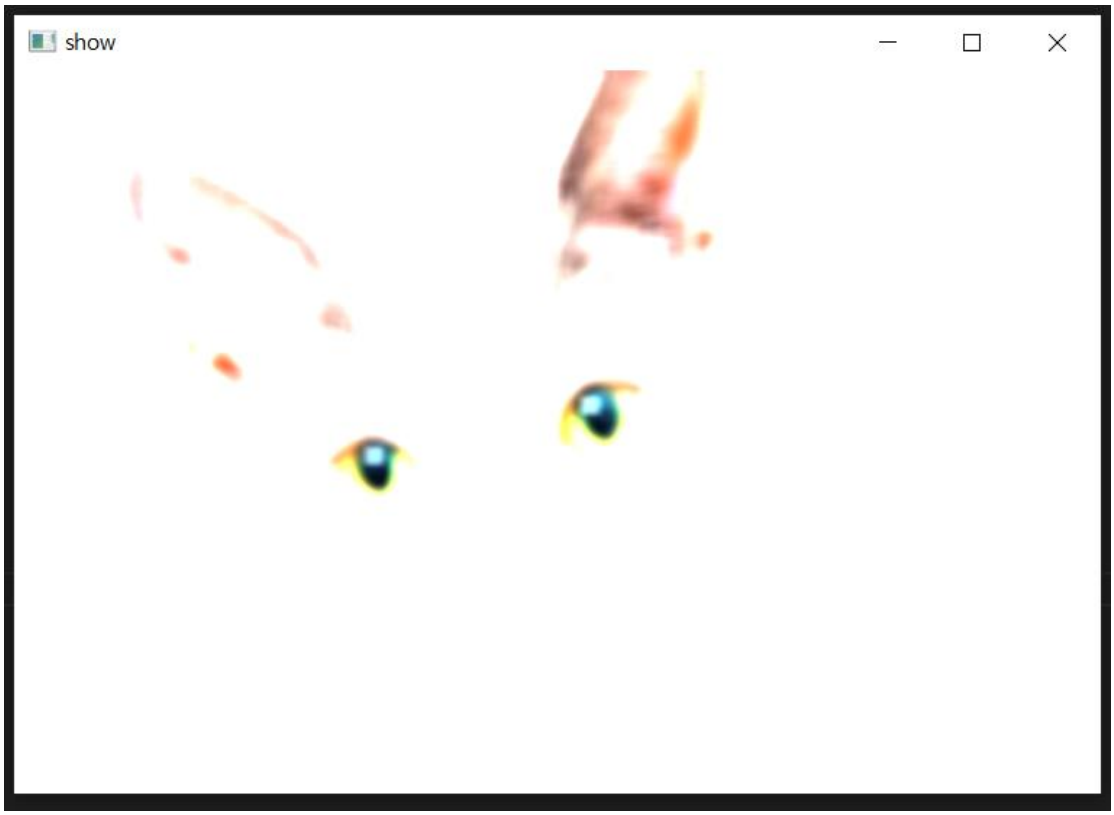
exp_01. Filter 2D

CODE

```python
#### 이미지 blur처리
#### filter 2d() 메소드 사용
import cv2
import numpy as np
from utils import image_show

# 이미지 경로
image_path = "./22.12.06_d45_image/data/cat.jpg"

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

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

```

RESULT



exp_02. Gaussian Blur 2D

CODE

```python
#### 이미지 blur처리
#### GaussianBlur 2d() 메소드 사용
import cv2
import numpy as np
from utils import image_show

# 이미지 경로
image_path = "./22.12.06_d45_image/data/cat.jpg"

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

# GaussianBlur(이미지, 커널, 표준편차)
image_g_blur = cv2.GaussianBlur(image, (9,9), 0)
image_show(image_g_blur)

