



Day64; 20221206

📅 날짜	@2022년 12월 6일
👤 유형	@2022년 12월 6일
☰ 태그	

GitHub - u8yes/AI

You can't perform that action at this time. You signed in with another tab or window. You signed out in another tab or window. Reload to refresh your session. Reload to refresh your session.

<https://github.com/u8yes/ai>

u8yes/AI



1 Contributor 0 Issues 0 Stars 0 Forks

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/7a3f4955-59a-4ffc-823f-b84f5128296f/02_%EC%82%AC%EC%9A%A9%EC%9E%90_%EC%9D%B8%ED%84%B0%ED%8E%98%EC%9D%B4%EC%8A%A4_%EB%B0%8F_IO%EC%B2%98%EB%A6%AC.pdf

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/daa4433c-a885-4780-8701-c8c784f6a811/03_%EB%B0%B0%EC%97%B4_%EC%97%B0%EC%82%B0_%ED%95%A8%EC%88%98.pdf

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/de86a59b-9e4d-4d5f-b400-45cf4971275f/01_move_window.py

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/1ddeaf18-a65e-4dfb-8457-903bde465997/02_resize_window.py

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/00277bd4-1127-4565-b1ca-36c327372bba/03_event_key.py

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/2a501b27-58d4-4c29-9294-d3d67f6341c7/04_event_mouse.py

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/c179e26a-2384-4915-b6d4-d55403dea2b3/05_event_trackbar.py

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/b5fea45d-edd7-4d3d-8f38-6e3c29baae15/06_event_mouse_trackbar.py

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/71be0edb-be1c-49d4-a94f-ded699a12ca2/07_draw_line_rect.py

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/43e7dab4-e1a4-4898-8210-70086b323f86/08_put_text.py

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/162eb54b-9af6-4734-92ad-d75cb0574d50/09.draw_circle.py

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/60793f50-46ed-4adb-85d0-ea53240d3e62/10.draw_ellipse.py

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/b67cd2ac-0607-4a23-85db-0e3f50225832/11.event_draw.py

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/cf6073a5-5e4b-4635-8d78-3e367b6a228e/12.read_image1.py

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/51022724-b6b3-4217-93f5-37f9cec612f1/13.read_image2.py

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/4ad9c187-016d-4ab8-887a-1871c2e0f73f/14.read_image3.py

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/3cd8f4b9-4a0b-49ad-8185-321d88fbc39e/15.write_image1.py

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/967124fb-0583-418e-a01c-118e85d1f3ad/16.write_image2.py

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/d7ab89fc-f394-4161-a363-124a349cda6f/17_matplotlib.py

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/363a9bb9-760f-446c-a77d-a1c7c6164843/18_plot.py

17_matplotlib.py



```
17_matplotlib.py × 18_plot.py ×

import cv2
import matplotlib.pyplot as plt

image = cv2.imread("images/matplot.jpg", cv2.IMREAD_COLOR)
if image is None:
    raise Exception("영상 파일 읽기 에러")

rows, cols = image.shape[:2]
rgb_img = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
# BGR을 RGB 컬러로 변환
# cvtColor - 컨버팅해서 데이터들을 읽어오라는 기능을 제공.
# BGR2의 2는 to로 표현.
gray_img = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
# BGR의 컬러공간을 흑백으로 converting해줌.

plt.figure(num=1, figsize=(3,4))
plt.imshow(image)
plt.title("figure1-original(bgr") # blue green red
plt.axis('off') # 축은 x축, y축을 보여주지 않겠다.
plt.tight_layout() # 여백을 없애주는 것.
```

```

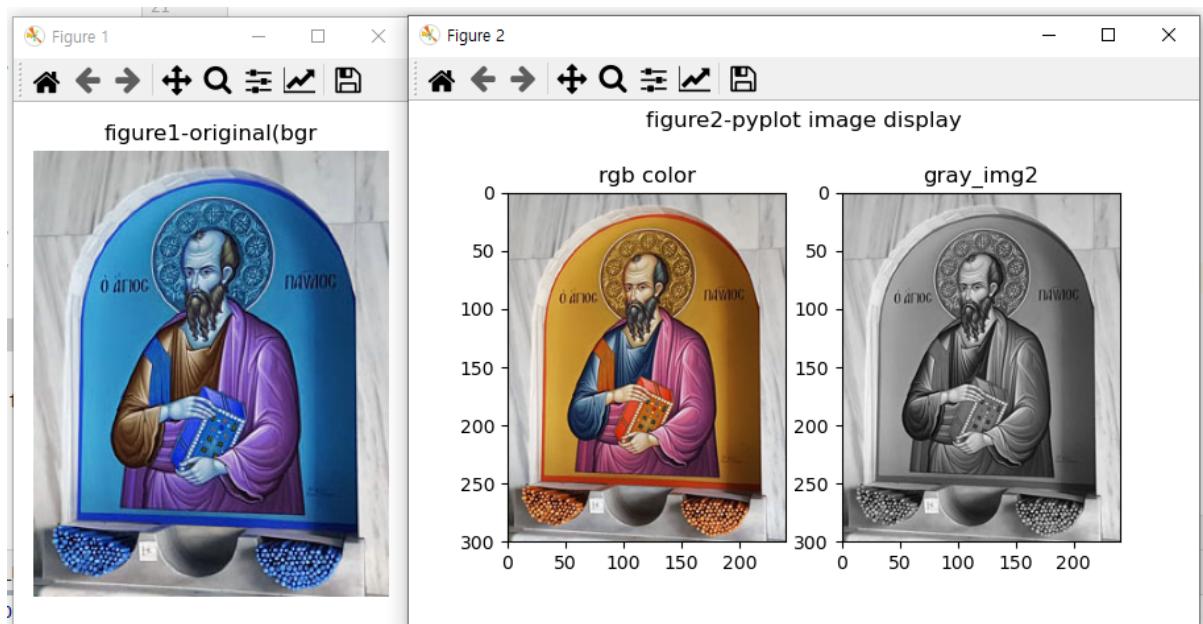
fig = plt.figure(num=2, figsize=(6,4))
plt.suptitle('figure2-pyplot image display')

plt.subplot(1,2,1)
plt.imshow(rgb_img)
plt.axis([0, cols, rows, 0])
plt.title("rgb color")

plt.subplot(1,2,2)
plt.imshow(gray_img, cmap='gray')
plt.axis([0, cols, rows, 0])
plt.title("gray_img2")

plt.show()

```



figure

미국식[ˈfɪgjər] ↗ 영국식[ˈfɪgə(r)] ↗

(명사)

1 (특히 공식적인 자료로 제시되는) 수치

the latest trade/sales/unemployment, etc. figures ↗

가장 최근의 무역/매출/실업률 등의 수치

2 숫자 (→double figures, single figures)

Write the figure '7' on the board. ↗

칠판에 숫자 7을 써라.

(동사)

1 (어떤 과정상황 등에서) 중요하다[중요한 부분이다] (=feature)

The question of the peace settlement is likely to figure prominently in the talks. ↗

평화 정착 문제가 그 회담에서 대단히 중요한 부분이 될 것 같다.

2 (...일 거라고) 생각[판단]하다

I figured (that) if I took the night train, I could be in Scotland by morning. ↗

나는 만약 밤 기차를 타면 아침이면 스코틀랜드에 가 있게 되리라고 판단했다.

