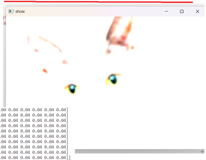
이성규 수강생 과제제출

코드

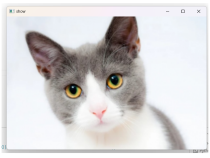
출력값

```python
1206 > ex01커널집접정의.py > ...
'''
커널을 직접 정의한 후 filter2D() 메소드를 통해 이미지에 적용하는 것도
생성된 커널을 이미지에 적용 시 중앙 원소가 변환되는 픽셀이며, 나머지는
'''
import cv2
from utils import image_show
import numpy as np

# 이미지 경로
image_path = "./cat.png"

# 이미지 읽기 처리
image = cv2.imread(image_path)

# 커널 생성 처리
kernel = np.ones((10,10)) / 25.0 # 모두 더하면 1이 되도록 정규화
image_kernel = cv2.filter2D(image, -1, kernel)
print(kernel)
image_show(image_kernel)
```

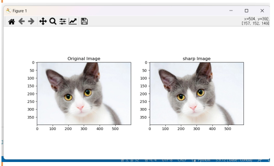


```
[[0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04]
 [0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04]
 [0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04]
 [0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04]
 [0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04]
 [0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04]
 [0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04]
 [0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04]
 [0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04]
 [0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04 0.04]]
```

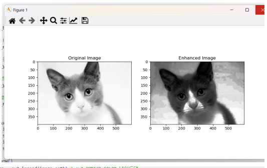
```python
import cv2
from utils import image_show
import numpy as np

# 이미지 경로
image_path = "./cat.png"

# 이미지 읽기 처리
image = cv2.imread(image_path)

# 가우시안 블러
image_very_blurry = cv2.GaussianBlur(image,(5,5),0)
image_show(image_very_blurry)
```

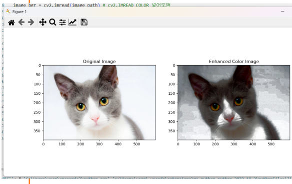


```python
import cv2
import numpy as np
import matplotlib.pyplot as plt

image_path = "./cat.png"

# 이미지 선명하게 ( 커널 생성 : 대상이 있는 픽셀을 강조)
# 컬러 이미지 로드
image_bgr = cv2.imread(image_path, cv2.IMREAD_COLOR)

#RGB 타입으로 변환
image_rgb = cv2.cvtColor(image_bgr, cv2.COLOR_BGR2RGB)

# 커널 생성
kernel = np.array([[0 ,-1, 0],
                   [-1, 5,-1],
                   [0 ,-1, 0]])

# 커널 적용
image_sharp = cv2.filter2D(image_rgb, -1, kernel)

fig, ax = plt.subplots(1, 2, figsize=(10,5))
ax[0].imshow(image_rgb)
ax[0].set_title("Original Image")
ax[1].imshow(image_sharp)
ax[1].set_title("sharp Image")
plt.show()
```

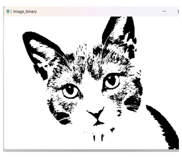


```python
1206 > ex04이미지대비높이기.py > ...
import cv2
import matplotlib.pyplot as plt

image_path = "./cat.png"

# 흑백 이미지 대비 높이기
image_gray = cv2.imread(image_path, cv2.IMREAD_GRAYSCALE)
image_enhanced = cv2.equalizeHist(image_gray)

# plot
fig, ax = plt.subplots(1, 2, figsize=(10,5))
ax[0].imshow(image_gray, cmap='gray')
ax[0].set_title("Original Image")
ax[1].imshow(image_enhanced, cmap='gray')
ax[1].set_title("Enhanced Image")

plt.show()

################################################################

# 컬러 이미지 대비 높이기
# 방법 : RGB -> YUV 컬러 포맷으로 변환 -> equlizeHist() -> RGB
# BGR
image_bgr = cv2.imread(image_path) # cv2.IMREAD_COLOR 넣어도댐

# RGB 타입으로 변환
image_rgb = cv2.cvtColor(image_bgr, cv2.COLOR_BGR2RGB) # matplot

# YUV 컬러 포멧으로 변환
image_yuv = cv2.cvtColor(image_rgb, cv2.COLOR_RGB2YUV)
# 히스토그램 평활화 적용
image_yuv[:,:,0] = cv2.equalizeHist(image_yuv[:,:,0])

# RGB로 변경
image_rgb_temp = cv2.cvtColor(image_yuv, cv2.COLOR_YUV2RGB)

# plot
fig, ax = plt.subplots(1, 2, figsize=(12,8))
ax[0].imshow(image_rgb)
ax[0].set_title("Original Image")
ax[1].imshow(image_rgb_temp)
ax[1].set_title("Enhanced Color Image")

plt.show()
```

