CHAPTER 04
인터페이스 기초

김 찬, 윤 영 선

ckim.esw@gmail.com, ysyun@hnu.kr

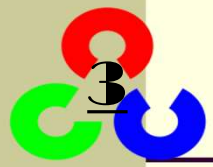
정보통신공학과

4. 인터페이스 기초

4.3 그리기 함수

4.3.1 직선 및 사각형 그리기

```
import numpy as np
import cv2

blue, green, red = (255, 0, 0), (0, 255, 0), (0, 0, 255)
image = np.zeros((400, 600, 3), np.uint8)
image[:] = (255, 255, 255)

pt1, pt2 = (50, 50), (250, 150)
pt3, pt4 = (400, 150), (500, 50)
roi = (50, 200, 200, 100)
```

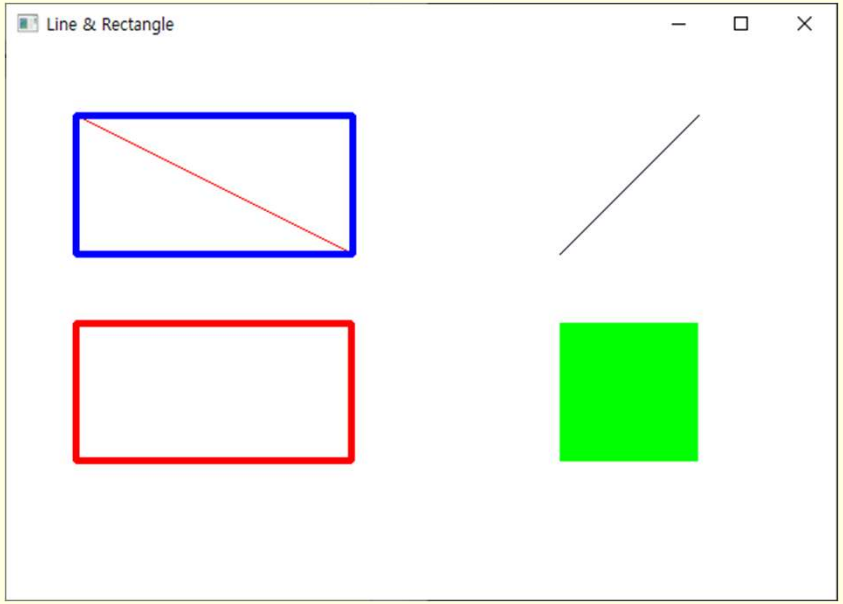


4. 인터페이스 기초

■ 4.3 그리기 함수

◆ 4.3.1 직선 및 사각형 그리기

```
# 사각형 그리기
cv2.rectangle(image, pt1, pt2, blue, 3, cv2.LINE_4)
cv2.rectangle(image, roi, red, 3, cv2.LINE_8)
cv2.rectangle(image, (400, 200, 100, 100), green, cv2.FILLED)

# 직선 그리기
cv2.line(image, pt1, pt2, red)
cv2.line(image, pt3, pt4, cv2.LINE_AA)

cv2.imshow("Line & Rectangle", image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```



4. 인터페이스 기초

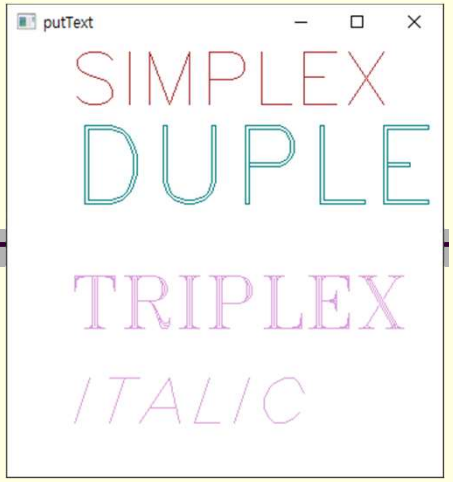
■ 4.3 그리기 함수

◆ 4.3.2 글자 쓰기

```
import numpy as np
import cv2

...

cv2.imshow('putText', image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

- 항상 고정이라고 생각 해 두면 편함.

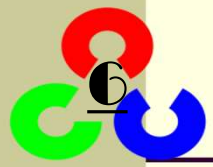
4. 인터페이스 기초



4.3 그리기 함수

4.3.2 글자 쓰기

```
olive, violet, brown = (128, 128, 0), (221, 160, 221), (42, 42, 165)
pt1, pt2 = (50, 230), (50, 310)

image = np.zeros((350, 350, 3), np.uint8)
image.fill(255)

cv2.putText(image, 'SIMPLEX', (50, 50), cv2.FONT_HERSHEY_SIMPLEX, 2, brown)
cv2.putText(image, 'DUPLEX', (50, 130), cv2.FONT_HERSHEY_DUPLEX, 3, olive)
cv2.putText(image, 'TRIPLEX', pt1, cv2.FONT_HERSHEY_TRIPLEX, 2, violet)

fontFace = cv2.FONT_HERSHEY_PLAIN | cv2.FONT_ITALIC
cv2.putText(image, 'ITALIC', pt2, fontFace, 4, violet)
```



