Lecture 2 - Image Fundamentals

This lecture will cover:

- Image acquisition
- Sampling and Quantization
- Pixels
- **Image operation**
- Color space

上海科技大学
ShanghaiTech University

Image Operations

- Array and Matrix Operation
- Vector and Matrix Operation
- Linear and Nonlinear Operation
- Set and Logical Operation
- Arithmetic Operation
- Spatial Operation
- Image Transformation
- Probabilistic Methods

Image Operations

- Array and Matrix Operation
- Vector and Matrix Operation
- Linear and Nonlinear Operation
- Set and Logical Operation
- **Arithmetic Operation**
- **Spatial Operation**
- Image Transformation
- Probabilistic Methods

Array and Matrix Operation

Consider two 2 x 2 image

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \text{ and } \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}$$

- **Array product**

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} = \begin{bmatrix} a_{11}b_{11} & a_{12}b_{11} \\ a_{21}b_{21} & a_{22}b_{22} \end{bmatrix}$$

- **Matrix product**

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} = \begin{bmatrix} a_{11}b_{11} + a_{12}b_{21} & a_{11}b_{12} + a_{12}b_{22} \\ a_{21}b_{11} + a_{22}b_{21} & a_{21}b_{12} + a_{22}b_{22} \end{bmatrix}$$

Vector and Matrix Operation

- **Multispectral image processing**

A pixel in a n-dimensional space can be expressed as a column vector $Z = [z_1, z_2 \ldots z_n]^T$, then a vector norm between two pixels Z and A

$$\|Z - A\| = [(Z - A)^T (Z - A)]^{\frac{1}{2}}$$
$$= [(z_1 - a_1)^2 + (z_2 - a_2)^2 + \cdots + (zn - an)^2]^{\frac{1}{2}}$$

