이성규 수강생 과제제출

코드 / 출력값

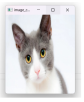
206 > ex10 이미지 모서리 감지된거 원 칠하기.py > ...

```python
# 감지 된 모서리에 흰색 원 처리하기
import cv2
import numpy as np
from utils import image_show

image_path = './edge.png'
image_read = cv2.imread(image_path)
image_gray = cv2.cvtColor(image_read, cv2.COLOR_BGR2GRAY)

# 감지할 모서리 개수
corners_to_detect = 4
minimum_quality_score = 0.05
mininum_distance = 25

# 모서리 감지
corners = cv2.goodFeaturesToTrack(image_gray, corners_to_detect,minimum_quality_score,mininum_distance)

for corner in corners:
    x, y = corner[0]
    cv2.circle(image_read, (int(x), int(y)), 10, (0,255,0), -1) # 모서리마다 회색 원을 그림(뒷배경이 하얀/

image_gray_temp =cv2.cvtColor(image_read, cv2.COLOR_BGR2GRAY) # 흑백 이미지로 변환

image_show(image_gray_temp,"check edge with circle")
```

```python
import cv2
from utils import image_show

image_gray = cv2.imread('./cat.png', cv2.IMREAD_GRAYSCALE)

image_10x10 = cv2.resize(image_gray, (10, 10)) # 이미지를 10x10 픽셀 크기로 변환
image_10x10.flatten() # 이미지 데이터를 1차원 벡터로 변환

image_show(image_10x10, "image_10x10")
```

06 > ex11머신러닝 특성 만들기(추가).py > ...

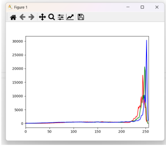
```python
import cv2
from utils import image_show

image = cv2.imread('./cat.png')

# image 10x10 픽셀 크기로 변환
image_color_10x10 = cv2.resize(image, (10,10))
image_shape_info = image_color_10x10.flatten().shape
image_color_10x10.flatten() # flatten이란 다차원 배열을 1차원 공간으로 변경
image_show(image_color_10x10 , "image_color_10x10")

# image 225x255 픽셀 크기로 변환
image_color_225x255 = cv2.resize(image, (225, 255))
image_color_225x255.flatten()
image_show(image_color_225x255 , "image_color_225x255")

print(image_color_225x255.shape)
# (255, 225, 3)
print(image_color_225x255.flatten().shape) # 이미지 데이터를 1차원 벡터로 
# (172125,)
```




```
(255, 225, 3)
(172125,)
```

1206 > ex12컬러 히스토그램을 특성으로 인코딩.py > ...

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

image_path = './cat.png'

image_bgr = cv2.imread(image_path)
image_rgb = cv2.cvtColor(image_bgr, cv2.COLOR_BGR2RGB)
features = [] # 특성 값을 담을 리스트
colors = ('r', 'g', 'b') # 각 컬러 채널에 대해 히스토그램을 계산

# 각 채널을 반복하면서 히스토그램을 계산하고 리스트에 추가
for i, channel in enumerate(colors):
    histogram = cv2.calcHist([image_rgb], # 이미지
                             [i], # 채널 인덱스
                             None, # 마스크. ( 없으므로 None )
                             [256], # 히스토그램 크기
                             [0,256] ) # 범위
    plt.plot(histogram, color=channel)
    plt.xlim([0, 256])

plt.show()
```



1206 > ex12평균 색을 특성으로 인코딩.py > ...

```python
# 평균색 특성 인코딩
import cv2
import numpy as np

image = cv2.imread('./cat.png')

channels = cv2.mean(image) # 각 채널의 평균 계산
print("channels :",channels)
# channels : (205.8482456140351, 205.4207560568087, 207.63455722639935, 0.0)


# 파랑과 빨강을 바꿉니다 ( BGR -> RGB )
observation = np.array([(channels[2], channels[1], channels[0])])
print("observation : ",observation)
# observation :  [[207.63455723 205.42075606 205.84824561]]
```

```
channels : (205.8482456140351, 205.4207560568087, 207.63455722639935, 0.0)
observation :  [[207.63455723 205.42075606 205.84824561]]
```

206 > exbase01이미지선명화(블러).py > ...

```python
# 기본적인 이미지 처리 기술을 이용한 이미지 선명화 -1

import cv2
import numpy as np

img = cv2.imread('./car.png' , 0)
print(img.shape)
img = cv2.resize(img, (320, 240)) # 노트북 해상도 이슈

blurred_1 = np.hstack([
    cv2.blur(img,(3,3)),
    cv2.blur(img,(5,5)),
    cv2.blur(img,(9,9))
])

cv2.imshow('blurred_1', blurred_1)
cv2.imwrite('./blur.png', blurred_1)
cv2.waitKey(0)
```

```
(640, 960)
```

blur.png

1206 > exbase02이미지선명화(가우시안 블러 필터).py > ...

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

img = cv2.imread('./car.png' , 0)
print(img.shape)
img_rsize = cv2.resize(img, (320, 240)) # 노트북 해상도

Gaussian_blurred_1 = np.hstack([
    cv2.GaussianBlur(img_rsize, (3, 3), 0),
    cv2.GaussianBlur(img_rsize, (5, 5), 0),
    cv2.GaussianBlur(img_rsize, (9, 9), 0),
]) # 홀수만 가능.

image_show(Gaussian_blurred_1,"Gaussian_blurred_1")
cv2.imwrite('gaussian_blur.png',Gaussian_blurred_1)
```

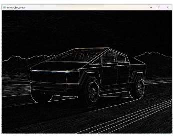
gaussian_blur.png

206 > exbase03이미지선명화(cv2.filter2D).py > ...

```python
#  cv2.filter2D()를 활용하여 필터 적용

import cv2
import numpy as np
from utils import image_show

image = cv2.imread('./car.png' )

# Creating out sharpening filter
filter = np.array([[-1,-1,-1] , [-1,9,-1], [-1,-1,-1]])

sharpen_img = cv2.filter2D(image, -1, filter)

# 비교해보기
cv2.imshow("org image", image)
image_show(sharpen_img, "sharpen image")
```

206 > exbase04이미지선명화(샤프닝필터).py > ...