4. 인터페이스 기초

■ 4.3 그리기 함수

◆ 4.3.3 원 그리기

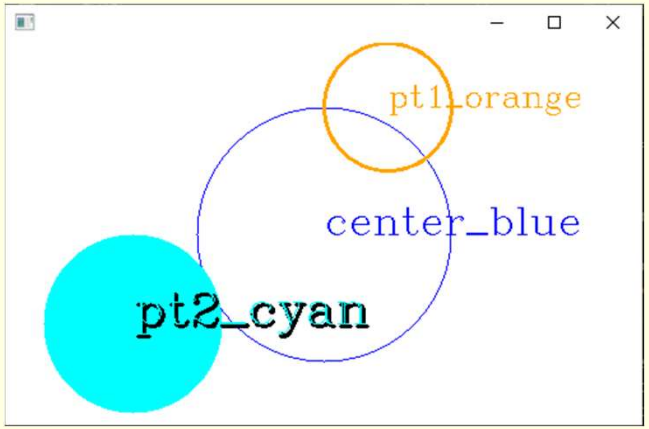
```
orange, blue, cyan = (0, 165, 255), (255, 0, 0), (255, 255, 0)
white, black = (255, 255, 255), (0, 0, 0)

image = np.full((300, 500, 3), white, np.uint8)

center = (image.shape[1]//2, image.shape[0]//2)
pt1, pt2 = (300, 50), (100, 220)
shade = (pt2[0] + 2, pt2[1] + 2)

cv2.circle(image, center, 100, blue)
cv2.circle(image, pt1, 50, orange, 2)
cv2.circle(image, pt2, 70, cyan, -1)
```

4. 인터페이스 기초

■ 4.3 그리기 함수

◆ 4.3.3 원 그리기

```
font = cv2.FONT_HERSHEY_COMPLEX

cv2.putText(image, 'center_blue', center, font, 1.0, blue)
cv2.putText(image, 'pt1_orange', pt1, font, 0.8, orange)
cv2.putText(image, 'pt2_cyan', shade, font, 1.2, black, 2)
# 폰트 두께를 더 두껍게하고, 위치를 약간 옮김으로 그림자 효과 생성
cv2.putText(image, 'pt2_cyan', pt2, font, 1.2, cyan, 1)
```

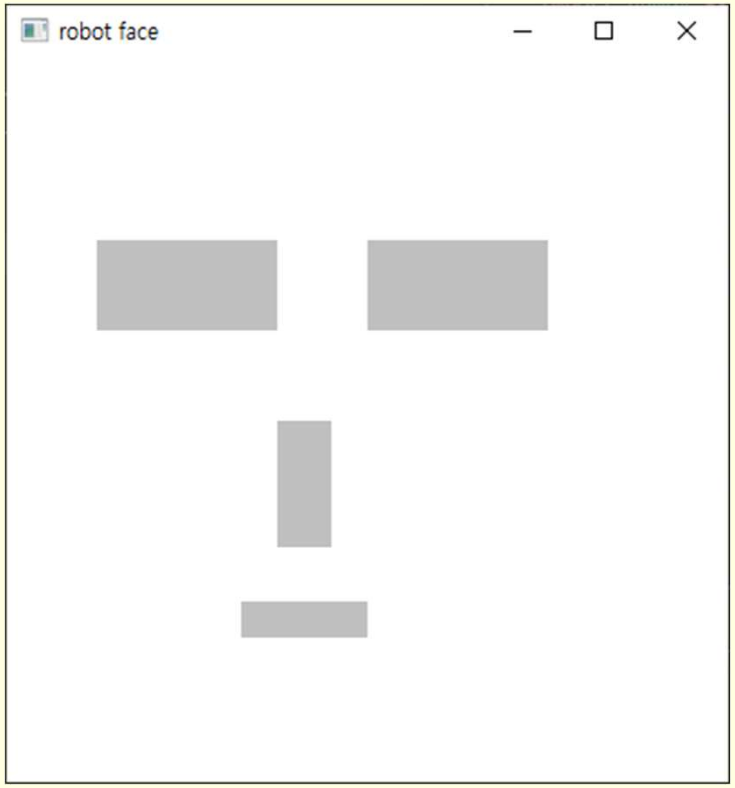


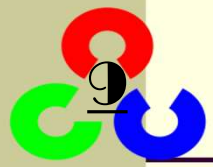
4. 인터페이스 기초

4.3 그리기 함수

4.3.4 얼굴 그리기

```
image = np.zeros((400, 400), np.uint8)
image.fill(255)

eye = np.full((50, 100), 192, np.uint8)
image[100:150, 50:150] = eye
image[100:150, 200:300] = eye

nouse = np.full((70, 30), 192, np.uint8)
image[200:270, 150:180] = nouse

mouse = np.full((20, 70), 192, np.uint8)
image[300:320, 130:200] = mouse
```





4. 인터페이스 기초

4.3 그리기 함수

4.3.5 마우스로 얼굴 그리기

```
import numpy as np
import cv2

def onMouse():
...

image = np.full((300, 300, 3), (255, 255, 255), np.uint8)
pt = (-1, -1)
title = "Draw Event"

cv2.imshow(title, image)
cv2.setMouseCallback(title, onMouse)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

4. 인터페이스 기초

4.3 그리기 함수

4.3.5 마우스로 얼굴 그리기(왼쪽 클릭)

