Lecture 7-1 Geometry Operations and interpolation (chapter 2.6.5)

Yuyao Zhang PhD
zhangyy8@shanghaitech.edu.cn
SIST Building 2 302-F

Outline

- Spatial Operations
  - Affine transform （仿射变换）
  - Projective transform
- Image interpolation
  - Nearest-neighbor interpolation
  - Linear & bi-linear interpolation

Geometric operations

- Geometric

$$J(x, y) = I\left(T(x, y)\right);$$

- Point operation

$$J(x, y) = T\left(I(x, y)\right);$$

Shift translation (Affine)

$J(x,y)$ $I(x,y)$

- Translate downwards by 2 pixels.

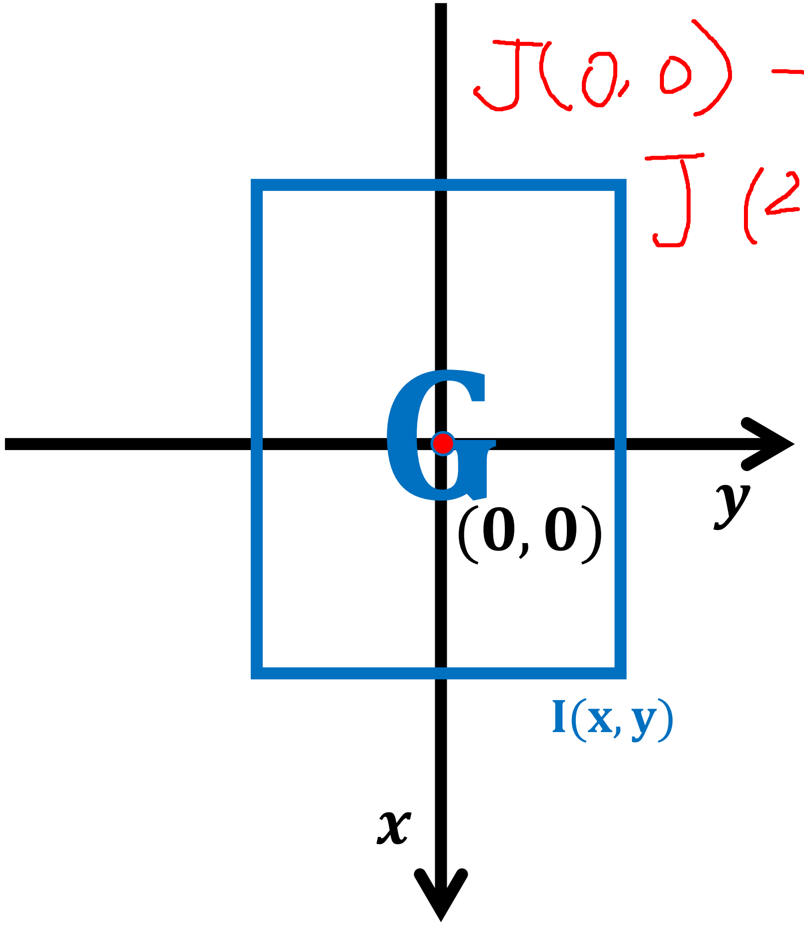
$$\mathbf{J(x,y) = I(x-2,y)}$$

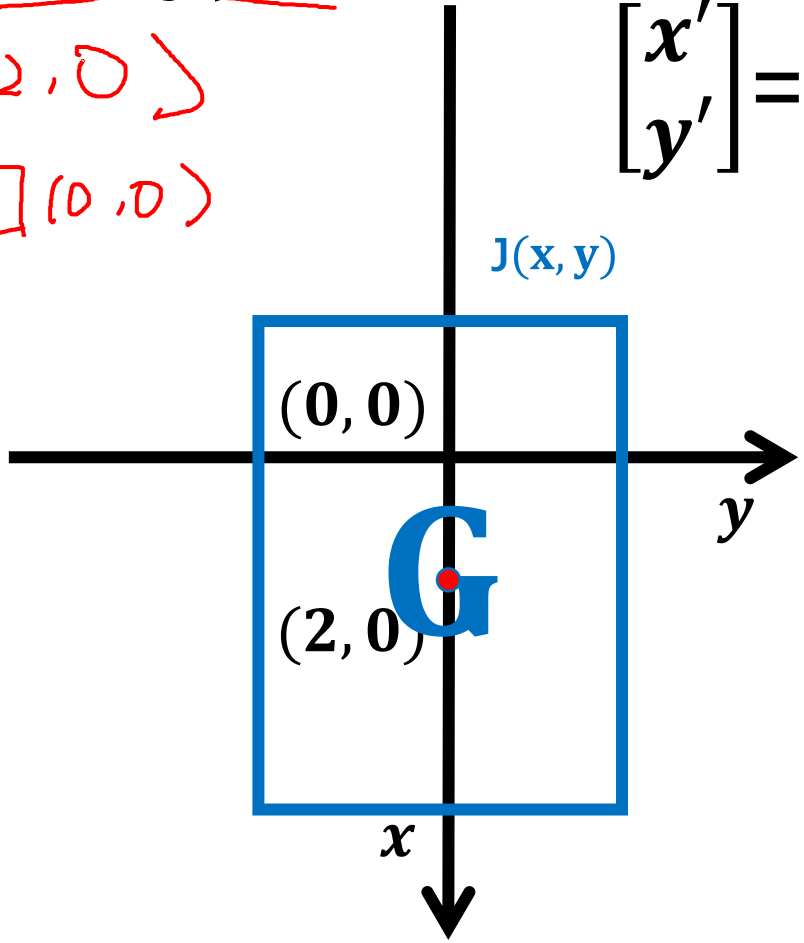
$J(0,0) \rightarrow I(-2,0)$

$J(2,0) \rightarrow I(0,0)$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x-2 \\ y \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} c_1 \\ c_2 \end{bmatrix}$$





Scaling translation

- Translate left by 2 pixels.

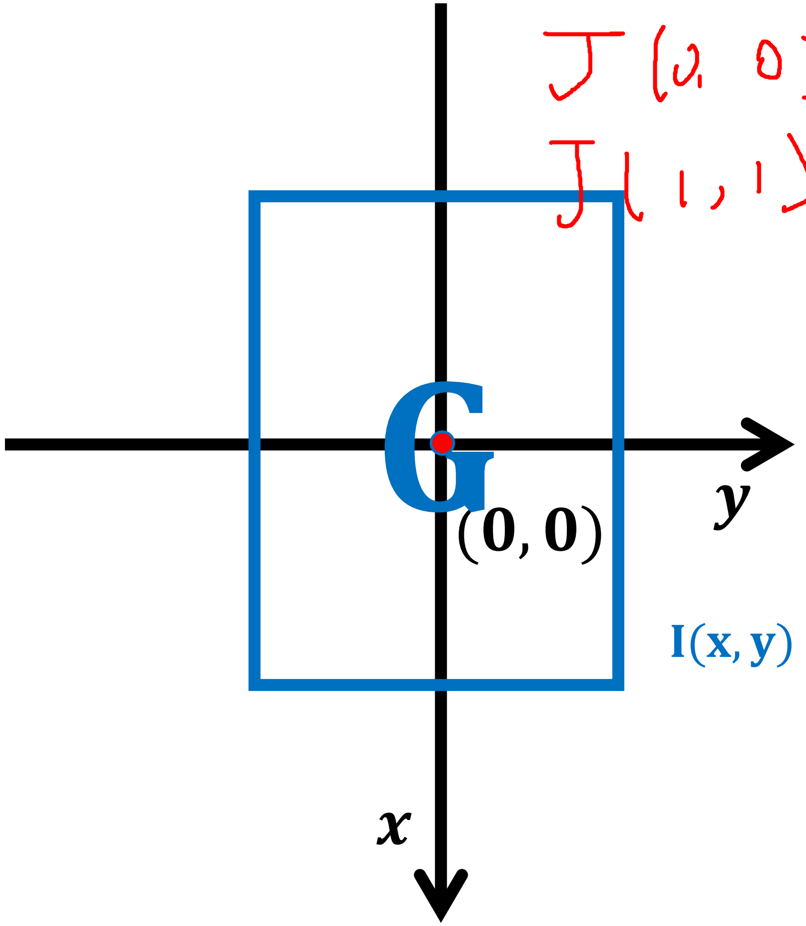
$$\mathrm{J}(\mathbf{x},\mathbf{y}) = \mathbf{I}(2*\mathbf{x},2*\mathbf{y})$$