- **Linear transformations**

$$g = Hf + n$$

上海科技大学 ShanghaiTech University

Linear and Nonlinear Operation

An operator

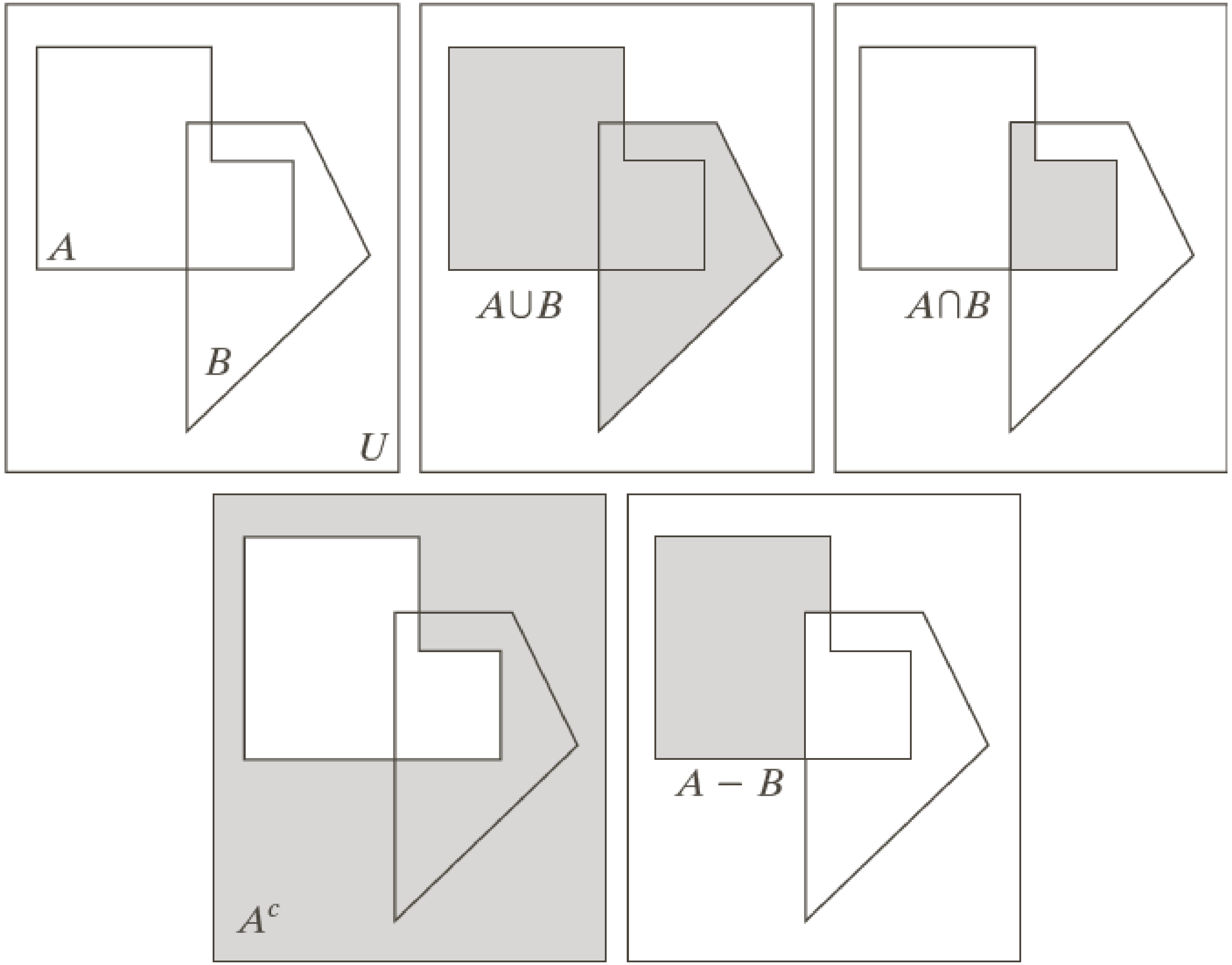
$$H[f(x,y)] = g(x,y)$$

is linear if

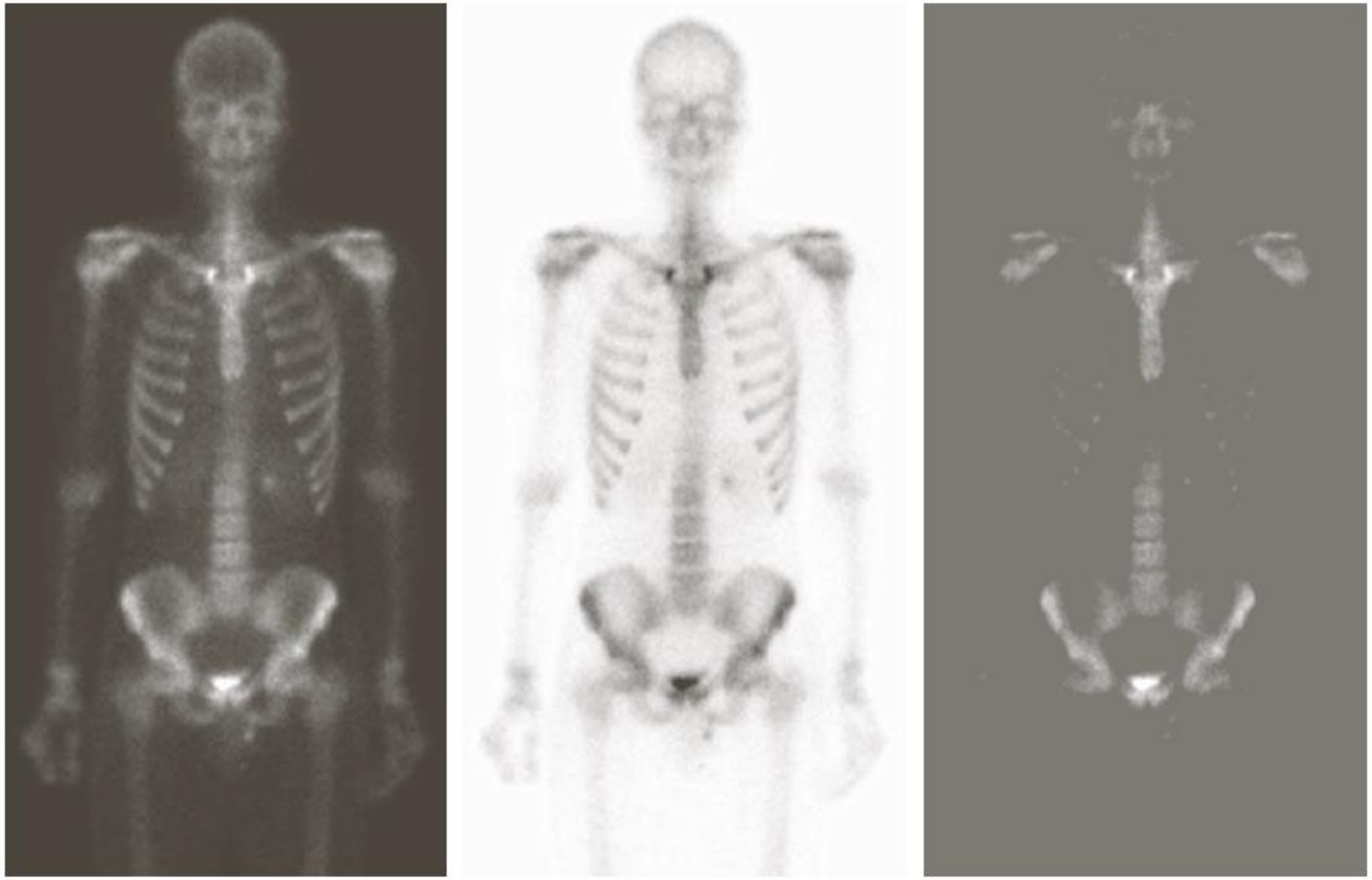
$$H[a_i f_i(x,y) + ajfj(x,y)] = a_i H[f_i(x,y)] + ajH[f_j(x,y)]$$
$$= a_i g_i(x,y) + a_j g_j(x,y)$$