```
def onMouse(event, x, y, flags, param):
    global title, pt

    if event == cv2.EVENT_LBUTTONDOWN:
        if pt[0] < 0:
            pt = (x, y)
        else:
            cv2.rectangle(image, pt, (x, y), (255, 0, 0), 2)
            cv2.imshow(title, image)
            pt = (-1, -1)
```

4. 인터페이스 기초

4.3 그리기 함수

4.3.5 마우스로 얼굴 그리기(오른쪽 클릭)

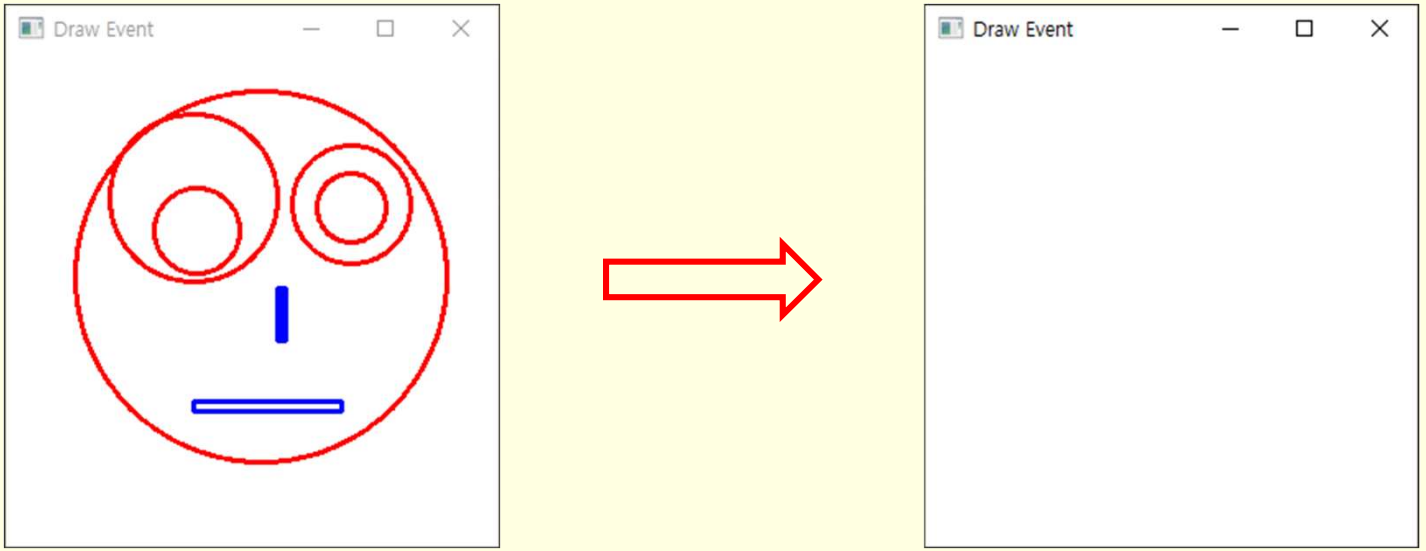

```
def onMouse(event, x, y, flags, param):

    ...

    elif event == cv2.EVENT_RBUTTONDOWN:
        if pt[0] < 0:
            pt = (x, y)
        else:
            dx, dy = pt[0] - x, pt[1] - y
            radius = int(np.sqrt(dx*dx + dy*dy))
            cv2.circle(image, pt, radius, (0, 0, 255), 2)
            cv2.imshow(title, image)
            pt = (-1, -1)
```

4. 인터페이스 기초

4.3 그리기 함수

과제1

- 조건:
  - 마우스 휠 을 조작 했을 때, 초기화 (지우개 역할) 하라.



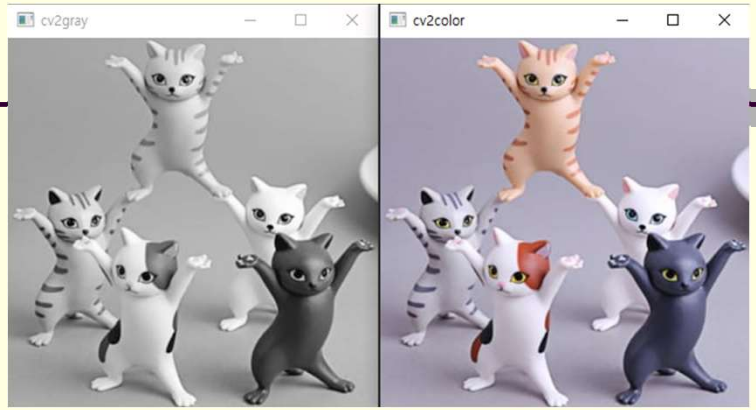
4. 인터페이스 기초

4.4 영상파일 처리

4.4.1 영상파일 읽기

```
import cv2

title1, title2 = 'cv2gray', 'cv2color'
cv2gray = cv2.imread('/images/test_image.jpg', cv2.IMREAD_GRAYSCALE)
cv2color = cv2.imread('/images/test_image.jpg', cv2.IMREAD_COLOR)

cv2.imshow(title1, cv2gray)
cv2.imshow(title2, cv2color)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

4. 인터페이스 기초



4.4 영상파일 처리

4.4.1 영상파일 읽기

```
import cv2

image_file_path = 'D:/github/OpenCV-Python/2022-10-04/images/test_image.jpg'

title1, title2 = 'cv2gray', 'cv2color'
cv2gray = cv2.imread(image_file_path, cv2.IMREAD_GRAYSCALE)
cv2color = cv2.imread(image_file_path, cv2.IMREAD_COLOR)

cv2.imshow(title1, cv2gray)
cv2.imshow(title2, cv2color)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

상대경로가 아닌, 절대경로를 넣어야 함

4. 인터페이스 기초

4.4 영상파일 처리

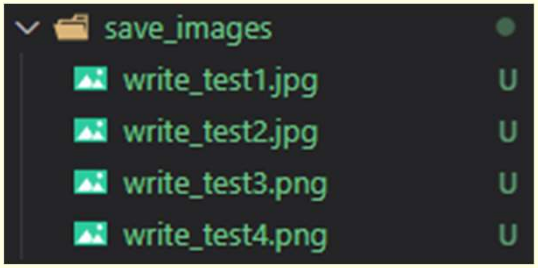
4.4.2 영상파일 저장