영어사전 다른 뜻 5

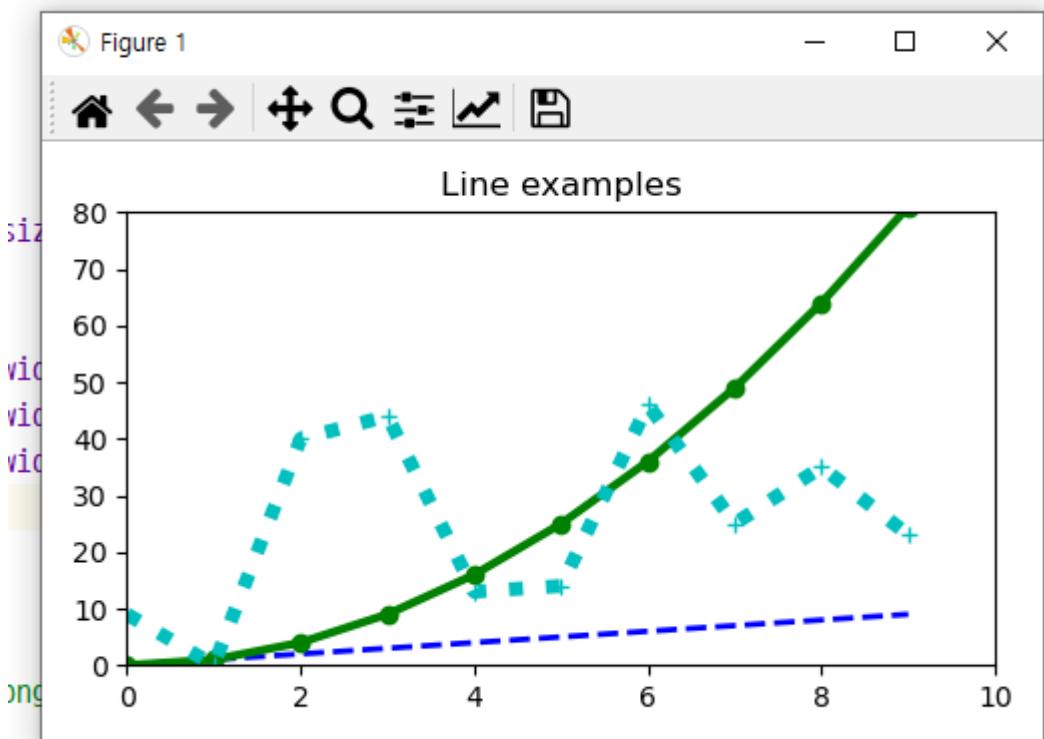
18_plot.py

```
7_matplotlib.py x 18_plot.py x
import matplotlib.pyplot as plt
import numpy as np

x = np.arange(10)
y1 = np.arange(10)
y2 = np.arange(10)**2
y3 = np.random.choice(50, size=10)

plt.figure(figsize=(5,3))
plt.plot(x,y1, 'b--', linewidth=2) # 긴 점선으로 출력
plt.plot(x,y2, 'go-', linewidth=3) # 선으로
plt.plot(x,y3, 'c+:', linewidth=5) # 점선으로 출력

plt.title("Line examples")
plt.axis([0,10,0,80])
plt.tight_layout()
plt.savefig(fname="sample.png", dpi=300) # 저장하고자 함
plt.show()
```



-->		
<< workspaces > openCV > src > chap02		
	이름	수정한 날짜
	PC 03_event_key.py	2022-12-05
	PC 04_event_mouse.py	2022-12-05
A)	PC 05_event_trackbar.py	2022-12-06
open	PC 06_event_mouse_trackbar.py	2022-12-05
	PC 07_draw_line_rect.py	2022-12-05
	PC 08_put_text.py	2022-12-05
	PC 09_draw_circle.py	2022-12-05
	PC 10.draw_ellipse.py	2022-12-05
	PC 11.event_draw.py	2022-12-05
	PC 12.read_image1.py	2022-12-05
	PC 13.read_image2.py	2022-12-05
	PC 14.read_image3.py	2022-12-05
	PC 15.write_image1.py	2022-12-05
	PC 16.write_image2.py	2022-12-05
	PC 17_matplotlib.py	2022-12-06
	PC 18_plot.py	2022-12-06
(C:)	sample.png	2022-12-06
(D:)	zoom주소_20221206.txt	2022-12-06
	PC 연습.py	2022-12-06

retrieve

미국·영국[ri'tri:v] ⓘ 영국식 ⓘ

[동사](#)

1 (특히 제자리가 아닌 곳에 있는 것을) 되찾아오다[회수하다] (=recover)

She bent to retrieve her comb from the floor. ⓘ

그녀가 바닥에 떨어진 빗을 주우려고 몸을 숙였다.

2 (정보를) 검색하다

to retrieve information from the database ⓘ

데이터베이스에서 정보를 검색하다

3 (좋지 않은 사태를) 수습[개선]하다, (잃은 것을) 되찾다

You can only retrieve the situation by apologizing. ⓘ

당신은 사과를 함으로써만 사태를 수습할 수 있다.

[영어사전 다른 뜻 2](#)

배열 연산 함수

y를 행으로, x를 열 순으로 인덱스

기본 배열 처리 함수

❖ 영상 속성과 픽셀 접근

- ✓ OpenCV 파이썬은 영상을 numpy.ndarray을 이용하여 표현
- ✓ 영상의 중요 속성(shape, dtype)을 확인하고 numpy의 astype(), reshape()로 속성을 변경하고 영상 픽셀을 y(행), x(열) 순으로 인덱스를 지정한 후 접근
- ✓ numpy는 다중 채널 영상을 모양(shape)에 의해 표현하고 OpenCV는 픽셀 자료형으로 표현

구분	numpy 자료형	OpenCV 자료형, 1-채널
8비트 unsigned 정수	np.uint8	cv2.CV_8U
8비트 signed 정수	np.int8	cv2.CV_8S
16비트 unsigned 정수	np.uint16	cv2.CV_16U
16비트 signed 정수	np.int16	cv2.CV_16S
32비트 signed 정수	np.int32	cv2.CV_32S
32비트 실수	np.float32	cv2.CV_32F
64비트 실수	np.float64	cv2.CV_64F

numpy 자료형

unsigned 0 ~ 255까지(음수 표현 X)

uint8비트 = 1바이트

OpenCV 자료형

8U = 1바이트 - unsigned 양수만 표현.

8S - signed로 표현한 것은 음수, 양수 다 포함시키겠다.

16은 short

32는 int 양수 음수

F는 float

64F는 double을 표현.

