# 제어지능SW개발

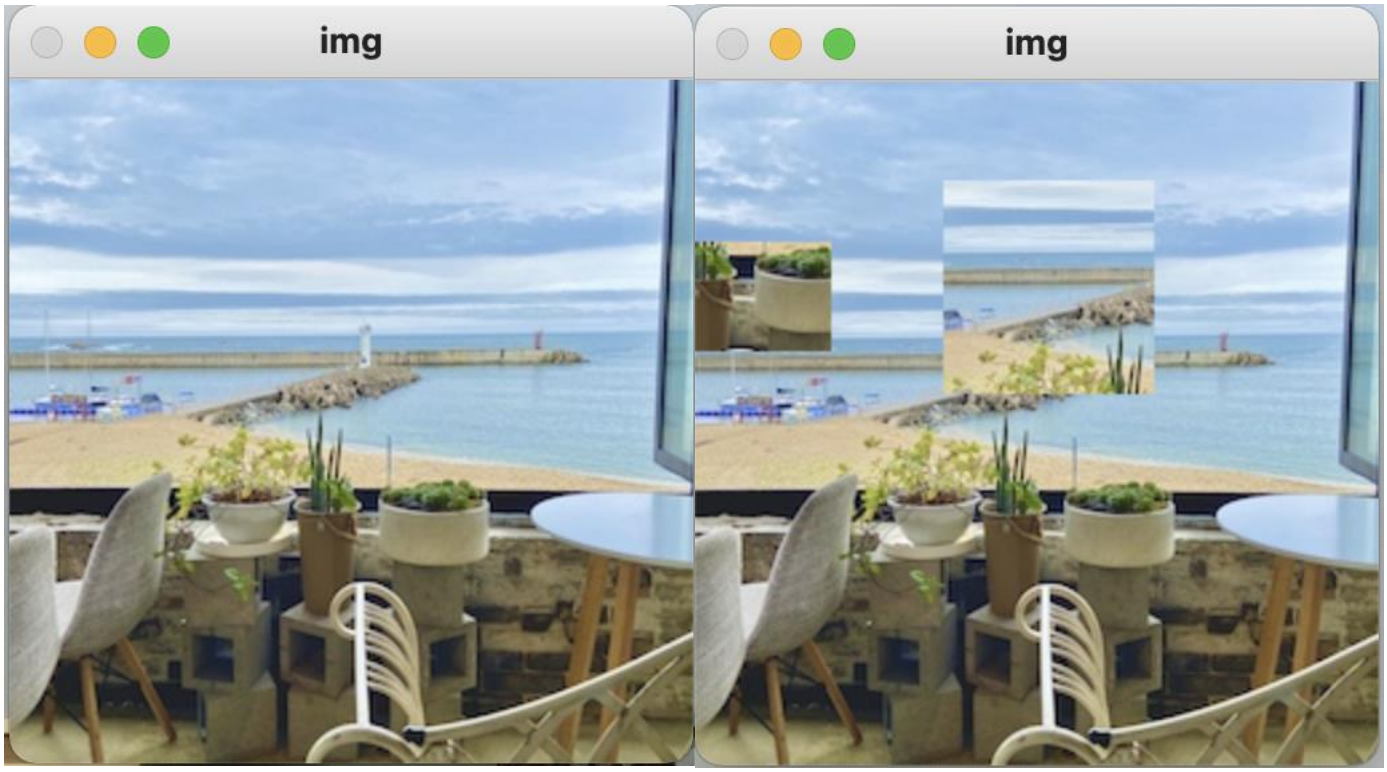
## 11주차 과제

201708015

A

**이한결**

# 과제 1.

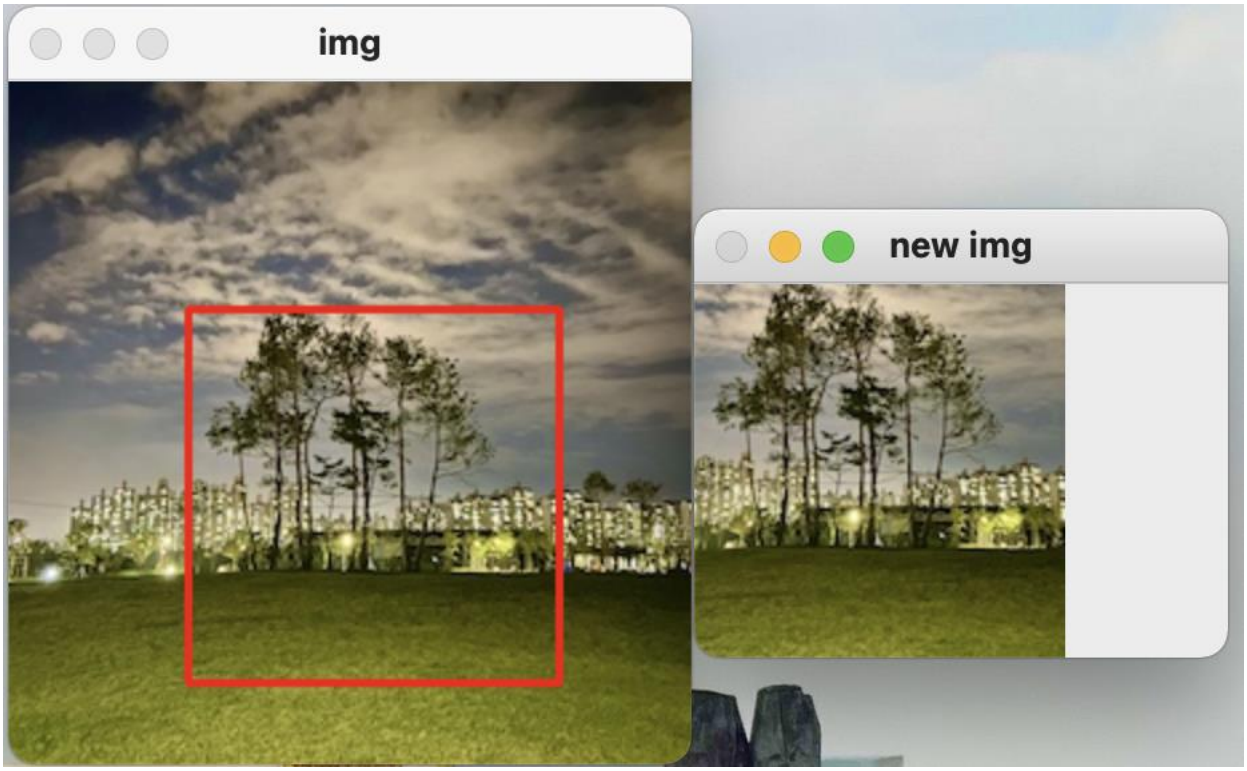



```
import numpy as np
import cv2 as cv

img_org = cv.imread("image1.jpg")
img = np.copy(img_org)

x_down, y_down, x_up, y_up = np.zeros(4, dtype=np.uint8)
def Click(event, x, y, flags, param):
        global x_down, y_down, x_up, y_up
        if event == 2:
                if x_down > x_up:
                        x_down, x_up = x_up, x_down
                if y_down > y_up:
                        y_down, y_up = y_up, y_down

                img_event = np.copy(img_org[y_down:y_up+1, x_down:x_up+1])
```

```
            x_len, y_len = x_up - x_down + 1, y_up - y_down + 1
            x_hlen, y_hlen = x_len // 2, y_len // 2
            y_left = y - y_hlen
            y_right = y + y_hlen + y_len % 2
            x_left = x - x_hlen
            x_right = x + x_hlen + x_len % 2
            y_ld, y_rd, x_ld, x_rd = np.zeros(4, dtype=np.int8)
            if y_left < 0:
                y_ld = -y_left
                y_left = 0
            if x_left < 0:
                x_ld = -x_left
                x_left = 0
            if x_right > 255:
                x_rd = x_right - 255
                x_right = 255
            if y_right > 255:
                y_rd = y_right - 255
                y_right = 255
            img[y_left : y_right , x_left : x_right] = img_event[y_ld:y_len-y_rd,x_ld:x_len-x_rd]
            cv.imshow("img", img)

        if event == 1:
            x_down, y_down = x, y
            img_org[:,:] = img[:,:]
            cv.setMouseCallback("img", Drag)

def Mouse_up(x, y):
        global x_up, y_up
        x_up = x
```

```
        y_up = y
        cv.imshow('img', img_org)
        cv.setMouseCallback("img", Click)

def Drag(event, x, y, flags, param):
        global x_down, y_down
        xx = x_down
        yy = y_down

        if event == 0:
                img = np.copy(img_org)
                cv.rectangle(img, (xx, yy), (x, y), (0, 255, 255))
                cv.imshow('img', img)

        if event == 4:
                Mouse_up(x, y)

def Mouse_down(event, x, y, flags, param):
        global x_down, y_down
        if event == 1:
                x_down, y_down = x, y
                cv.setMouseCallback("img", Drag)

cv.namedWindow("img")
cv.setMouseCallback("img", Mouse_down)
cv.imshow("img",img_org)
cv.waitKey(0)

cv.destroyAllWindows()
```