```
import cv2

image_file_path = 'D:/github/OpenCV-Python/2022-10-04/images/test_image.jpg'
save_file_path = 'D:/github/OpenCV-Python/2022-10-04/save_images/'

image = cv2.imread(image_file_path, cv2.IMREAD_COLOR)

params_jpg = (cv2.IMWRITE_JPEG_QUALITY, 10)     # JPEG 화질 설정
params_png = (cv2.IMWRITE_PNG_COMPRESSION, 9)   # PNG 압축율 설정
```

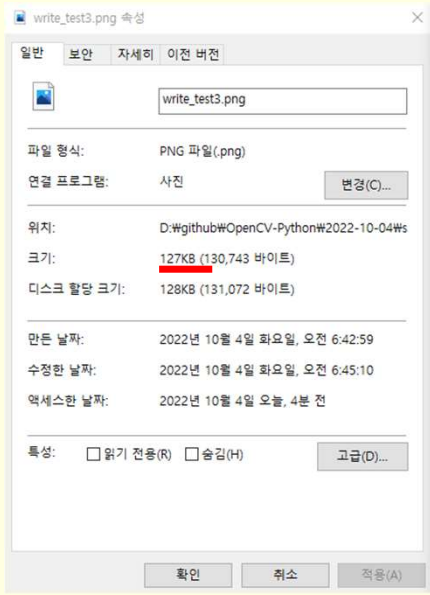
4. 인터페이스 기초

4.4 영상파일 처리

4.4.2 영상파일 저장



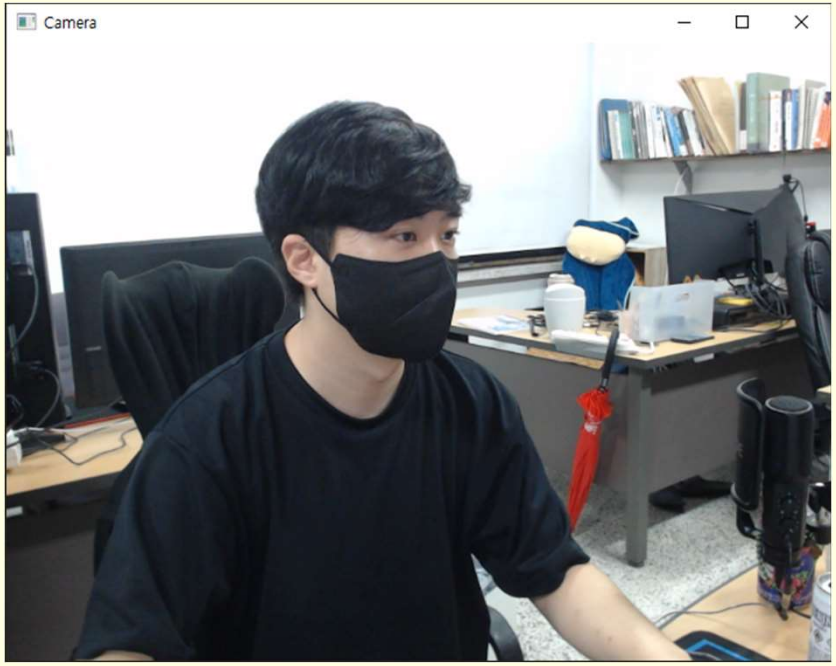

```
cv2.imwrite(f'{save_file_path}write_test1.jpg', image)
cv2.imwrite(f'{save_file_path}write_test2.jpg', image, params_jpg)
cv2.imwrite(f'{save_file_path}write_test3.png', image)
cv2.imwrite(f'{save_file_path}write_test4.png', image, params_png)

print('저장완료')
```



4. 인터페이스 기초

4.5 비디오 처리

4.5.1 카메라에서 프레임 읽기




```
import cv2

capture = cv2.VideoCapture(0)
while True:
    ret, frame = capture.read()
    if cv2.waitKey(30) >= 0:
        break # 스페이스바로 종료

    exposure = capture.get(cv2.CAP_PROP_EXPOSURE) # 노출 속성
    title = 'Camera'
    cv2.imshow(title, frame)
capture.release()
```

4. 인터페이스 기초

4.5 비디오 처리

◆ 4.5.2 카메라 프레임을 동영상 파일로 저장

```
import cv2

capture = cv2.VideoCapture(0)

fps = 29.97
delay = round(1000/fps)
size = (640, 360)
fourcc = cv2.VideoWriter_fourcc(*'DX50’)

## 카메라 속성을 실행창에 출력
print('width x height: ',size)
print('VideoWriterfourcc: ', fourcc)
print(f'delay: {delay}')
print('fps: ',fps)
```

4. 인터페이스 기초

4.5 비디오 처리

4.5.2 카메라 프레임을 동영상 파일로 저장

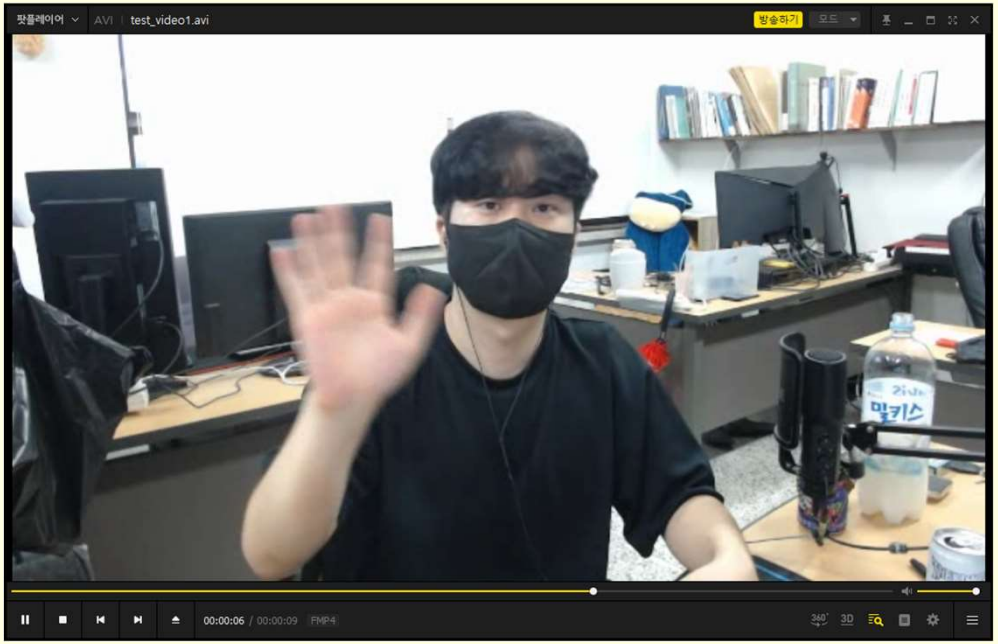
```
## 카메라 속성 지정
capture.set(cv2.CAP_PROP_ZOOM, 1)
capture.set(cv2.CAP_PROP_FOCUS, 0)
capture.set(cv2.CAP_PROP_FRAME_WIDTH, size[0])
capture.set(cv2.CAP_PROP_FRAME_HEIGHT, size[1])


