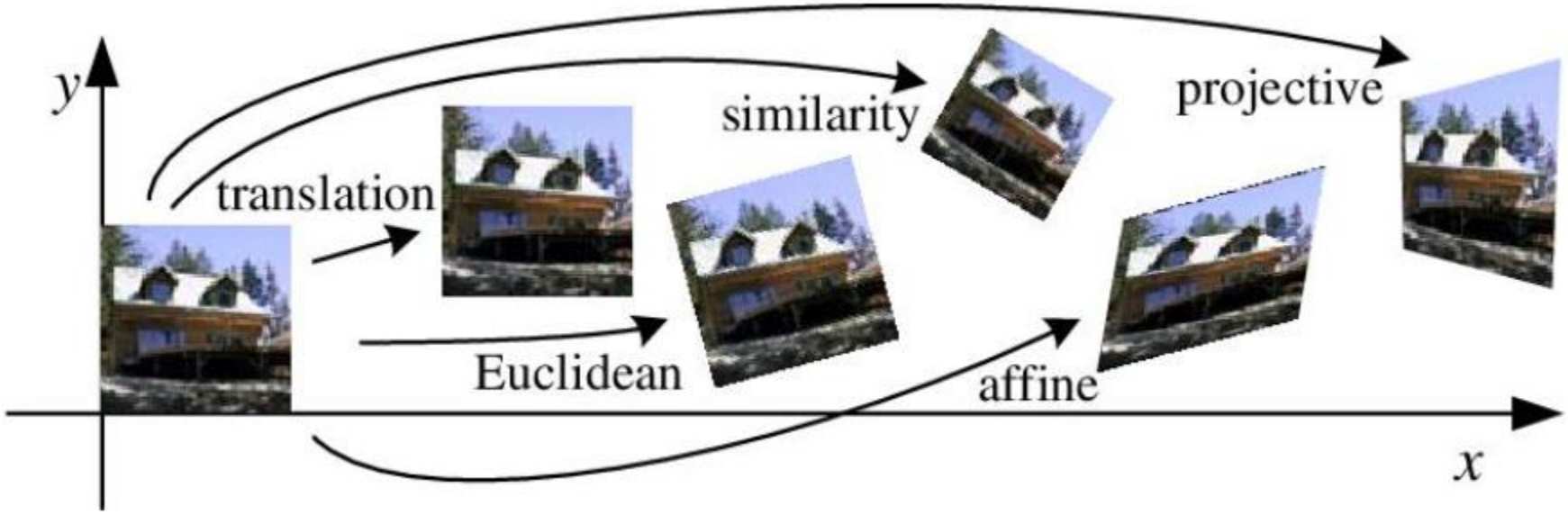
기하학적 변환

영상의 기하학적 변환

영상의 기하학적 변환 (geometric transformation) 이란?

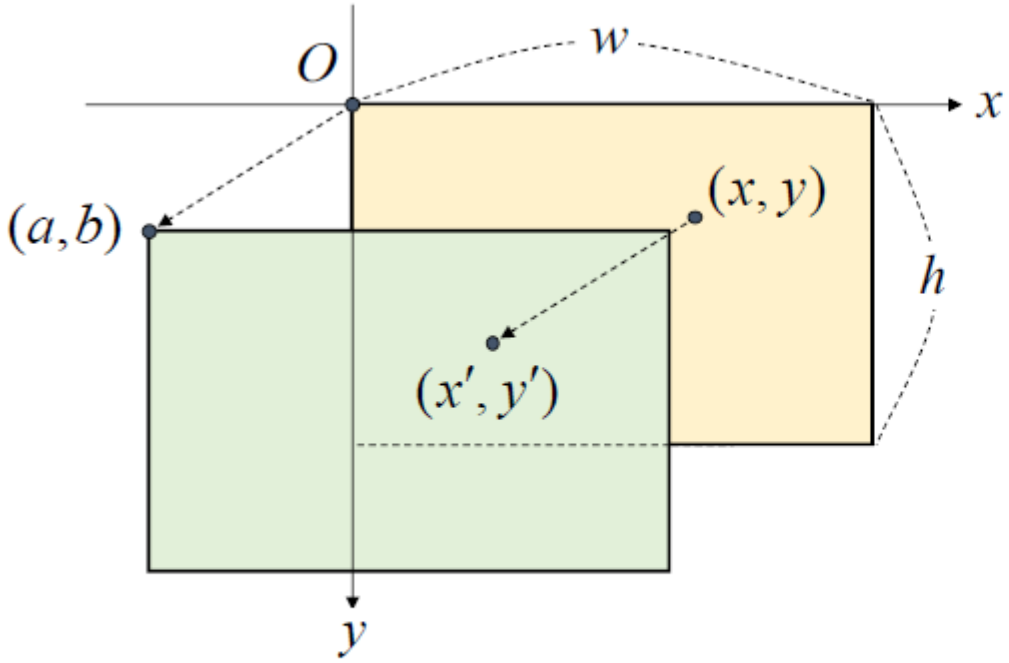
- 영상을 구성하는 픽셀의 배치 구조를 변경함으로써 전체 영상의 모양을 바꾸는 작업
- Image registration, removal of geometric distortion, etc.



영상의 이동 변환

이동 변환(Translation transformation)

- 가로 또는 세로 방향으로 영상을 특정 크기만큼 이동시키는 변환
- x축과 y축 방향으로의 이동 변위를 지정



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

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

2x3 어파인 변환 행렬

영상의 이동 변환

영상의 어파인 변환 함수

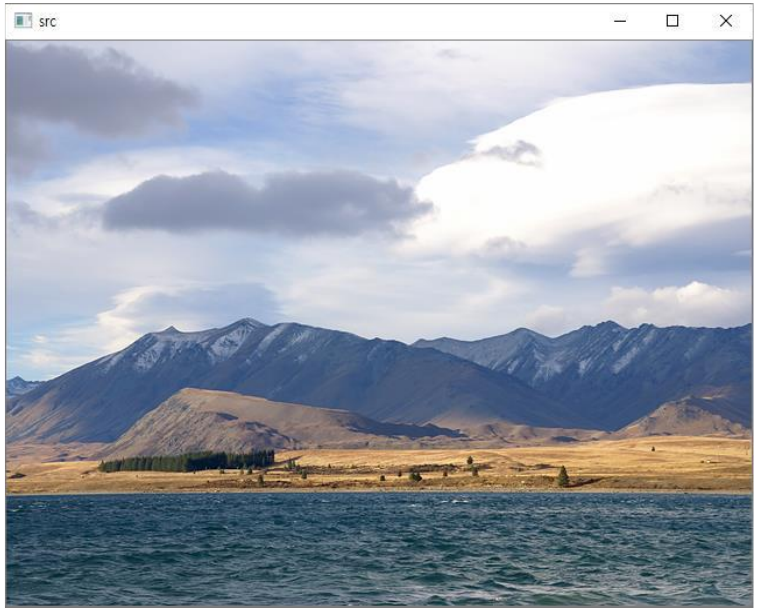
```
cv2.warpAffine(src, M, dsize, dst=None, flags=None, borderMode=None, borderValue=None ) --> dst
```

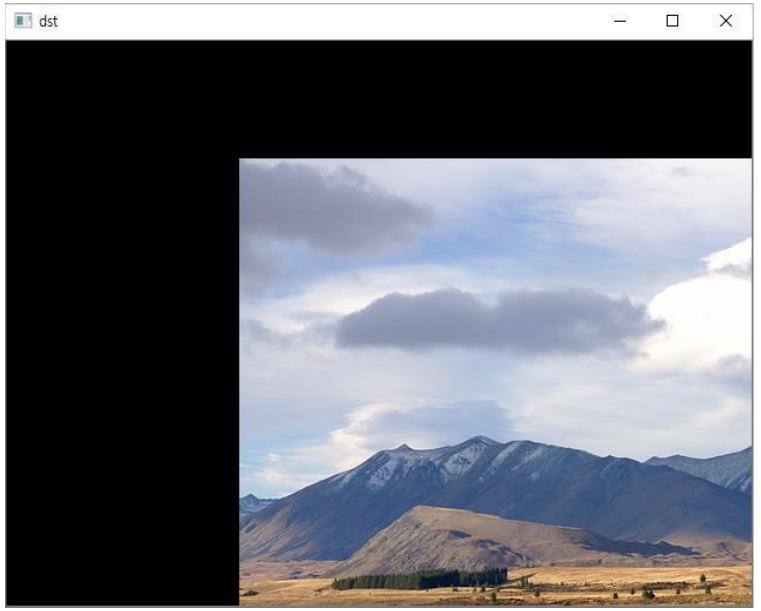
- src: 입력 영상
- M: 2x3 어파인 변환 행렬. 실수형
- dsize: 결과 영상 크기. (w, h) 튜플. (0, 0)이면 src 와 같은 크기로 설정
- dst: 출력 영상
- flags: 보간법. 기본값은 cv2.INTER_LINEAR.
- borderMode: 가장자리 픽셀 확장 방식. 기본값은 cv2.BORDER_CONSTANT.
- borderValue: cv2.BORDER_CONSTANT 일 때 사용할 상수 값. 기본값은 0.