- **Additivity**
- **Homogeneity**

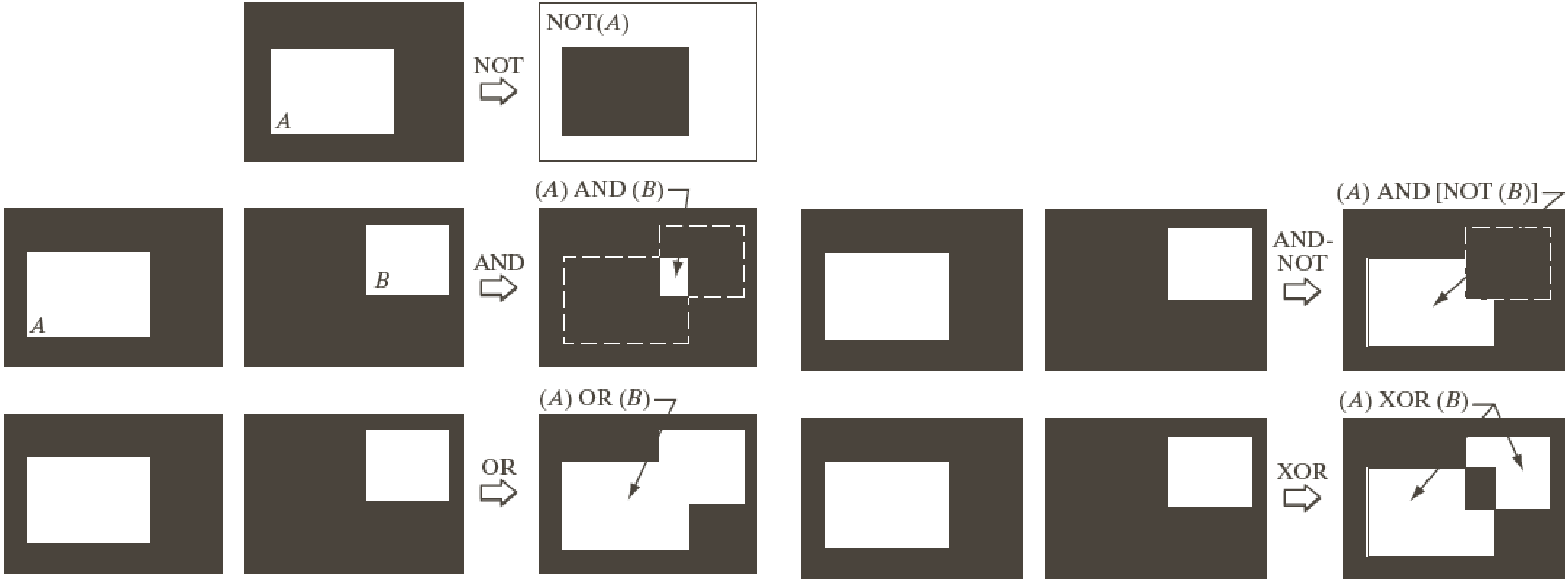
Set Operation (Coordinates)



Set Operation (Intensity)