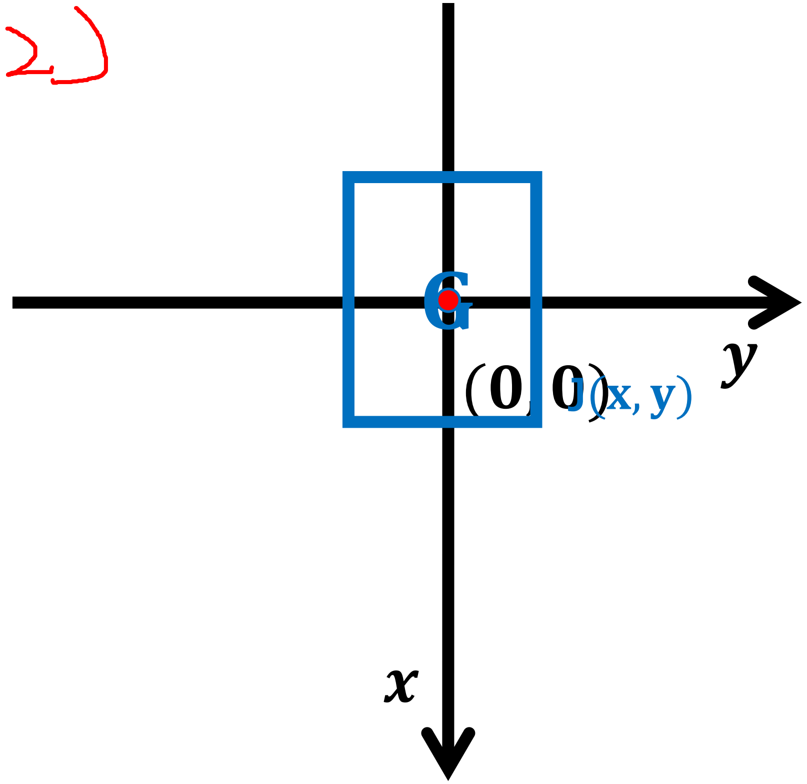
J(0, 0) = I(0, 0)

J(1, 1) = I(2, 2)

$$\mathrm{J}\begin{bmatrix}x'\\y'\end{bmatrix}=\overset{\mathrm{I}}{\begin{bmatrix}2*x\\2*y\end{bmatrix}}=\begin{bmatrix}2&0\\0&2\end{bmatrix}\begin{bmatrix}x\\y\end{bmatrix}+\begin{bmatrix}0\\0\end{bmatrix}$$

$$\begin{bmatrix}x'\\y'\end{bmatrix}=\begin{bmatrix}a&0\\0&b\end{bmatrix}\boxed{\begin{bmatrix}x\\y\end{bmatrix}}+\underline{\begin{bmatrix}0\\0\end{bmatrix}}\ 2$$

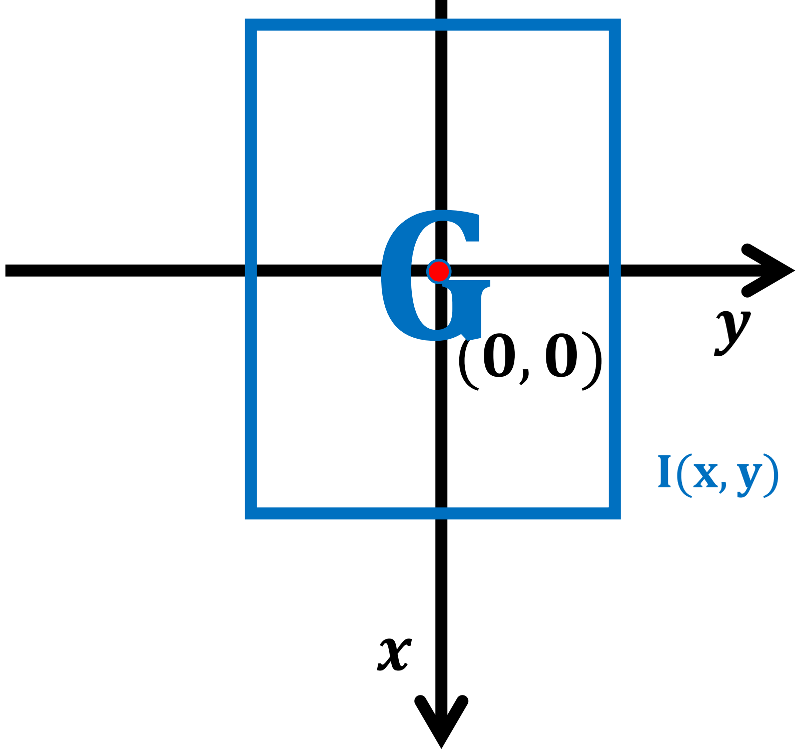


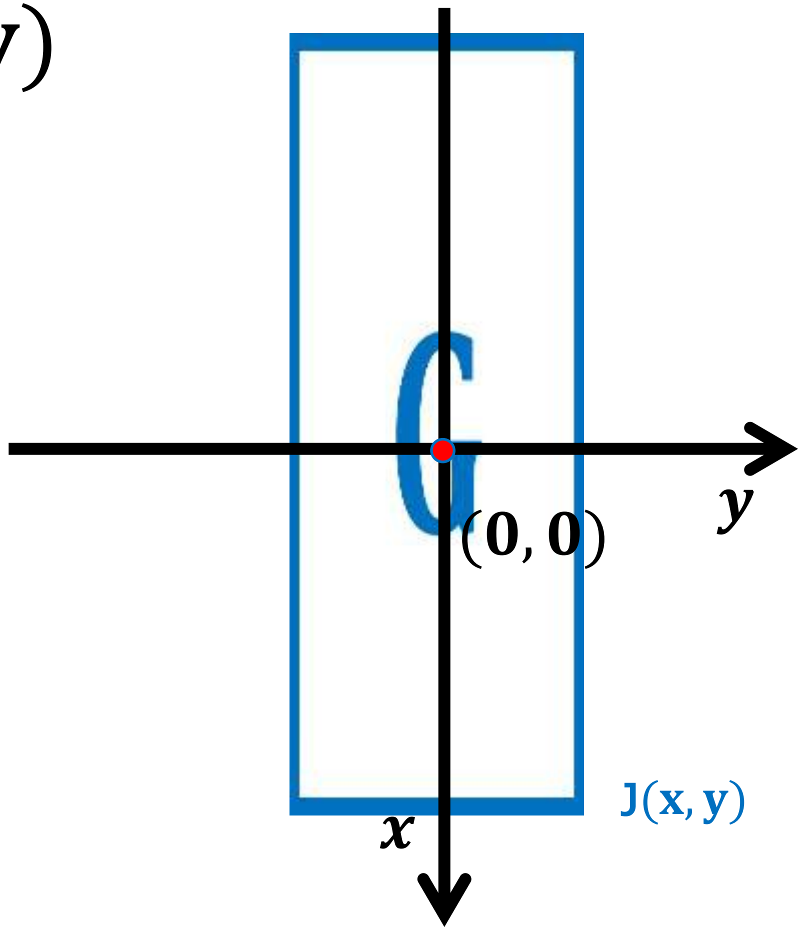


Scaling translation

- Translate left by 2 pixels.