영상의 이동 변환

영상의 이동 변환 예제

```python
import sys
import numpy as np
import cv2


src = cv2.imread('tekapo.bmp')

if src is None:
    print('Image load failed!')
    sys.exit()

aff = np.array( object: [[1, 0, 200],
                [0, 1, 100]], dtype=np.float32)

dst = cv2.warpAffine(src, aff, dsize: (0, 0))

cv2.imshow( winname: 'src', src)
cv2.imshow( winname: 'dst', dst)
cv2.waitKey()
cv2.destroyAllWindows()
```

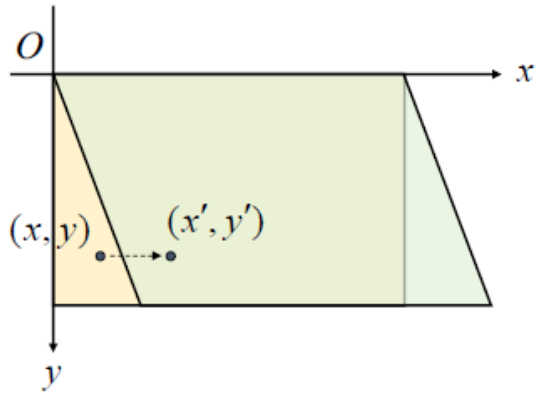




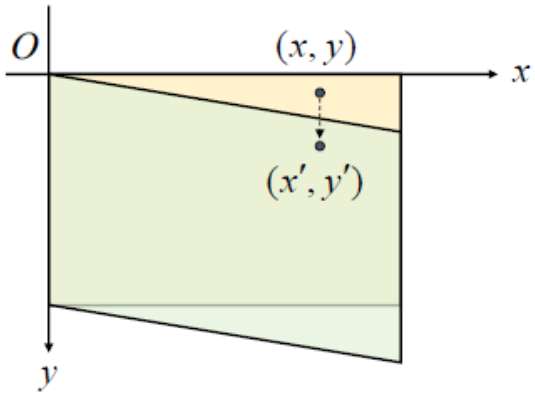
영상의 전단 변환

전단 변환 (Shear transformation)

- 층 밀림 변환. x축과 y축 방향에 대해 따로 정의.



$$\begin{cases} x' = x + my \\ y' = y \end{cases} \quad \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & m & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

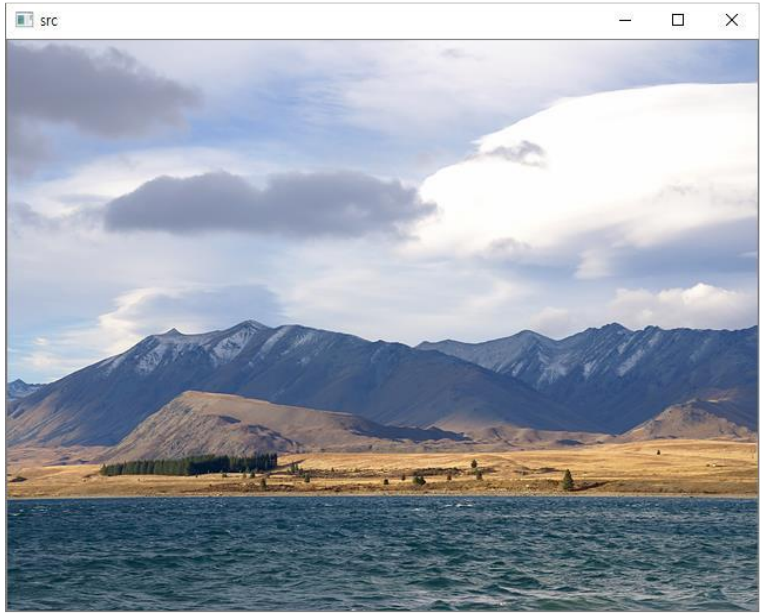


$$\begin{cases} x' = x \\ y' = mx + y \end{cases} \quad \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ m & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

영상의 전단 변환

영상의 전단 변환 예제

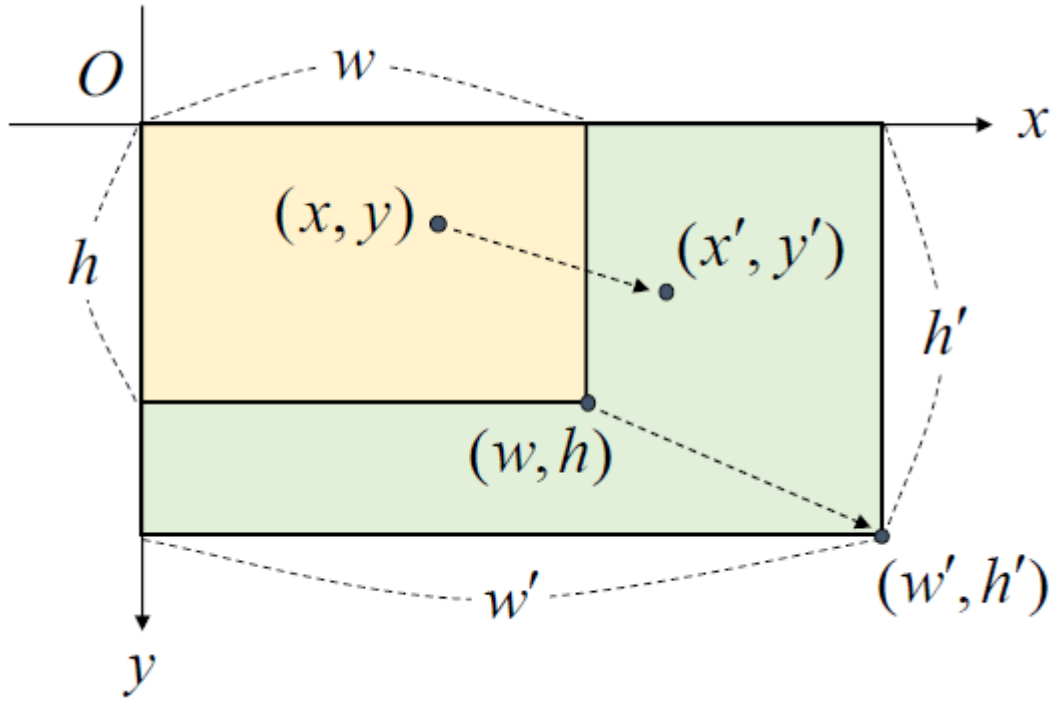
```python
import sys
import numpy as np
import cv2

src = cv2.imread('tekapo.bmp')

if src is None:
    print('Image load failed!')
    sys.exit()

aff = np.array( object: [[1, 0.5, 0],
                         [0, 1, 0]], dtype=np.float32)

h, w = src.shape[:2]
dst = cv2.warpAffine(src, aff,  dsize: (w + int(h * 0.5), h))

cv2.imshow( winname: 'src', src)
cv2.imshow( winname: 'dst', dst)
cv2.waitKey()
cv2.destroyAllWindows()
```



영상의 확대와 축소

크기 변환(Scale transformation)

- 영상의 크기를 원본 영상보다 크게 또는 작게 만드는 변환
- x축과 y축 방향으로의 스케일 비율(scale factor)를 지정



$$\begin{cases} x' = s_x x \\ y' = s_y y \end{cases} \qquad \begin{cases} s_x = w'/w \\ s_y = h'/h \end{cases}$$

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

영상의 확대와 축소

영상의 크기 변환

cv2.resize(src, dsize, dst=None, fx=None, fy=None, interpolation=None) --> dst

- src: 입력 영상
- dsize: 결과 영상 크기. (w, h) 튜플. (0, 0)이면 fx 와 fy 값을 이용하여 결정
- dst: 출력 영상
- fx, fy : x 와 y 방향 스케일 비율 (scale factor). dsize 값이 0 일 때 유효
- interpolation: 보간법 지정. 기본값은 cv2.INTER_LINEAR

cv2.INTER_NEAREST	최근방 이웃 보간법
cv2.INTER_LINEAR	양선형 보간법 (2x2 이웃 픽셀 참조)
cv2.INTER_CUBIC	3차회선 보간법 (4x4 이웃 픽셀 참조)
cv2.INTER_LANCZOS4	Lanczos 보간법 (8x8 이웃 픽셀 참조)
cv2.INTER_AREA	영상 축소 시 효과적

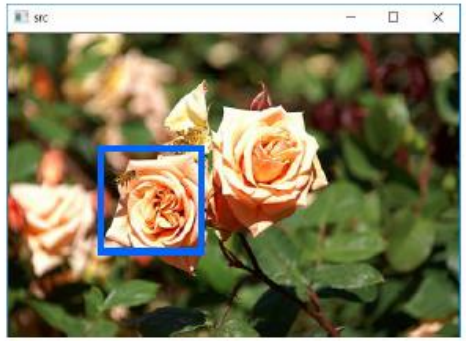
영상의 확대와 축소

영상의 크기 변환 예제

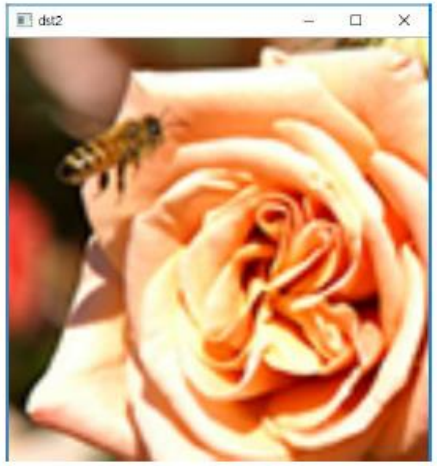
```python
import sys
import numpy as np
import cv2

src = cv2.imread('rose.bmp') # src.shape=(320, 480)

if src is None:
    print('Image load failed!')
    sys.exit()

dst1 = cv2.resize(src, dsize=(0, 0), fx=4, fy=4, interpolation=cv2.INTER_NEAREST)
dst2 = cv2.resize(src, dsize=(1920, 1280))  # cv2.INTER_LINEAR
dst3 = cv2.resize(src, dsize=(1920, 1280), interpolation=cv2.INTER_CUBIC)
dst4 = cv2.resize(src, dsize=(1920, 1280), interpolation=cv2.INTER_LANCZOS4)

cv2.imshow(winname='src', src)
cv2.imshow(winname='dst1', dst1[500:900, 400:800])
cv2.imshow(winname='dst2', dst2[500:900, 400:800])
cv2.imshow(winname='dst3', dst3[500:900, 400:800])
cv2.imshow(winname='dst4', dst4[500:900, 400:800])
cv2.waitKey()
cv2.destroyAllWindows()
```