Logical Operation



Arithmetic Operation

- **Addition**

$$s(x, y) = f(x, y) + g(x, y)$$

- **Subtraction**

$$d(x, y) = f(x, y) - g(x, y)$$

- **Multiplication**

$$p(x, y) = f(x, y) \times g(x, y)$$

- **Division**

$$v(x, y) = f(x, y) \div g(x, y)$$

Image Addition

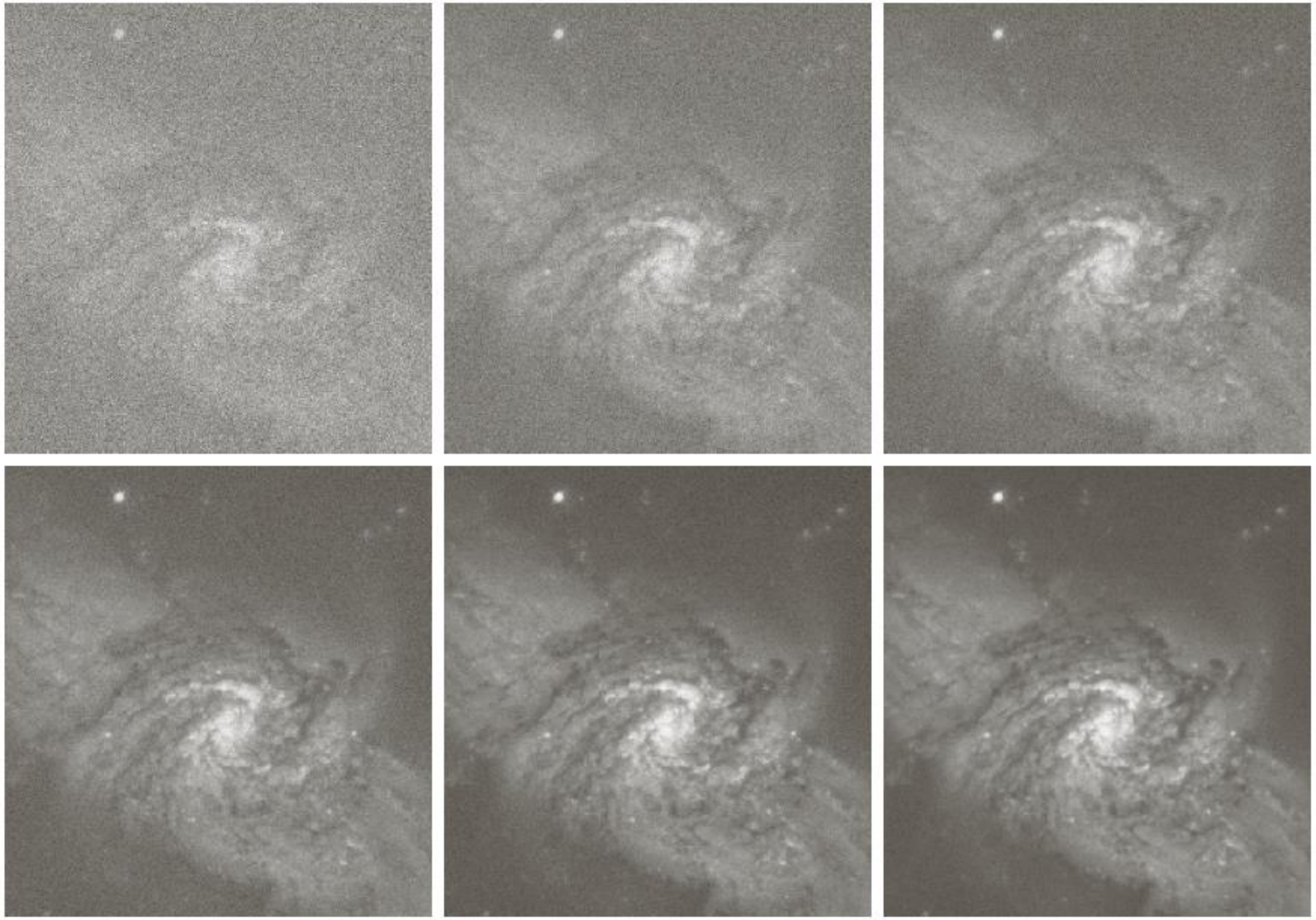


Image Addition

If $f(x, y)+g(x, y) > L_{\max}$, $s(x, y)$ can be calculated as

- Average

$$s(x, y) = \frac{f(x, y) + g(x, y)}{2}$$

- Scale

$$\{\min[s(x, y)], \max[s(x, y)]\} = \{0, L_{\max}\}$$

- Max intensity value

$$\text{If } s(x, y) > L_{\max}, \qquad s(x, y) = L_{\max}$$

上海科技大学 ShanghaiTech University

Image Subtraction

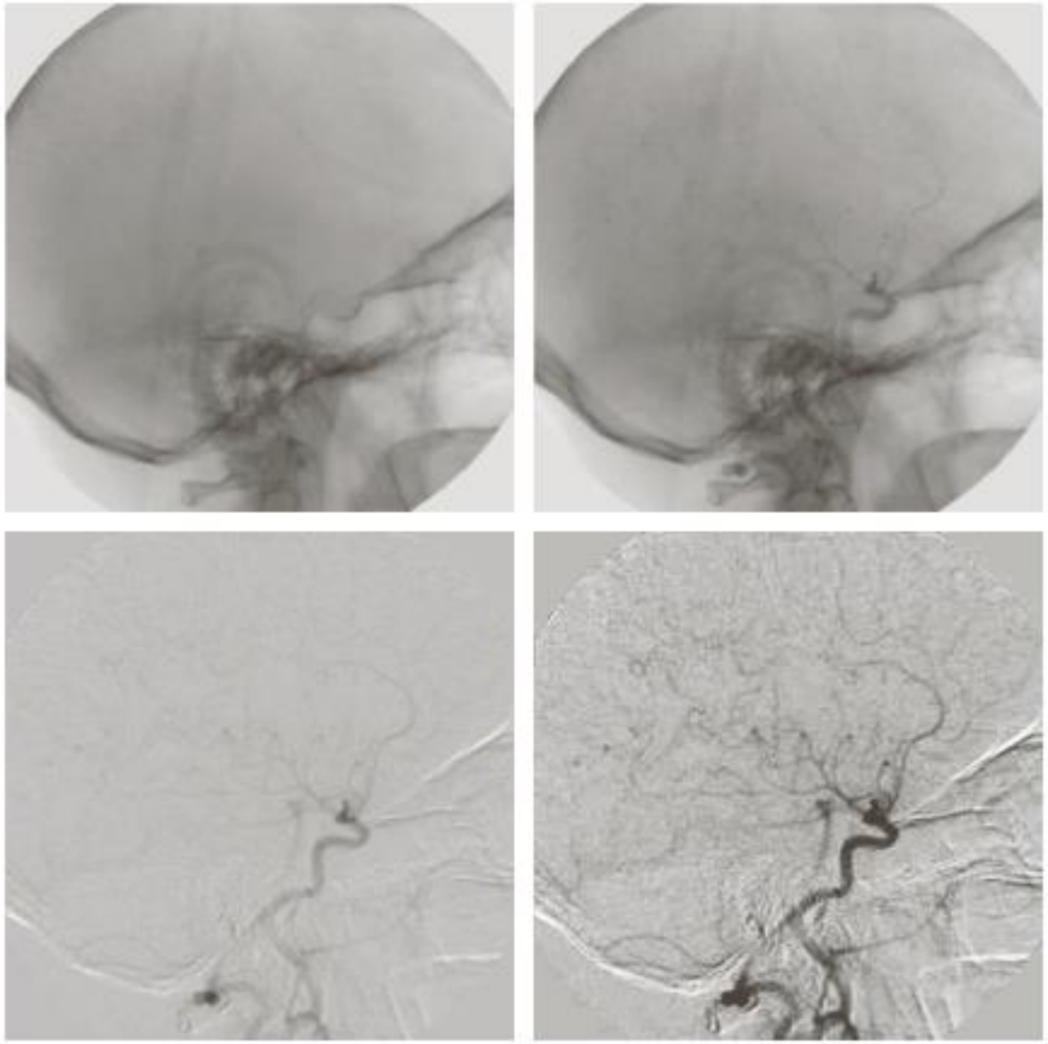


Image Multiplication

