Geometric Transforms

김성영교수
금오공과대학교
컴퓨터공학과

학습 내용

- Geometric Transforms 개요
- Spatial transform
- Gray-level interpolation

기하학적 변환

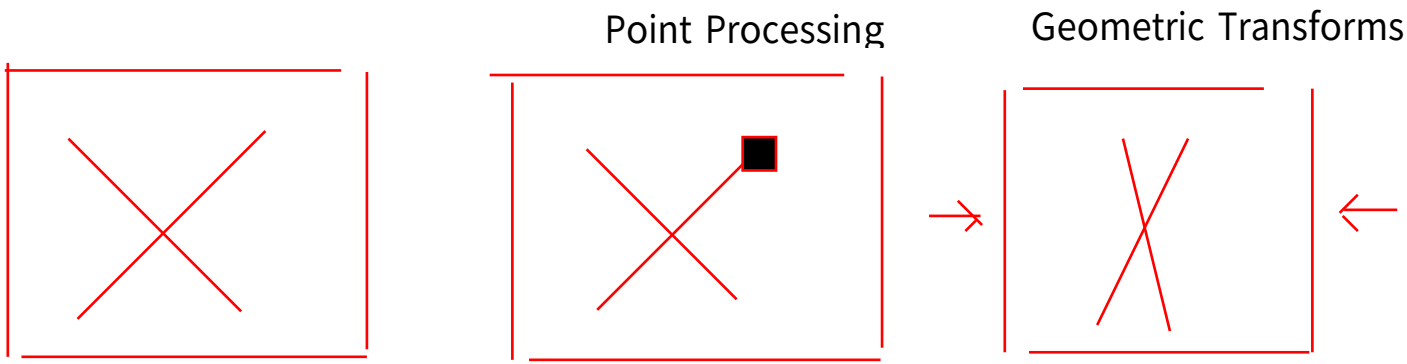
공간 변환

cv2.resize(img,(),interpolation = cv2.

Geometric Transforms 개요

- 수식이나 변환 관계에 의해 픽셀들의 위치를 변경하는 변환
- 두 단계의 처리 단계로 구성
  - ① **mapping by spatial transform**
  - ② **gray-level interpolation**

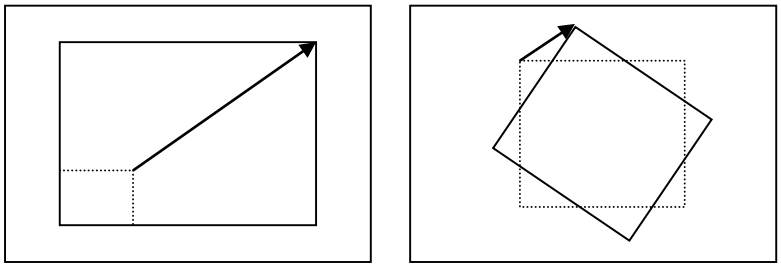
$$\mathbf{D}(x,y) = \mathbf{I}(T_x(x,y), T_y(x,y))$$



Spatial Transform

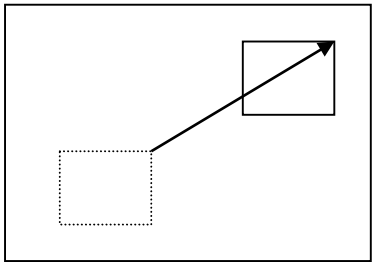
Map the input image location to a location in the output image