영상의 확대와 축소

영상의 크기 변환 예제 결과

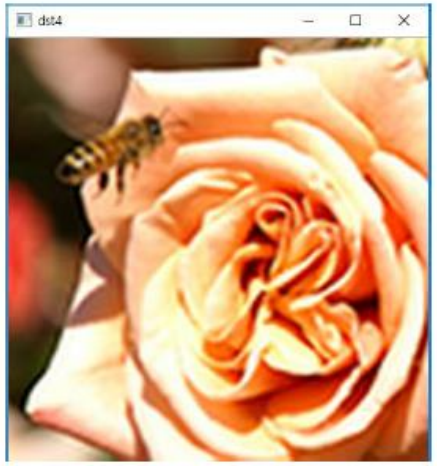


cv2.INTER_NEAREST
(by one neighbors)



cv2.INTER_LINEAR
(by 2x2 neighbors)

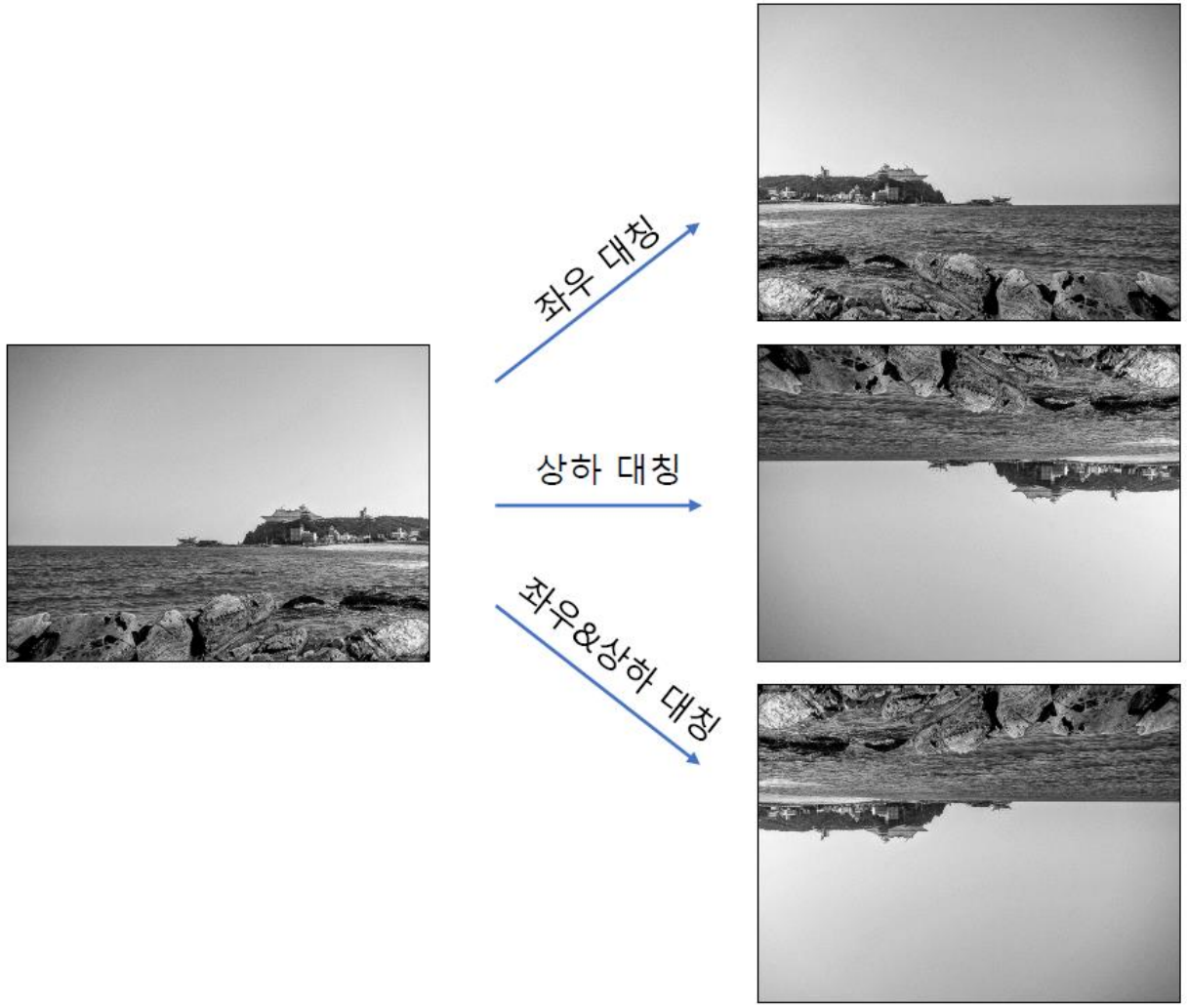
cv2.INTER_CUBIC
(by 4x4 neighbors)



cv2.INTER_LANCZOS4
(by 8x8 neighbors)

영상의 대칭

영상의 대칭 변환 (flip, reflection)



영상의 대칭

영상의 대칭 변환

cv2.flip(src, flipCode, dst=None) --> dst

- src: 입력 영상
- flipCode: 대칭 방향 지정

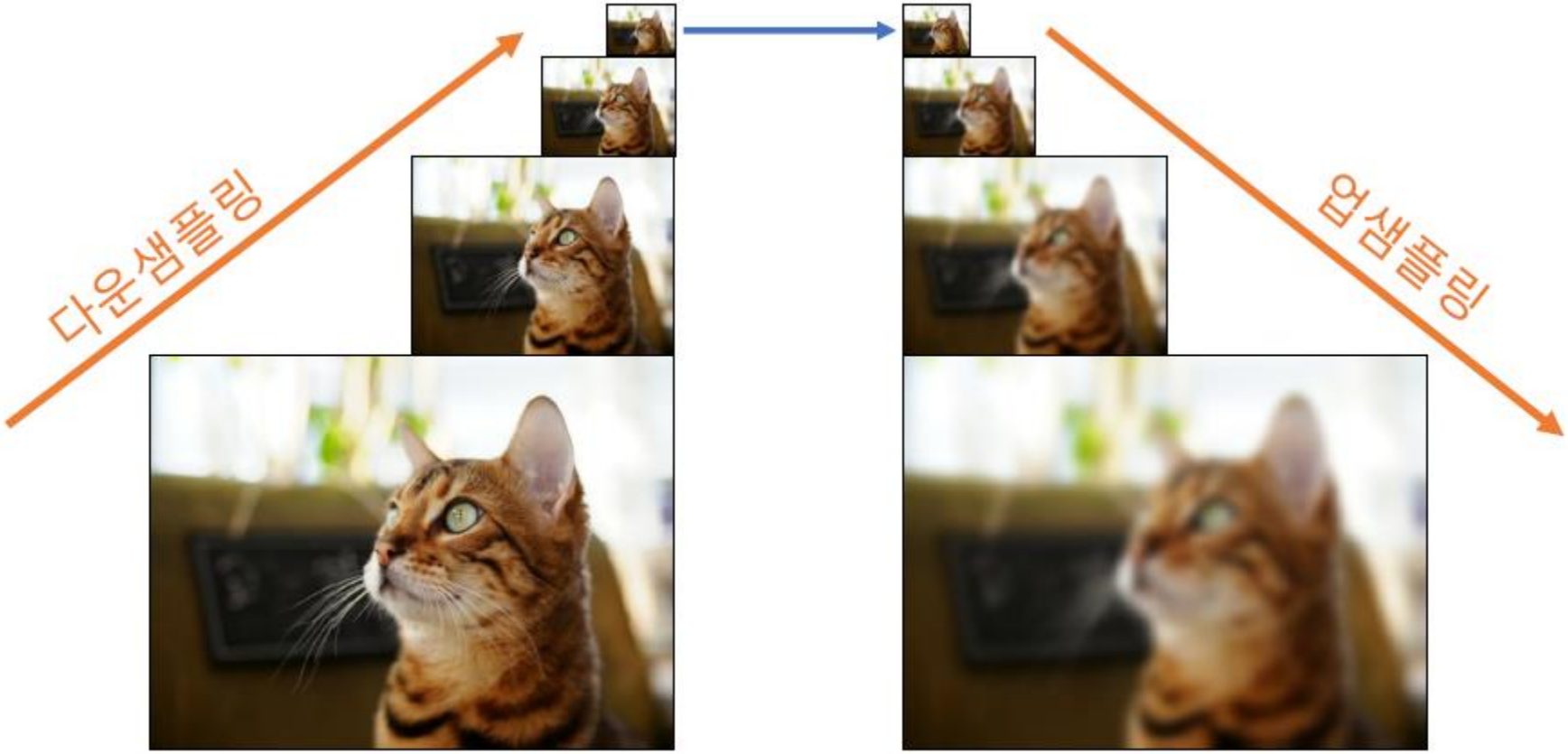
양수 (+1)	좌우 대칭
0	상하 대칭
음수 (-1)	좌우 & 상하 대칭

- dst 출력 영상

이미지 피라미드

이미지 피라미드 (Image pyramid)란?

- 하나의 영상에 대해 다양한 해상도의 영상 세트를 구성한 것
- 보통 가우시안 블러링 & 다운샘플링 형태로 축소하여 구성



이미지 피라미드

영상 피라미드 다운샘플링

cv2.pyrDown(src, dst=None, dstsize=None, borderType=None)) --> dst

- src: 입력 영상
- dst: 출력 영상
- dstsize: 출력 영상 크기 .