Image Division

$g(x,y) = f(x,y)\,h(x,y)$ $\quad$ $h(x,y)$ $\quad$ $f(x,y)$

$$f(x,y) = g(x,y)/h(x,y)$$

Spatial Operation

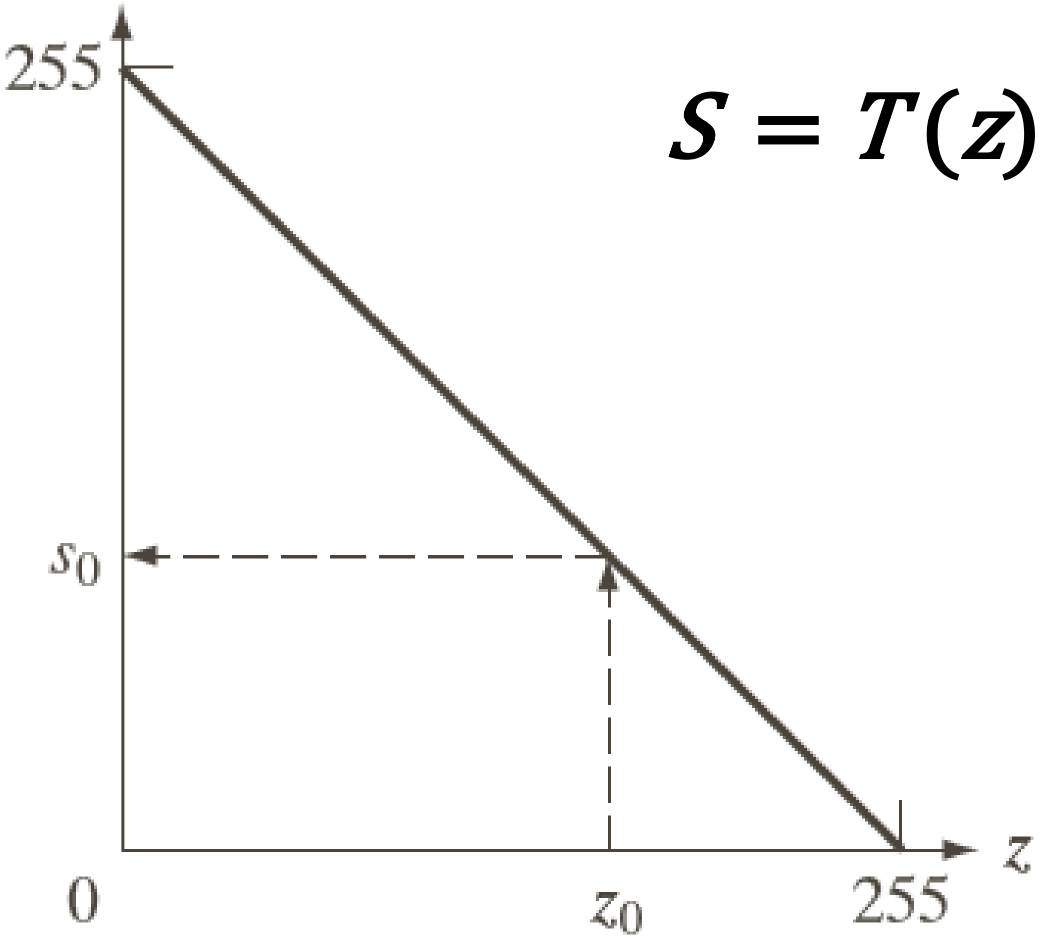
Performed directly on the pixels of the image

- Single-pixel operations
- Neighborhood operations
- Image geometry

  Scale, Rotate, Translate, Mirror, Transpose, Shear, etc.

- Interpolation

Single-pixel Operation



$$S = T(z)$$

Region operation

S_{xy} is a region with center (x, y), $g(x,y) = \frac{1}{mn}\sum_{(r,c)\in Sx_y} f(r,c)$

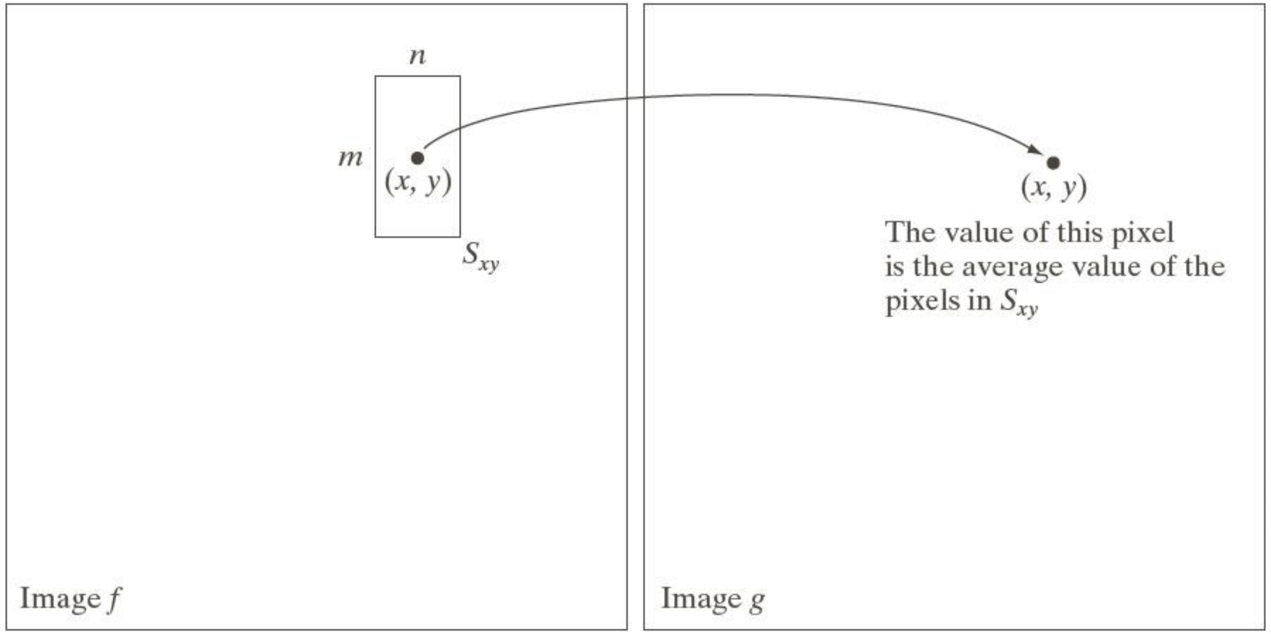


Image geometry

- **Modify spatial relationship between pixels –** *rubber-sheet*
  - Forward mapping: $(x\ y) = T(v\ w)$
  - Inverse mapping: $(v\ w) = T^{-1}(x\ y)$