$$x' = T_x(x, y), \qquad y' = T_y(x, y)$$

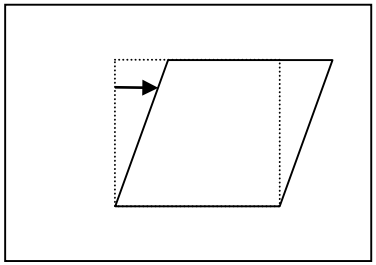


Scaling

Rotation



Translation



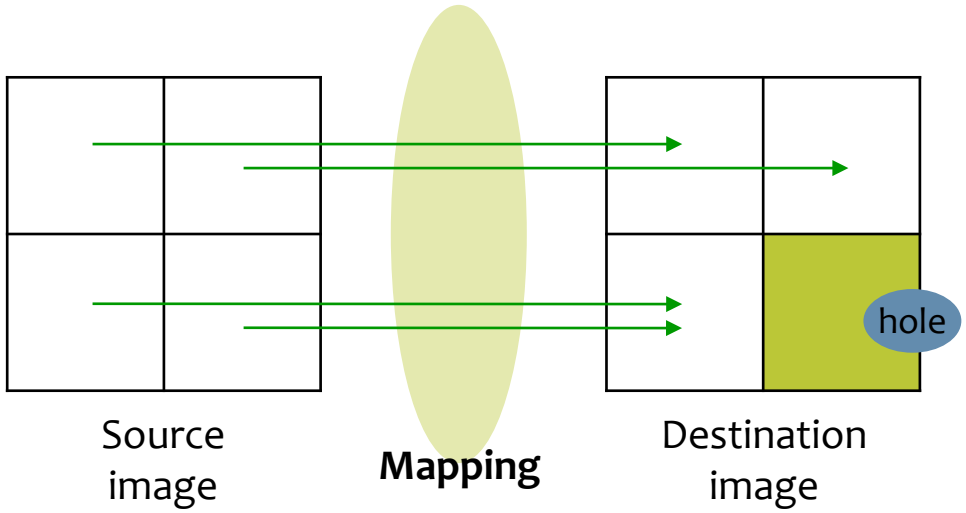
Skew

shear

Affine transformation

forward mapping

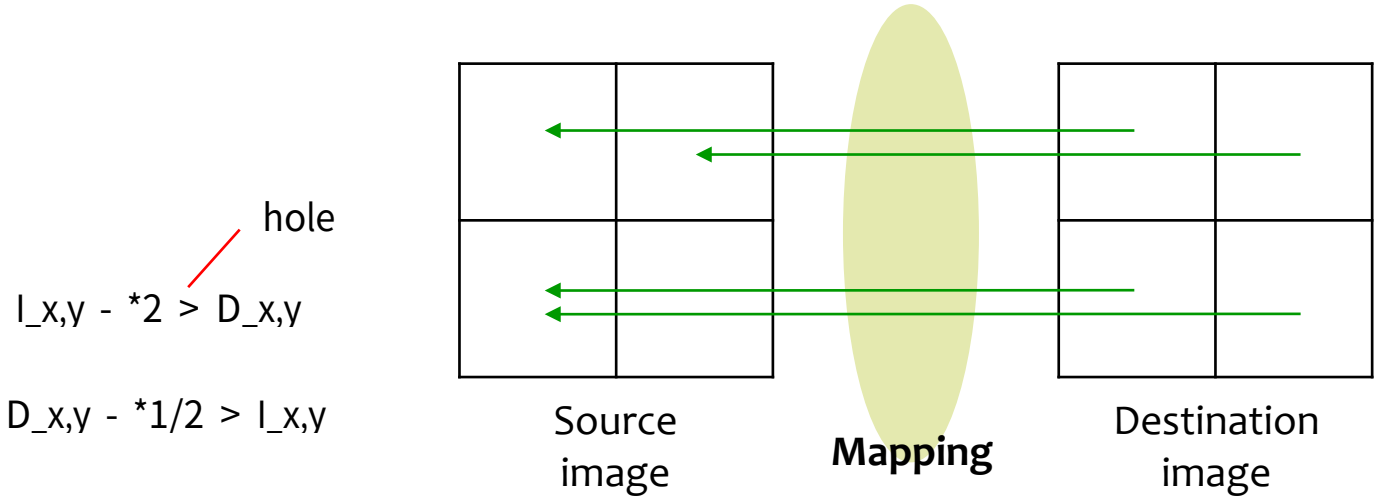
- 변환 수식에 의해 **입력좌표**를 **출력좌표**로 변환하는 과정
- 출력 영상에서 정의되지 않은 픽셀(hole) 발생



I(x, y) - > 2* I(x, y)

backward mapping

- 출력 영상의 각 픽셀 좌표에 대응하는 원본 영상의 좌표를 계산하여 해당 픽셀의 밝기 값을 결정하는 방법
- 출력 영상에서 정의되지 않은 픽셀 발생 방지
- 계산된 좌표가 정수가 아닌 경우 발생 → interpolation 적용



