Lecture 5

黃正民
Cheng-Ming Huang

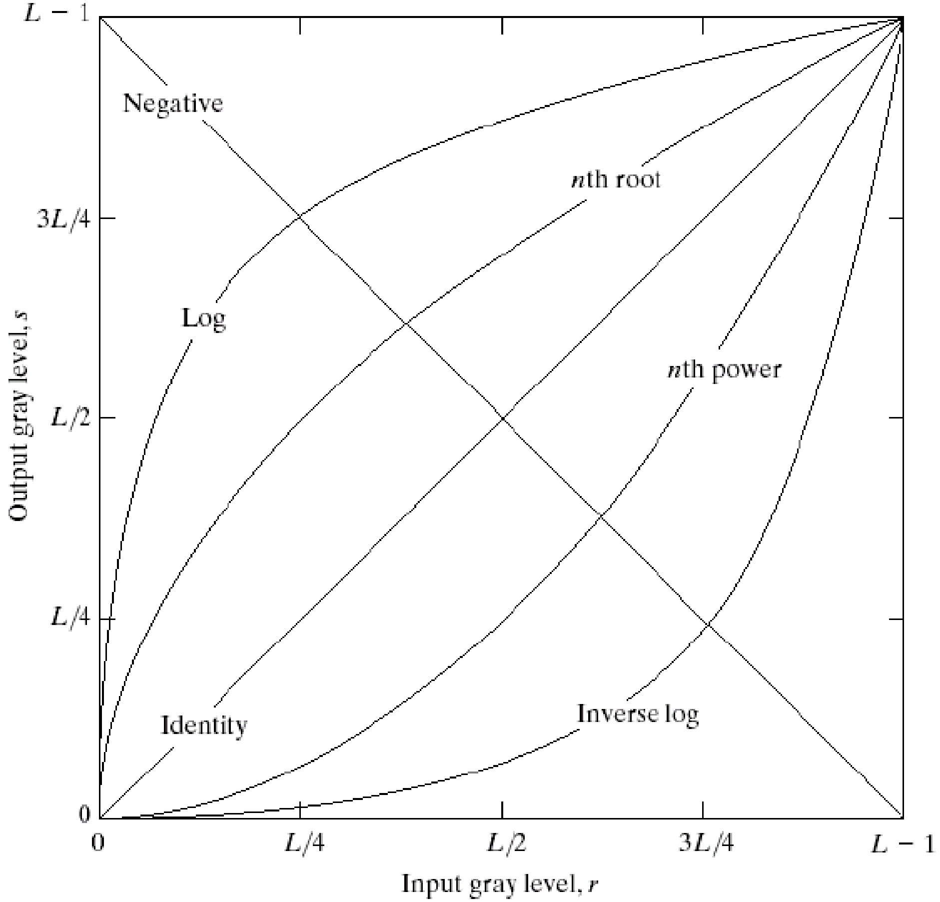
EE, NTUT

Point Operators

- Color spaces transformation
- Gray-level transformation
- Histogram processing
- Color description and matching
- Geometric transformation

Gray-Level Transformation

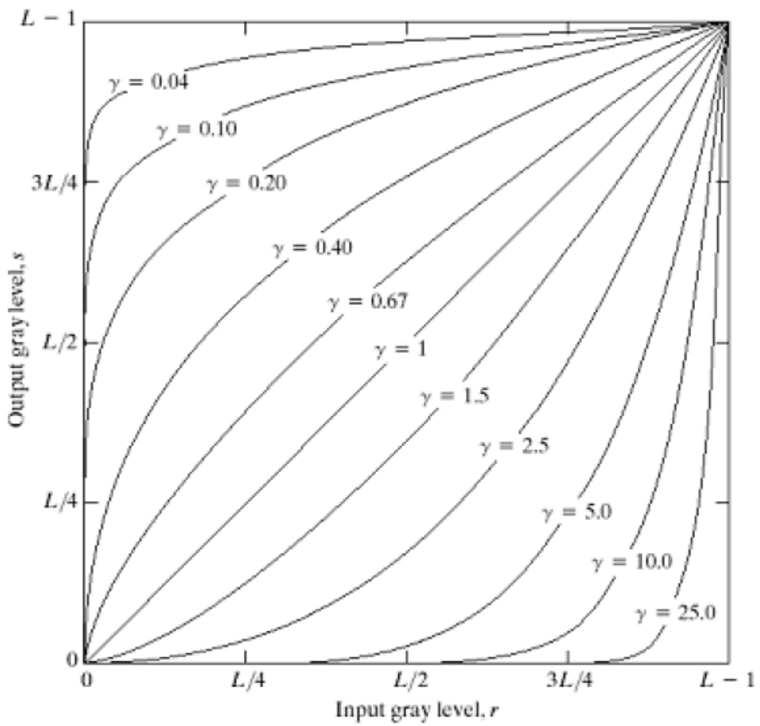
- Input gray level v.s. output gray level