  따로 지정하지 않으면 입력 영상의 가로 , 세로 크기의 1/2 로 설정
- borderType: 가장자리 픽셀 확장 방식
- 참고 사항
  - 먼저 5x5 크기의 가우시안 필터를 적용
  - 이후 짝수 행과 열을 제거하여 작은 크기의 영상을 생성

이미지 피라미드

영상 피라미드 업샘플링

cv2.pyrUp(src, dst=None, dstsize=None, borderType=None)) --> dst

- src: 입력 영상
- dst: 출력 영상
- dstsize: 출력 영상 크기 .

  따로 지정하지 않으면 입력 영상의 가로, 세로 크기의 2배로 설정
- borderType: 가장자리 픽셀 확장 방식

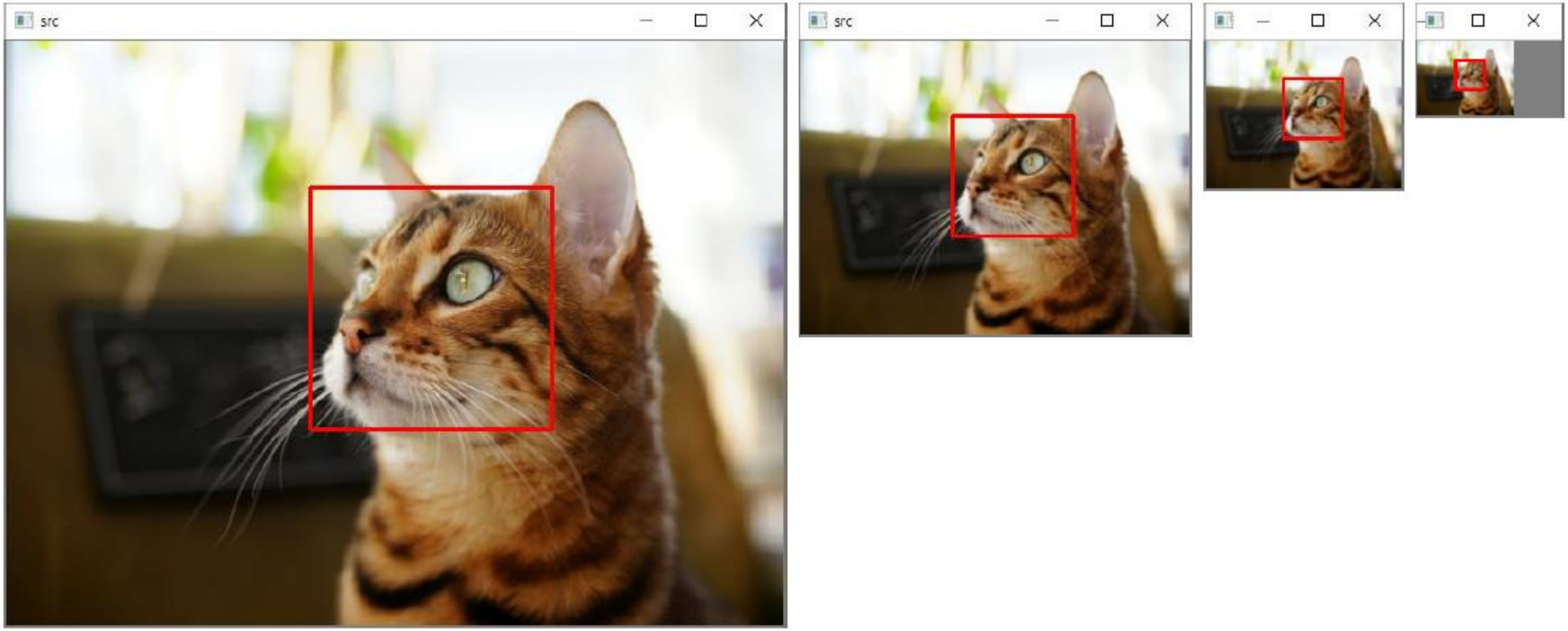
이미지 피라미드

피라미드 영상에 사각형 그리기 예제

```python
import sys
import numpy as np
import cv2

src = cv2.imread('cat.bmp')

if src is None:
    print('Image load failed!')
    sys.exit()

rc = (250, 120, 200, 200)  # rectangle tuple

# 원본 영상에 그리기
cpy = src.copy()
cv2.rectangle(cpy, rc, (0, 0, 255), 2)
cv2.imshow( winname: 'src', cpy)
cv2.waitKey()

# 피라미드 영상에 그리기
for i in range(1, 4):
    src = cv2.pyrDown(src)
    cpy = src.copy()
    cv2.rectangle(cpy, rc, (0, 0, 255), 2, shift=i)
    cv2.imshow( winname: 'src', cpy)
    cv2.waitKey()
    cv2.destroyWindow('src')

cv2.destroyAllWindows()
```

이미지 피라미드

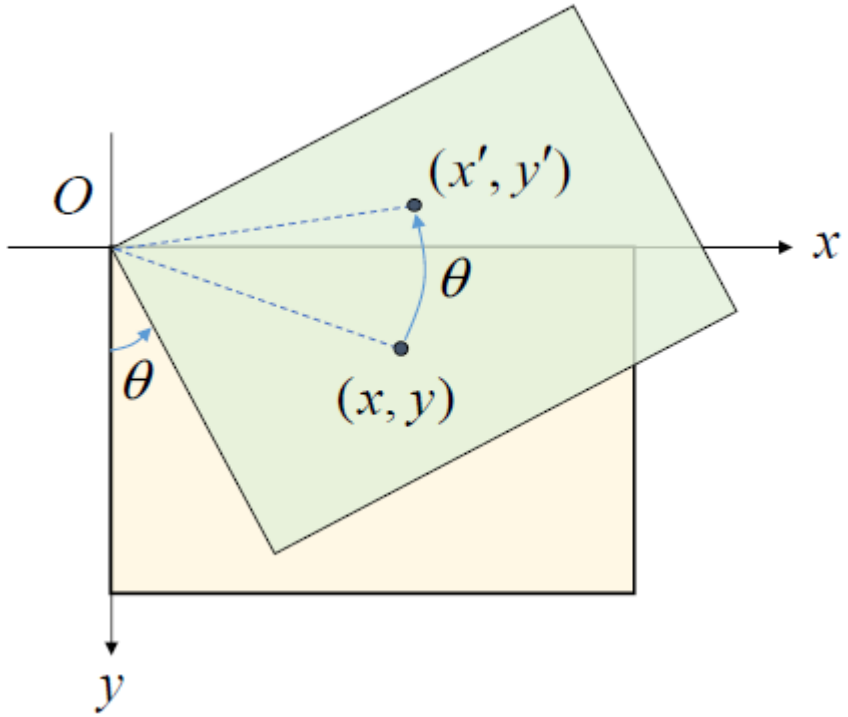
피라미드 영상에 사각형 그리기 예제 실행 결과



영상의 회전

회전 변환(rotation transformation)

- 영상을 특정 각도만큼 회전시키는 변환 (반시계 방향)



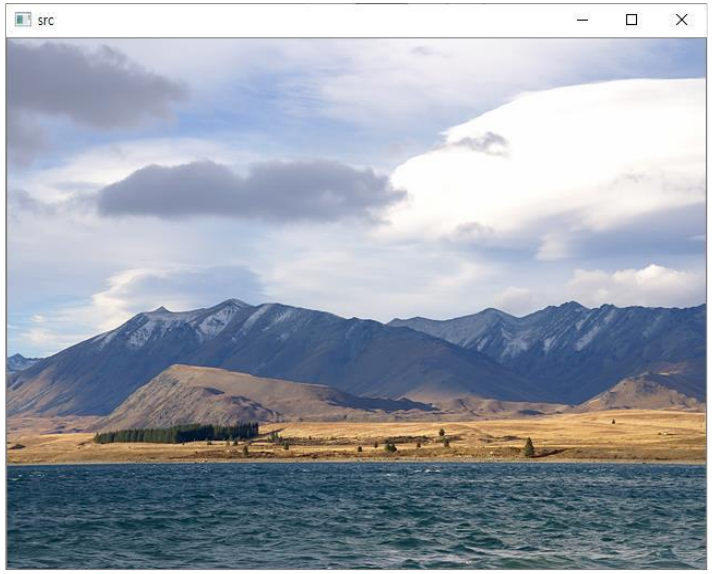
$$\begin{cases} x' = \cos\theta \cdot x + \sin\theta \cdot y \\ y' = -\sin\theta \cdot x + \cos\theta \cdot y \end{cases}$$

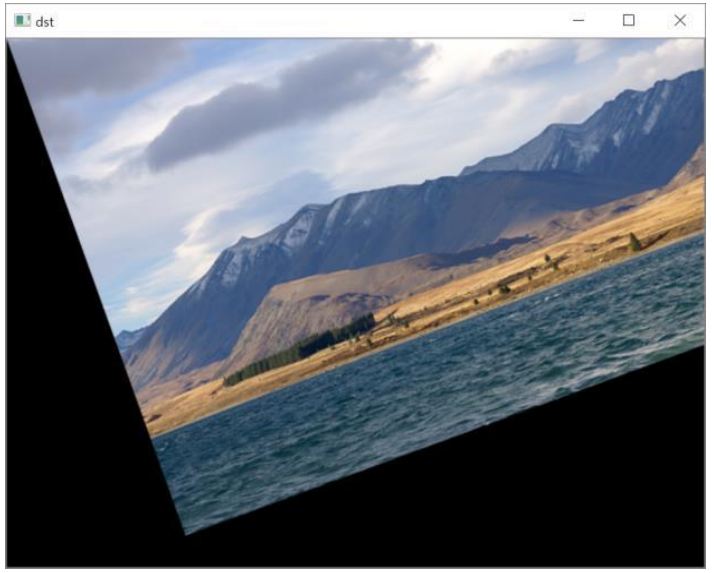
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