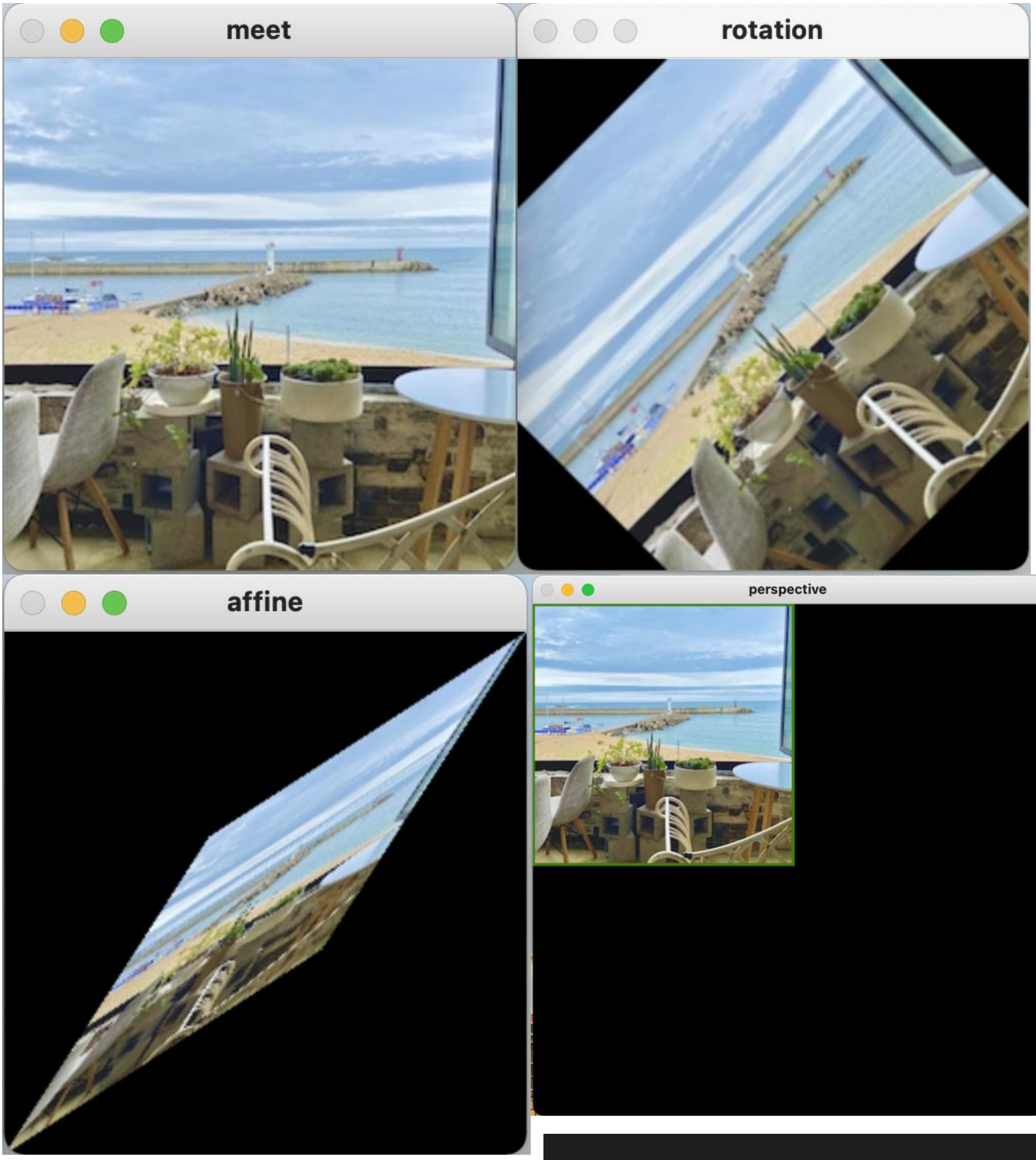
# 과제 2.




```
import cv2 as cv
import numpy as np
import time
isDragging = False
x0, y0, w, h = -1, -1, -1, -1
blue, red = (255, 0, 0), (0, 0, 255)
turn = 0
img = cv.imread('image2.jpg')
def onMouse(event, x, y, flags, param):
    global isDragging, x0, y0, img, turn, roi, w, h
    if event == cv.EVENT_LBUTTONDOWN:
        isDragging = True
        x0, y0 = x, y
    elif event == cv.EVENT_MOUSEMOVE:
        if isDragging:
            img_draw = img.copy()
            cv.rectangle(img_draw, (x0,y0),(x,y),blue,2)
```

```
                cv.imshow('img', img_draw)
        elif event == cv.EVENT_LBUTTONUP:
            if isDragging:
                isDragging = False
                w = x-x0
                h = y-y0
                if w>0 and h>0:
                    img_draw = img.copy()
                    cv.rectangle(img_draw, (x0,y0),(x, y), red, 2)
                    cv.imshow('img', img_draw)
                    roi = img[y0:y0+h, x0:x0+w]
                    cv.imwrite("new img.jpg", roi)
                    cv.imshow("new img",roi)
                else:
                    cv.imshow('img',img)
                    print('왼쪽부터  드래그를  하세요')
cv.imshow('img',img)
cv.setMouseCallback('img', onMouse)
cv.waitKey()
cv.destroyAllWindows()
```