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

image = cv2.imread('./car.png' )

# Creating maxican hot filter
# 5x5 필터
filter = np.array([[0,0,-1,0,0] , [0,-1,-2,-1,0], [-1,-2,16,-2,-1], [0,-1,-2,-1,0], [0,0,-1,0,0]])

# 3x3 필터
# filter = np.array([[-1, -1, -1], [-1, 8, -1], [-1, -1, -1]])

# Applying cv2.filter2D function on our Cybertruck image
mexican_hot_image = cv2.filter2D(image, -1, filter)

image_show(mexican_hot_image, "mexican_hot_image")
# 저장해서 비교 가능
# cv2.imwrite('./mexican_hat_5x5.png', mexican_hat_image)
```

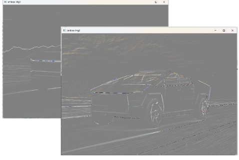


206 > exbase05이미지선명화(커스텀필터).py > ...

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

image = cv2.imread('./car.png' )

# 커스텀 필터 제작
filter = np.array([[-1, -8, -1 ], [-8 , 38, -8], [-1, -8, -1 ]])

custom_image_filter = cv2.filter2D(image, -1, filter)

image_show(custom_image_filter, "custom_image_filter")
```

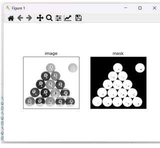
1206 > exbase07이미지선명화(엠보스필터).py > ...

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

image = cv2.imread('./car.png' )

# 엠보싱 효과
filter1 = np.array([[0, 1, 0],
                    [0, 0, 0],
                    [0,-1, 0]])

filter2 = np.array([[-1,-1, 0],
                    [-1, 0, 1],
                    [ 0, 1, 1]])


emboss_img1 = cv2.filter2D(image,-1, filter1)
emboss_img2 = cv2.filter2D(image,-1, filter2)
# 어두우므로 회색추가
emboss_img1 = emboss_img1 + 128 # 128이 회색
emboss_img2 = emboss_img2 + 128

# 비교해보기
cv2.imshow("emboss_img1", emboss_img1)
image_show(emboss_img2, "emboss_img2")
```



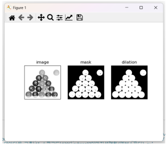
206 > exbase08이미지 팽창 및 침식.py > ...

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

# image loading and input image -> gray
img = cv2.imread('./Billiards.png', cv2.IMREAD_GRAYSCALE)

# 임계값 연산자의 출력을 마스크라는 변수에 저장
# 230보다 작으면 모든 값을 흰색 처리/ 230 보다 큰 모든 값은 검은색 처리
_, mask = cv2.threshold(img, 230, 255, cv2.THRESH_BINARY_INV)

'''
우리가 사용하는 추가 매개변수는 임계 강도 픽셀 값이며, 이 경우에는 230과 255 값으로 설정됩니
이는 230보다 큰 모든 값이 255 값으로 설정됨을 의미합니다. 우리가 사용하는 마지막 매개변수는 
단순히 값을 반전시키는 임계값 알고리즘 THRESH_BINARY_INV입니다
(230보다 작은 모든 값은 흰색이 되고 230보다 큰 모든 값은 검은 색이 됩니다.)
'''

titles =['image', 'mask']
images = [img, mask]

for i in range(2) :
    plt.subplot(1, 2, i+1)
    plt.imshow(images[i],'gray')
    plt.title(titles[i])
    plt.xticks([])
    plt.yticks([])
plt.show()
```



1206 > exbase09이미지 팽창 및 침식 2.py > ...

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

# image loading and input image -> gray
img_gray = cv2.imread('./Billiards.png', cv2.IMREAD_GRAYSCALE)
_, mask = cv2.threshold(img_gray, 230, 255, cv2.THRESH_BINARY_INV)

# 3x3 커널 생성
kernel = np.ones((3,3), np.uint8)

dilation = cv2.dilate(mask, kernel)
'''
[[1 1 1]
 [1 1 1]
 [1 1 1]]
'''
titles = ['image', 'mask', 'dilation']
images = [img_gray, mask, dilation]

for i in range(3):
    plt.subplot(1, 3, i+1)
    plt.imshow(images[i], 'gray')
    plt.title(titles[i])
    plt.xticks([])
    plt.yticks([])
plt.show()
```



1206 > exbase10이미지 팽창 및 침식 3.py > ...

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

img_gray = cv2.imread('./Billiards.png', cv2.IMREAD_GRAYSCALE)
_, mask = cv2.threshold(img_gray, 230, 255, cv2.THRESH_BINARY_INV)

# 3x3 커널 생성
kernel = np.ones((3,3), np.uint8)

dilation = cv2.dilate(mask, kernel, iterations=10)

titles = ['image', 'mask', 'dilation']
images = [img_gray, mask, dilation]

for i in range(3):
    plt.subplot(1, 3, i+1)
    plt.imshow(images[i], 'gray')
    plt.title(titles[i])
    plt.xticks([])
    plt.yticks([])
plt.show()
```