forward vs. backward mapping

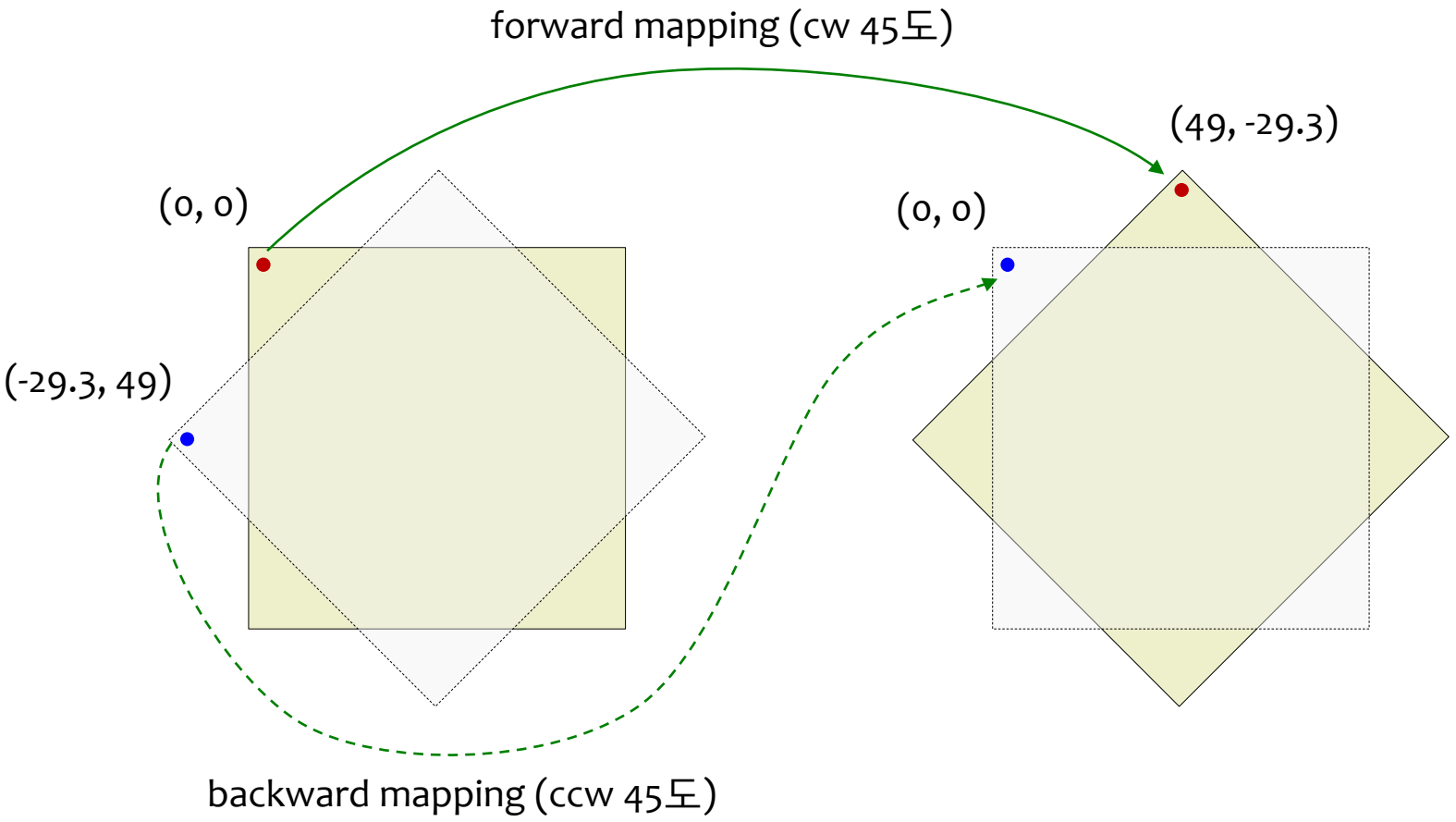


image size: 99 x 99, CW 45도 회전

Affine transform

Rigid - body transform (shearing , skew가 빠짐)

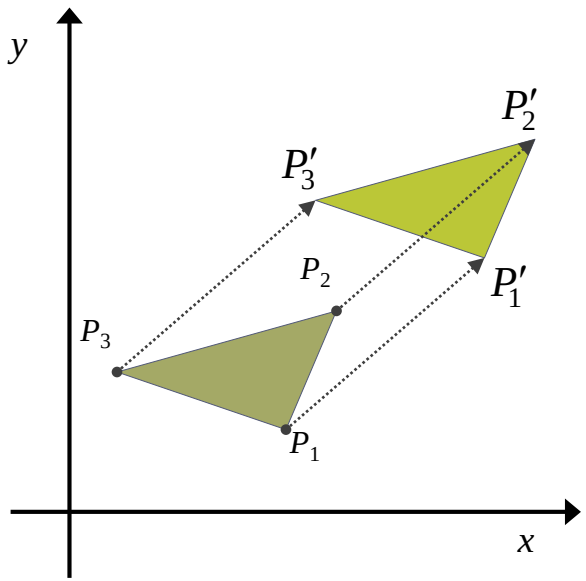
- Linear transform
- 휘어짐이 없고 평행한 선들은 평행을 유지하는 변환
- 이동, 회전, 스케일 및 이들의 조합에 의한 변환
- $x' = a_0x + a_1y + a_2,\ y' = b_0x + b_1y + b_2$
- homogeneous coordinate system 사용

동차 좌표계

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a_0 & a_1 \\ b_0 & b_1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} a_2 \\ b_2 \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a_o & a_1 & a_2 \\ b_o & b_1 & b_2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

translation



forward mapping

$$\begin{cases} x' = x + T_x \\ y' = y + T_y \end{cases} \qquad \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & T_x \\ 0 & 1 & T_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