- ❖ 영상 속성과 픽셀 접근
 - ✓ 이미지 크기 변경
 - resize 함수를 이용해서 이미지 크기 변경
 resize(이미지 데이터, (가로 크기, 세로 크기))
 - 이미지들은 제각기 다양한 크기를 가지는데 이를 특성으로 사용하려면 동일한 차원으로 만들어야 하기 때문에 이미지 크기를 변경하는데 이미지는 행렬에 정보를 담고 있기 때문에 이미지 크기를 줄이면 행렬 크기와 거기에 담긴 정보도 줄어 듬
 - 머신 러닝은 수 천에서 수십만 개의 이미지가 필요한데 이미지가 클수록 메모리를 많이 차지하기 때문에 이미지 크기를 줄여서 메모리 사용량을 줄일 수 있음
 - 머신 러닝에서 많이 사용하는 이미지의 크기는 32X32, 64X64, 96X96, 256X256 등

Chap03_ 01_mat_array.py



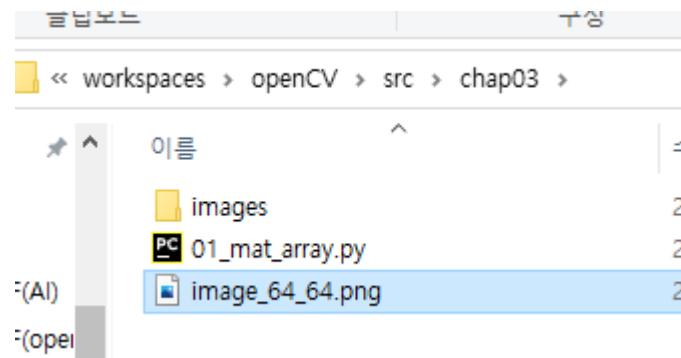
```
01_mat_array.py ×
1 import cv2
2 import numpy as np
3 import matplotlib.pyplot as plt
4
5 # 흑백 이미지로 로드
6 image = cv2.imread("./images/plane_256x256.jpg", cv2.IMREAD_GRAYSCALE)
7 # IMREAD_GRAYSCALE 회색으로 바로 바꿔줌.
8
9 # 64 X 64 크기로 변경
10 image_64_64 = cv2.resize(image, (64, 64))
11
12 plt.imshow(image_64_64, cmap="gray") # cmap 컬러출력
13 plt.axis("off")
14 plt.savefig(fname="image_64_64.png", dpi=300)
15 plt.show()
```

- □ ×

x=58,7 y=14,1
[132,0]



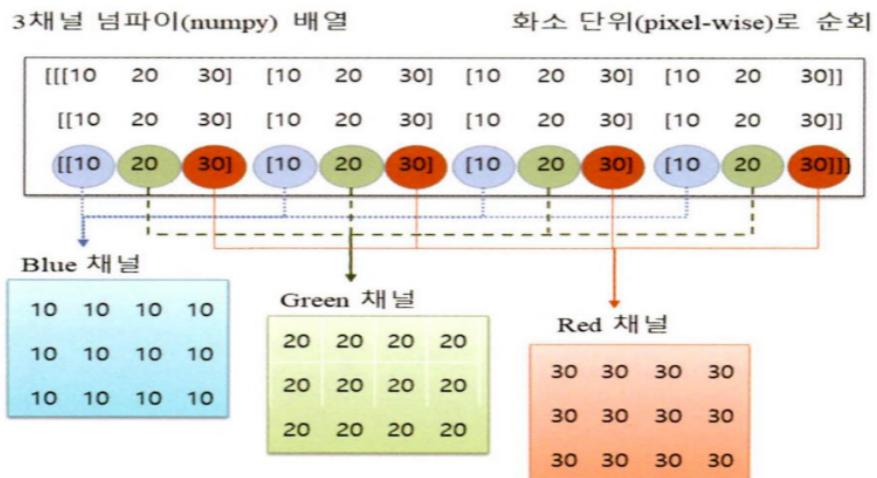
ay.py



cv는 BGR로 처리(CV의 이상한 특징)

채널 처리 함수

- ❖ 영상 채널 분리와 병합
 - ✓ 3채널 numpy 배열 및 컬러 영상의 각 채널



- ✓ cv2.split()는 다중 채널 영상을 리스트에 단일 채널 영상으로 분리하고 cv2.merge()는 단일 채널 영상을 병합하여 다중 채널 영상을 생성
 - cv2.split(m[, mv]) -> mv
 - cv2.merge(mv[, dst]) -> dst

채널 처리 함수

- ❖ 영상 채널 분리와 병합
 - ✓ 채널 분리
 - dst = cv2.split(src)는 3-채널 BGR 컬러 영상 src를 채널 분리하여 리스트 dst에 저장
 - type(dst)는 'list'이고 type(dst[0]), type(dst[1]), type(dst[2])는 'numpy.ndarray'
 - 채널 순서는 0-채널은 Blue, 1-채널은 Green, 2-채널은 Red

회전 처리 함수

- ❖ 회전
 - ✓ 입력된 2차원 배열을 수직, 수평 양축으로 뒤집기
 - cv2.flip(src, flipCode[, dst]) -> dst
 - src, dst: 입력 배열, 출력 배열
 - flipCode: 배열을 뒤집는 축
 - 0 : X축을 기준으로 위아래로 뒤집기
 - 1: y축을 기준으로 좌우로 뒤집기
 - -1: 양축(x축, y축 모두)을 기준으로 뒤집기
 - ✓ 입력된 배열을 복사
 - cv2.repeat(src, ny, nx[, dst]) -> dst
 - src, dst: 입력 배열, 출력 배열
 - ny, nx: 수직 방향, 수평 방향 반복 횟수
 - ✓ 입력 행렬의 전치 행렬을 출력으로 리턴
 - cv2.transpose(src[, dst]) -> dst
 - src, dst: 입력 배열, 출력 배열

02_rotation.py

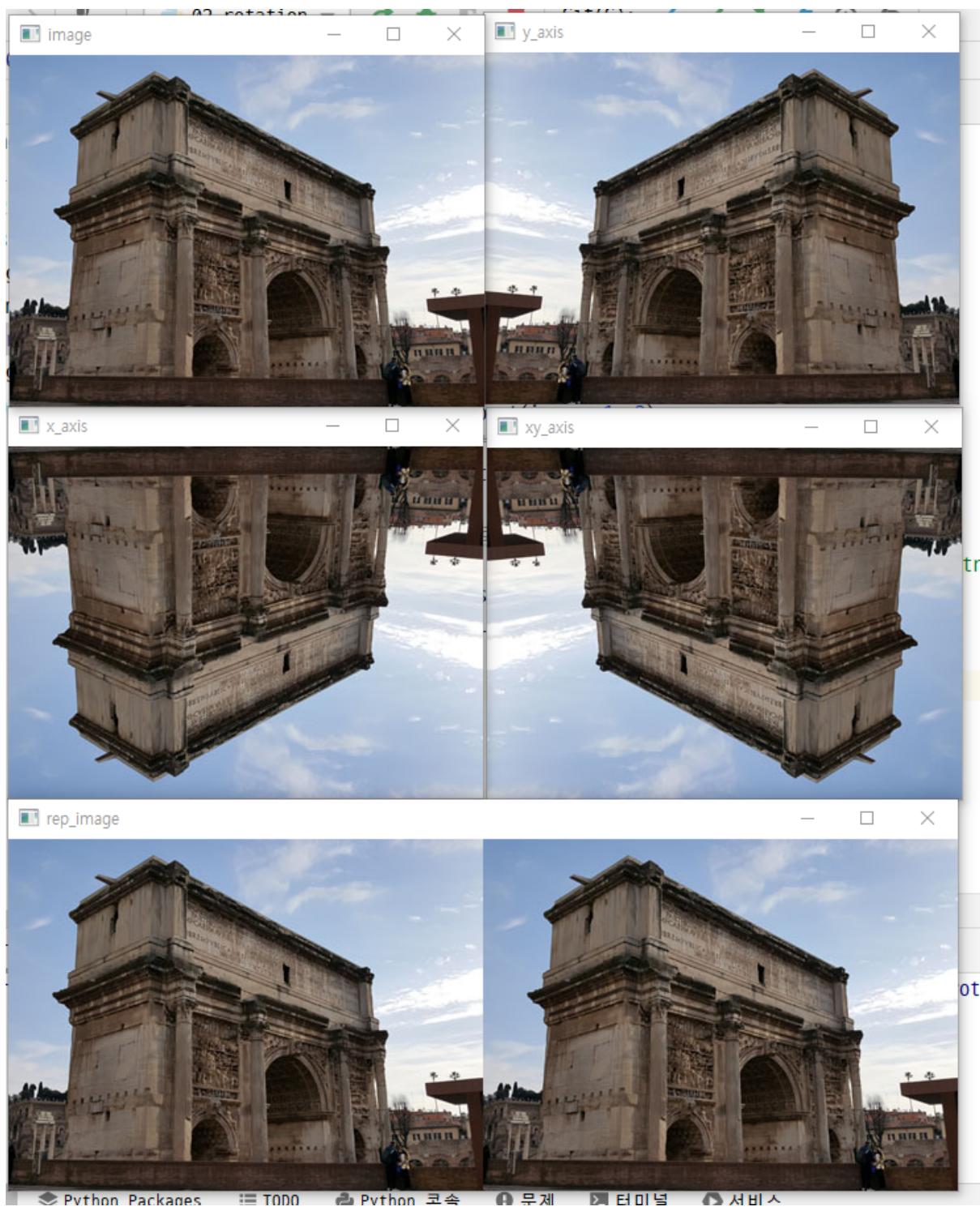
```
01_mat_array.py × 02_rotation.py ×
import cv2

image = cv2.imread("images/flip_test.jpg", cv2.IMREAD_COLOR)
if image is None:
    raise Exception("영상 파일 읽기 오류 발생")

x_axis = cv2.flip(image, 0) # x축
y_axis = cv2.flip(image, 1) # y축
xy_axis = cv2.flip(image, -1)
rep_image = cv2.repeat(image, 1, 2)
# ny, nx: 수직 방향, 수평 방향 반복 횟수
trans_image = cv2.transpose(image)

# 각 행렬을 영상으로 표시
titles = ['image', 'x_axis', 'y_axis', 'xy_axis', 'rep_image', 'trans_image']
for title in titles:
    cv2.imshow(title, eval(title))

cv2.waitKey(0)
```