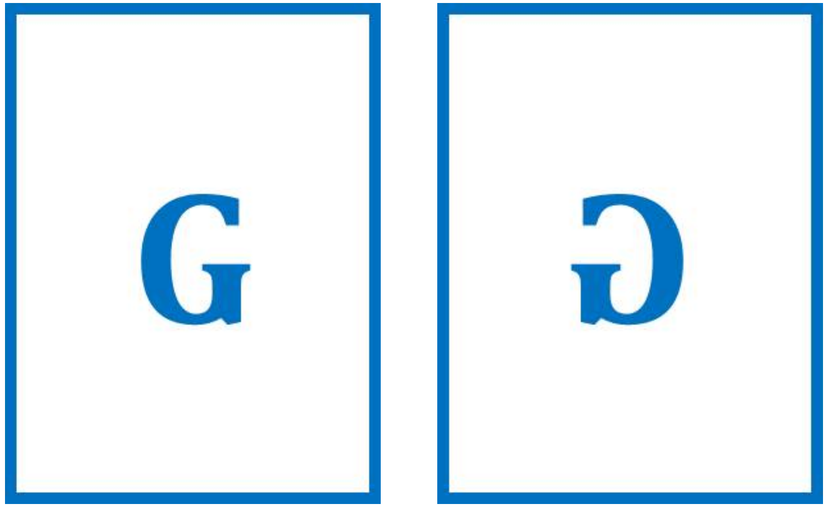
$$J(x, y) = I(2 * x, \frac{1}{2} * y)$$



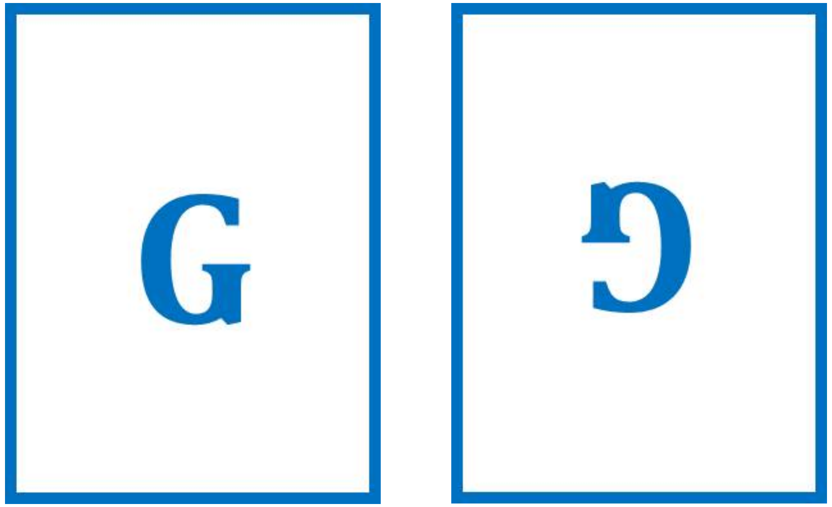


Flip translation

- $\mathbf{J}(\mathbf{x}, \mathbf{y}) = \mathbf{I}(\mathbf{x}, -\mathbf{y})$
- $\mathbf{J}(\mathbf{x}, \mathbf{y}) = \mathbf{I}(-\mathbf{x}, -\mathbf{y})$



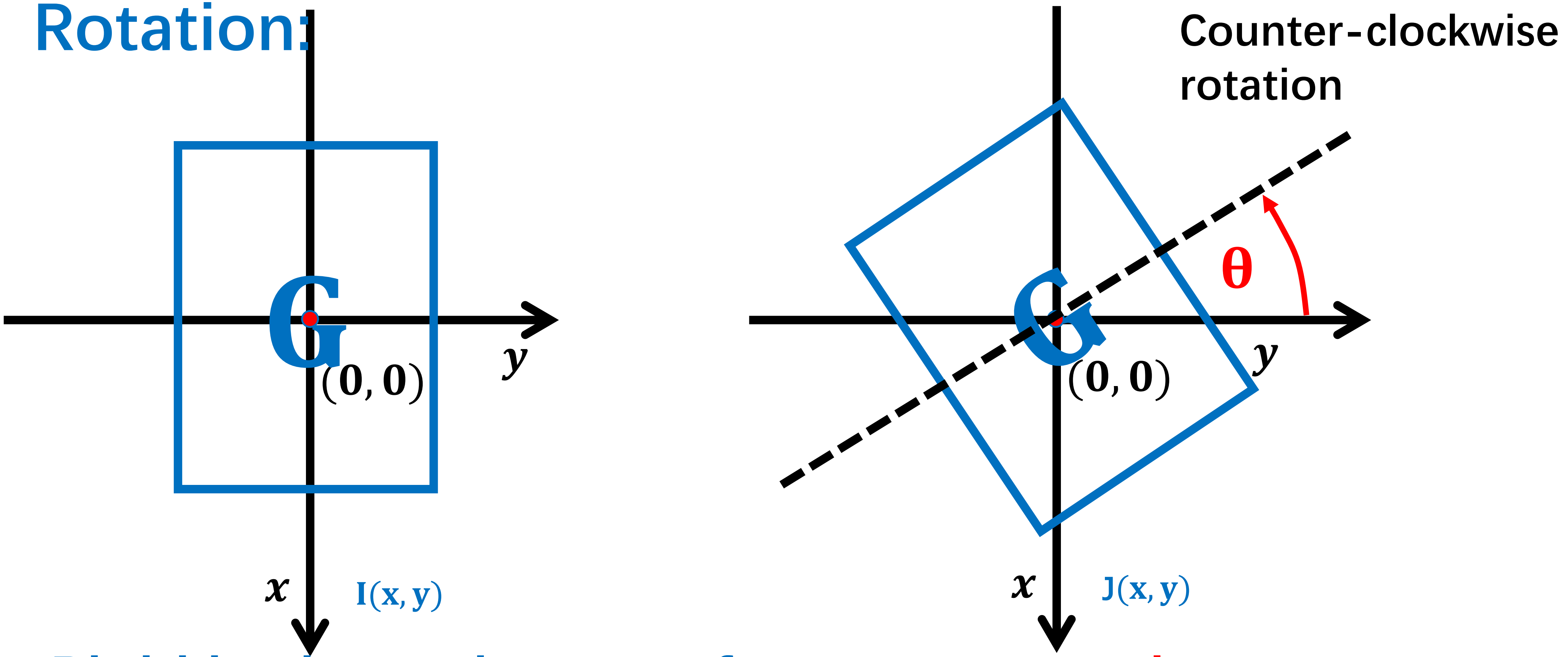
$\mathbf{I}(\mathbf{x}, \mathbf{y})$ $\mathbf{J}(\mathbf{x}, \mathbf{y})$