- **Affine transform**

$$[x\ y\ 1] = [v\ w\ 1]T = [v\ w\ 1]\begin{bmatrix} t_1 & t_4 & 0 \\ t_2 & t_5 & 0 \\ t_3 & t_6 & 1 \end{bmatrix}$$

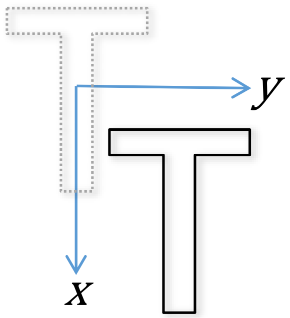
or

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = T\begin{bmatrix} v \\ w \\ 1 \end{bmatrix} = \begin{bmatrix} t_1 & t_2 & t_3 \\ t_4 & t_5 & t_6 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} v \\ w \\ 1 \end{bmatrix}$$

Affine Transform

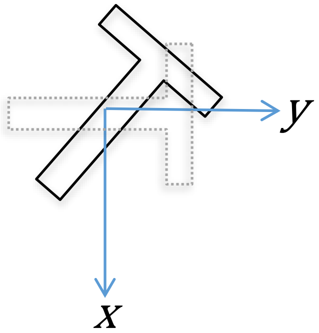
➢ Translation

$$\begin{cases} x = v + \Delta v \\ y = w + \Delta w \end{cases} \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & \Delta v \\ 0 & 1 & \Delta w \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ w \\ 1 \end{bmatrix}$$



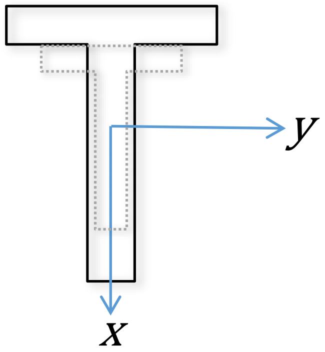
➢ Rotation

$$\begin{cases} x = v\cos\beta - w\sin\beta \\ y = v\sin\beta + w\cos\beta \end{cases} \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\beta & -\sin\beta & 0 \\ \sin\beta & \cos\beta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ w \\ 1 \end{bmatrix}$$



➢ Scaling

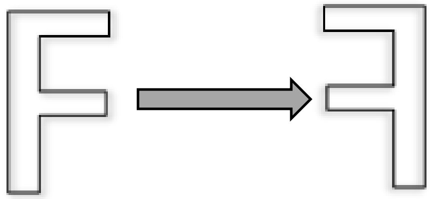
$$\begin{cases} x = c_x v \\ y = c_y w \end{cases} \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} c_x & 0 & 0 \\ 0 & c_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ w \\ 1 \end{bmatrix}$$



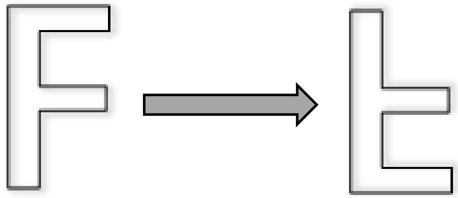
Affine Transform

- **Mirror**

Horizontal: $\begin{cases} x = W - v \\ y = w \end{cases} \Longrightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & W \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ w \\ 1 \end{bmatrix}$

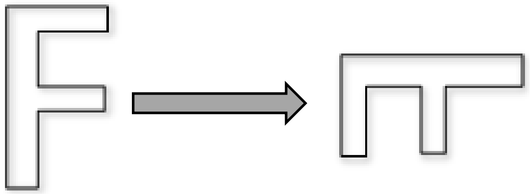


Vertical: $\begin{cases} x = v \\ y = H - w \end{cases} \Longrightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & H \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ w \\ 1 \end{bmatrix}$



- **Transpose**

$$\begin{cases} x = w \\ y = v \end{cases} \Longrightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ w \\ 1 \end{bmatrix}$$