영상의 회전

영상의 회전 예제

```python
import sys
import math
import numpy as np
import cv2

src = cv2.imread('tekapo.bmp')

if src is None:
    print('Image load failed!')
    sys.exit()

rad = 20 * math.pi / 180
aff = np.array( object: [[math.cos(rad), math.sin(rad), 0],
                [-math.sin(rad), math.cos(rad), 0]], dtype=np.float32)

dst = cv2.warpAffine(src, aff,  dsize: (0, 0))

cv2.imshow( winname: 'src', src)
cv2.imshow( winname: 'dst', dst)
cv2.waitKey()

cv2.destroyAllWindows()
```





영상의 회전

영상의 회전 변환 행렬 구하기

cv2.getRotationMatrix2D(center, angle, scale) -> retval

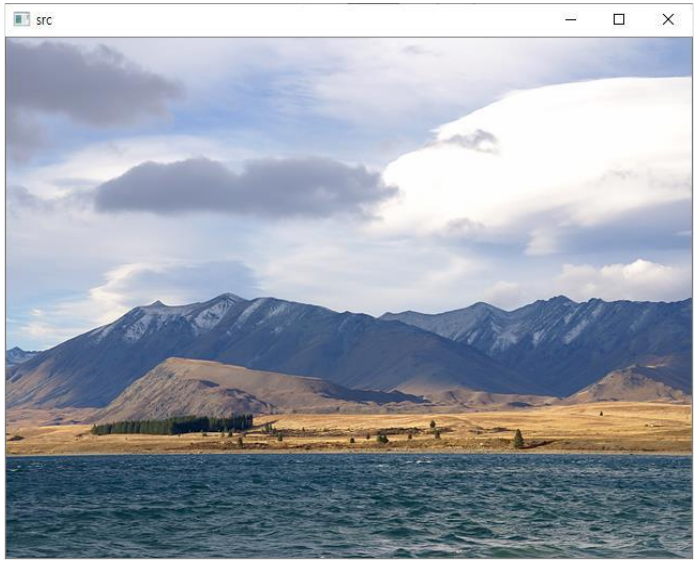
- center: 회전 중심 좌표. (x, y) 튜플.
- angle: (반시계 방향) 회전 각도(degree). 음수는 시계 방향.
- scale: 추가적인 확대 비율
- retval: 2x3 어파인 변환 행렬. 실수형.

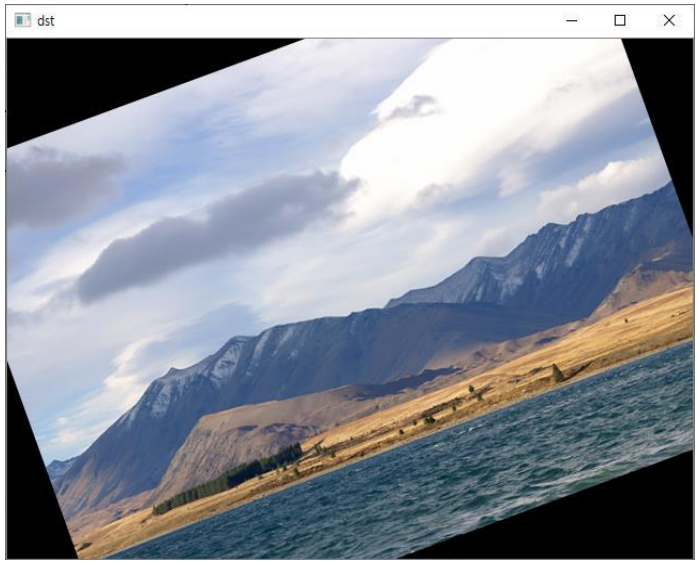
$$\begin{bmatrix} \alpha & \beta & (1-\alpha)\cdot \text{center.x} - \beta \cdot \text{center.y} \\ -\beta & \alpha & \beta \cdot \text{center.x} + (1-\alpha)\cdot \text{center.y} \end{bmatrix} \quad \text{where} \quad \begin{cases} \alpha = \text{scale} \cdot \cos(\text{angle}) \\ \beta = \text{scale} \cdot \sin(\text{angle}) \end{cases}$$

영상의 회전

영상의 중앙 기준 회전 예제

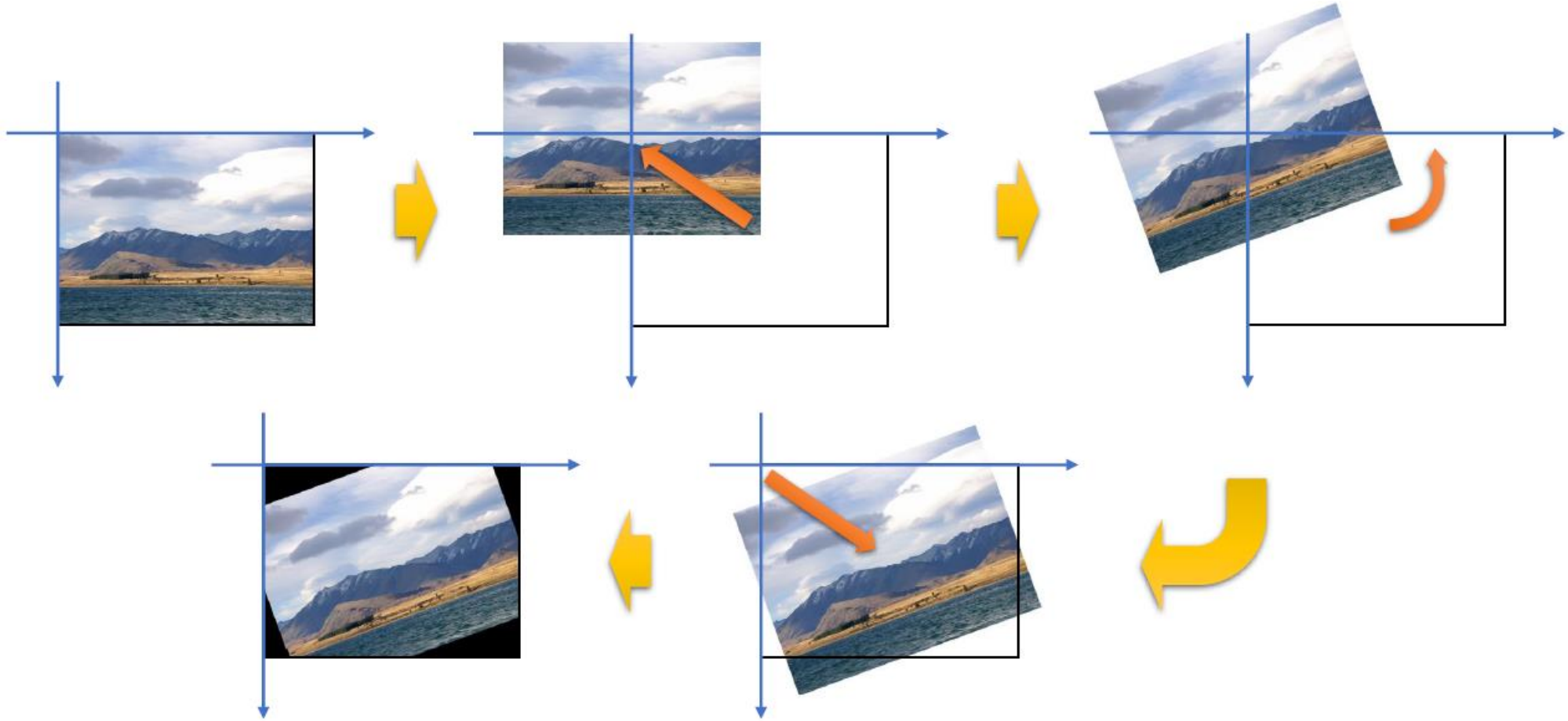
```python
import sys
import numpy as np
import cv2

src = cv2.imread('tekapo.bmp')

if src is None:
    print('Image load failed!')
    sys.exit()

cp = (src.shape[1] / 2, src.shape[0] / 2)
rot = cv2.getRotationMatrix2D(cp, angle: 20, scale: 0.5)

dst = cv2.warpAffine(src, rot, dsize: (0, 0))

cv2.imshow( winname: 'src', src)
cv2.imshow( winname: 'dst', dst)
cv2.waitKey()

cv2.destroyAllWindows()
```