## 동영상파일 개방 및 코덱, 해상도 설정
writer = cv2.VideoWriter('D:/github/OpenCV-Python/2022-10-
04/save_videos/test_video1.avi', fourcc, fps, size)
```

4. 인터페이스 기초

■ 4.5 비디오 처리

◆ 4.5.2 카메라 프레임을 동영상 파일로 저장

```
while True:
    ret, frame = capture.read()
    if cv2.waitKey(30) >= 0:
        break # 스페이스바로 종료

    writer.write(frame)

    title = 'Camera'
    cv2.imshow(title, frame)

writer.release()  # 쓰기 종료
capture.release() # 카메라 종료
```



4. 인터페이스 기초

4.5 비디오 처리

4.5.3 동영상 파일 읽기

```
import cv2

capture = cv2.VideoCapture('D:/github/OpenCV-Python/2022-10-
04/save_videos/test_video1.avi')

fps = capture.get(cv2.CAP_PROP_FPS)
delay = int(1000/fps)
fps_cnt = 0 # 현재 프레임 번호
```

4. 인터페이스 기초

4.5 비디오 처리

4.5.3 동영상 파일 읽기

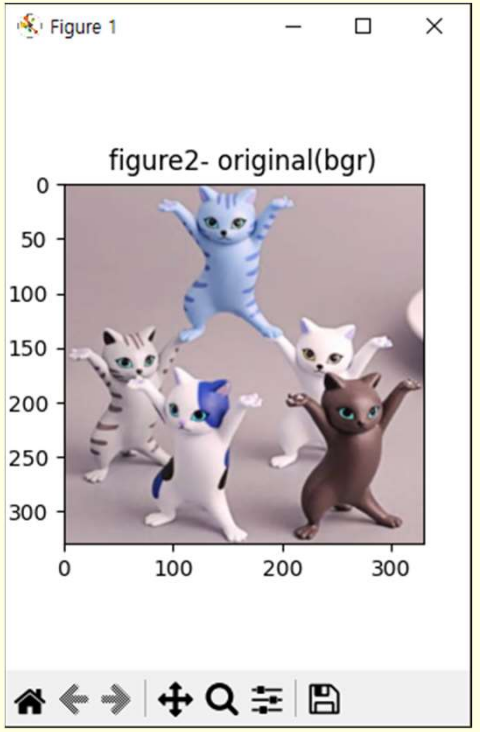
```
while True:
    ret, frame = capture.read()
    if cv2.waitKey(30) >= 0:
        break # 스페이스바로 종료

    fps_cnt += 1
    print(fps_cnt)

    title = 'Camera'
    cv2.imshow(title, frame)

capture.release()
```

4. 인터페이스 기초

4.6 matplotlib

4.6.1 matplotlib로 이미지 출력하기

```
import cv2
import matplotlib.pyplot as plt

image = cv2.imread("절대경로", cv2.IMREAD_COLOR)

print(image.shape)
```



4. 인터페이스 기초

4.6 matplotlib

4.6.1 matplotlib로 이미지 출력하기

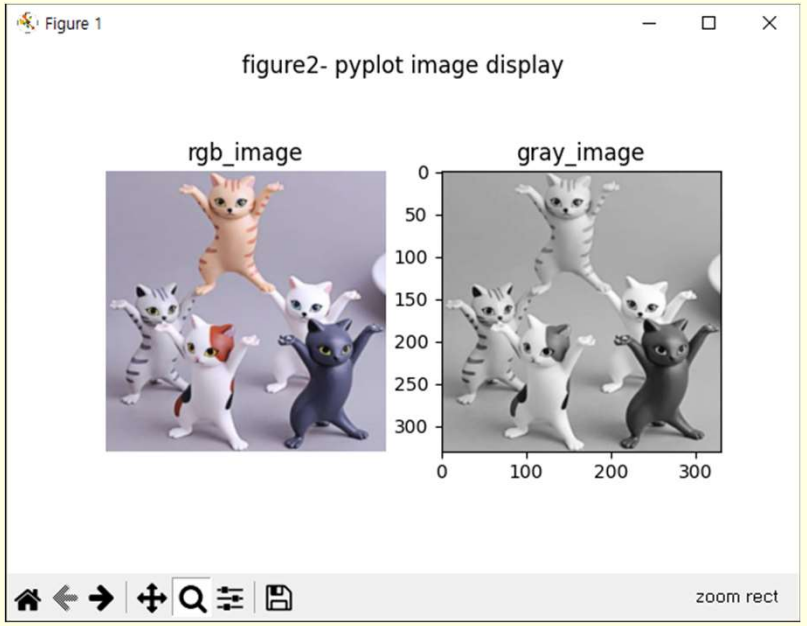
```
import cv2
import matplotlib.pyplot as plt

image = cv2.imread("절대경로", cv2.IMREAD_COLOR)

print(image.shape)