```

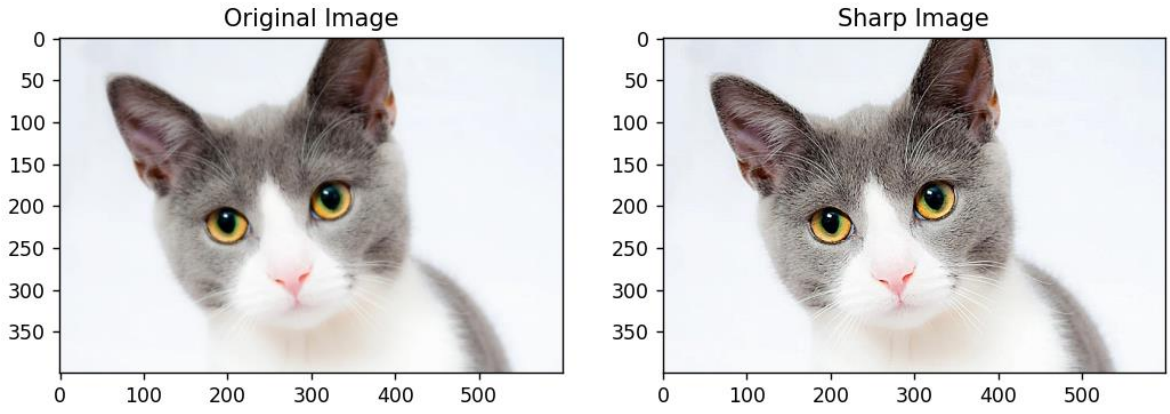
RESULT

exp_03. Image 선명하게 (Sharpen)

CODE

```python
#### 이미지 선명하게
import cv2
import numpy as np
import matplotlib.pyplot as plt
# from utils import image_show

# 이미지 경로
image_path = "./22.12.06_d45_image/data/cat.jpg"

# 이미지 읽기 처리
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)

flg, 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()
```

RESULT



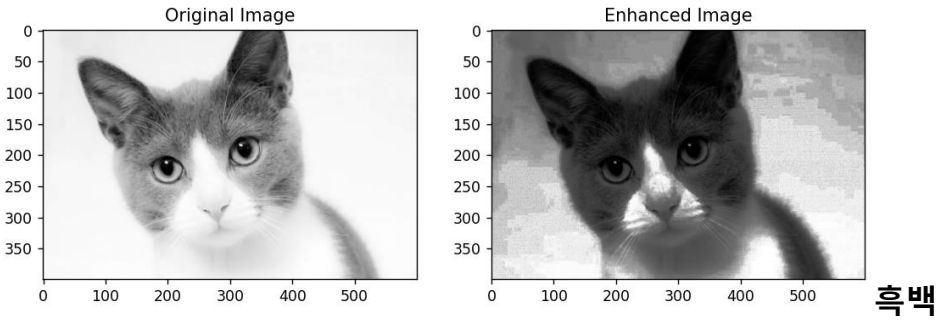
exp_04. 이미지 대비 높이기 (equalizeHist)

CODE

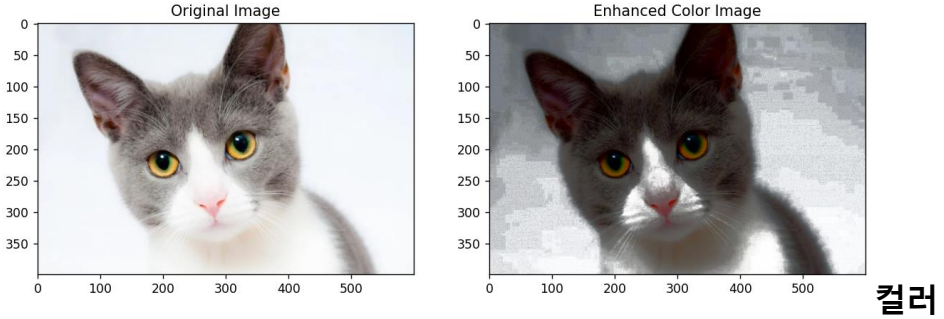
```python
#### 이미지 대비
import cv2
import matplotlib.pyplot as plt

# 이미지 경로
image_path = "./22.12.06_d45_image/data/cat.jpg"

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

flg, 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()
```

```python

### 컬러 이미지 대비 높이기
### 방법 : RGB -> YUV -> equalizeHist() -> RGB
# 1. BGR
image_bgr = cv2.imread(image_path)
# 2. RGB
image_rgb = cv2.cvtColor(image_bgr, cv2.COLOR_BGR2RGB)
# 3. YUV
image_yuv = cv2.cvtColor(image_rgb, cv2.COLOR_RGB2YUV)
# 4. 히스토그램 평활화 적용
image_yuv[:,:,0] = cv2.equalizeHist(image_yuv[:,:,0])
# 5. RGB 변경
image_rgb_temp = cv2.cvtColor(image_yuv, cv2.COLOR_YUV2RGB)

flg, 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()
```