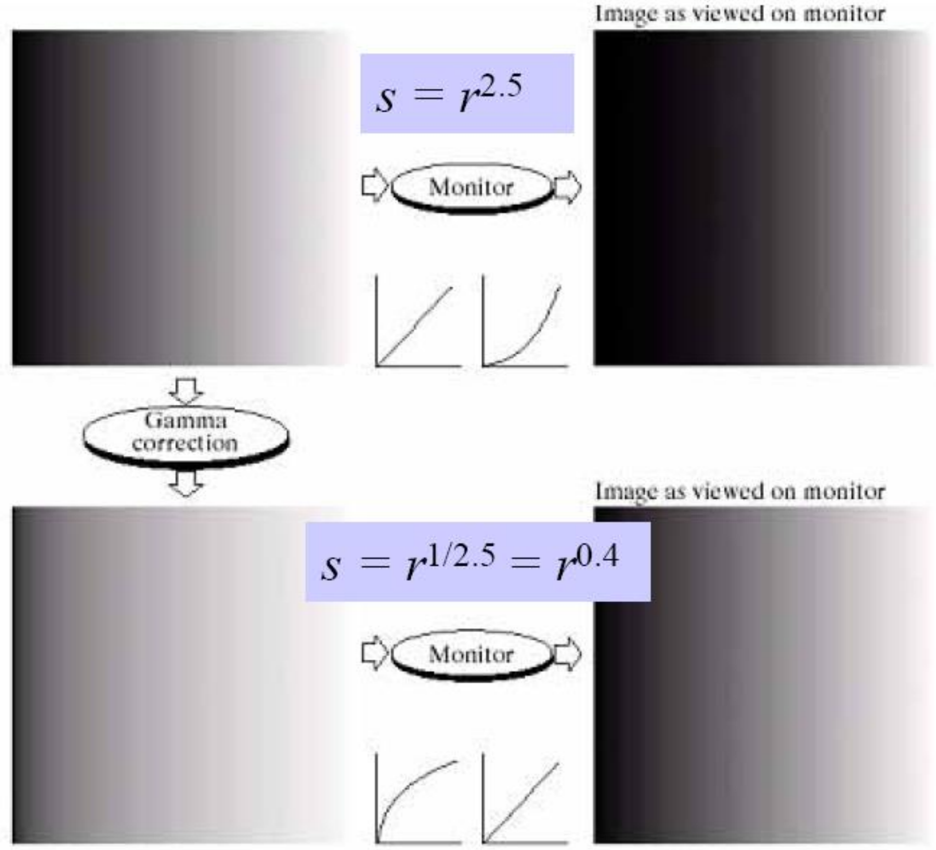


Gamma Correction

- Remove non-linear mapping between image and input/output devices
- Power-Law Transform