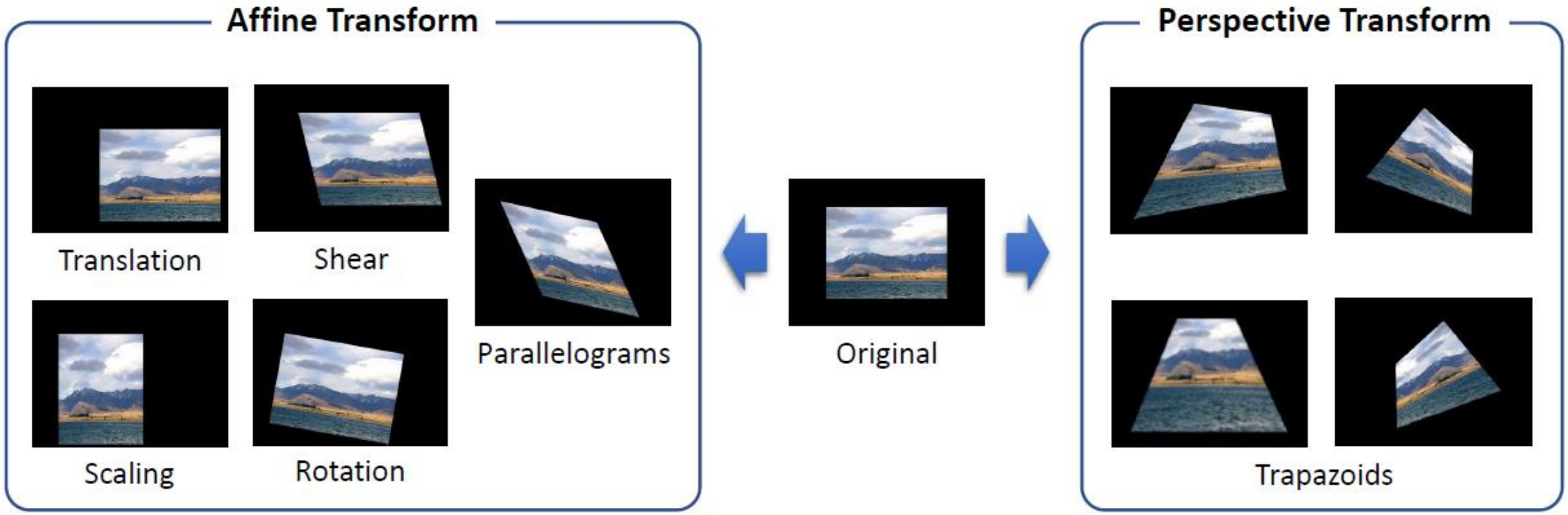
영상의 회전

영상의 중앙 기준 회전 예제 동작



어파인 변환과 투시 변환

어파인 변환 vs. 투시 변환



$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

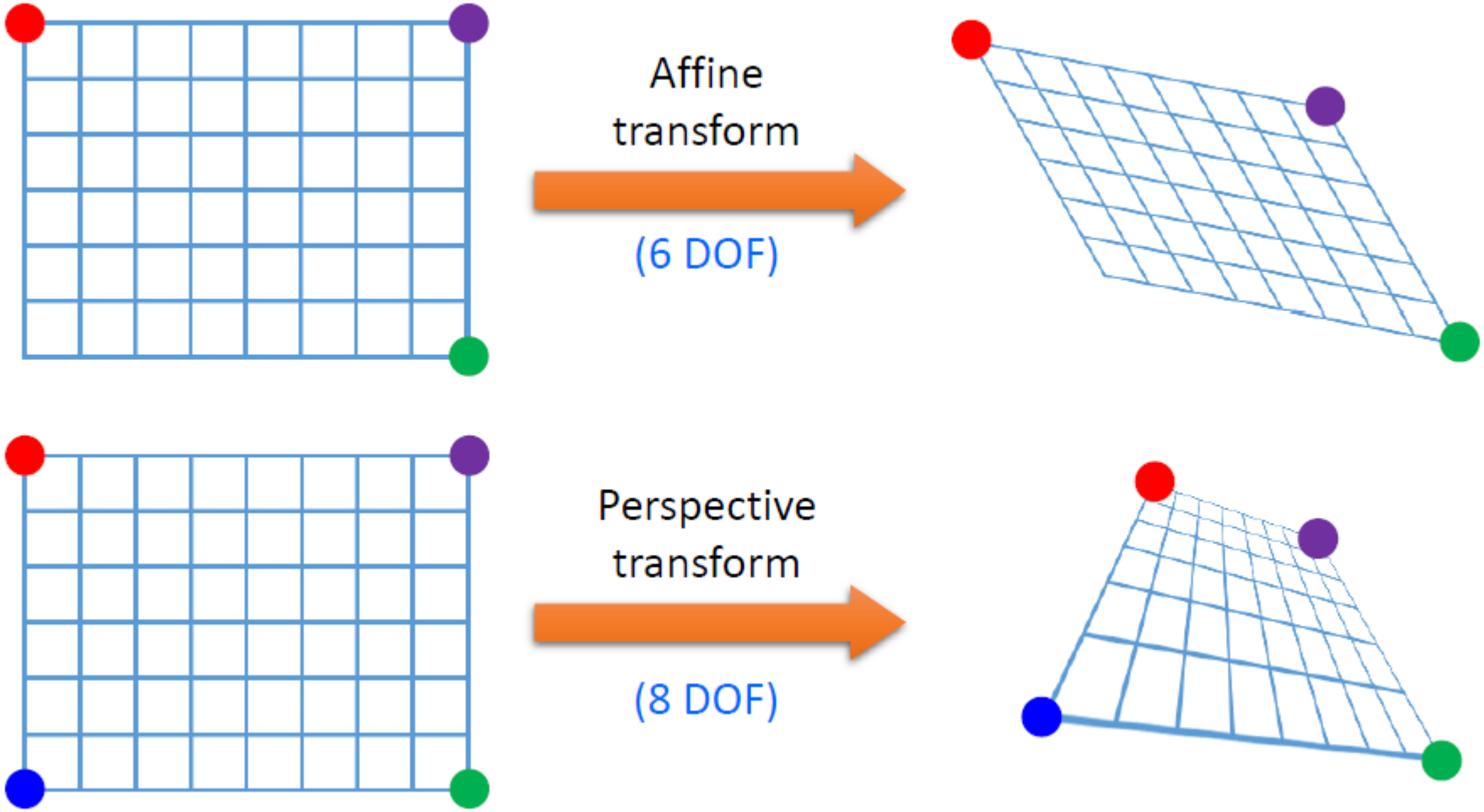
2x3 matrix
(6 DOF)

3x3 matrix
(8 DOF)

$$\begin{pmatrix} wx' \\ wy' \\ w \end{pmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \\ p_{31} & p_{32} & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

어파인 변환과 투시 변환

어파인 변환 vs. 투시 변환



어파인 변환과 투시 변환

어파인 변환 행렬 구하기

cv2.getAffineTransform(src, dst) --> retval

- src: 3개의 원본 좌표점. numpy.ndarray.shape=(3, 2)

  e.g) np.array ([[x_1, y_1], [x_2, y_2], [x_3, y_3]], np.float32)
- dst: 3개의 결과 좌표점. numpy.ndarray.shape=(3, 2)
- retval : 2x3 투시 변환 행렬

$$\begin{bmatrix} x_i' \\ y_i' \end{bmatrix} = \texttt{map\_matrix} \cdot \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

where $dst(i) = (x_i', y_i'), src(i) = (x_i, y_i), i = 0, 1, 2$

어파인 변환과 투시 변환

투시 변환 행렬 구하기

cv2.getPerspectiveTransform(src, dst, solveMethod=None) --> retval

- src: 4개의 원본 좌표점. numpy.ndarray.shape=(4, 2)

  e.g) np.array ([[x_1, y_1], [x_2, y_2], [x_3, y_3], [x_4, y_4]], np.float32)

- dst: 4개의 결과 좌표점. numpy.ndarray.shape=(4, 2)
- retval : 3x3 투시 변환 행렬

$$\begin{bmatrix} t_i x'_i \\ t_i y'_i \\ t_i \end{bmatrix} = \texttt{map\_matrix} \cdot \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

where $dst(i) = (x'_i, y'_i), src(i) = (x_i, y_i), i = 0, 1, 2, 3$

어파인 변환과 투시 변환

영상의 어파인 변환 함수

cv2.warpAffine(src, M, dsize, dst=None, flags=None, borderMode=None, borderValue=None) --> dst

- src: 입력 영상
- M: 2x3 어파인 변환 행렬. 실수형
- dsize: 결과 영상 크기. (w, h) 튜플. (0, 0)이면 src 와 같은 크기로 설정
- dst: 출력 영상
- flags: 보간법. 기본값은 cv2.INTER_LINEAR.
- borderMode: 가장자리 픽셀 확장 방식. 기본값은 cv2.BORDER_CONSTANT.
- borderValue: cv2.BORDER_CONSTANT 일 때 사용할 상수 값. 기본값은 0.

어파인 변환과 투시 변환

영상의 투시 변환 함수

cv2.warpPerspective(src, M, dsize, dst=None, flags=None, borderMode=None, borderValue=None) --> dst

- src: 입력 영상
- M: 3x3 투시 변환 행렬. 실수형
- dsize: 결과 영상 크기. (w, h) 튜플. (0, 0)이면 src 와 같은 크기로 설정
- dst: 출력 영상
- flags: 보간법. 기본값은 cv2.INTER_LINEAR.
- borderMode: 가장자리 픽셀 확장 방식. 기본값은 cv2.BORDER_CONSTANT.
- borderValue: cv2.BORDER_CONSTANT 일 때 사용할 상수 값. 기본값은 0.