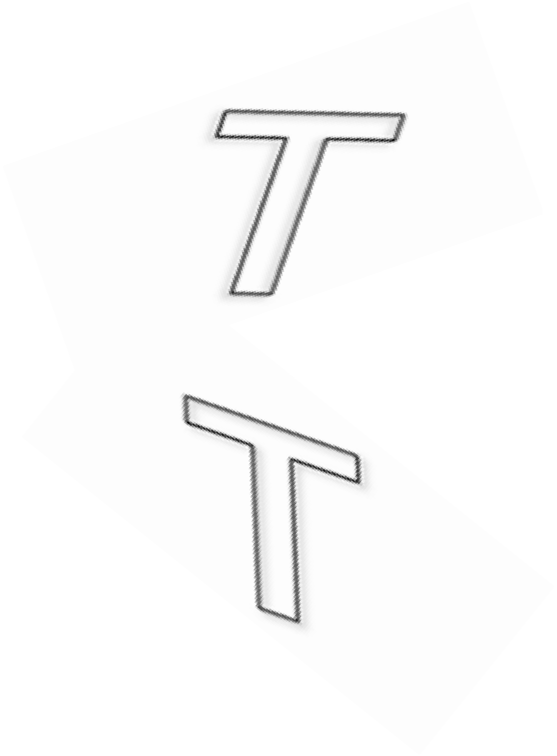


Affine Transform

- **Shear**

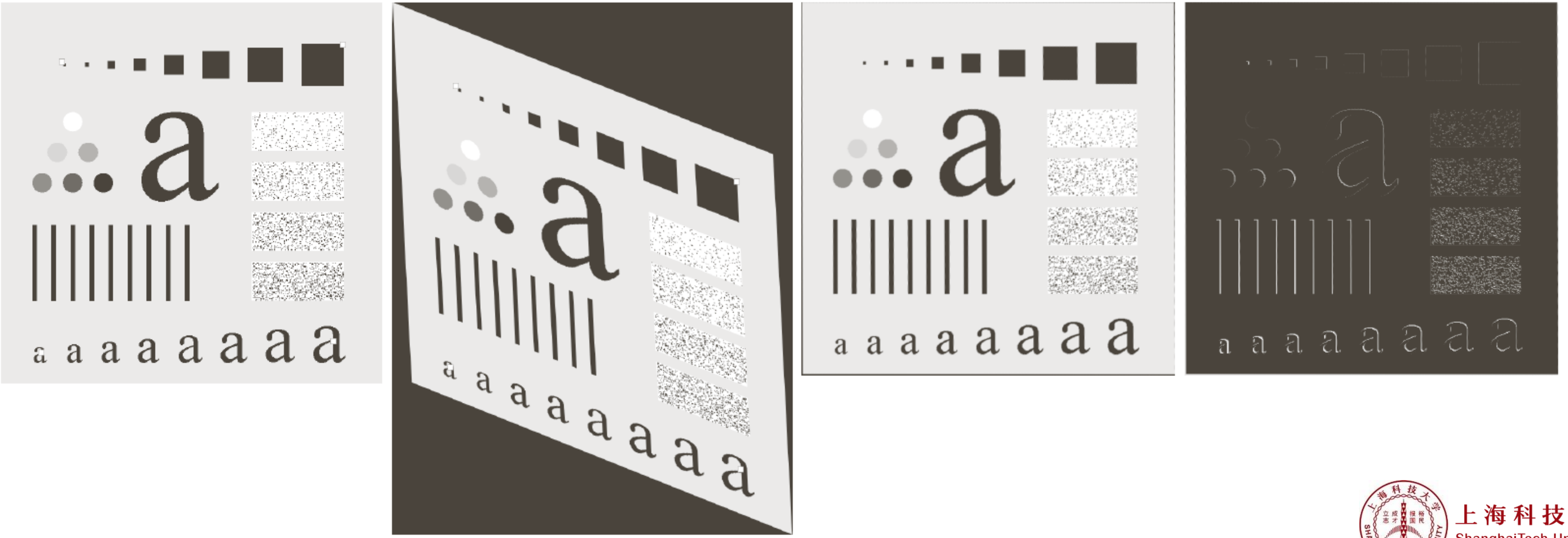
Horizontal: $\begin{cases} x = v + c_y w \\ y = w \end{cases} \Longrightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & c_y & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ w \\ 1 \end{bmatrix}$

Vertical: $\begin{cases} x = v \\ y = c_x v + w \end{cases} \Longrightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ c_x & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ w \\ 1 \end{bmatrix}$