$$s = cr^{\gamma}$$

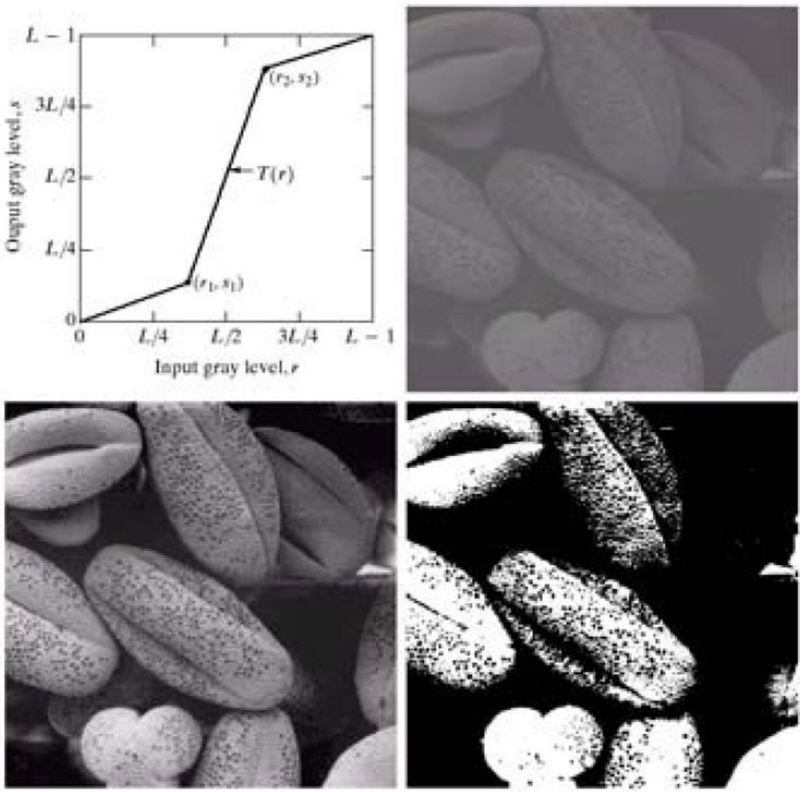




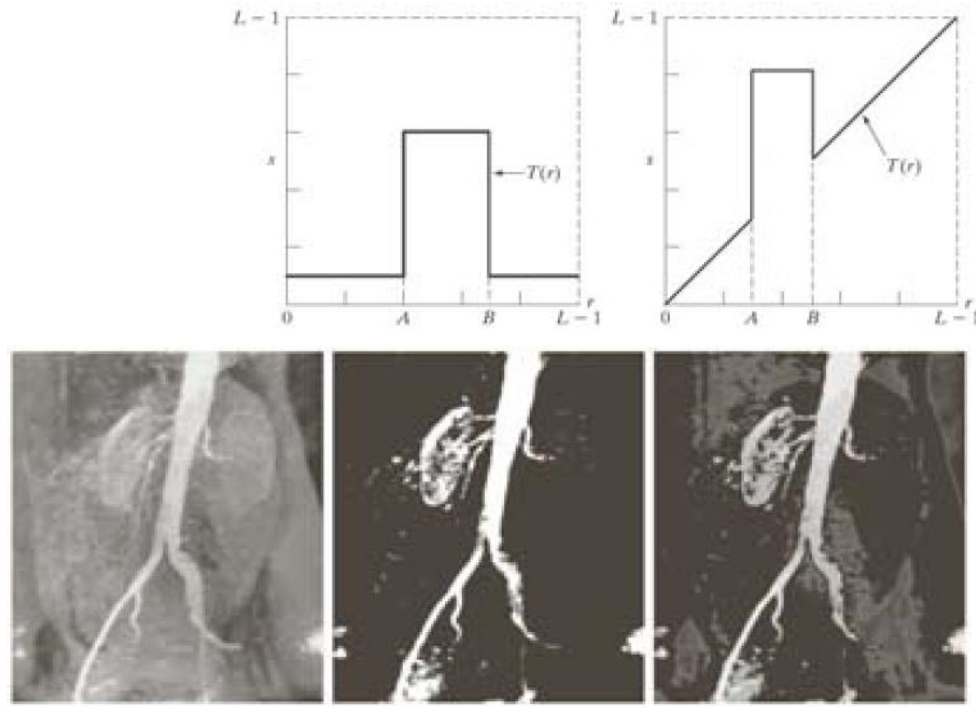
Gray-Level Transformation

- Contrast stretching
- Gray-level slicing

original



contrast stretching thresholding



original

Histogram Processing

n : total number of image pixels

n_j : number of pixels with intensity value j

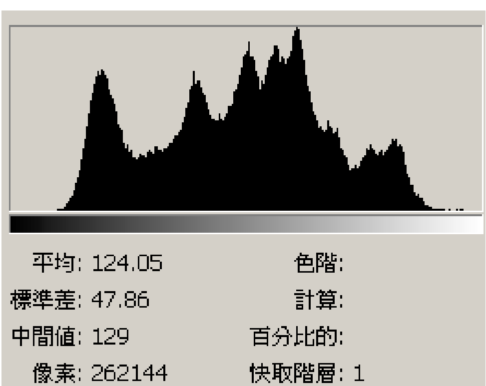
Histogram equalization

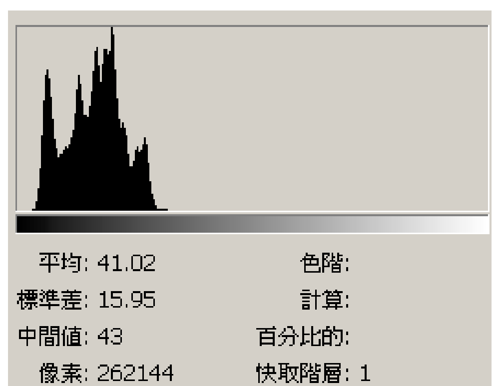
pixel intensity value $i \rightarrow c_i = 255\sum_{j=0}^{i}\frac{n_j}{n}$