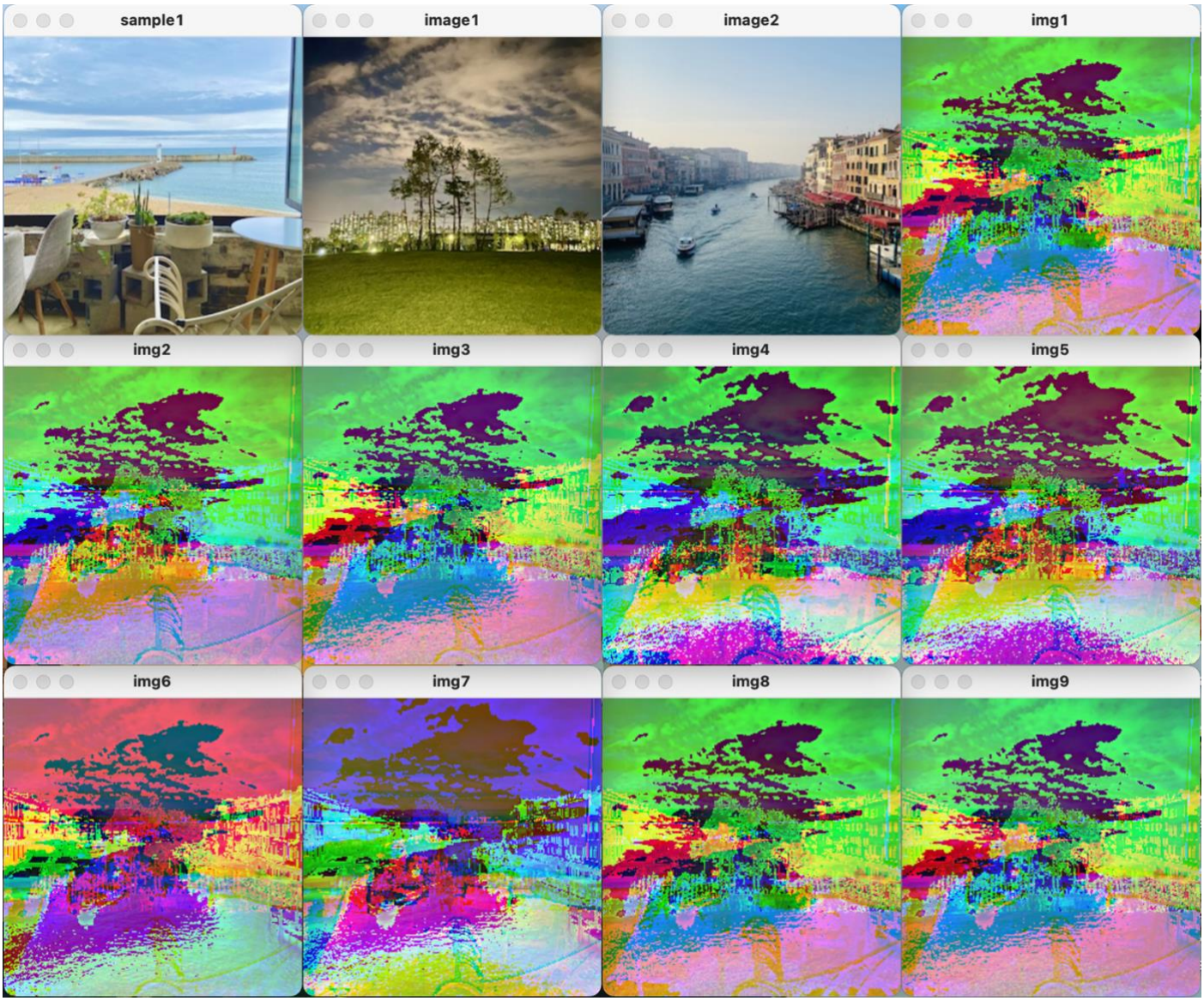
## 과제 3.




```
import cv2 as cv
import numpy as np
import time
def myShow(title, img):
        cv.imshow(title, img)
        if cv.waitKey(0) & 0XFF == 27:
                cv.destroyAllwindows()
```

```
            exit(0)
        else:
            cv.destroyWindow(title)
img = cv.imread('image1.jpg')
cv.imshow("meet",img)
row, cols, ch = img.shape
M = cv.getRotationMatrix2D(((cols-1)/2.0, (row-1)/2.0), 45, 1)
rotationImg = cv.warpAffine(img, M, (cols, row))
myShow("rotation", rotationImg)
pts1 = np.float32([[0,0], [0,255], [255,0]])
pts2 = np.float32([[100,100], [0,255],[255,0]])
M = cv.getAffineTransform(pts1, pts2)
affineImg = cv.warpAffine(img, M, (cols, row))
myShow("affine", affineImg)
pts1 = np.float32([[0,0], [0,255], [255,0], [255,255]])
pts2 = np.float32([[0,0], [0,255], [255,0], [255,255]])
M = cv.getPerspectiveTransform(pts1, pts2)
perspctiveImg = cv.warpPerspective(img, M, (500,500))
cv.rectangle(perspctiveImg, (0,0), (254,254), (0,128,64), 2)
myShow("perspective", perspctiveImg)
cv.destroyAllWindows()
```

# 과제 4.




```
import numpy as np
import cv2 as cv
sample1 = cv.resize(cv.imread("image1.jpg"),(256,256))
image1 = cv.resize(cv.imread("image2.jpg"),(256,256))
image2 = cv.resize(cv.imread("image4.jpg"),(256,256))
cv.imshow('sample1', sample1)
cv.imshow('image1', image1)
```

```
cv.imshow('image2', image2)
np.random.seed(seed=5)
for i in range(9):
        img = np.zeros((256, 256, 3), dtype=np.int8)
        rimg = [0, 1, 2]
        np.random.shuffle(rimg)
        rn = np.random.randint(low=0, high=2, size=3)
        img[:,:,rimg[0]] = sample1[:,:,rn[0]]
        img[:,:,rimg[1]] = image1[:,:,rn[1]]
        img[:,:,rimg[2]] = image2[:,:,rn[2]]
        cv.imshow('img{0}'.format(i+1), img)
cv.waitKey(0)
cv.destroyAllWindows()
```