plt.figure(figsize=(3,4))
plt.imshow(image)
plt.title('figure2- original(bgr)')
plt.axis('off')
plt.show()
```

4. 인터페이스 기초

4.6 matplotlib

◆ 4.6.2 matplotlib로 이미지 두개 동시에 출력하기

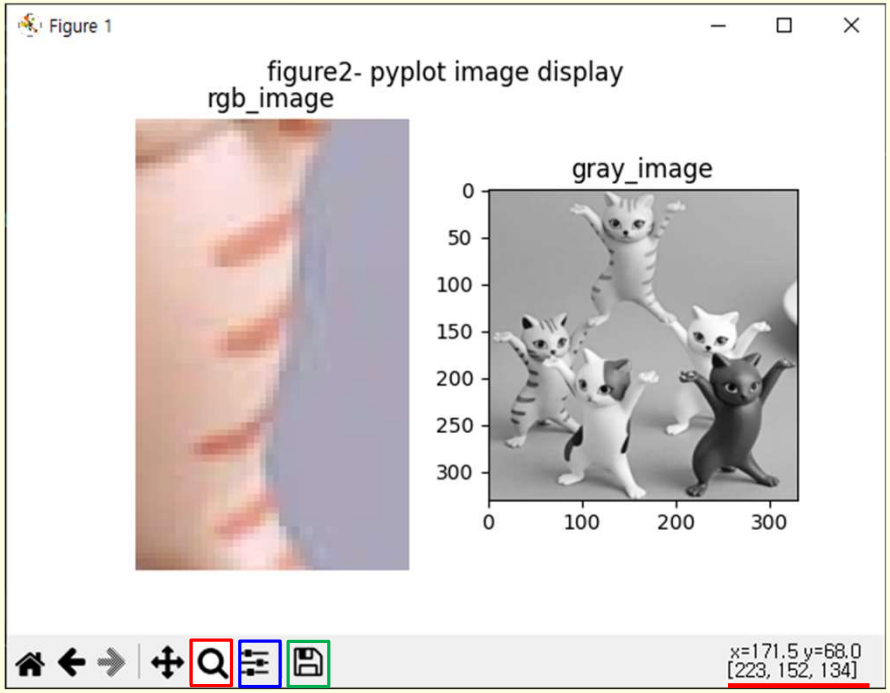
```
rgb_img = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
gray_img = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

plt.figure(figsize=(6,4))
plt.suptitle('figure2- pyplot image display')
plt.subplot(1, 2, 1) # 1행 2열의 첫번째 이미지
plt.title('rgb_image')
plt.axis('off')
plt.imshow(rgb_img)

plt.subplot(1, 2, 2) # 1행 2열의 두번째 이미지
plt.title('gray_image')
plt.imshow(gray_img, cmap='gray')
plt.show()
```



4. 인터페이스 기초

4.6 matplotlib

◆ 4.6.2 matplotlib 기능 들



확대/축소기능 이미지 저장

세부 설정

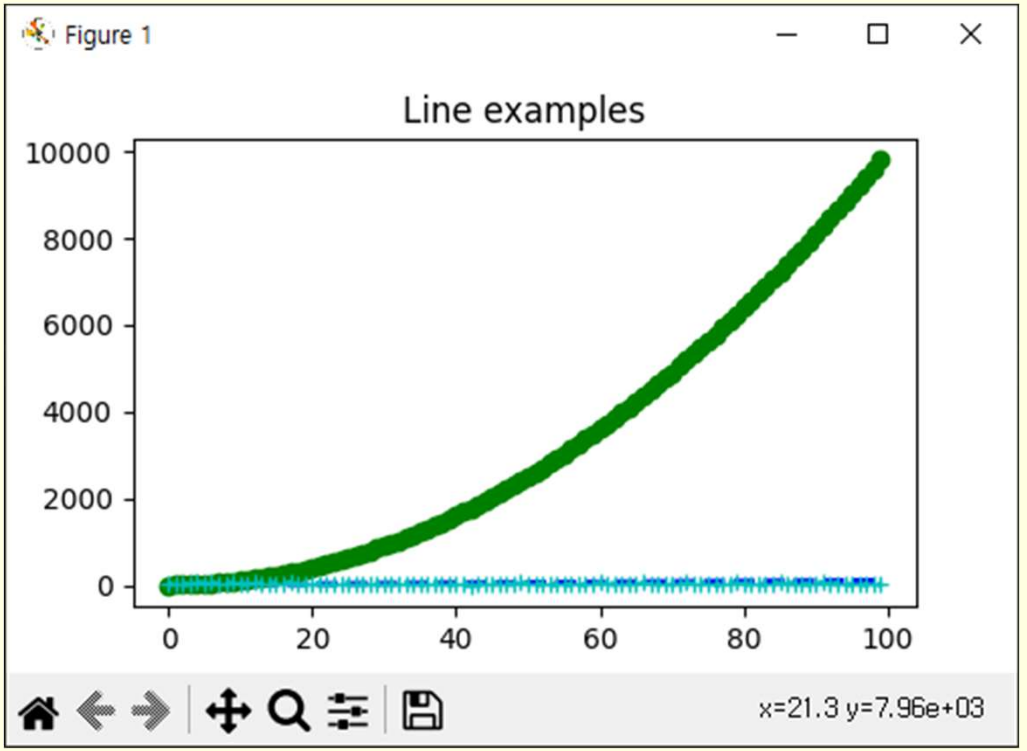
현재 위치 r, g, b 컬러 값

4. 인터페이스 기초

4.6 matplotlib

◆ 4.6.3 matplotlib로 그래프 그리기

```
import matplotlib.pyplot as plt
import numpy as np

x = np.arange(100)
y1 = np.arange(100)
y2 = np.arange(100)**2
y3 = np.random.choice(50, size=100)