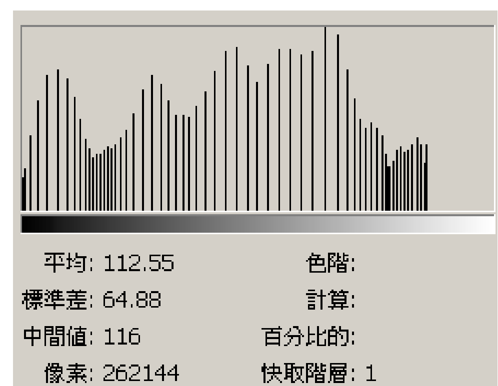
a

b=a/3

c= equalization of b



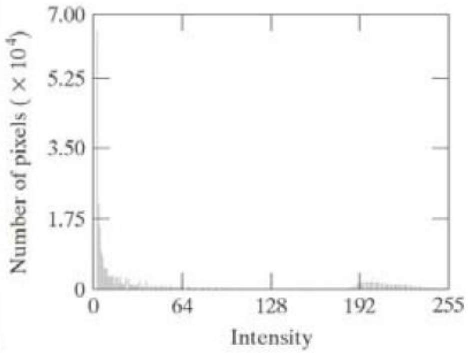




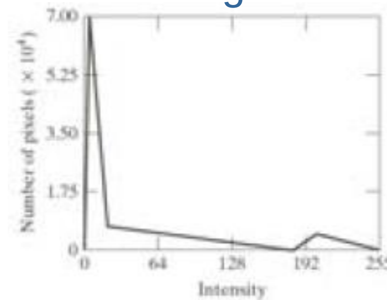
Histogram Processing

- Histogram matching (specification)