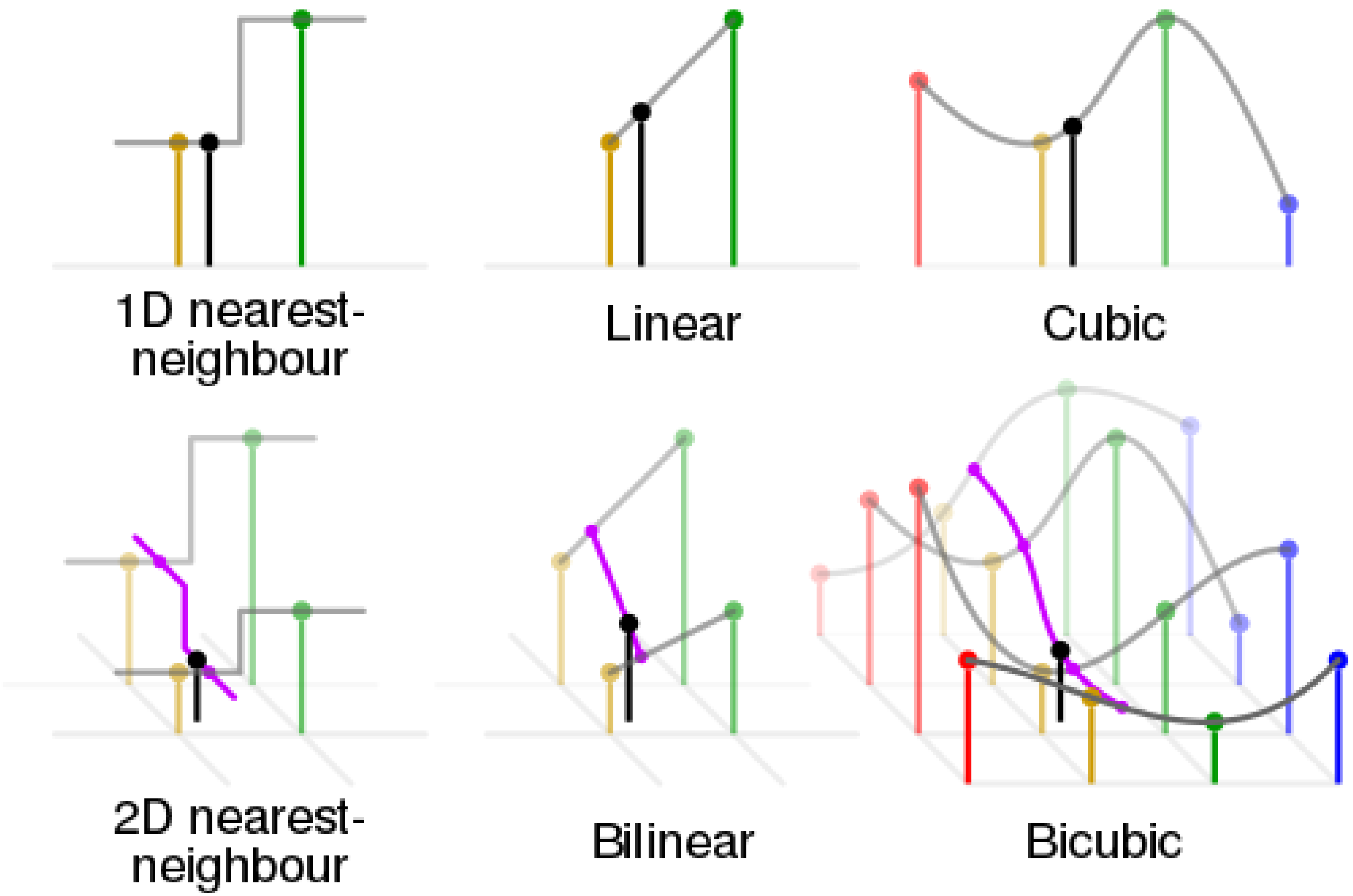


Registration

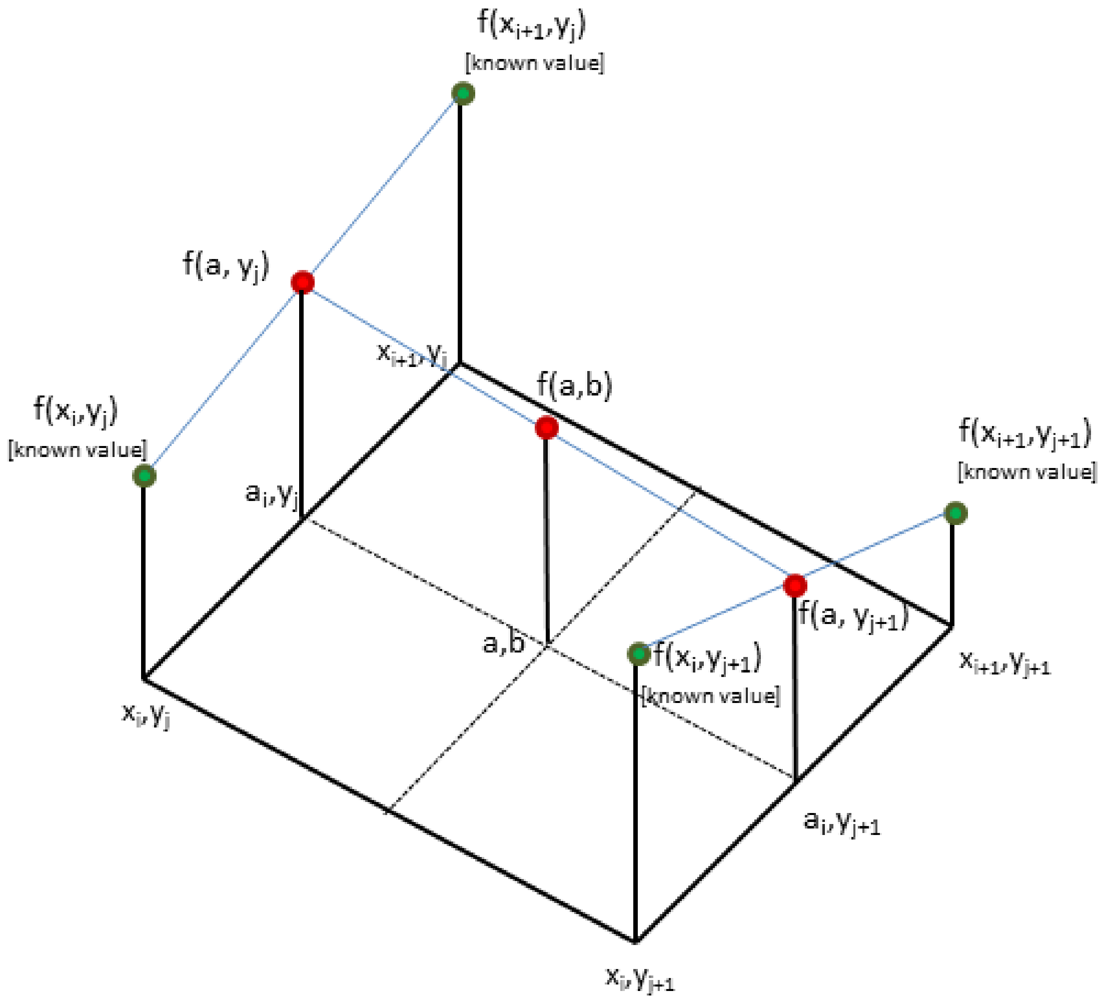
- To align two or more images of the same scene
- Given input and output images, to estimate the transformation functions and then use it to register the two images



Interpolation



Bilinear interpolation



Interpolation