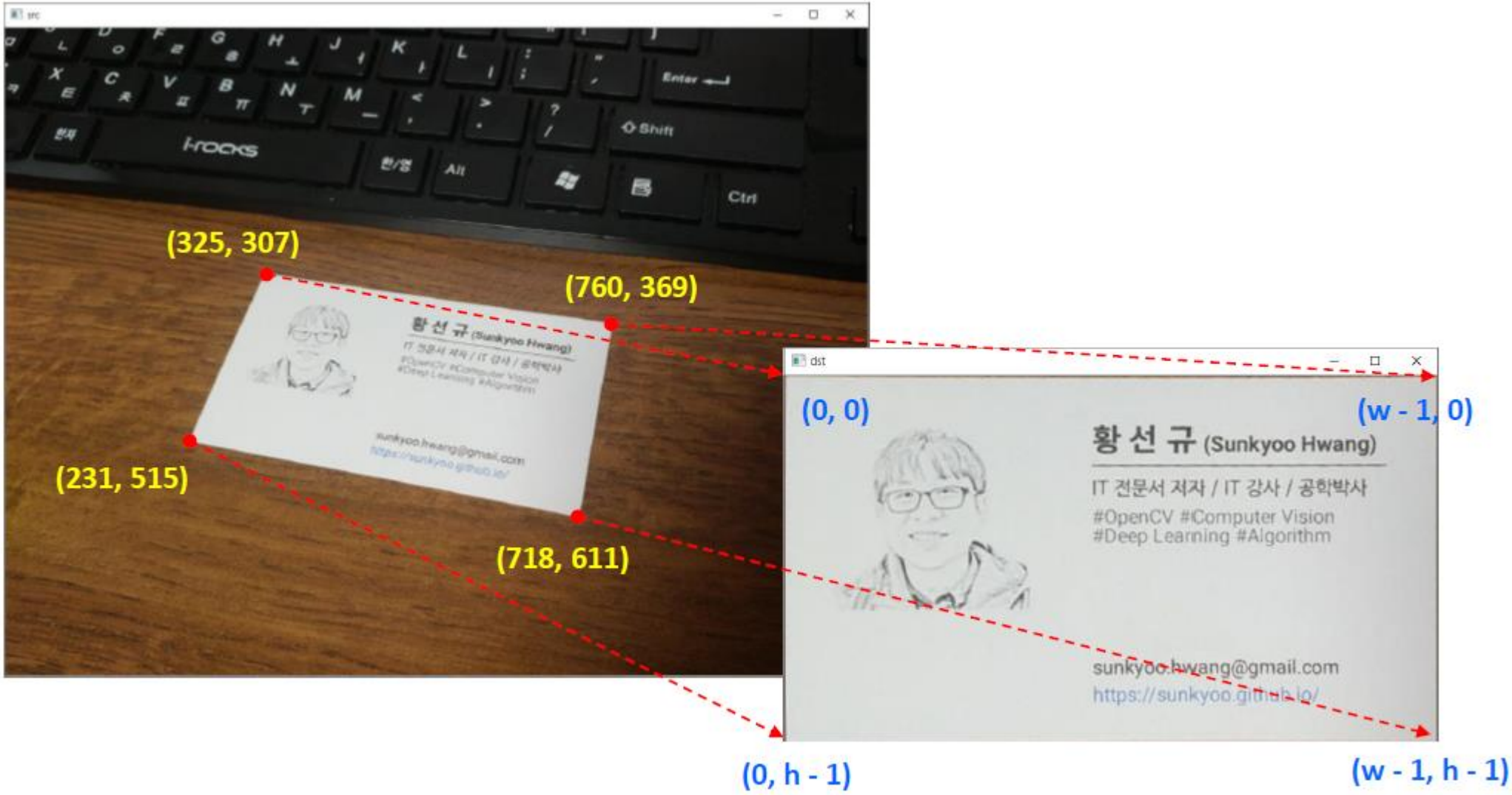
어파인 변환과 투시 변환

투시 변환 예제 찌그러진 명함 펴기

```python
import sys
import numpy as np
import cv2

src = cv2.imread('namecard.jpg')

if src is None:
    print('Image load failed!')
    sys.exit()

w, h = 720, 400
srcQuad = np.array( object: [[325, 307], [760, 369], [718, 611], [231, 515]], np.float32)
dstQuad = np.array( object: [[0, 0], [w-1, 0], [w-1, h-1], [0, h-1]], np.float32)

pers = cv2.getPerspectiveTransform(srcQuad, dstQuad)
dst = cv2.warpPerspective(src, pers,  dsize: (w, h))

cv2.imshow( winname: 'src', src)
cv2.imshow( winname: 'dst', dst)
cv2.waitKey()
cv2.destroyAllWindows()
```

어파인 변환과 투시 변환

투시 변환 예제 찌그러진 명함 펴기



리매핑

리매핑(remapping)

- 영상의 특정 위치 픽셀을 다른 위치에 재배치하는 일반적인 프로세스

dst(x, y) = src(map_x(x, y), map_y(x, y))

- 어파인 변환, 투시 변환을 포함한 다양한 변환을 리매핑으로 표현 가능
- examples)

$$\begin{cases} \text{map}_x(x,y) = x - 200 \\ \text{map}_y(x,y) = y - 100 \end{cases} \qquad \begin{cases} \text{map}_x(x,y) = w - 1 - x \\ \text{map}_y(x,y) = y \end{cases}$$

리매핑

리매핑 함수

cv2.remap(src, map1, map2, interpolation, dst=None, borderMode=None, borderValue=None)) --> dst

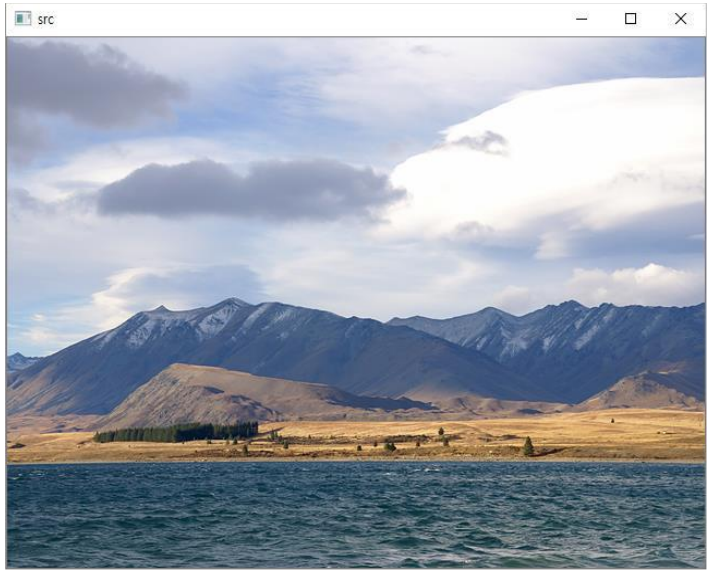
- src: 입력 영상
- map1: 결과 영상의 (x, y) 좌표가 참조할 입력 영상의 x 좌표

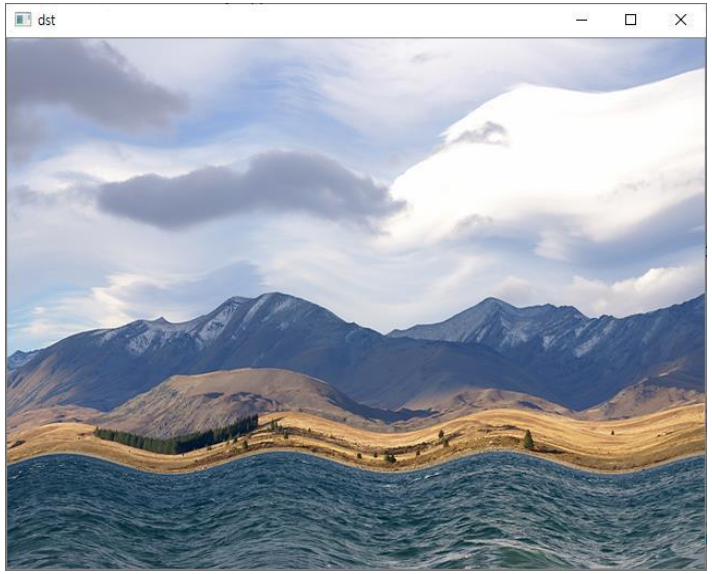
  입력 영상과 크기는 같고 타입은 np.float32 인 numpy.ndarray
- map2: 결과 영상의 (x, y) 좌표가 참조할 입력 영상의 y 좌표
- interpolation: 보간법
- dst: 출력 영상
- borderMode: 가장자리 픽셀 확장 방식. 기본값은 cv2.BORDER_CONSTANT.
- borderValue: cv2.BORDER_CONSTANT 일 때 사용할 상수 값. 기본값은 0.

리매핑

삼각함수를 이용한 리매핑 예제

```python
import sys
import numpy as np
import cv2

src = cv2.imread('tekapo.bmp')

if src is None:
    print('Image load failed!')
    sys.exit()

h, w = src.shape[:2]

map2, map1 = np.indices( dimensions: (h, w), dtype=np.float32)
map2 = map2 + 10 * np.sin(map1 / 32)

dst = cv2.remap(src, map1, map2, cv2.INTER_CUBIC, borderMode=cv2.BORDER_DEFAULT)

cv2.imshow( winname: 'src', src)
cv2.imshow( winname: 'dst', dst)
cv2.waitKey()

cv2.destroyAllWindows()
```





리매핑

삼각함수를 이용한 리매핑 예제

```
src = cv2.imread('tekapo.bmp')

h, w = src.shape[:2]

map2, map1 = np.indices((h, w), dtype=np.float32)
map2 = map2 + 10 * np.sin(map1 / 32)

dst = cv2.remap(src, map1, map2, cv2.INTER_CUBIC,
                borderMode=cv2.BORDER_DEFAULT)
```

map1

0	1	2	3
0	1	2	3
0	1	2	3
0	1	2	3

...

⋮

map2

0	0.312	0.625	0.936
1	1.312	1.625	1.936
2	2.312	2.625	2.936
3	3.312	3.625	3.936

...