original

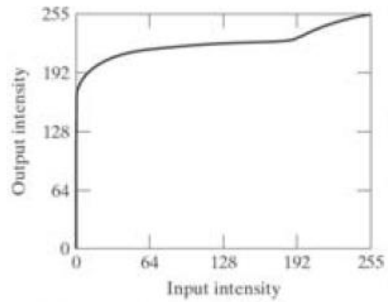


specified histogram

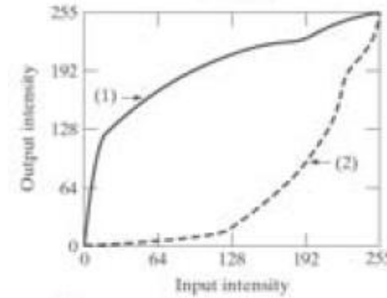


e.g. 2

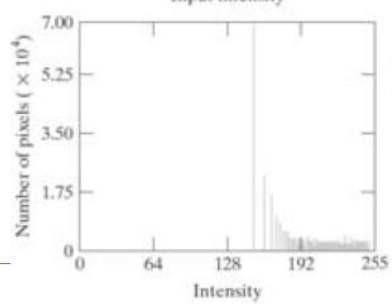
transformation function

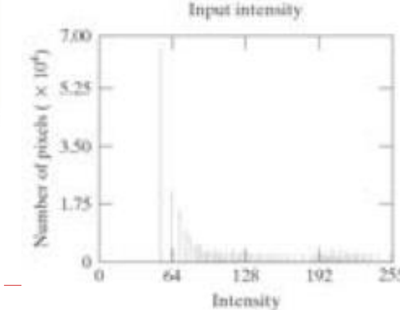


e.g. 1



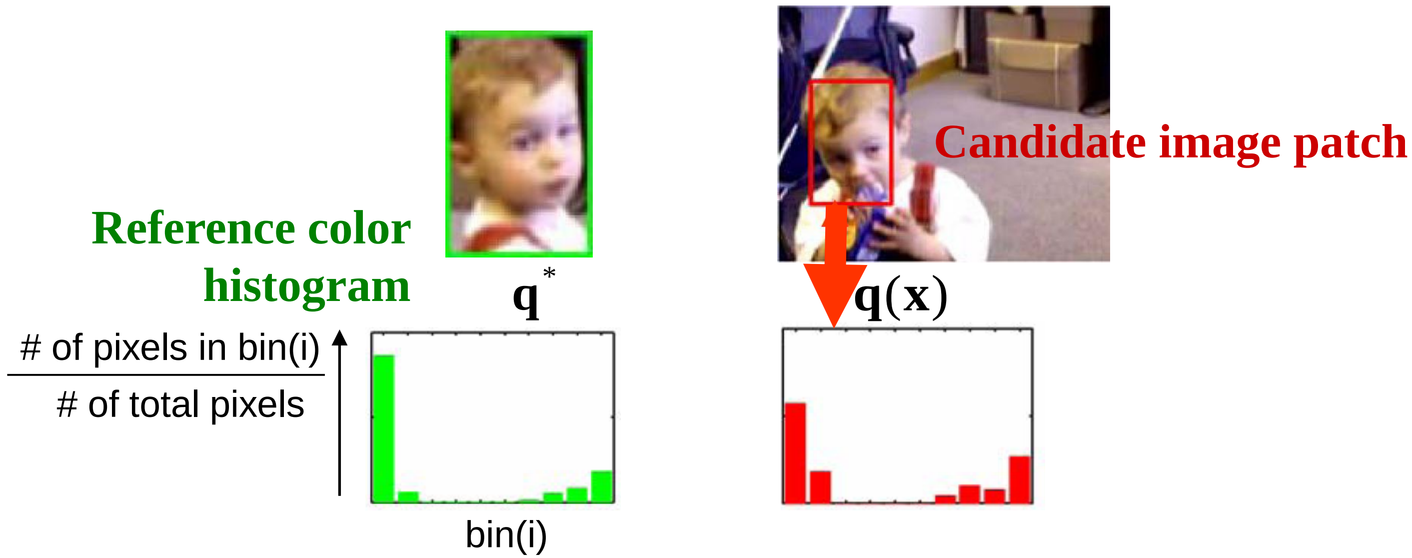
transformation function





Color Description and Matching

- Color description
  - Histogram (in selected color space)
- Color similarity matching
  - Compares the dense histograms of reference and candidate
  - Normalize the histograms as probability distributions



Color Description and Matching

- Compares two dense histograms H_1 & H_2

  (Σ_I: summation over all bins of image I)

  - Correlation

$$d(H_1, H_2) = \frac{\sum_I (H_1'(I) \cdot H_2'(I))}{\sqrt{\sum_I (H_1'(I)^2) \cdot \sum_I (H_2'(I)^2)}}$$

$$H_k'(I) = \frac{H_k(I) - 1}{N \cdot \sum_J H_k(J)}$$

  - Chi-Square

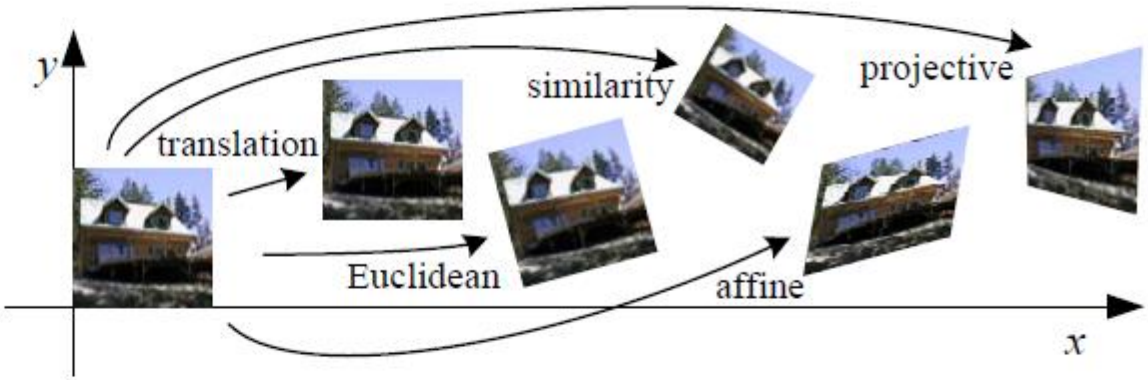
$$d(H_1, H_2) = \sum_I \frac{(H_1(I) - H_2(I))^2}{H_1(I) + H_2(I)}$$

  - Bhattacharyya distance

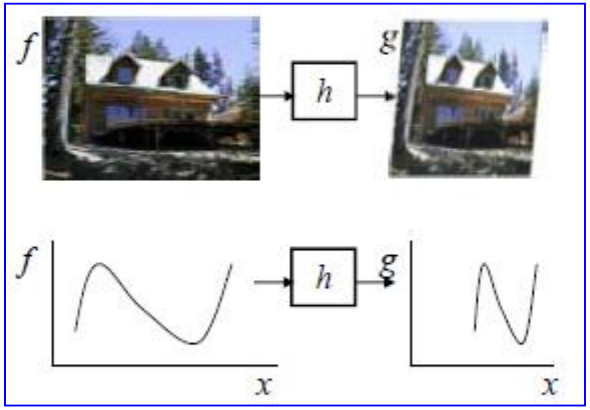
$$d(H_1, H_2) = \sqrt{1 - \sum_I \frac{\sqrt{H_1(I) \cdot H_2(I)}}{\sqrt{\sum_I H_1(I) \cdot \sum_I H_2(I)}}}$$