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

plt.title('Line examples')
plt.show()
```

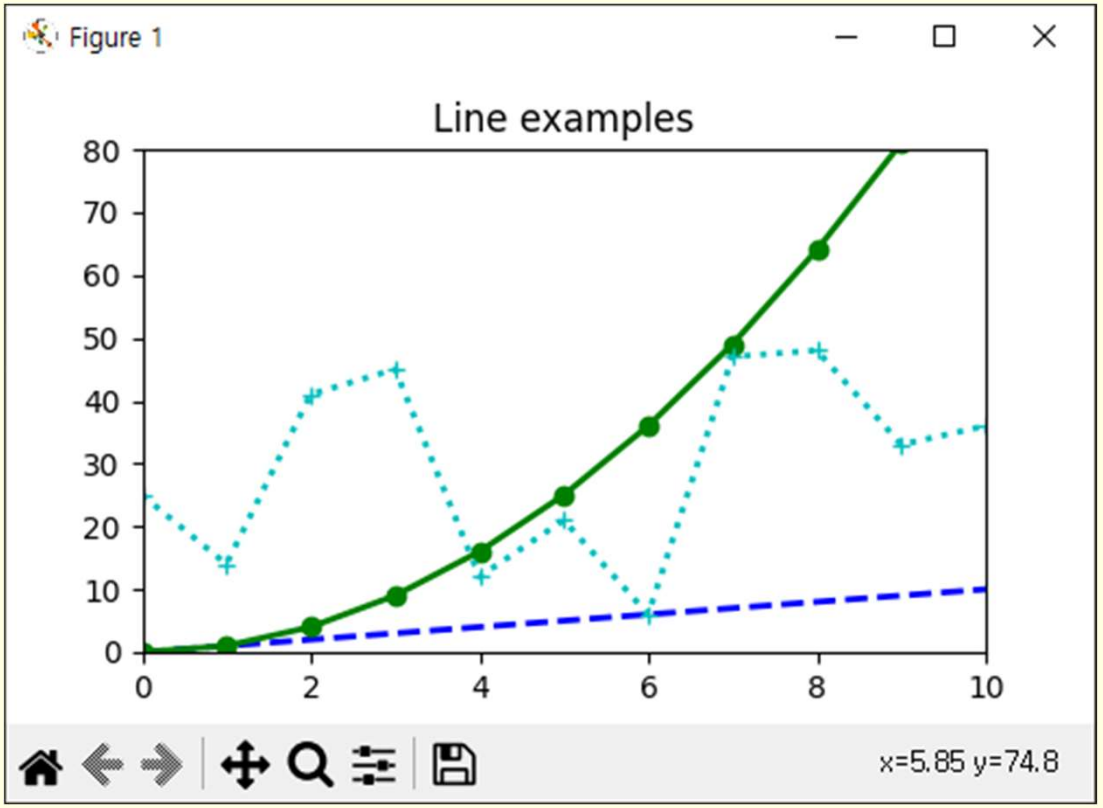


4. 인터페이스 기초

4.6 matplotlib

◆ 4.6.3 matplotlib로 그래프 그리기

```
...

plt.axis([0,10, 0,80])
plt.show()
```



CHAPTER 05
기본 배열 연산 함수

김 찬, 윤 영 선

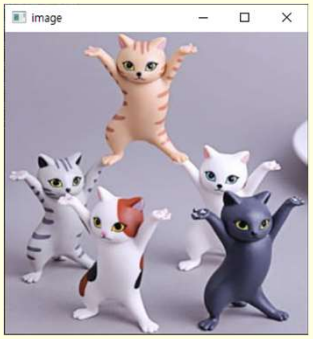
ckim.esw@gmail.com, ysyun@hnu.kr

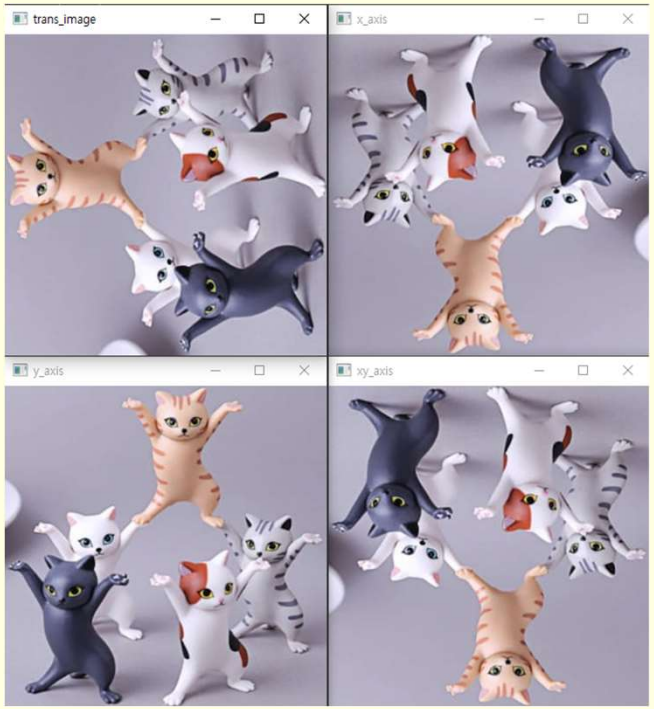
정보통신공학과

5. 기본 배열 연산

5.1 기본 배열 처리 함수

5.1.1 행렬 처리 함수