⋮

[프로젝트] 문서 스캐너

문서 스캐너

- 카메라로 촬영한 문서 영상을 똑바로 펴서 저장해주는 프로그램

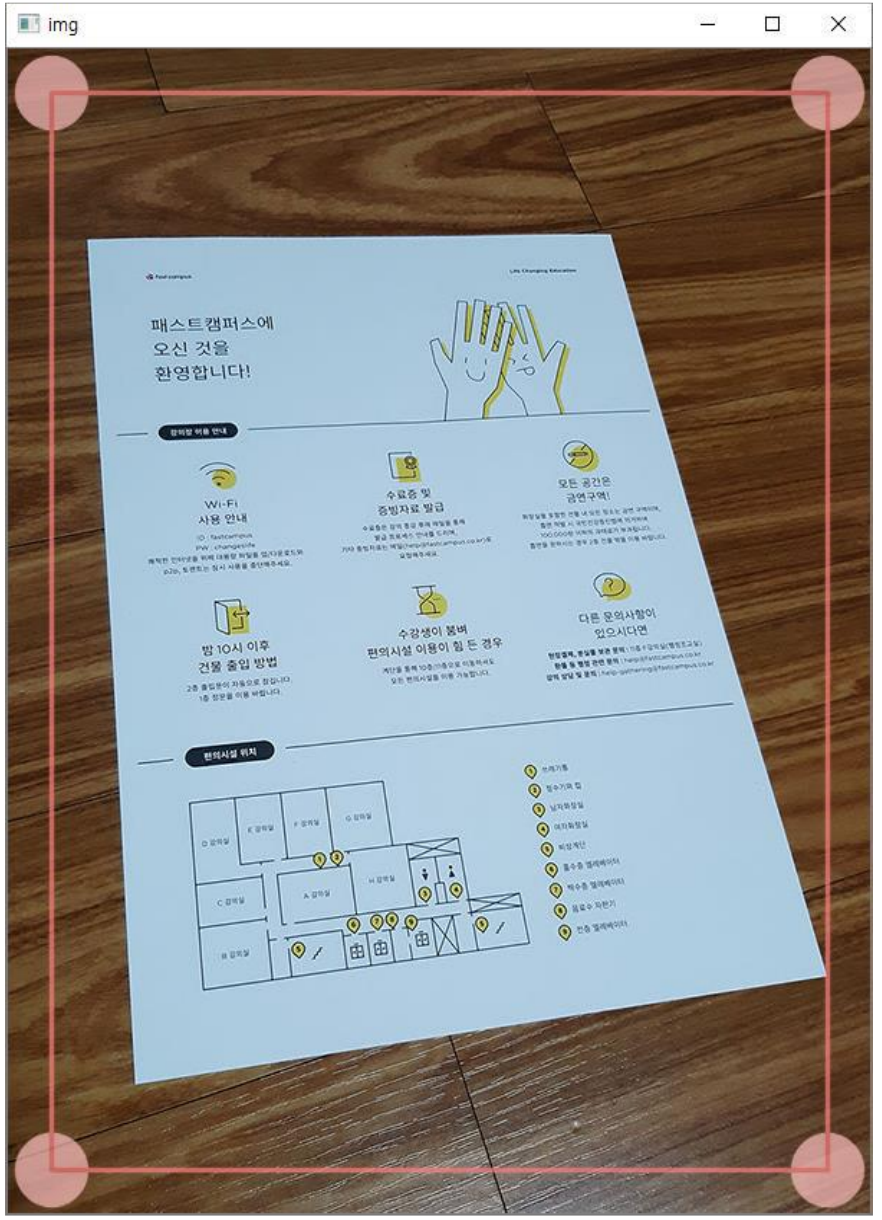
구현할 기능

- 마우스로 문서 모서리 선택 & 이동하기
- 키보드 ENTER 키 인식
- 왜곡된 문서 영상을 직사각형 형태로 똑바로 펴기 투시 변환

[프로젝트] 문서 스캐너

마우스로 문서 모서리 선택 & 이동하기

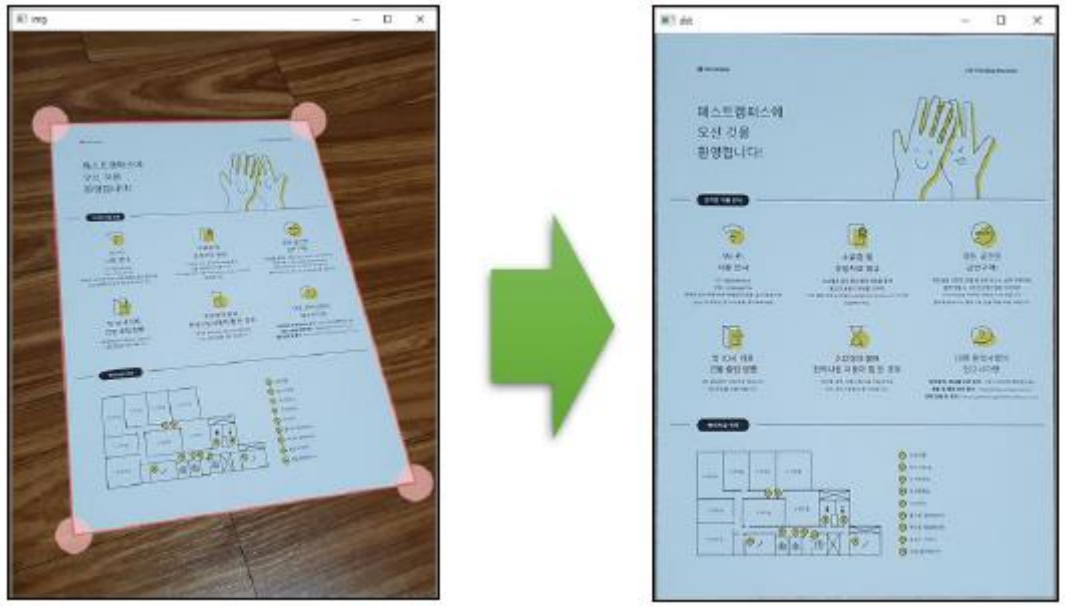
- 마우스 왼쪽 버튼이 눌린 좌표가
  네 개의 모서리와 근접해 있는지를 검사
- 특정 모서리를 선택했다면 마우스 드래그를 검사
- 마우스 드래그 시 좌표 이동 & 화면 표시
- 마우스 왼쪽 버튼이 떼어졌을 때의 좌표를 기록



[프로젝트] 문서 스캐너

왜곡된 문서 영상을 직사각형 형태로 똑바로 펴기 투시 변환

- 네 개의 모서리 좌표를 순서대로 srcQuad 배열에 추가
- dstQuad 배열에는 미리 정의한 A4 용지 크기의 네 모서리 좌표를 저장(A4 용지 크기 : 210x297cm)
- srcQuad 점들로부터 dstQuad 점들로 이동하는 투시 변환 계산
- 계산된 투시 변환 행렬을 이용하여 영상을 투시 변환하여 화면 출력



[프로젝트] 문서 스캐너

왜곡된 문서 영상을 직사각형 형태로 똑바로 펴기 투시 변환

```python
import sys
import numpy as np
import cv2


def drawROI(img, corners):
    cpy = img.copy()

    c1 = (192, 192, 255)
    c2 = (128, 128, 255)

    for pt in corners:
        cv2.circle(cpy, tuple(pt.astype(int)), radius: 25, c1, -1, cv2.LINE_AA)

    cv2.line(cpy, tuple(corners[0].astype(int)), tuple(corners[1].astype(int)), c2, thickness: 2, cv2.LINE_AA)
    cv2.line(cpy, tuple(corners[1].astype(int)), tuple(corners[2].astype(int)), c2, thickness: 2, cv2.LINE_AA)
    cv2.line(cpy, tuple(corners[2].astype(int)), tuple(corners[3].astype(int)), c2, thickness: 2, cv2.LINE_AA)
    cv2.line(cpy, tuple(corners[3].astype(int)), tuple(corners[0].astype(int)), c2, thickness: 2, cv2.LINE_AA)

    disp = cv2.addWeighted(img, alpha: 0.3, cpy, beta: 0.7, gamma: 0)

    return disp
```

[프로젝트] 문서 스캐너

- **왜곡된 문서 영상을 직사**

```
23
24  def onMouse(event, x, y, flags, param):
25      global srcQuad, dragSrc, ptOld, src
26
27      if event == cv2.EVENT_LBUTTONDOWN:
28          for i in range(4):
29              if cv2.norm(srcQuad[i] - (x, y)) < 25:
30                  dragSrc[i] = True
31                  ptOld = (x, y)
32                  break
33
34      if event == cv2.EVENT_LBUTTONUP:
35          for i in range(4):
36              dragSrc[i] = False
37
38      if event == cv2.EVENT_MOUSEMOVE:
39          for i in range(4):
40              if dragSrc[i]:
41                  dx = x - ptOld[0]
42                  dy = y - ptOld[1]
43
44                  srcQuad[i] += (dx, dy)
45
46                  cpy = drawROI(src, srcQuad)
47                  cv2.imshow( winname: 'img', cpy)
48                  ptOld = (x, y)
49                  break
```

[프로젝트] 문서 스캐너

왜곡된 문서 영상을 직사각형 형태로 똑바로 펴기 투시 변환

```
50
51  # 입력 이미지 불러오기
52  src = cv2.imread('scanned.jpg')
53
54  if src is None:
55      print('Image open failed!')
56      sys.exit()
57
58  # 입력 영상 크기 및 출력 영상 크기
59  h, w = src.shape[:2]
60  dw = 500
61  dh = round(dw * 297 / 210)  # A4 용지 크기: 210x297cm
62
63  # 모서리 점들의 좌표, 드래그 상태 여부
64  srcQuad = np.array( object: [[30, 30], [30, h-30], [w-30, h-30], [w-30, 30]], np.float32)
65  dstQuad = np.array( object: [[0, 0], [0, dh-1], [dw-1, dh-1], [dw-1, 0]], np.float32)
66  dragSrc = [False, False, False, False]
67
68  # 모서리점, 사각형 그리기
69  disp = drawROI(src, srcQuad)
70
```

[프로젝트] 문서 스캐너

왜곡된 문서 영상을 직사각형 형태로 똑바로 펴기 투시 변환

```
71    cv2.imshow( winname: 'img', disp)
72    cv2.setMouseCallback( windowName: 'img', onMouse)
73
74    while True:
75        key = cv2.waitKey()
76        if key == 13:  # ENTER 키
77            break
78        elif key == 27:  # ESC 키
79            cv2.destroyWindow('img')
80            sys.exit()
81
82    # 투시 변환
83    pers = cv2.getPerspectiveTransform(srcQuad, dstQuad)
84    dst = cv2.warpPerspective(src, pers,  dsize: (dw, dh), flags=cv2.INTER_CUBIC)
85
86    # 결과 영상 출력
87    cv2.imshow( winname: 'dst', dst)
88    cv2.waitKey()
89    cv2.destroyAllWindows()
```