키값 입력하면 창이 전부 사라짐.

03_mat_channel.py

The screenshot shows the PyCharm IDE interface. The top navigation bar has tabs for three files: 01_mat_array.py, 02_rotation.py, and 03_mat_channel.py (which is currently selected). The left sidebar shows a project structure with a src folder containing chap01, chap02, and chap03. chap03 contains images, Common, and external libraries. The right pane displays the code for 03_mat_channel.py:

```
import numpy as np
import cv2

ch0 = np.zeros((2, 4), np.uint8) + 10
ch1 = np.ones((2, 4), np.uint8) * 20
ch2 = np.zeros((2, 4), np.uint8); ch2[:] = 30
# 30으로 채워진다.

list_bar = [ch0, ch1, ch2]
merge_bgr = cv2.merge(list_bar) # 채널 합성
split_bar = cv2.split(merge_bgr) # 채널 분리 : 컬러영상 -> 3채널 분리

print('split_bgr 행렬 형태 :', np.array(split_bar).shape)
print('merge_bgr 행렬 형태 :', merge_bgr.shape)
```

The bottom panel shows the run configuration and output. It says "실행: 03_mat_channel x 17_matplotlib x". The output window shows the results of the print statements:

```
C:\ProgramData\Anaconda3\envs\tf_cpu\python.exe D:/heaven_dev/workspaces/openCV/src/chap03/03_mat_chann
split_bgr 행렬 형태 : (3, 2, 4)
merge_bgr 행렬 형태 : (2, 4, 3)
```

A status message at the bottom says "종료 코드 0(으)로 완료된 프로세스".

```
import numpy as np
import cv2

ch0 = np.zeros((2, 4), np.uint8) + 10
ch1 = np.ones((2, 4), np.uint8) * 20
ch2 = np.zeros((2, 4), np.uint8); ch2[:] = 30
# 30으로 채워지도록 함.

list_bar = [ch0, ch1, ch2]
merge_bgr = cv2.merge(list_bar) # 채널 합성
split_bgr = cv2.split(merge_bgr) # 채널 분리 : 컬러영상 -> 3채널 분리

print('split_bgr 행렬 형태 :', np.array(split_bgr).shape)
# split_bgr 행렬 형태 : (3, 2, 4)
print('merge_bgr 행렬 형태 :', merge_bgr.shape)
# merge_bgr 행렬 형태 : (2, 4, 3)

print("[ch0] = \n%s" % ch0) # 단일 채널 원소 출력
print("[ch1] = \n%s" % ch1)
print("[ch2] = \n%s" % ch2)

print('[merge_bgr] = \n %s\n' % merge_bgr) # %s 자리에 출력
# 다중 채널 원소 출력 # 2면 4행 3열로 출력됨.

print('[split_bgr[0]] = \n%s' % split_bgr[0])
print('[split_bgr[1]] = \n%s' % split_bgr[1])
print('[split_bgr[2]] = \n%s' % split_bgr[2])
```

```
C:\Python\lib\imagedata\imagechannels\test_04_
split_bgr 행렬 형태 : (3, 2, 4)
merge_bgr 행렬 형태 : (2, 4, 3)
[ch0] =
[[10 10 10 10]
 [10 10 10 10]]
[ch1] =
[[20 20 20 20]
 [20 20 20 20]]
[ch2] =
[[30 30 30 30]
 [30 30 30 30]]
[merge_bgr] =
[[[10 20 30]
 [10 20 30]
 [10 20 30]
 [10 20 30]]]

[split_bgr[0]] =
[[10 10 10 10]
 [10 10 10 10]]
[split_bgr[0]] =
[[20 20 20 20]
 [20 20 20 20]]
[split_bgr[0]] =
[[30 30 30 30]
 [30 30 30 30]]
```