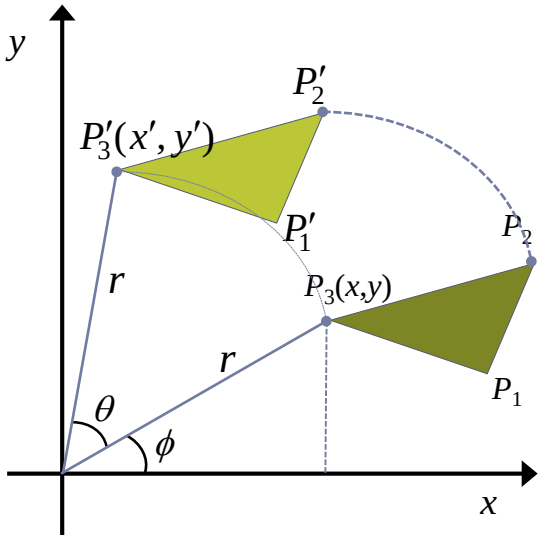
backward mapping

$$\begin{cases} x = x' - T_x \\ y = y' - T_y \end{cases} \qquad \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & -T_x \\ 0 & 1 & -T_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}$$

rotation

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

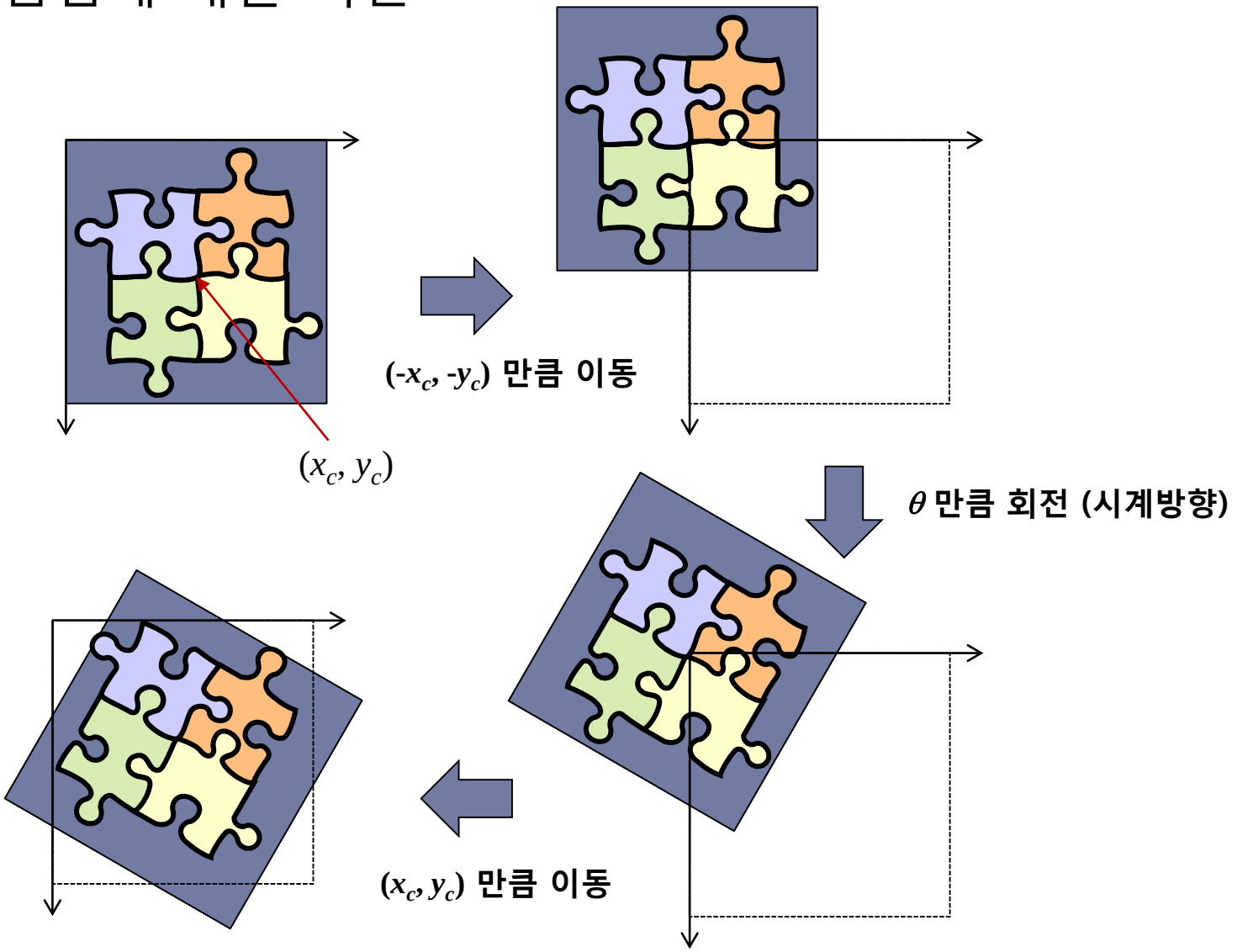
$$x = r\cdot\cos\phi,\quad y = r\cdot\sin\phi$$