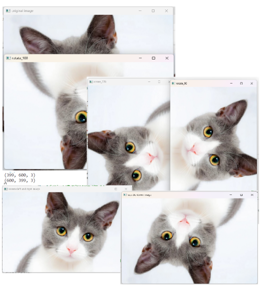




```python
1206 > ex05이미지이진화.py > ...
이는 이미지 안의 영역마다 빛 조건이 달라질 때 도움이 된다.
'''
# 이진화를 할때 그레이로 해야함. 컬러로 하면 적용이 안됨

import cv2
from utils import image_show

# 이미지 경로
image_path = "./cat.png"

# 이미지 읽기
image = cv2.imread(image_path)

# 이미지 이진화
image_gray = cv2.imread(image_path, cv2.IMREAD_GRAYSCALE)
max_output_value = 255 # 출력 픽셀 강도의 최대값
neighborhood_size = 99
subtract_from_mean = 10 # 평균값 ( 높으면 하얀색이 비중 높아짐 )

image_binary = cv2.adaptiveThreshold(image_gray,
                                     max_output_value,
                                     cv2.ADAPTIVE_THRESH_GAUSSIAN_C,
                                     cv2.THRESH_BINARY, neighborhood_size,
                                     subtract_from_mean) # cv2.THRESH_BINA

image_show(image_binary,"Image_binary")
```

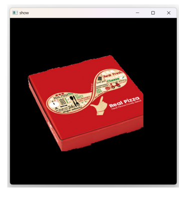


```python
# 이미지 경로
image_path = "./cat.png"

# 이미지 읽기
image = cv2.imread(image_path)

img90 = cv2.rotate(image, cv2.ROTATE_90_CLOCKWISE) # /
img180 = cv2.rotate(image, cv2.ROTATE_180) # 180도 회전
img270 = cv2.rotate(image, cv2.ROTATE_90_COUNTERCLOCKW

# 회전은 크기를 변화시키지는 않습니다.
print(image.shape)
print(img90.shape)

cv2.imshow("original image", image)
cv2.imshow("rotate_90", img90)
cv2.imshow("rotate_180", img180)
cv2.imshow("rotate_270", img270)

cv2.waitKey(0)

# 이미지 좌우 및 상하 반전
# Flip(이미지, key) : key 1은 좌우 0은 상하

dst_temp1 = cv2.flip(image,1)
dst_temp2 = cv2.flip(image,0)

cv2.imshow("reverse left and right image", dst_temp1)
cv2.imshow("upside down image", dst_temp2) # 상하반전
cv2.waitKey(0)
```

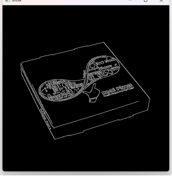


```
(399, 600, 3)
(600, 399, 3)
```

```python
1206 > ex07이미지배경 제거.py > ...
# 이미지 경로
image_path ='./pizza.png'
# 이미지 읽기
image = cv2.imread(image_path)

# 사각형 좌표 : 시작점의 x, y , 넓이, 높이
rectangle = (0, 0, 400, 400)

# 초기 마스크 생성
mask = np.zeros(image.shape[:2], np.uint8)

# grabCut에 사용할 임시 배열 생성
bgdModel = np.zeros((1, 65), np.float64)
fgdModel = np.zeros((1, 65), np.float64)

# grabCut 실행
cv2.grabCut(image,    # 원본 이미지
            mask,     # 마스크
            rectangle,
            bgdModel, # 사각형
            fgdModel, # 배경을 위한 임시 배열
            5,        # 반복 횟수
            cv2.GC_INIT_WITH_RECT) # 사각형을 위한 초기화

# 배경인 곳은 0, 그 외에는 1로 설정한 마스크 생성
mask_2 = np.where((mask == 2) | ( mask == 0), 0, 1).astype('uint8')

# 이미지에 새로운 마스크를 곱해서 -> 배경을 제거
image_nobg = image * mask_2[:,:,np.newaxis]
image_show(image_nobg)
```



```python
import cv2
import numpy as np
from utils import image_show

# 이미지 읽기
image = cv2.imread('./pizza.png')

# 경계선 찾기 ( canny 할 땐 항상 그레이로 해야함 )
image_gray = cv2.imread('./pizza.png', cv2.IMREAD_GRAYSCALE)

# 픽셀 강도의 중간값을 계산
mdeian_intensity = np.median(image_gray) # 이 이미지의 중간값을 찾음
print(mdeian_intensity)

# 중간 픽셀 강도에서 위아래 1표준편차 떨어진 값을 임계값으로 설정
lower_threshold = int(max(0, (1.0 - 0.33) * mdeian_intensity ))
upper_threshold = int(min(255, (1.0 + 0.33) * mdeian_intensity))

# Canny edge Detection 적용
Image_canny = cv2.Canny(image_gray, lower_threshold, upper_threshold)
image_show(Image_canny)
```



```python
ex09이미지 모서리 감지.py > ...

import cv2
import numpy as np
from utils import image_show

image_path = './edge.png'

# 이미지 읽기
image_read = cv2.imread(image_path)
print(image_read.shape) # (523, 862, 3)

# 모서리 찾기
image_gray = cv2.cvtColor(image_read, cv2.COLOR_BGR2GRAY)
image_gray = np.float32(image_gray)

block_size = 2 # 모서리 감지 매개변수를 설정
aperture = 29
free_parameter = 0.04 # 이웃한 픽셀과 얼마만큼 근접히 할것인지.

detector_response = cv2.cornerHarris( image_gray, block_size, aperture, free_parameter )

# 임계값보다 큰 감지 결과만 남기고 흰색으로 표시합니다.
threshold = 0.02
image_read[detector_response > threshold * detector_response.max()] = [255,255,255]

image_gray = cv2.cvtColor(image_read, cv2.COLOR_BGR2GRAY) # 흑백으로 변환
image_show(image_gray,"edge(spot)")
```