf{P}_4$$

- Point P is transformed to point P' after transformation $T_1, T_2, \ldots, T_n$, and the transformation equation is:

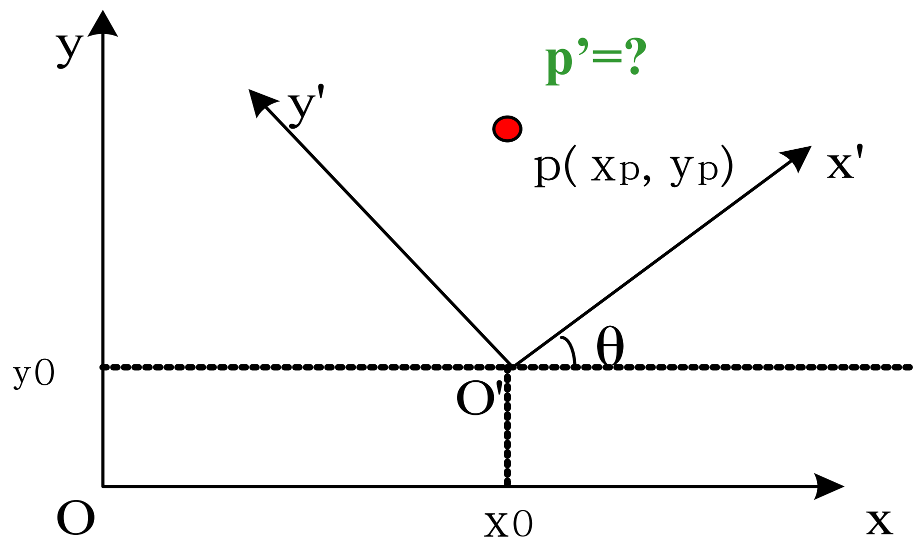
  (点P经过变换$T_1, T_2, \ldots, T_n$变换到点P’,变换方程为)

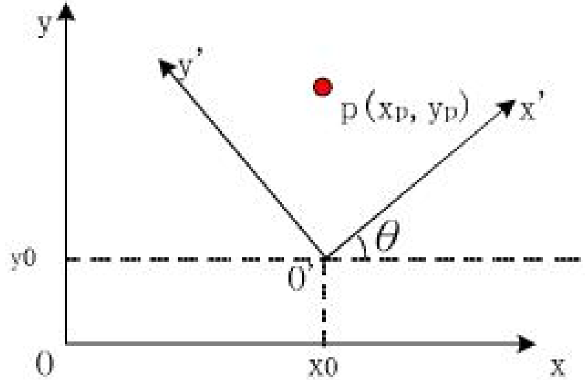
- The old coordinate system is transformed into the new one through the transformation of T1, T2, ..., Tn, and the coordinates of point P are transformed from P_{old} to P_{new} (where P_{old} is the coordinate of the point in the original coordinate system, and P_{new} is the point in the transformed coordinate system. coordinates), the transformation equation is:

  (旧坐标系经过$T_1, T_2, \ldots, T_n$的变换变换到新坐标系，点P的坐标由P_{old}变换到P_{new}（其中P_{old}为点在原始坐标系中的坐标， P_{new}为点在变换后坐标系中的坐标），变换方程为)

- The transformation of the point and the transformation of the coordinate system are inverse transformations of each other. (点的变换与坐标系的变换互为逆变换)

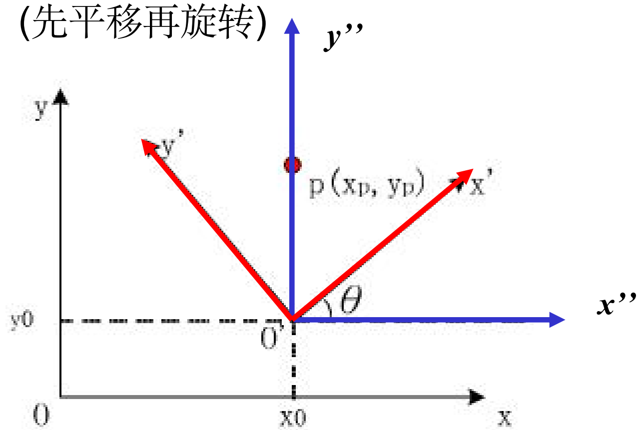
Practice:





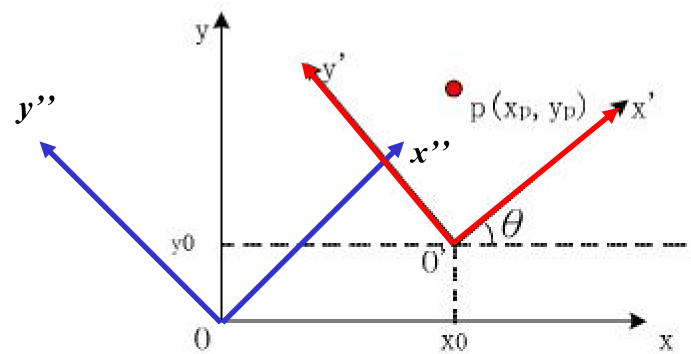
First translate then rotate

(先平移再旋转)



First rotate then translate

(先旋转再平移)



Note: Transforms in which coordinate system?

(注意：各变换是在那个坐标系中的进行的？)