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

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

영상 중심점에 대한 회전



임의 점 (x, y)에 대한 회전

1) 임의의 점을 원점으로 평행 위치 이동

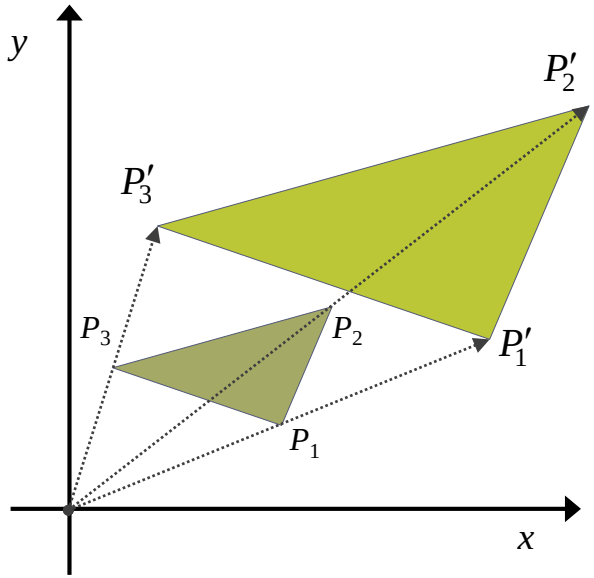
2) 원점을 중심으로 회전

3) 단계 1)에서 수행한 이동의 반대 방향으로 평행 위치 이동

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

$$= \begin{bmatrix} \cos\theta & -\sin\theta & T_x(1-\cos\theta)+T_y\sin\theta \\ \sin\theta & \cos\theta & T_y(1-\cos\theta)-T_x\sin\theta \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

scaling



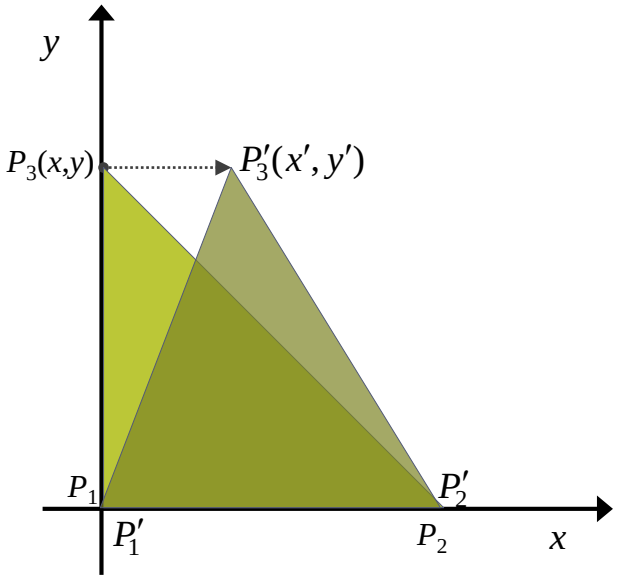
forward mapping

$$\begin{cases} x' = S_x \cdot x \\ y' = S_y \cdot y \end{cases} \qquad \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

backward mapping

$$\begin{cases} x = S_x^{-1} \cdot x' \\ y = S_y^{-1} \cdot y' \end{cases} \qquad \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} S_x^{-1} & 0 & 0 \\ 0 & S_y^{-1} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}$$

skew



forward mapping (about x)

$$\begin{cases} x' = x + u \cdot y \\ y' = y \end{cases} \qquad \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & u & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

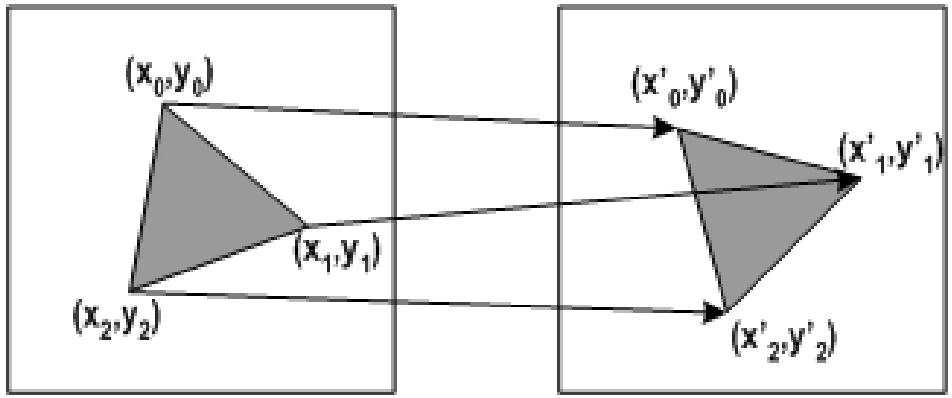
backward mapping (about x)

$$\begin{cases} x = x' - u \cdot y' \\ y = y' \end{cases} \qquad \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & -u & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}$$

To calculate an affine transform

- 6개의 미지수를 결정해야 함
- 3개의 좌표 점이 필요

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a_o & a_1 & a_2 \\ b_o & b_1 & b_2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$