$\mathbf{I}(\mathbf{x}, \mathbf{y})$ $\mathbf{J}(\mathbf{x}, \mathbf{y})$

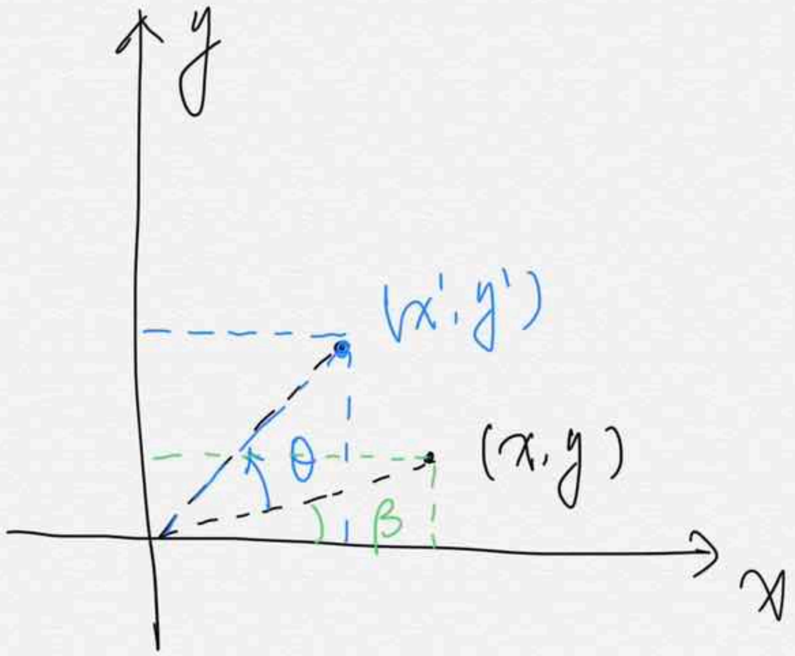
Rotation Transform

- Rotation:



Rigid body motion transform: preserve shapes and angles.

Rotation Transform



$$\begin{cases} x = \cos\beta\,|r| \\ y = \sin\beta\,|r| \end{cases}$$

$$\begin{cases} x' = \cos(\theta+\beta)\,|r| \\ \quad = (\cos\theta\cos\beta - \sin\theta\sin\beta)\,|r| \\ y' = \sin(\theta+\beta)\,|r| \\ \quad = (\sin\theta\cos\beta + \cos\theta\sin\beta)\,|r| \end{cases}$$

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

2D Linear Transform

- It is common for scale + rotate + shift to be considered into a 2D linear transformation.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} * \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} shift_x \\ shift_y \end{bmatrix}$$

$J(x, y)$ $\qquad\qquad$ $I(x, y)$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 2x \\ \frac{1}{2}y \end{bmatrix} = \begin{bmatrix} a & 0 \\ 0 & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

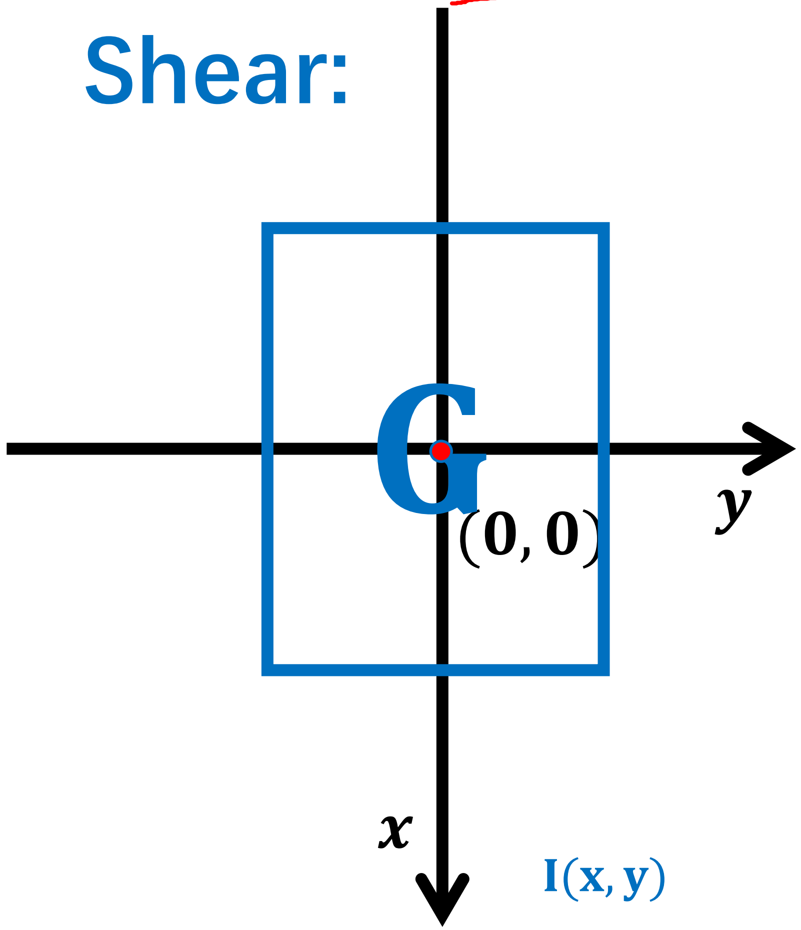
$$\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