04_image_channels.py

```
import cv2

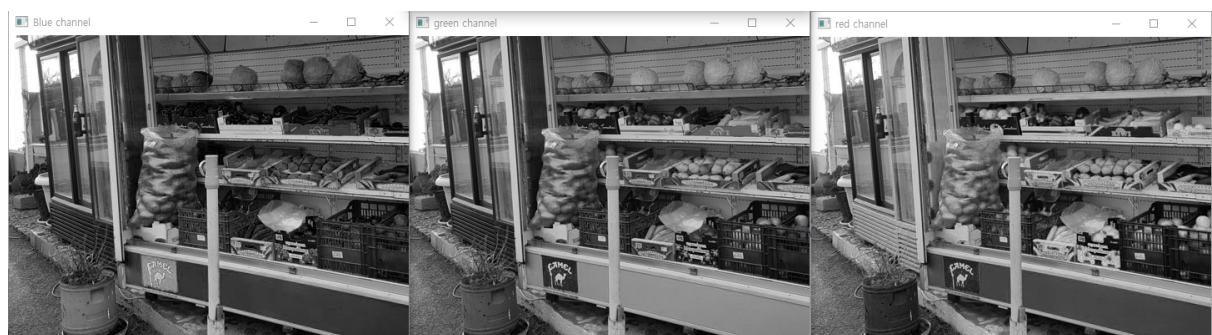
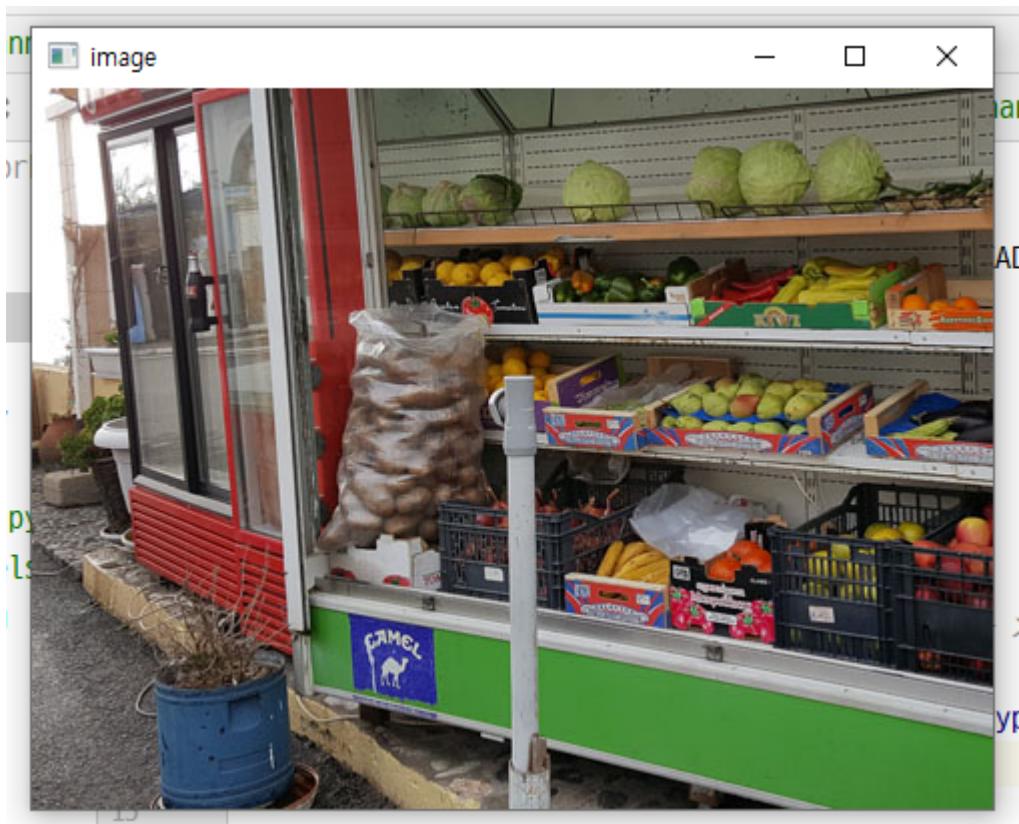
image = cv2.imread('images/color.jpg', cv2.IMREAD_COLOR)
if image is None:
    raise Exception("영상 파일 읽기 오류 발생")

if image.ndim != 3:
    raise Exception("컬러 영상 아님")

bgr = cv2.split(image) # 채널 분리: 컬러 영상 -> 3채널 분리
|
print('bgr 자료형:', type(bgr), type(bgr[0]), type(bgr[0][0][0]))
print('bgr 원소 개수:', len(bgr))

# 각 채널을 윈도우에 띄우기
cv2.imshow('image', image)
cv2.imshow('Blue channel', bgr[0])
cv2.imshow('green channel', bgr[1])
cv2.imshow('red channel', bgr[2])

cv2.waitKey(0)
```



B, G, R 각각의 색깔만 밟은 톤 - 아예 컬러에서 흑백으로 바꿔어야 하는 게 정상

05_arithmetic_op.py

Region Of Interest

arithmetic

미국·영국 [ə'riθmətik] ↗ 영국식 ↗

영사

1 산수, 연산

He's not very good at arithmetic. ↗

그는 산수를 별로 잘 하지 못한다.

2 산술, 계산

a quick bit of mental arithmetic ↗

재빠른 암산

영어사전 다른 뜻 1

```
import numpy as np, cv2

m1 = np.full((3, 6), 10, np.uint8)
m2 = np.full((3, 6), 50, np.uint8)
m_mask = np.zeros(m1.shape, np.uint8)
m_mask[:, 3:] = 1 # ROI - Region Of Interest 관심영역을 통해 강조

m_add1 = cv2.add(m1, m2)
m_add2 = cv2.add(m1, m2, mask=m_mask)
# mask 부분의 이미지를 더 강조하겠다는 것(밝은색)

# 행렬 나눗셈 수행
m_div1 = cv2.divide(m1, m2) # 정수형으로 나누기, 그래서 0
m1 = m1.astype(np.float32)
m2 = np.float32(m2)
m_div2 = cv2.divide(m1, m2)

titles = ['m1', 'm2', 'm_mask', 'm_add1', 'm_add2', 'm_div1', 'm_div2']
for title in titles:
    print("[%s] = \n%s \n" % (title, eval(title)))
```

```
C:\ProgramData\Anaconda3\envs\t1

[m1] =
[[10. 10. 10. 10. 10. 10.]
 [10. 10. 10. 10. 10. 10.]
 [10. 10. 10. 10. 10. 10.]]

[m2] =
[[50. 50. 50. 50. 50. 50.]
 [50. 50. 50. 50. 50. 50.]
 [50. 50. 50. 50. 50. 50.]]

[m_mask] =
[[0 0 0 1 1 1]
 [0 0 0 1 1 1]
 [0 0 0 1 1 1]]

[m_add1] =
[[60 60 60 60 60 60]
 [60 60 60 60 60 60]
 [60 60 60 60 60 60]]

[m_add2] =
[[ 0  0  0 60 60 60]
 [ 0  0  0 60 60 60]
 [ 0  0  0 60 60 60]]

[m_div1] =
[[0 0 0 0 0 0]
 [0 0 0 0 0 0]
 [0 0 0 0 0 0]]

[m_div2] =
[[0.2 0.2 0.2 0.2 0.2 0.2]
 [0.2 0.2 0.2 0.2 0.2 0.2]
 [0.2 0.2 0.2 0.2 0.2 0.2]]
```

06_exp_log.py

```
import numpy as np, cv2

v1 = np.array([1, 2, 3], np.float32)
v2 = np.array([[1], [2], [3]], np.float32) # 3행 1열
v3 = np.array([[1, 2, 3]], np.float32) # 1행 3열

# OpenCV 산술 연산 함수는 numpy array만 가능함
v1_exp = cv2.exp(v1)
v2_exp = cv2.exp(v2)
v3_exp = cv2.exp(v3)
log = cv2.log(v1)
sqrt = cv2.sqrt(v2) # 루트
pow = cv2.pow(v3, 3) # 거듭제곱(3승)

# 결과 출력
print("[v1] 형태: %s 원소: %s" % (v1.shape, v1))
print("[v2] 형태: %s 원소: \n%s" % (v2.shape, v2))
print("[v3] 형태: %s 원소: %s" % (v3.shape, v3))
print()

# 행렬 정보 출력 - OpenCV 결과는 행렬로 반환됨 - 행벡터는 열벡터로 반환됨
print("[v1_exp] 자료형: %s 형태: %s" % (type(v1_exp), v1_exp.shape))
print("[v2_exp] 자료형: %s 형태: %s" % (type(v2_exp), v2_exp.shape))
print("[v3_exp] 자료형: %s 형태: %s" % (type(v3_exp), v3_exp.shape))
print()

# 열벡터를 1 행에 출력하는 예시 - 행벡터로 변환
print("[log] =", log.T) # .T는 전치행렬에 대한 결과
print("[sqrt] =", np.ravel(sqrt)) # 평평하게 펴줌.
print("[pow] =", pow.flatten()) |
```

```
C:\ProgramData\Anaconda3\envs\tf_cpu\python.exe D:/head.py  
[v1] 형태: (3,) 원소: [1. 2. 3.]  
[v2] 형태: (3, 1) 원소:  
[[1.]  
 [2.]  
 [3.]]  
[v3] 형태: (1, 3) 원소: [[1. 2. 3.]]  
  
[v1_exp] 자료형: <class 'numpy.ndarray'> 형태: (3, 1)  
[v2_exp] 자료형: <class 'numpy.ndarray'> 형태: (3, 1)  
[v3_exp] 자료형: <class 'numpy.ndarray'> 형태: (1, 3)  
  
[log] = [[0.          0.6931472 1.0986123]]  
[sqrt] = [1.          1.4142135 1.7320508]  
[pow] = [ 1.   8.  27.]
```

07_bitwise_op.py

1과 0으로 계산.