Warping

- Nonlinear transform (rubber sheet transform)
- pixel별로 이동 정도를 다르게 할 수 있어서 고무판 위에 그려진 영상을 임의대로 구부리는 것과 같은 효과를 낼 수 있음
- 고차항을 사용하여 일반화된 다항식으로 표현

$$x' = \sum_{i=0}^{N} \sum_{j=0}^{N-i} a_{ij} x^i y^j$$

$$y' = \sum_{i=0}^{N} \sum_{j=0}^{N-i} b_{ij} x^i y^j$$

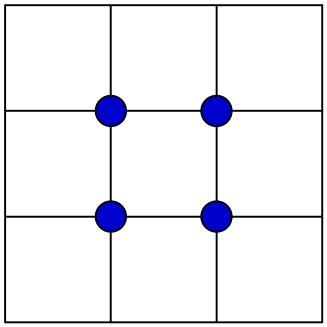
1st order warping

$$x' = \sum_{i=0}^{N} \sum_{j=0}^{N-i} a_{ij} x^i y^j$$

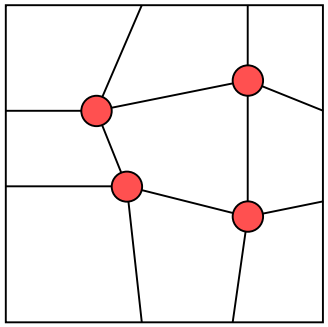
$$y' = \sum_{i=0}^{N} \sum_{j=0}^{N-i} b_{ij} x^i y^j$$