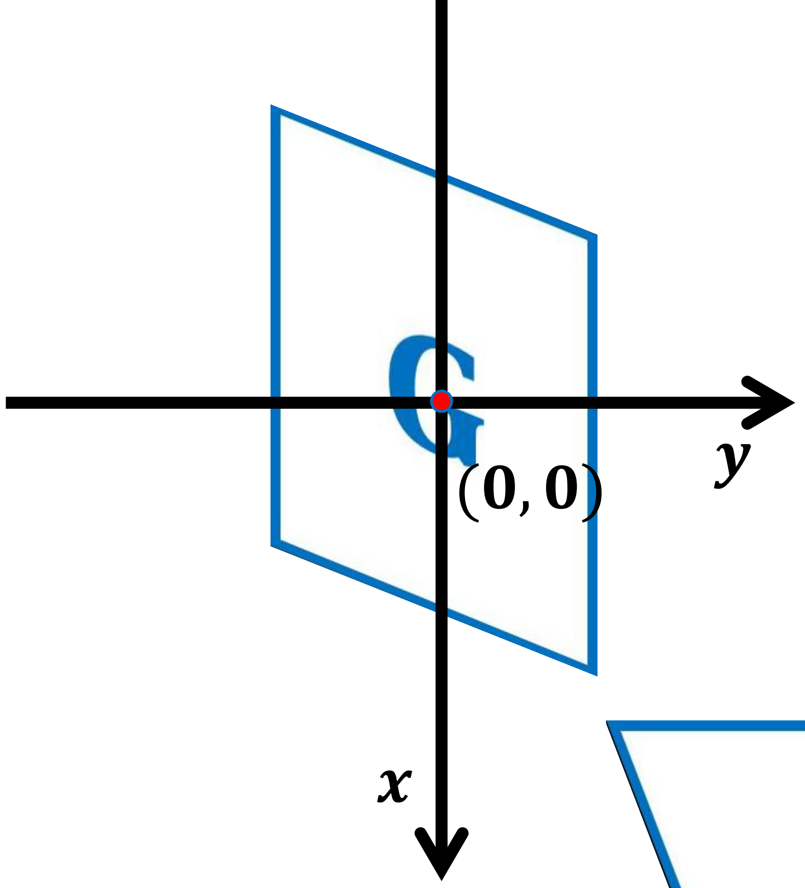
2D Linear Transform

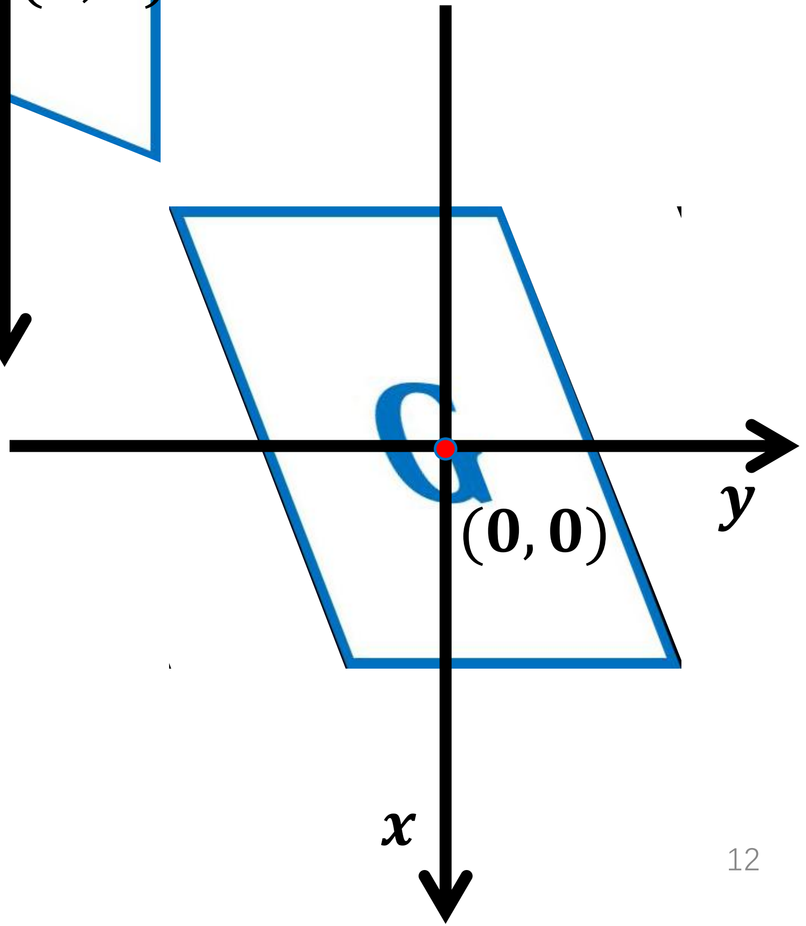
- Scale: $$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix} * \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$
- rotate: $$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} cos\theta & sin\theta \\ -sin\theta & cos\theta \end{bmatrix} * \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$
- Shift: $$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} * \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} shift_x \\ shift_y \end{bmatrix}$$

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

- **Shear:**

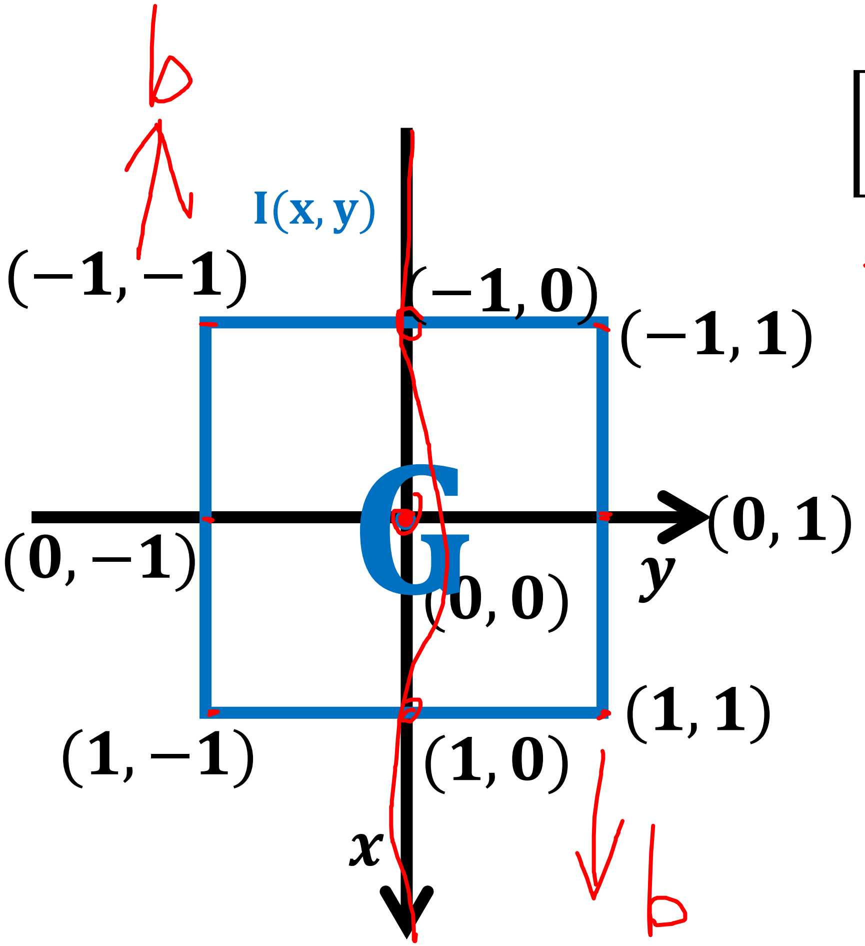






Shear in x-axis

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & b \\ 0 & 1 \end{bmatrix} * \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$