```
import cv2

image = cv2.imread('절대경로', cv2.IMREAD_COLOR)

x_axis = cv2.flip(image, 0)
y_axis = cv2.flip(image, 1)
xy_axis = cv2.flip(image, -1)
trans_image = cv2.transpose(image)

titles = ['image', 'x_axis', 'y_axis', 'xy_axis', 'trans_image']
for title in titles:
    cv2.imshow(title, eval(title))
cv2.waitKey(0)
```

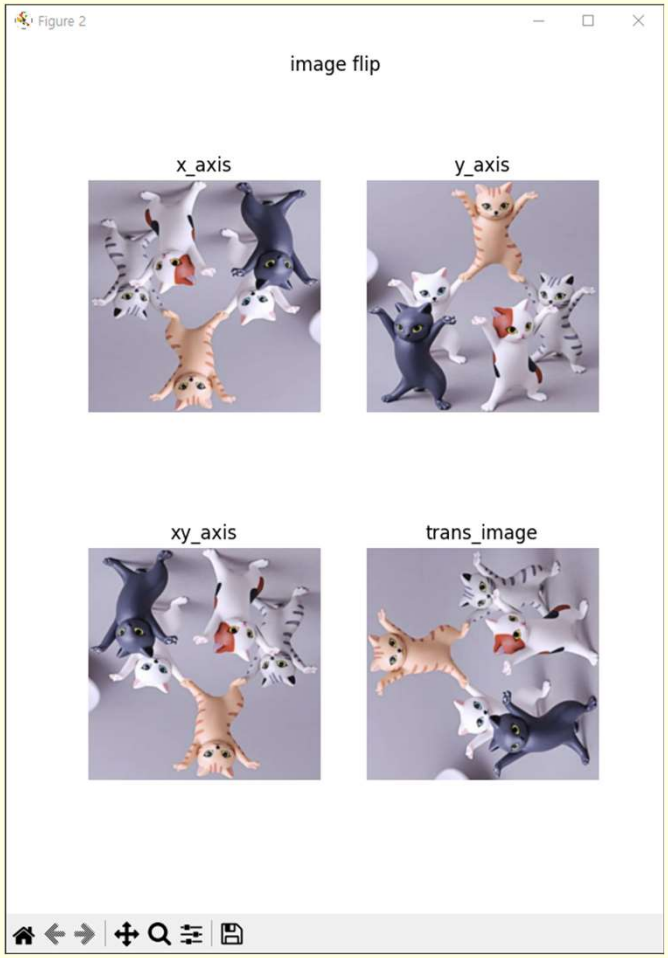
5. 기본 배열 연산

5.1 기본 배열 처리 함수

◆ 과제2

- 조건:
  - OpenCV가 아닌 matplotlib을 이용하여 두 개의 Figure를 출력하라.

- 세부조건:
  - 이미지1
    - 원본 이미지
  - 이미지2
    - 축 변경을 통한 이미지 4개를 한번에 출력



5. 기본 배열 연산

5.1 기본 배열 처리 함수

◆5.2.2 컬러 채널 분리

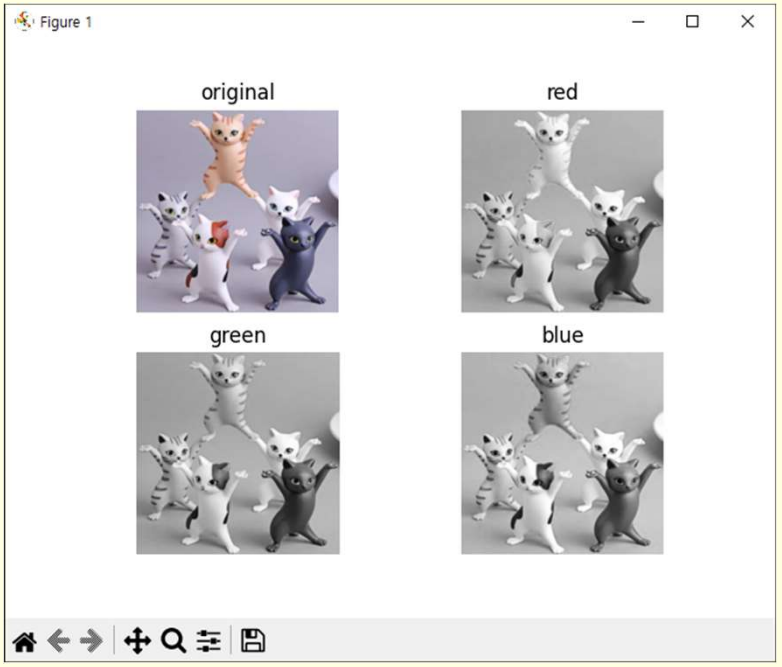
```
import cv2
import matplotlib.pyplot as plt

image = cv2.imread('절대경로', cv2.IMREAD_COLOR)
image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
red, green, blue = cv2.split(image)

plt.subplot(2, 2, 1) # 2행 2열의 1번째 이미지
plt.title('original')
plt.axis('off')
plt.imshow(image)
```

5. 기본 배열 연산

5.1 기본 배열 처리 함수

5.2.2 컬러 채널 분리

```
plt.subplot(2, 2, 2) # 2행 2열의 2번째 이미지
plt.title('red')
plt.axis('off')
plt.imshow(red, cmap='gray')

plt.subplot(2, 2, 3) # 2행 2열의 3번째 이미지
plt.title('green')
plt.axis('off')
plt.imshow(green, cmap='gray')

plt.subplot(2, 2, 4) # 2행 2열의 4번째 이미지
plt.title('blue')
plt.axis('off')
plt.imshow(blue, cmap='gray')
plt.show()
```



5. 기본 배열 연산

5.1 기본 배열 처리 함수

5.2.2 Tip!

- 파이썬 내장함수 사용으로 코드 재사용

```
titles = ['original', 'red', 'green', 'blue']
images = [image, red, green, blue]
cmaps = [None, 'gray', 'gray', 'gray']

for idx, (title, img, c) in enumerate(zip(titles, images, cmaps)):
    plt.subplot(2, 2, idx+1) # 2행 2열의 이미지 순서
    plt.title(title)
    plt.axis('off')
    plt.imshow(img, cmap=c)
```