$$x' = a_0 x + a_1 y + a_2 xy + a_3,$$

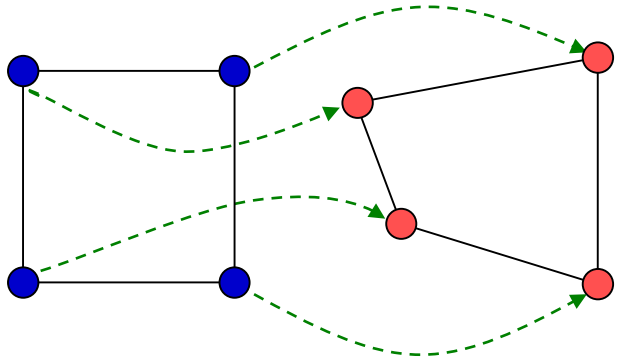
$$y' = b_0 x + b_1 y + b_2 xy + b_3$$

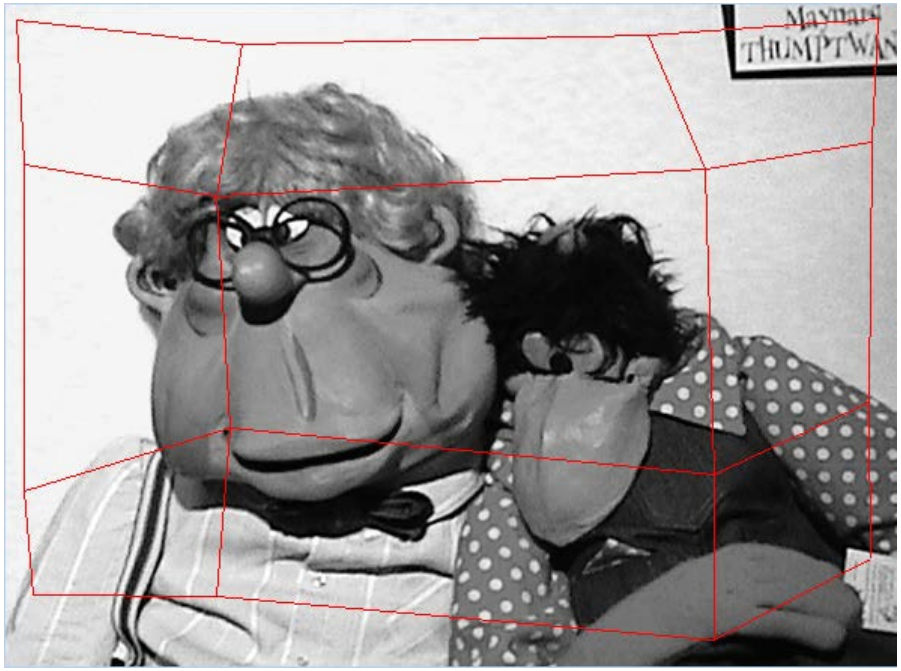


$d(x', y')$



$I(x, y)$





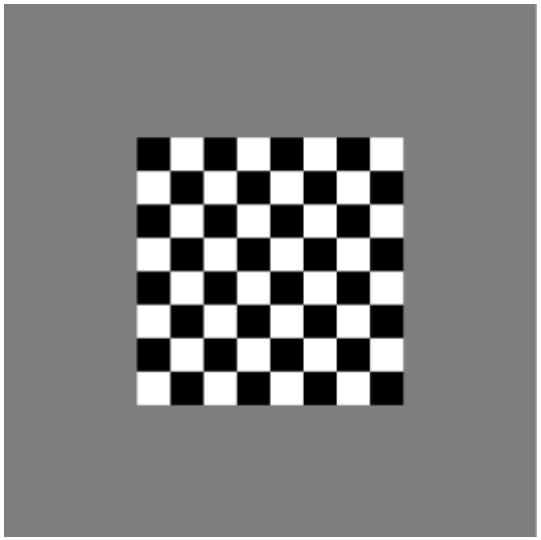
2nd order warping

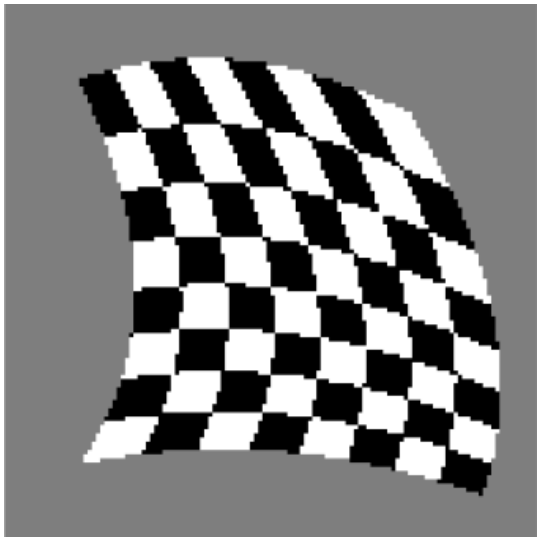
$$x' = \sum_{i=0}^{N} \sum_{j=0}^{N-i} a_{ij} x^i y^j$$

$$y' = \sum_{i=0}^{N} \sum_{j=0}^{N-i} b_{ij} x^i y^j$$

$$x' = a_0x^2 + a_1y^2 + a_2xy + a_3x + a_4y + a_5,$$

$$y' = b_0x^2 + b_1y^2 + b_2xy + b_3x + b_4y + b_5$$



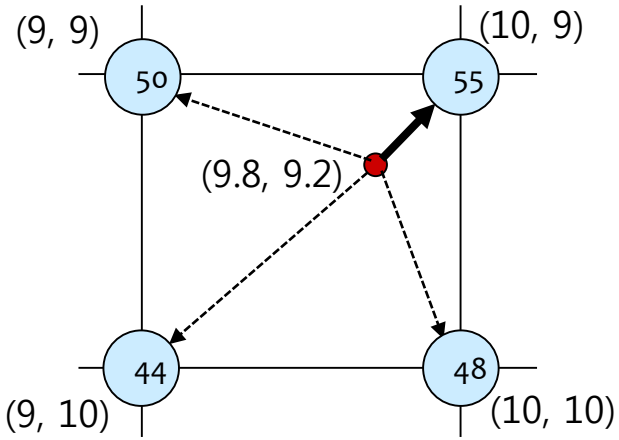


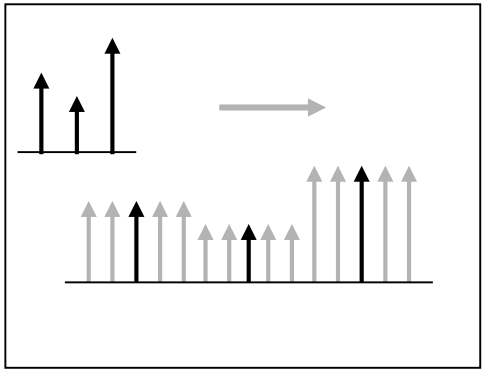
Interpolation

- Interpolation 이란?
  - 결과 픽셀에 정확하게 대응되는 입력 픽셀이 없는 경우 주변 픽셀들을 고려하여 새로운 값을 생성하는 방법
- Interpolation 종류
  - Nearest neighbor interpolation
  - Neighbor averaging interpolation
  - Bilinear interpolation
  - Higher order interpolation

Nearest neighbor interpolation

- 계산한 위치에서 가장 가까운 원시 픽셀을 선택하는 방법
- (e.g.) $I(10,11)$ ➔ $d(9.8,9.2) \approx d(10,9)=55$
- 처리 속도는 빠르지만 결과 영상의 질이 좋지 않음

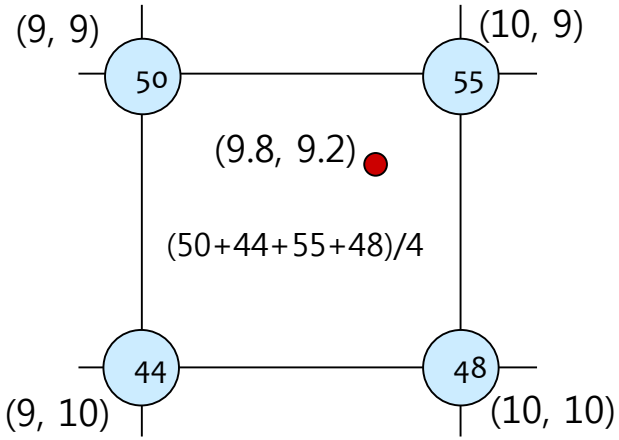


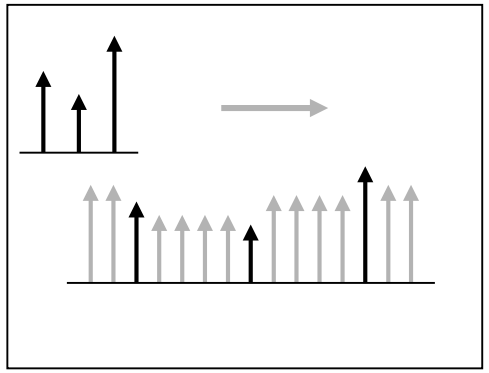


4배 확대

Neighbor averaging interpolation

- Find a neighborhood average
- Two dimensionally using all four neighbors
- More computationally intensive but provide more visually pleasing result
- (e.g.) $I(10,11)$ ➔ $d(9.8,9.2) = (50+44+55+48)/4 \approx 49$