$x' = x + by, \; y' = y$

$(1, 0) \rightarrow \cancel{(1, 0)}$

$(0, 0) \rightarrow (0, 0)$ $\quad y = 0$

$(-1, 0) \rightarrow (-1, 0)$

$(1, -1) \rightarrow (1 - b, -1)$

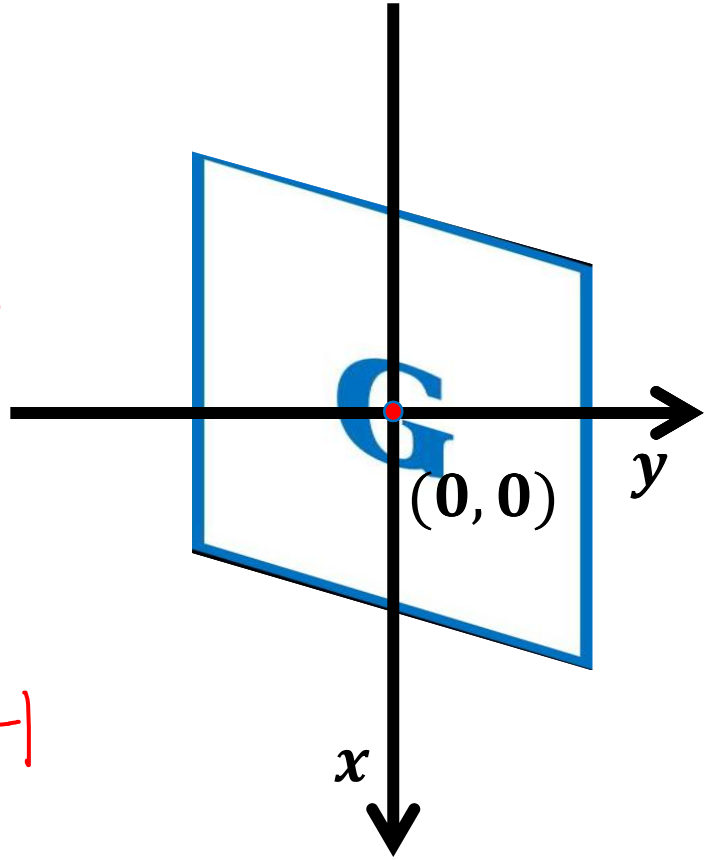
$(-1, -1) \rightarrow (-b - 1, -1)$

$(0, -1) \rightarrow (-b, -1)$ $\quad y = -1$

$(1, 1) \rightarrow (b + 1, 1)$

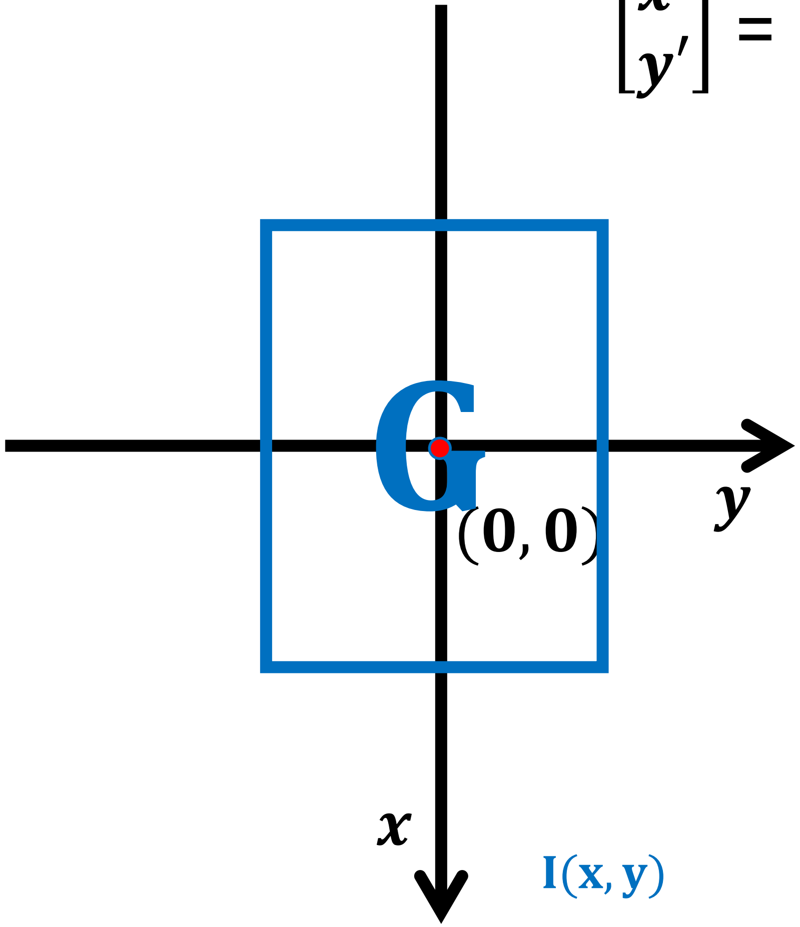
$(0, 1) \rightarrow (b, 1)$ $\quad y = 1$

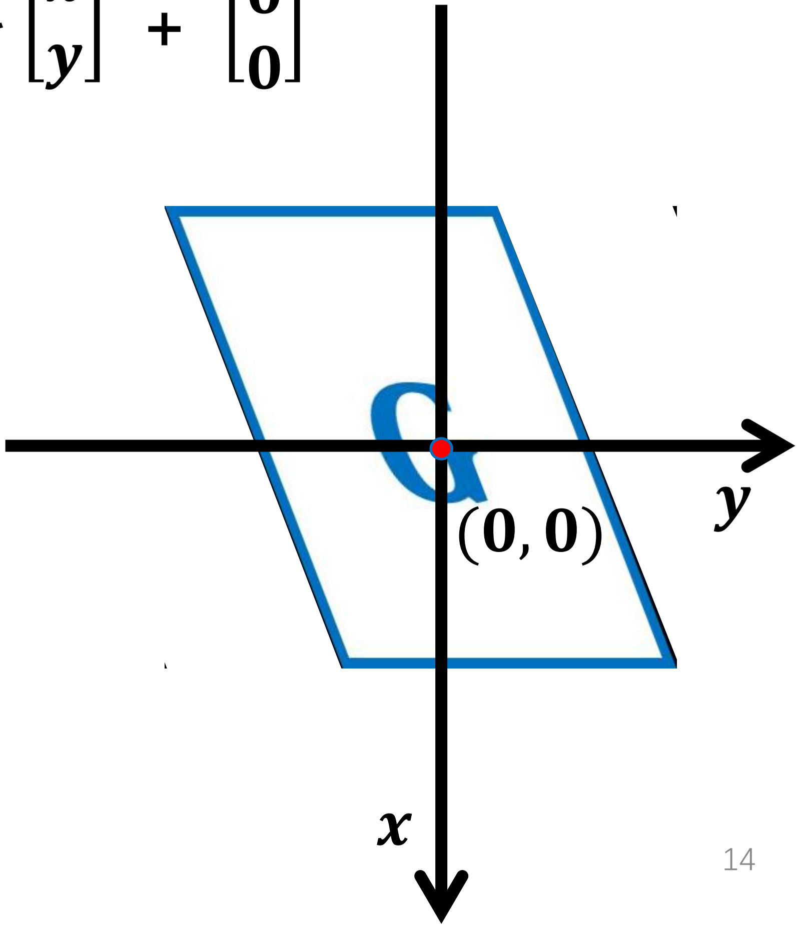
$(-1, 1) \rightarrow (b - 1, 1)$



Shear in y-axis

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ b & 1 \end{bmatrix} * \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$





$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} * \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} shift_x \\ shift_y \end{bmatrix}$$

2D Linear Transform

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} * \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} shift_x \\ shift_y \end{bmatrix}$$

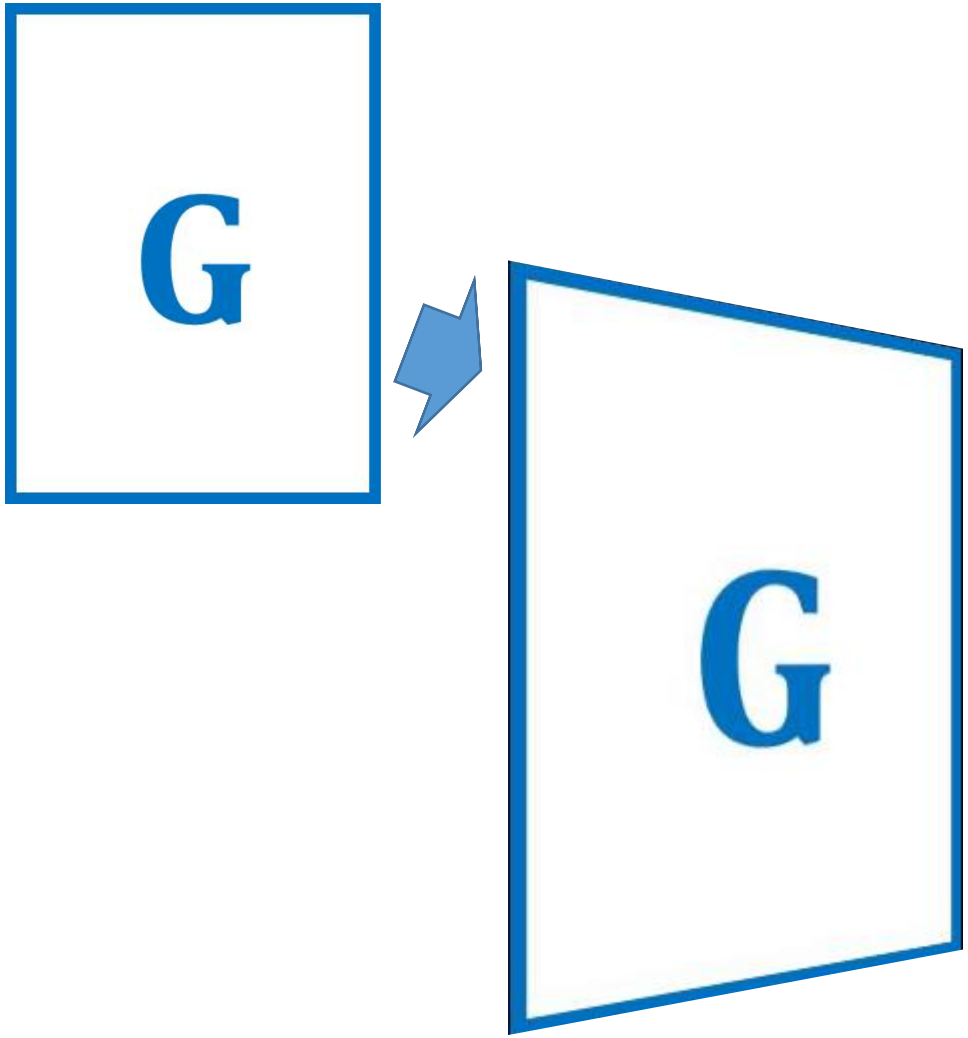
$\mathbf{J}(\mathbf{x}, \mathbf{y})$ $\qquad\qquad$ $\mathbf{I}(\mathbf{x}, \mathbf{y})$