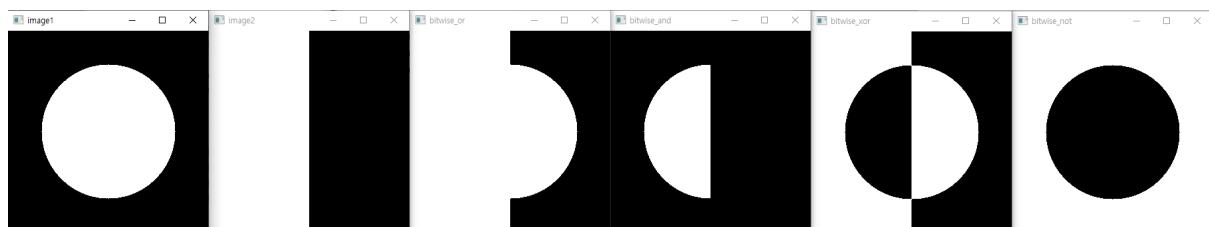
```
05_arithmetic_op.py × 06_exp_log.py × 07_bitwise_op.py >
import numpy as np, cv2

image1 = np.zeros((300, 300), np.uint8)
# 300행, 300열 검은색 영상 생성
image2 = image1.copy() # 똑같인 검은색

h, w = image1.shape[:2]
cx, cy = w//2, h//2 # cx=150, cy=150
cv2.circle(image1, (cx,cy), 100, 255, -1) # 중심에 원 그리기
cv2.rectangle(image2, (0,0, cx, h), 255, -1)

image3 = cv2.bitwise_or(image1, image2)
image4 = cv2.bitwise_and(image1, image2) # 둘 다 1일때만 반환
image5 = cv2.bitwise_xor(image1, image2)
image6 = cv2.bitwise_not(image1)

cv2.imshow("image1", image1)
cv2.imshow("image2", image2)
cv2.imshow("bitwise_or", image3)
cv2.imshow("bitwise_and", image4)
cv2.imshow("bitwise_xor", image5)
cv2.imshow("bitwise_not", image6)
cv2.waitKey(0)
```



08_mat_min_max.py

```
04_image_channels.py × 05_arithmethic_op.py × 06_exp_log.py ×
import numpy as np, cv2

data = [10, 200, 5, 7, 9,
        15, 35, 60, 80, 170,
        100, 2, 55, 37, 70]
m1 = np.reshape(data, (3, 5))
m2 = np.full((3, 5), 50)

m_min = cv2.min(m1, 30)
m_max = cv2.max(m1, m2)

## 행렬의 최솟값/최댓값과 그 좌표들을 반환
min_val, max_val, min_loc, max_loc = cv2.minMaxLoc(m1)

print("[m1] = \n%s\n" % m1)
print("[m_min] = \n%s\n" % m_min)
print("[m_max] = \n%s\n" % m_max)

# min_loc와 max_loc 좌표는 (y, x)이므로 행렬의 좌표 위치와 반대임
print("m1 행렬 최솟값 좌표 %s, 최솟값: %d" %(min_loc, min_val))
print("m1 행렬 최댓값 좌표 %s, 최댓값: %d" %(max_loc, max_val))
```

```
C:\ProgramData\Anaconda3\envs\tf_cpu\python
[m1] =
[[ 10 200   5   7   9]
 [ 15  35   60   80 170]
 [100   2  55   37  70]]

[m_min] =
[[10 30   5   7   9]
 [15 30  30  30 30]
 [30  2 30  30 30]]

[m_max] =
[[ 50 200  50  50  50]
 [ 50  50  60  80 170]
 [100  50  55  50  70]]

m1 행렬 최솟값 좌표 (1, 2), 최솟값: 2
m1 행렬 최댓값 좌표 (1, 0), 최댓값: 200
```

09_image_min_max.py

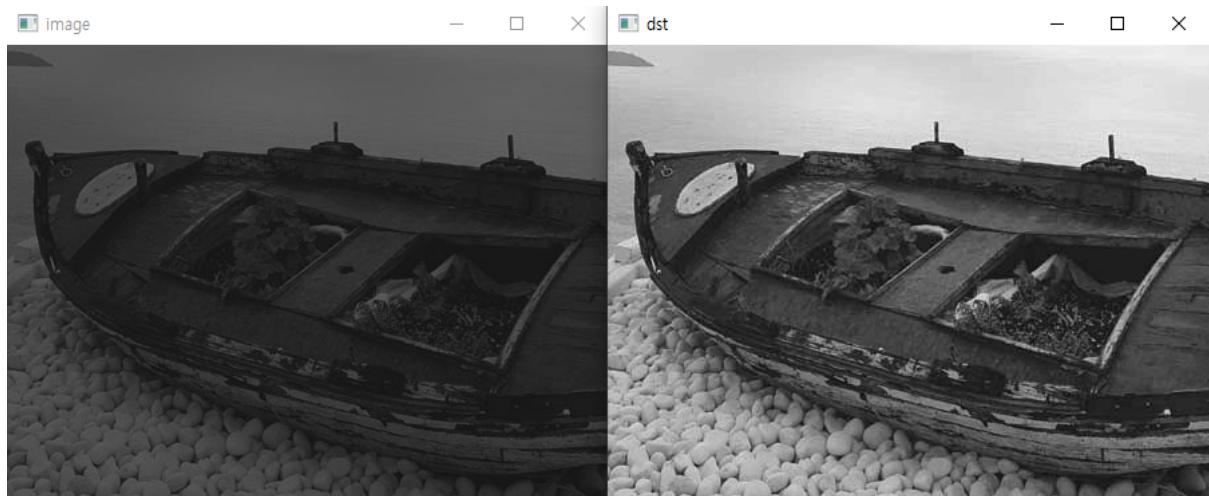
```
05_arithmethic_op.py × 06_exp_log.py × 07_bitwise_op.py ×
import numpy as np, cv2

image = cv2.imread("images/minMax.jpg", cv2.IMREAD_GRAYSCALE)
if image is None: raise Exception("영상 파일 읽기 오류 발생")

(min_val, max_val, _, _) = cv2.minMaxLoc(image)

ratio = 255/(max_val - min_val)
dst = np.round((image - min_val) * ratio).astype('uint8')
# uint8 - 양수의 값
(min_dst, max_dst, _, _) = cv2.minMaxLoc(dst)
# 1,2번째 값만 사용하려고 나머지는 매개변수는 _로 표현.

print("원본 영상 최솟값= %d , 최댓값= %d" % (min_val, max_val))
print("수정 영상 최솟값= %d , 최댓값= %d" % (min_dst, max_dst))
cv2.imshow("image", image)
cv2.imshow("dst", dst)
cv2.waitKey(0)
```