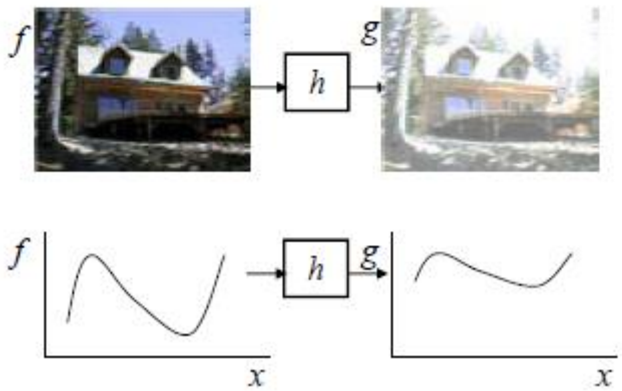
Geometric Transformations

- Basic set of 2D geometric image transformations



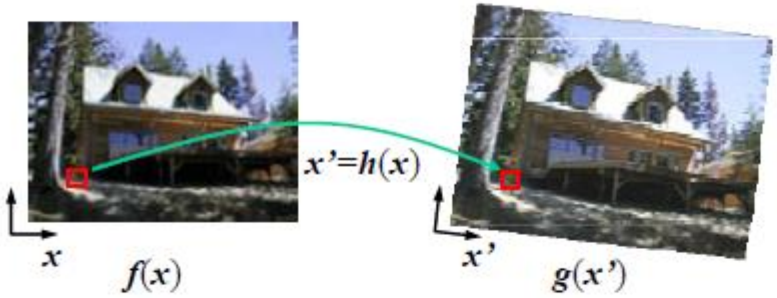
- Image warping involves modifying the **domain** of an image function rather than its range.





Geometric Transformations

- Forward warping algorithm
  - a pixel f(x) is copied to its corresponding location x′=h(x) in image g(x′)

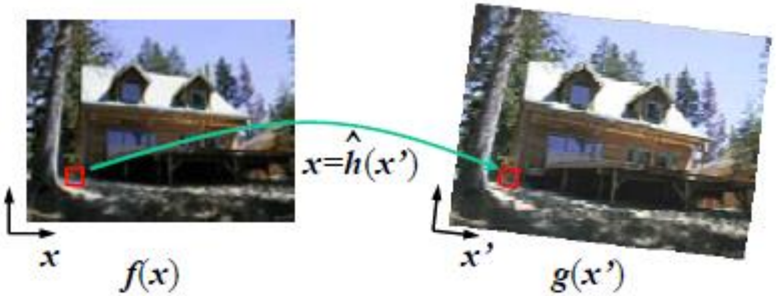


procedure *forwardWarp*$(f, \boldsymbol{h}, \textbf{out } g)$:

For every pixel x in $f(x)$

1. Compute the destination location $x' = \boldsymbol{h}(x)$.
2. Copy the pixel $f(x)$ to $g(x')$.

- Inverse warping algorithm
  - creating an image g(x′) from an image f(x) using the parametric transform x′=h(x)



procedure *inverseWarp*$(f, \boldsymbol{h}, \textbf{out } g)$:

For every pixel x' in $g(x')$

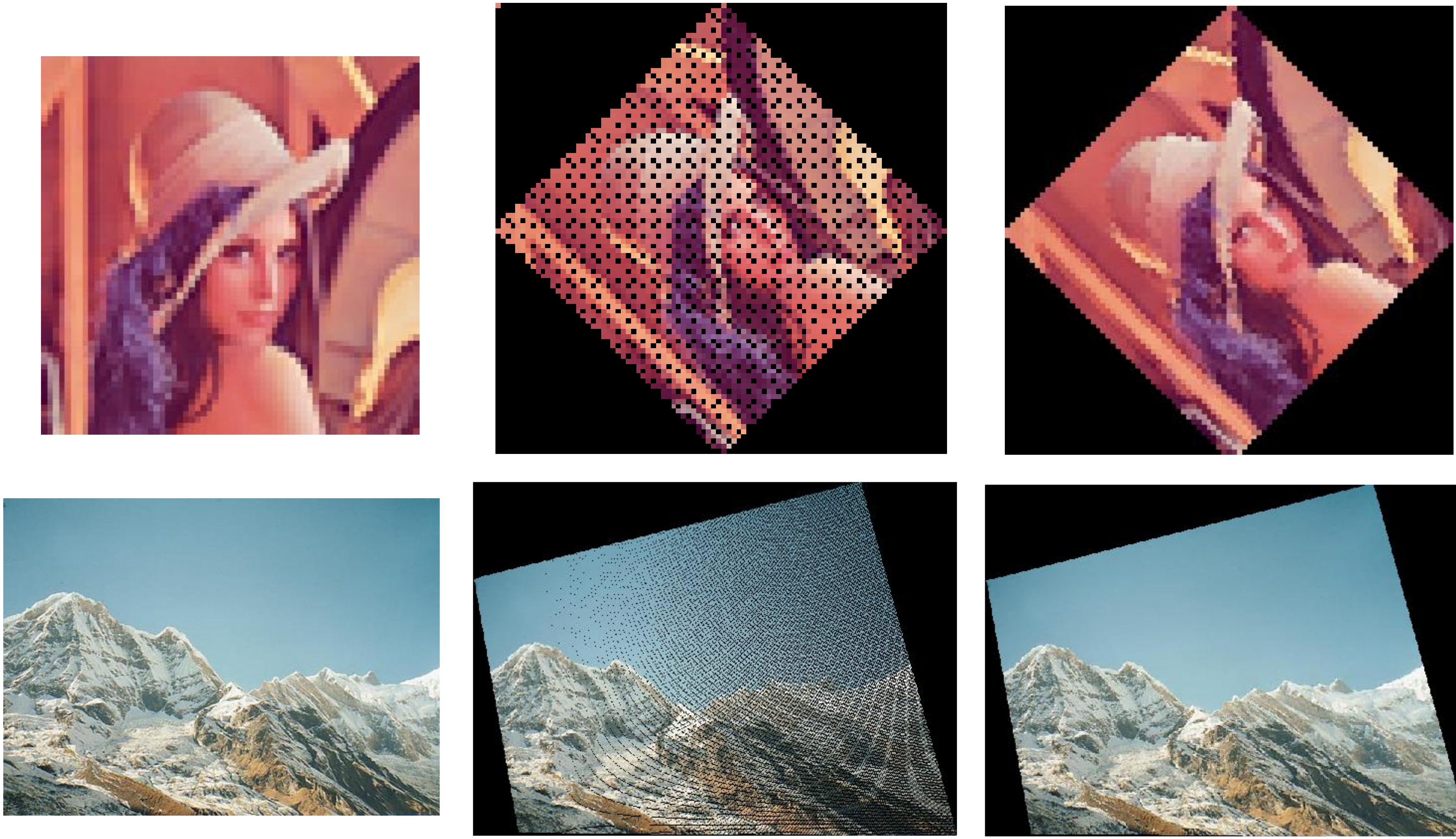
1. Compute the source location $x = \hat{\boldsymbol{h}}(x')$
2. Resample $f(x)$ at location x and copy to $g(x')$

Geometric Transformations

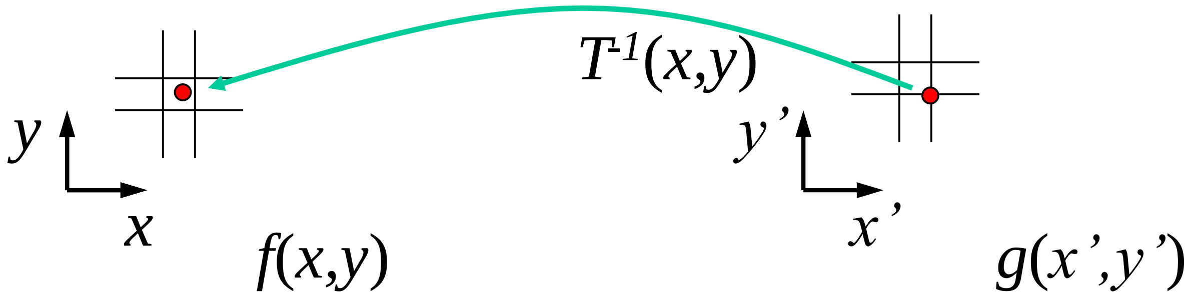
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\Theta & -\sin\Theta \\ \sin\Theta & \cos\Theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

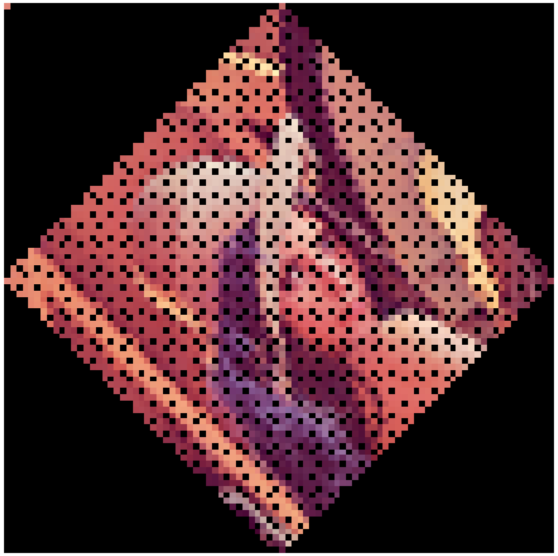
Forward warping

Sol 1: Inverse warping



Geometric Transformations





Get each pixel $g(x',y')$ from its corresponding location $(x,y) = T^{-1}(x',y')$ in the first image

Q: what if pixel comes from "between" two pixels?