- Rotation, Scaling, Shifting are all restricted 2D linear transform and are combined as rigid body transform.
- Shear transform preserves parallel lines from the original image.
- Shear + Rotation+ Scaling+ Shifting combined all the 2D linear/affine transform. (DOF = 6)

Projective Transform

- $$\begin{bmatrix}\tilde{x}'\\ \tilde{y}'\\ \tilde{z}'\end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & b_1\\ a_{21} & a_{22} & b_2\\ c_1 & c_2 & 1\end{bmatrix}\begin{bmatrix}x\\ y\\ 1\end{bmatrix}$$

- $$\begin{bmatrix}x'\\ y'\end{bmatrix} = \begin{bmatrix}\dfrac{a_{11}x+a_{12}y+b_1}{c_1x+c_2y+1}\\ \dfrac{a_{21}x+a_{22}y+b_2}{c_1x+c_2y+1}\end{bmatrix} = \begin{bmatrix}\dfrac{\tilde{x}'}{\tilde{z}'}\\ \dfrac{\tilde{y}'}{\tilde{z}'}\end{bmatrix}$$



Matlab commends

- A = [1 0 0; 0.4 1 0; 0 0 1]; % vertical shear
- tf = affine2d(A);
- im2 = imwarp(im,tf);
- figure;imshow(im2);

$$\begin{bmatrix} 1 & 0 & 0 \\ 0.4 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Practice

- **Using the zombie.jpg image, make a transform with $35°$ clock-wise rotation, 0.6 scaling and 50 shift on X-axial; 0.8 scaling and 15 shift on Y-axial.**

Outline

- **Spatial Operations**
  - Affine transform （仿射变换）
  - Projective transform
- **Image interpolation**
  - Nearest-neighbor interpolation
  - Linear & bi-linear interpolation

A real example

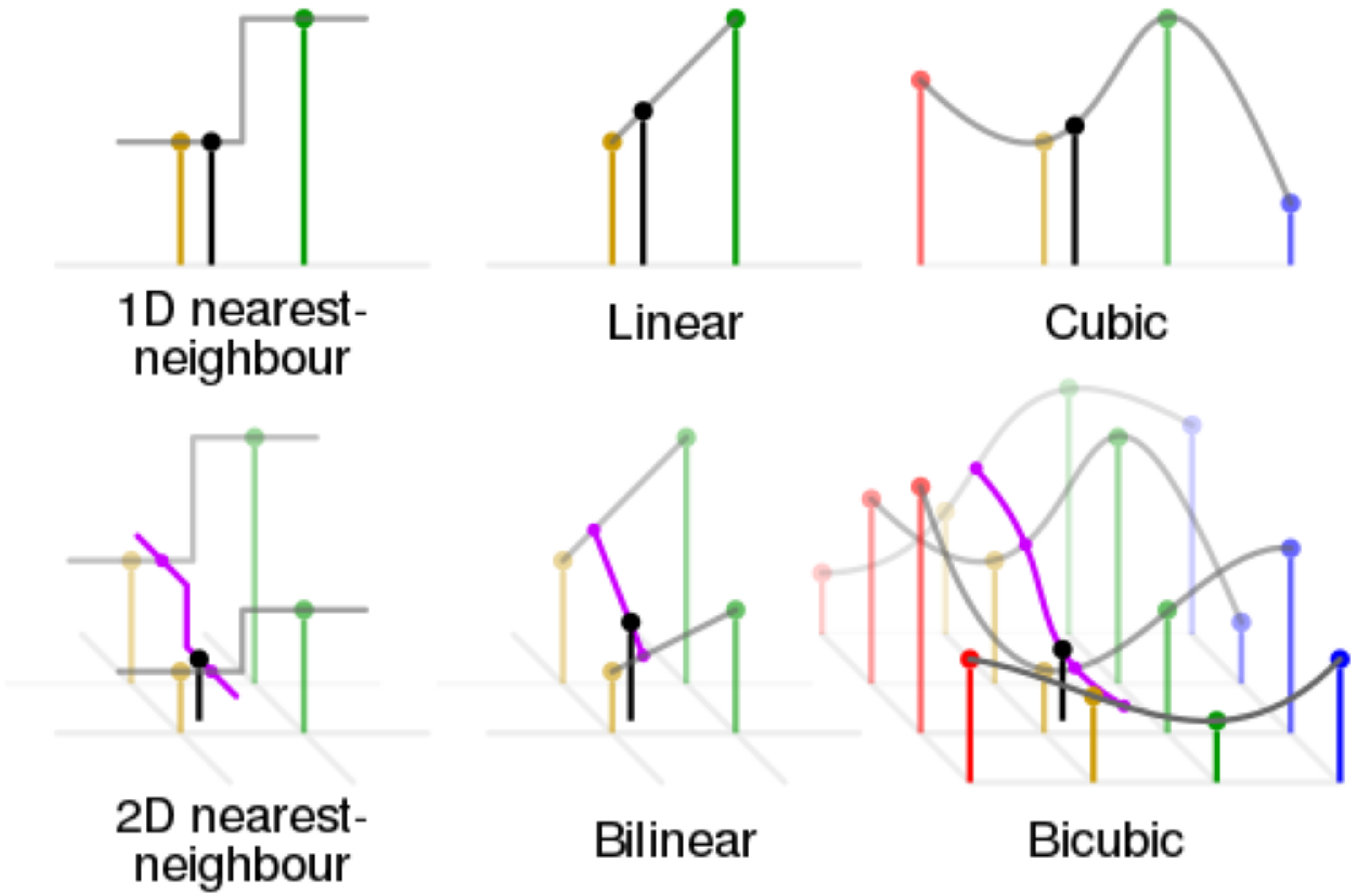
- $$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} * \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} shift_x \\ shift_y \end{bmatrix}$$

  a =1.2, $b = 1$, c =1, d =0.8, $shift_x = 3.2$, $shift_y = -1.6$

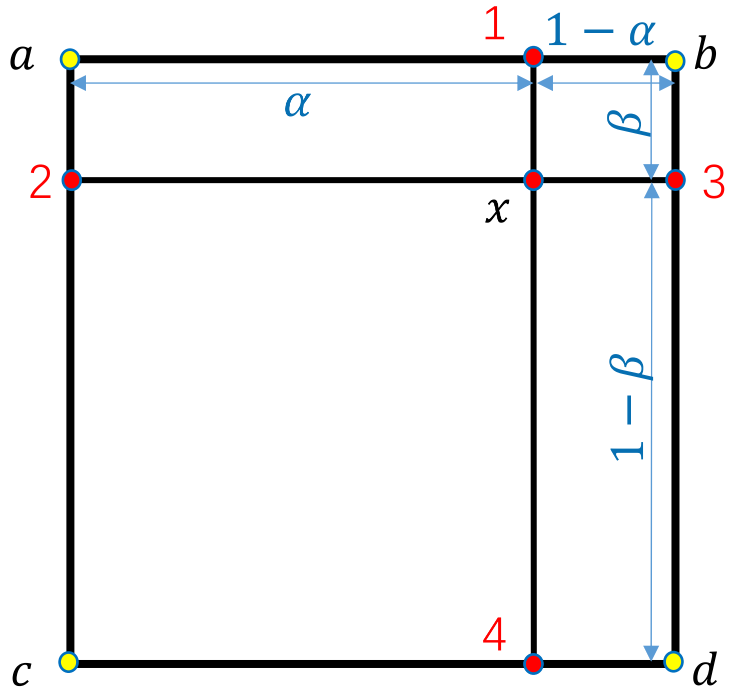
- $$\begin{bmatrix} 1 \\ 1 \end{bmatrix} \rightarrow \begin{bmatrix} 5.4 \\ 0.2 \end{bmatrix}$$

- Coordinate is not integer!! Then how to determinate intensity?