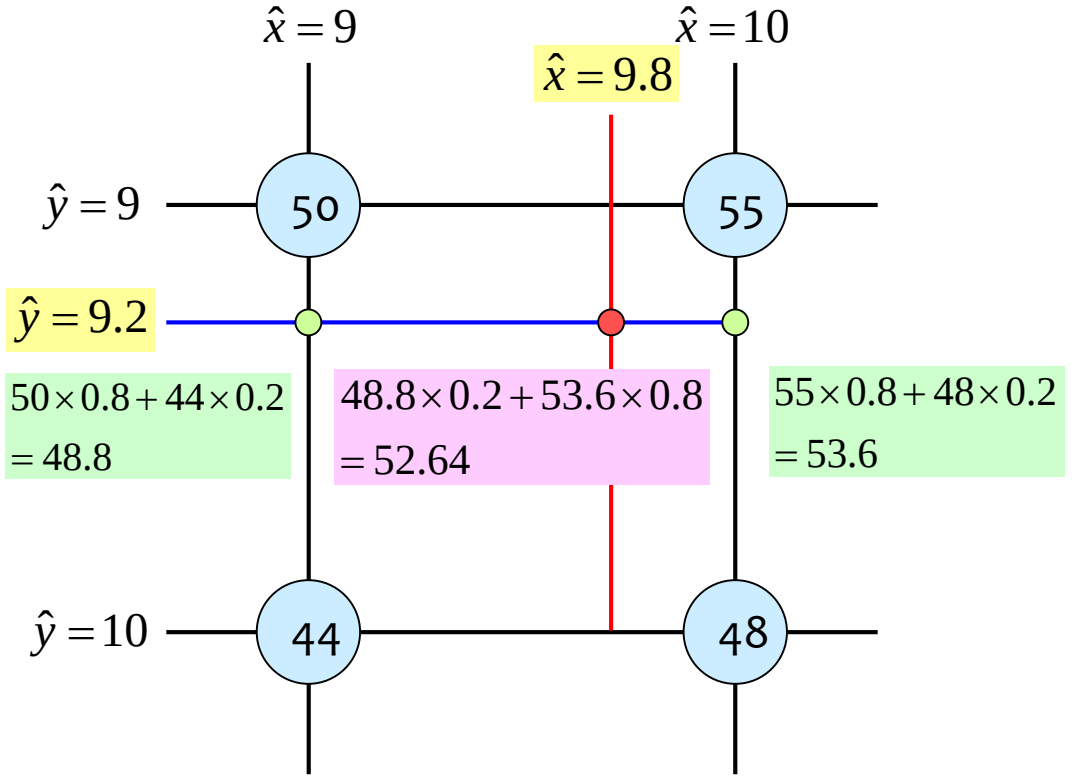


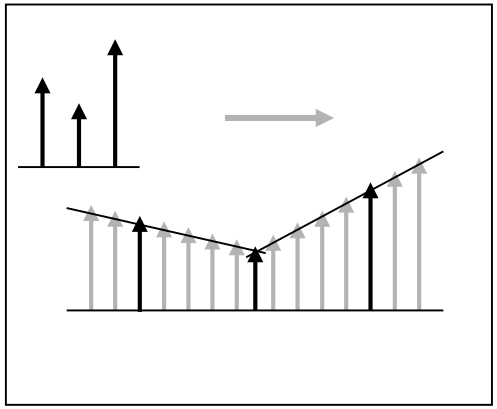


4배 확대

Bilinear interpolation

- 새로운 픽셀을 생성하기 위해 네 개의 가장 가까운 픽셀들에 가중치를 곱한 값들의 합을 사용
- 보다 자연스러운 영상을 산출





4배 확대

Nearest neighbor

Bilinear

요약

- Geometric Transforms
  - 수식이나 변환 관계에 의해 픽셀들의 위치를 변경하는 변환
- Mapping by spatial transform
  - 방식: forward 및 backward mapping
  - 종류: Affine transform 및 Warping (Perspective transform)
- Gray-level interpolation
  - Nearest neighbor interpolation
  - Neighbor averaging interpolation
  - Bilinear interpolation

Reference

- R. Gonzalez, R. Woods, **Digital Image Processing (2nd Edition),** Prentice Hall, 2002
- Scott E Umbaugh, **Computer Imaging**, CRC Press, 2005