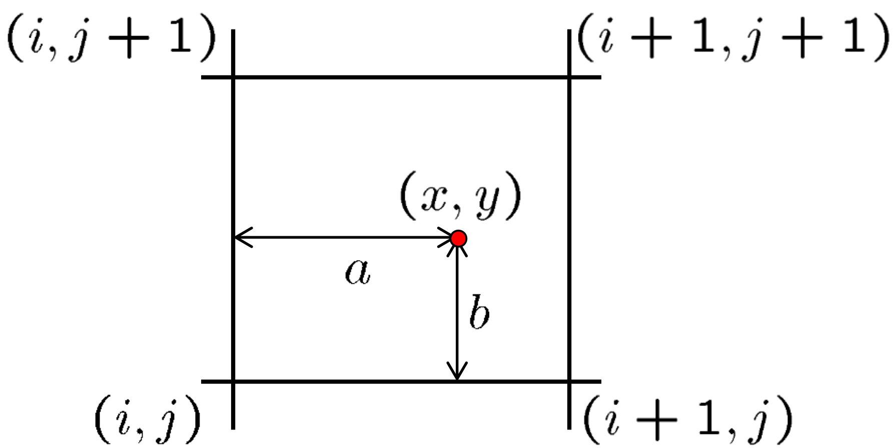
Sol 2: *Interpolate* color value from neighbors

- nearest neighbor, bilinear…

Geometric Transformations

- Bilinear interpolation

  Sampling at *f(x,y):*



$$\begin{aligned} f(x,y) = \quad & (1-a)(1-b) && f[i,j] \\ & +a(1-b) && f[i+1,j] \\ & +ab && f[i+1,j+1] \\ & +(1-a)b && f[i,j+1] \end{aligned}$$

Geometric Transformations

- Translation

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

- Rotation + translation $x' = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \end{bmatrix} \bar{x} \quad \boldsymbol{R} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$

- Scaled rotation (similarity transform)

$$x' = \begin{bmatrix} s\boldsymbol{R} & \boldsymbol{t} \end{bmatrix} \bar{x} = \begin{bmatrix} a & -b & t_x \\ b & a & t_y \end{bmatrix} \bar{x},$$

- Affine $x' = \begin{bmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \end{bmatrix} \bar{x}.$

- Projective (homography)

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

Geometric Transformations

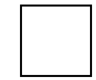
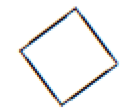
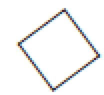
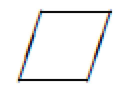
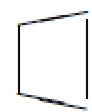
Transformation	Matrix	# DoF	Preserves	Icon
translation	$\begin{bmatrix} \boldsymbol{I} \mid \boldsymbol{t} \end{bmatrix}_{2\times3}$	2	orientation	
rigid (Euclidean)	$\begin{bmatrix} \boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{2\times3}$	3	lengths	
similarity	$\begin{bmatrix} s\boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{2\times3}$	4	angles	
affine	$\begin{bmatrix} \boldsymbol{A} \end{bmatrix}_{2\times3}$	6	parallelism	
projective	$\begin{bmatrix} \tilde{\boldsymbol{H}} \end{bmatrix}_{3\times3}$	8	straight lines	

Table 2.1 Hierarchy of 2D coordinate transformations. 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.

2D Transformations

- Basic 2D transformations as 3x3 matrices

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

Translate

$$\begin{bmatrix} \boldsymbol{x}' \\ \boldsymbol{y}' \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{s}_x & 0 & 0 \\ 0 & \boldsymbol{s}_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \\ 1 \end{bmatrix}$$

Scale

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

Rotate

$$\begin{bmatrix} \boldsymbol{x}' \\ \boldsymbol{y}' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & \boldsymbol{sh}_x & 0 \\ \boldsymbol{sh}_y & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \\ 1 \end{bmatrix}$$

Shear

2D Transformations

- 2D Affine transformations
  - Combinations of linear transformations and translations
  - Parallel lines remain parallel

$$\begin{bmatrix} x' \\ y' \\ w \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

- Projective transformations
  - Affine transformations and projective warps
  - Parallel lines do not necessarily remain parallel

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$