Interpolation



Bilinear interpolation



$$x_{1} = (1-\alpha)\,a + \alpha\cdot b \quad \text{or} \quad \cancel{\alpha\cdot a + (1-\alpha)b} \ ?$$

$$\text{if} \quad \alpha = 0.7$$

$$x_{2} = (1-\beta)\,a + \beta\cdot c$$

$$x_{3} = (1-\beta)\,b + \beta\cdot b$$

$$x_{4} = (1-\alpha)\,c + \alpha\cdot d$$

$$x = (1-\beta)\cdot x_{1} + \beta\cdot x_{4}$$

$$\text{or} = (1-\alpha)\,x_{2} + \alpha\cdot x_{3}$$

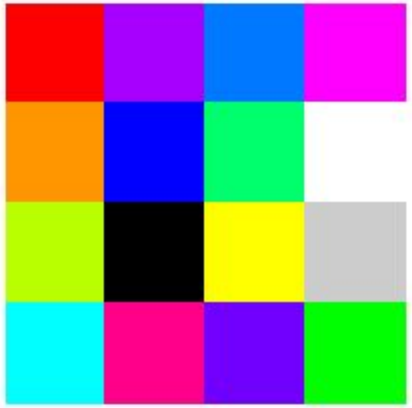
$$= (1-\beta)\left[(1-\alpha)a + \alpha b\right] + \beta\left[(1-\alpha)c + \alpha d\right]$$

$$= (1-\beta)(1-\alpha)\cdot a + \alpha(1-\beta)b + \beta(1-\alpha)\cdot c + \alpha\beta\cdot d$$

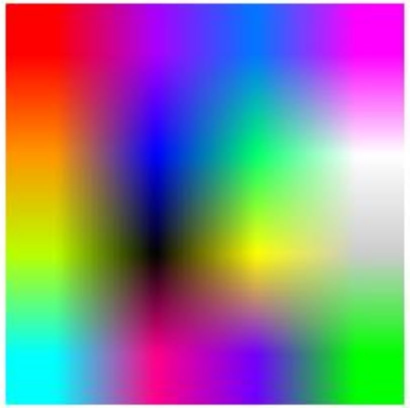
$$\alpha = 0.7,\ \beta = 0.2$$

$$= 0.24\,a + 0.56\,b + 0.06\,c + 0.14\,d$$

Nearest-neighbor vs Bilinear vs Bicubic interpolation



Nearest neighbor



Bilinear

Bicubic

Image Registration

- To align two or more images of the same scene
- Given input and output images, to estimate the transformation functions and then use it to register the two images

Image 1

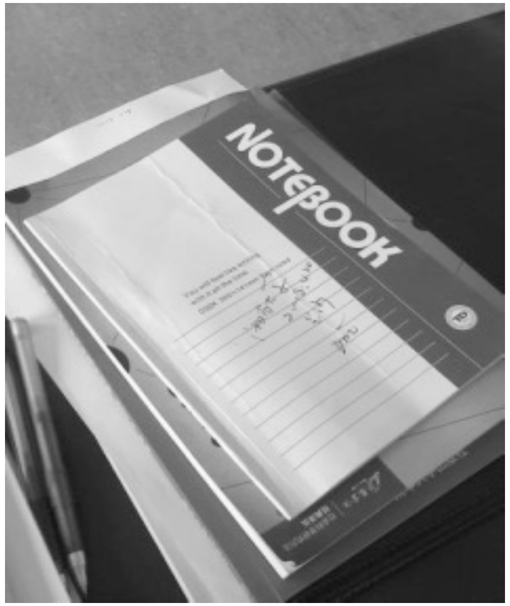
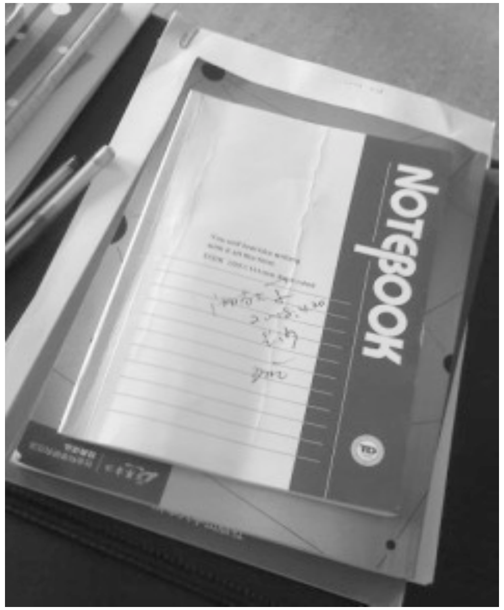
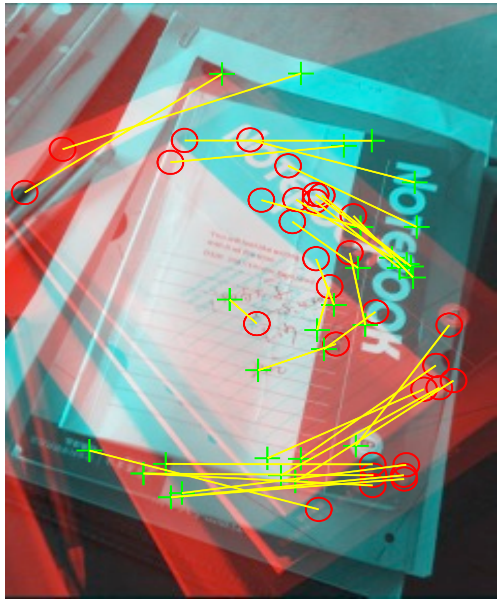


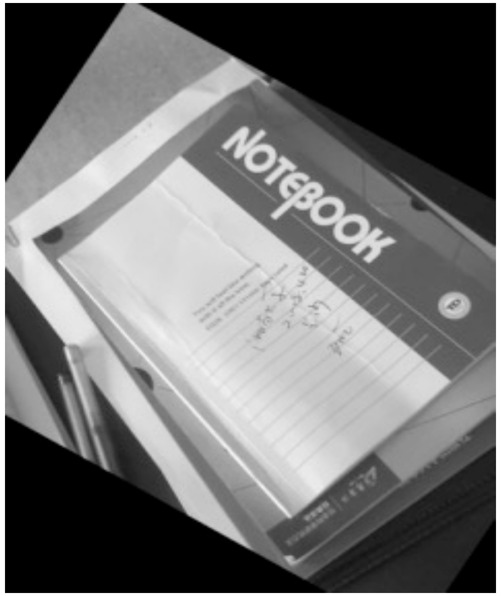
Image 2



Key points matching



Estimated rotated image



SIFT- Scale Invariant Feature Transform

Approach :

- ➢Create a scale space of images
  - Construct a set of progressive Gaussian blurred images
  - Take differences to get a difference of Gaussian pyramid (similar to a Laplacian)
- ➢Find local extrema in this space. Choose the key points from extrema.
- ➢For each keypoint, in a 16x16 window, find histograms of gradient directions
- ➢Create a feature vector out of these.

Take home message

- **Spatial transform： how does parameter effect on special transform. Rigid-body, shear, projective.**
- **Interpolation: To look like the nearest-neighbors and to involve more close-by neighbors.**