RESULT



흑백



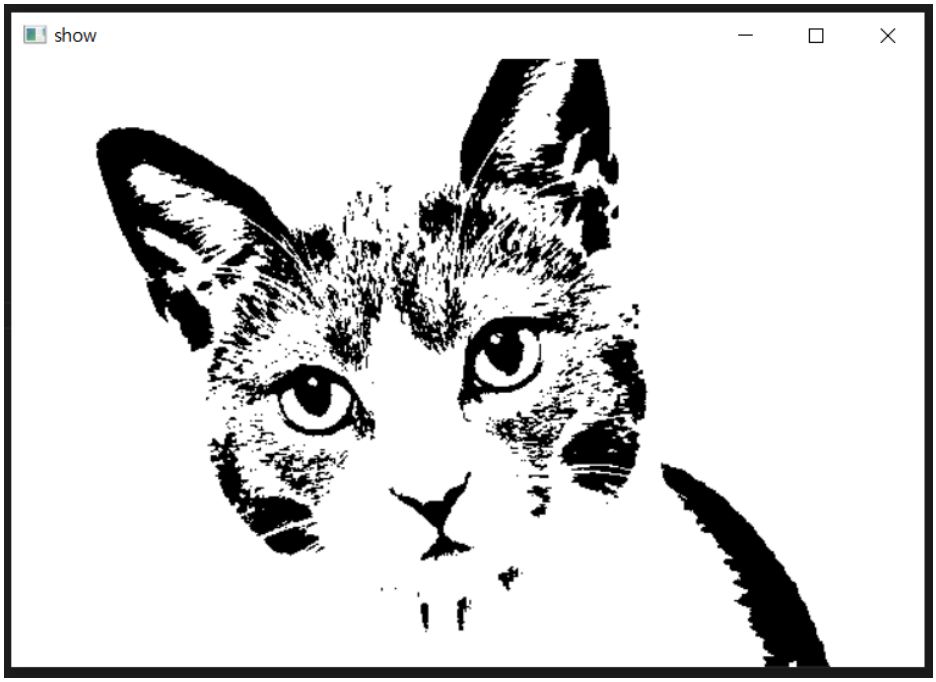
컬러

exp_05. 이미지 이진화 (adaptiveThreshold)

CODE

```python
#### adaptiveThreshold
import cv2
from utils import image_show

# 이미지 경로
image_path = "./22.12.06_d45_image/data/cat.jpg"

# 이미지 이진화
image_gray = cv2.imread(image_path, cv2.IMREAD_GRAYSCALE)
MAX_OUTPUT_VALUE   = 255
NEIGHBORHOOD_SIZE  = 99
SUBTRACT_FROM_MEAN = 10

image_binary = cv2.adaptiveThreshold(image_gray,
                                     MAX_OUTPUT_VALUE,
                                     cv2.ADAPTIVE_THRESH_GAUSSIAN_C,
                                     cv2.THRESH_BINARY,
                                     # cv2.THRESH_BINARY_INV, # 검정색(반전)
                                     NEIGHBORHOOD_SIZE,
                                     SUBTRACT_FROM_MEAN)
image_show(image_binary)
```

RESULT

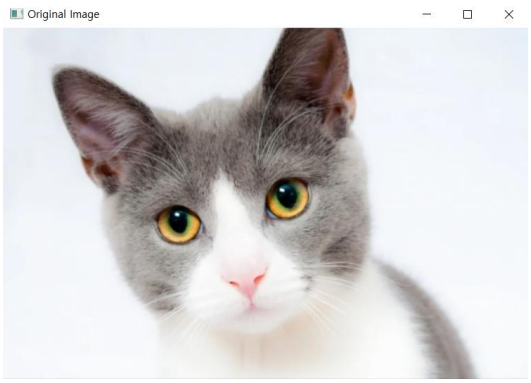


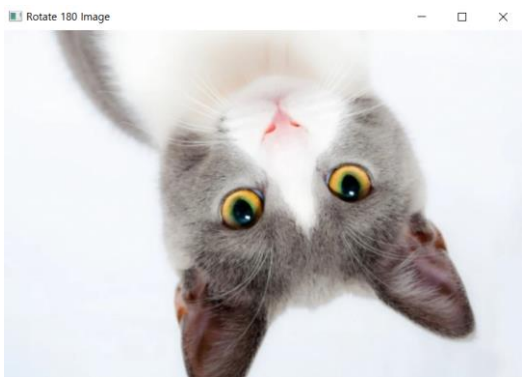
exp_06. Image 회전 (rotate)