10_sort.py

```
10_sort.py × 11_mat_gemm.py × 12_point_transform.py × 13_equation.py ×
import numpy as np, cv2
from Common.utils import ck_time

m = np.random.randint(0,100, 1000000).reshape(1000,1000) # 임의 난수 생성

# 행렬 원소 정렬
ck_time(0)
sort1 = cv2.sort(m, cv2.SORT_EVERY_ROW) # 행단위 오름차순
sort2 = cv2.sort(m, cv2.SORT_EVERY_COLUMN) # 열단위(세로) 오름차순
sort3 = cv2.sort(m, cv2.SORT_EVERY_ROW + cv2.SORT_DESCENDING) # 행단위(가로) 내림차순
ck_time(1)

ck_time(0)
sort4 = np.sort(m, axis=1) # 세로축 정렬
sort5 = np.sort(m, axis=0) # 가로축 정렬
sort6 = np.sort(m, axis=1)[:, ::-1] # 가로축 내림차순 정렬
ck_time(1)

sort7 = np.sort(m, axis=0)[::-1, :]
#
# titles= ['m', 'sort1', 'sort2', 'sort3', 'sort4', 'sort5', 'sort6', 'sort7']
# for title in titles:
#     print("[%s] = \n%s\n" %(title, eval(title)))
```

```

# 수행시간 체크 함수
stime = 0
def ck_time(mode = 0 , msg = ""):
    global stime

    if (mode == 0):
        stime = time.perf_counter()

    elif (mode==1):
        etime = time.perf_counter()
        elapsed = (etime - stime)
        print("수행시간 = %.5f sec" % elapsed) # 초 단위 경과 시간

    elif (mode == 2):
        etime = time.perf_counter()
        return (etime - stime)

    elif (mode== 3 ):
        etime = time.perf_counter()
        elapsed = (etime - stime)
        print("%s = %.5f sec" %(msg, elapsed)) # 초 단위 경과 시간

```

```

C:\ProgramData\Anaconda3\
수행시간 = 0.10630 sec
수행시간 = 0.09350 sec

```

```

수행시간 = 0.11639 sec
수행시간 = 0.10581 sec
[m] =
[[13 15 62 ... 46 78 54]
 [97 46 77 ... 35 38 15]
 [40 36 15 ... 57 50 37]
 ...
 [65 67 80 ... 9 59 98]
 [ 2 43 24 ... 32 97 1]
 [63 43 55 ... 24 90 58]]

[sort1] =
[[ 0  0  0 ... 99 99 99]
 [ 0  0  0 ... 99 99 99]
 [ 0  0  0 ... 99 99 99]
 ...
 [ 0  0  0 ... 99 99 99]

```

```

[ 0  0  0 ... 99 99 99]
[ 0  0  0 ... 99 99 99]]]

[sort2] =
[[ 0  0  0 ... 0  0  0]
 [ 0  0  0 ... 0  0  0]
 [ 0  0  0 ... 0  0  0]
 ...
 [99 99 99 ... 99 99 99]
 [99 99 99 ... 99 99 99]
 [99 99 99 ... 99 99 99]]]

[sort3] =
[[99 99 99 ... 0  0  0]
 [99 99 99 ... 0  0  0]
 [99 99 99 ... 0  0  0]
 ...
 [99 99 99 ... 0  0  0]
 [99 99 99 ... 0  0  0]
 [99 99 99 ... 0  0  0]]]

[sort4] =
[[ 0  0  0 ... 99 99 99]
 [ 0  0  0 ... 99 99 99]
 [ 0  0  0 ... 99 99 99]
 ...
 [ 0  0  0 ... 99 99 99]
 [ 0  0  0 ... 99 99 99]
 [ 0  0  0 ... 99 99 99]]]

[sort5] =
[[ 0  0  0 ... 0  0  0]
 [ 0  0  0 ... 0  0  0]
 [ 0  0  0 ... 0  0  0]
 ...
 [99 99 99 ... 99 99 99]
 [99 99 99 ... 99 99 99]
 [99 99 99 ... 99 99 99]]]

[sort6] =
[[99 99 99 ... 0  0  0]
 [99 99 99 ... 0  0  0]
 [99 99 99 ... 0  0  0]
 ...
 [99 99 99 ... 0  0  0]
 [99 99 99 ... 0  0  0]
 [99 99 99 ... 0  0  0]]]

[sort7] =
[[99 99 99 ... 99 99 99]
 [99 99 99 ... 99 99 99]
 [99 99 99 ... 99 99 99]
 ...
 [ 0  0  0 ... 0  0  0]
 [ 0  0  0 ... 0  0  0]
 [ 0  0  0 ... 0  0  0]]]

```

11_mat_gemm.py

```
import numpy as np, cv2

src1 = np.array([1, 2, 3, 1, 2, 3], np.float32).reshape(2, 3) # 2x3 행렬 선언
src2 = np.array([1, 2, 3, 4, 5, 6], np.float32).reshape(2, 3)
src3 = np.array([1, 2, 1, 2, 1, 2], np.float32).reshape(3, 2) # 3x2 행렬 선언
alpha, beta = 1.0, 1.0

# 행렬의 내적을 계산해주는 함수(gemm)
dst1 = cv2.gemm(src1, src2, alpha, None, beta, flags=cv2.GEMM_1_T)
dst2 = cv2.gemm(src1, src2, alpha, None, beta, flags=cv2.GEMM_2_T)
dst3 = cv2.gemm(src1, src3, alpha, None, beta)

titles = ['src1', 'src2', 'src3', 'dst1', 'dst2', 'dst3']
for title in titles:
    print("[%s] = \n%s\n" % (title, eval(title)))
```

```
C:\ProgramData\Anaconda3\lib\site-packages\numpy\linalg\__init__.py:14: UserWarning: 
  This module is part of the NumPy API and is subject to change in
  future versions.
  from .linalg import *
[src1] =
[[1. 2. 3.]
 [1. 2. 3.]]
```

```
[src2] =
[[1. 2. 3.]
 [4. 5. 6.]]
```

```
[src3] =
[[1. 2.]
 [1. 2.]
 [1. 2.]]
```

```
[dst1] =
[[ 5.  7.  9.]
 [10. 14. 18.]
 [15. 21. 27.]]
```

```
[dst2] =
[[14. 32.]
 [14. 32.]]
```

```
[dst3] =
[[ 6. 12.]
 [ 6. 12.]]
```

❖ 행렬 연산

✓ 행렬의 곱

cv2.gemm(src1, src2, alpha, src3, beta[, dst[, flags]]) -> dst

□ 수식: $dst = \alpha \cdot src1^T \cdot src2 + \beta \cdot src3^T$

□ 파라미터

- src1, src2: 행렬 곱을 위한 두 입력 행렬(np.float32/np.float64형 2채널까지 가능)
- alpha: 행렬 곱에 대한 가중치
- src3: 행렬 곱에 더해지는 델타 행렬
- beta: src3 행렬에 곱해지는 가중치
- dst: 출력 행렬
- flags: 옵션을 조합하여 입력 행렬들을 전치

cv2.GEMM_1_T -> 1, src1을 전치

cv2.GEMM_2_T -> 2, src2을 전치

cv2.GEMM_3_T -> 4, src3을 전치

12_point_transform.py

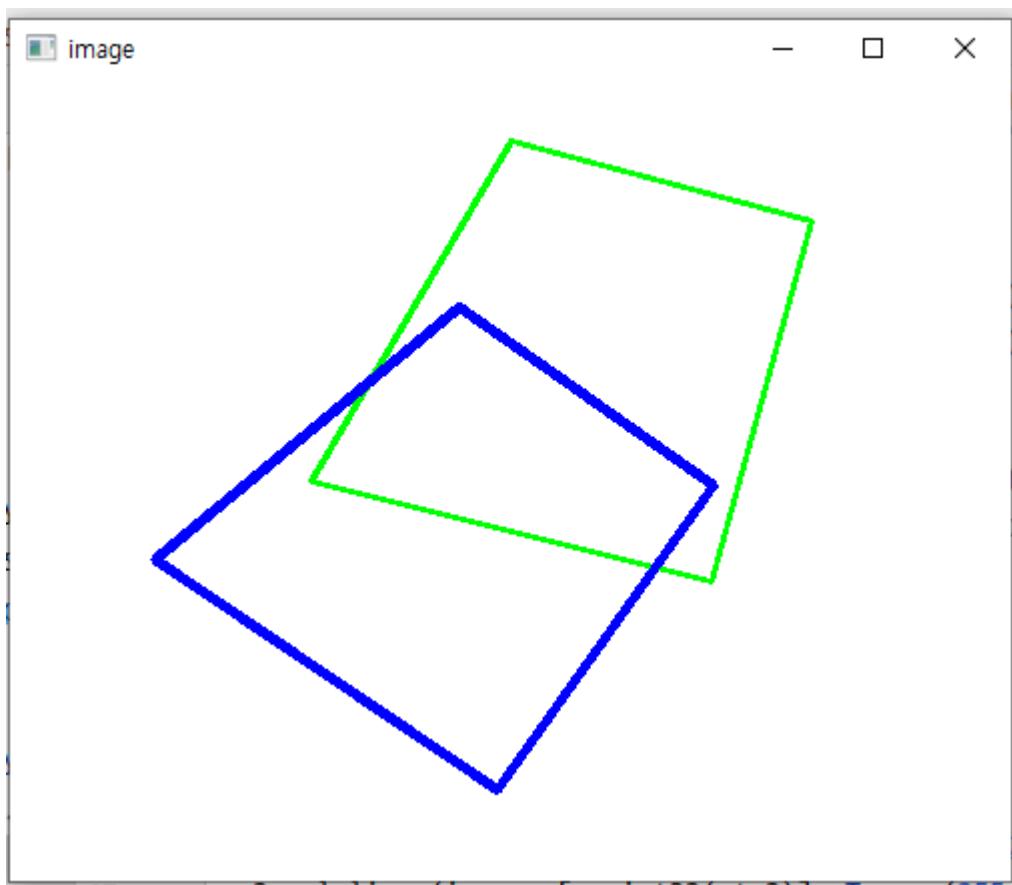
```
import numpy as np, cv2

theta = 20 * np.pi / 180 # 회전할 라디안 각도 계산
rot_mat = np.array([[np.cos(theta), -np.sin(theta)],
                    [np.sin(theta), np.cos(theta)]], np.float32) # 넘파이 행렬 생성

pts1 = np.array([(250, 30), (400, 70),
                 (350, 250), (150, 200)], np.float32)
pts2 = cv2.gemm(pts1, rot_mat, 1, None, 1, flags=cv2.GEMM_2_T)

for i, (pt1, pt2) in enumerate(zip(pts1, pts2)):
    print("pts1[%d] = %s, pts2[%d]= %s" %(i, pt1, i, pt2))

## 영상 생성 및 4개의 좌표 그리기
image = np.full((400, 500, 3), 255, np.uint8)
cv2.polyline(image, [np.int32(pts1)], True, (0, 255, 0), 2)
cv2.polyline(image, [np.int32(pts2)], True, (255, 0, 0), 3)
cv2.imshow("image", image)
cv2.waitKey(0)
```



그린에서 파란색으로 움직이는 것임

첫 기준은 왼쪽 상단에서 두번째 모양은 첫번째 좌표를 기준.

```
C:\ProgramData\Anaconda3\envs\tt_cpu\python.exe D:/heaver  
pts1[0] = [250. 30.], pts2[0]= [224.66255 113.695816]  
pts1[1] = [400. 70.], pts2[1]= [351.93564 202.58655]  
pts1[2] = [350. 250.], pts2[2]= [243.38737 354.63022]  
pts1[3] = [150. 200.], pts2[3]= [ 72.54986 239.24155]
```

13_equation.py

역행렬 = `equation`

equation

미국·영국 [r'kweɪʒn] 영국식 [r'keɪʒn]

(명사)

1 방정식, 등식

2 동일시

The equation of wealth with happiness can be dangerous.

부와 행복의 동일시는 위험할 수 있다.

3 (여러 가지 요소들을 고려해야 하는) 상황[문제]

When children enter the equation, further tensions may arise within a marriage.

자녀 문제까지 개입되면 결혼 생활 내에 긴장감이 추가적으로 발생할 수 있다.

decomposition

(명사)

the decomposition of organic waste

유기물 폐기물의 분해

```
import numpy as np, cv2

data = [-3, 0, 6, -3, 4, 2, -5, -1, 9]
m1 = np.array(data, np.float32).reshape(3,3)
m2 = np.array([36, 10, 28], np.float32)

# 역행렬 계산
ret, inv = cv2.invert(m1, cv2.DECOMP_LU)
# decomposition(분해) # DECOMP_LU 는 디폴트 옵션
if ret: # 역행렬이 존재할 경우
    dst1 = inv.dot(m2) # 넘파이에 의한 m2 내적을 곱함.
    dst2 = cv2.gemm(inv, m2, 1, None, 1) # cv에 의한 내적 계산.
    ret, dst3 = cv2.solve(m1, m2, cv2.DECOMP_LU)
    # 연립방정식에 대한 풀이 결과

    print("[inv] = \n%s\n" % inv)
    print("[dst1] =", dst1.flatten())
    print("[dst2] =", dst2.flatten())
    print("[dst3] =", dst3.flatten())
else:
    print("역행렬이 존재하지 않습니다.")
```

```
C:\ProgramData\Anaconda3\envs\tf_cpu\python
[inv] =
[[ 0.15079366 -0.02380952 -0.0952381 ]
 [ 0.06746032  0.22619048 -0.0952381 ]
 [ 0.09126984  0.01190476  0.04761905]]

[dst1] = [2.5238097 2.0238097 4.7380953]
[dst2] = [2.5238097 2.0238097 4.7380953]
[dst3] = [2.5238094 2.0238094 4.7380953]
```

```
if ret: # 역행렬이 존재할 경우
    dst1 = inv.dot(m2) # 넘파이에 의한 m2 내적을 곱함.
    dst2 = cv2.gemm(inv, m2, 1, None, 1) # cv에 의한 내적 계산.
    ret, dst3 = cv2.solve(m1, m2, cv2.DECOMP_LU)
    # 연립방정식에 대한 풀이 결과

[dst1] = [2.5238097 2.0238097 4.7380953]
[dst2] = [2.5238097 2.0238097 4.7380953]
[dst3] = [2.5238094 2.0238094 4.7380953]
```