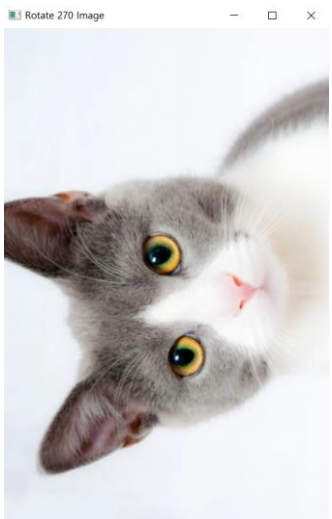
CODE

```python
#### Image Rotate
import cv2
image_path = "./22.12.06_d45_image/data/cat.jpg"  # 이미지 경로

# 이미지 회전
image  = cv2.imread(image_path)
img90  = cv2.rotate(image, cv2.ROTATE_90_CLOCKWISE) # 시계방향 90도
img180 = cv2.rotate(image, cv2.ROTATE_180)  # 180 회전
img270 = cv2.rotate(image, cv2.ROTATE_90_COUNTERCLOCKWISE)  # 270 회전

# 이미지 크기
print(image.shape)
print(img90.shape)
print(img180.shape)
print(img270.shape)

# 이미지 출력
cv2.imshow("Original Image"  , image)
cv2.imshow("Rotate 90 Image" , img90)
cv2.imshow("Rotate 180 Image", img180)
cv2.imshow("Rotate 270 Image", img270)
cv2.waitKey(0)

# 이미지 좌우 및 상하 반전 (1 좌우 반전 0 상하 반전
dst_temp1 = cv2.flip(image, 1)
dst_temp2 = cv2.flip(image, 0)

cv2.imshow("dst_temp1", dst_temp1)
cv2.imshow("dst_temp2", dst_temp2)
cv2.waitKey(0)
```

RESULT





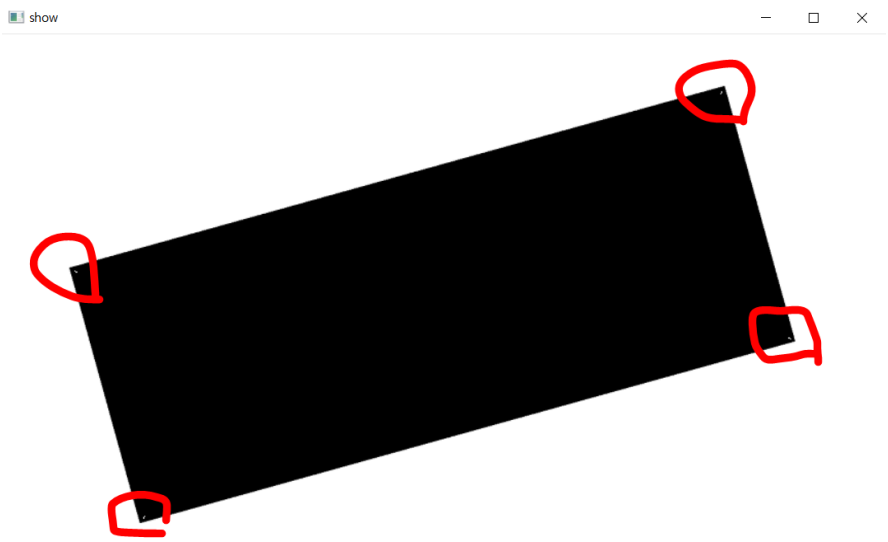


exp_07. 모서리 감지 (cornerHarris)

CODE

```python
#### 모서리 감지
import cv2
import numpy as np
from utils import image_show

# 이미지 경로
image_path = "./22.12.06_d45_image/data/test1.jpg"
image_read = cv2.imread(image_path)

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

BLOCK_SIZE = 2  # 모서리 감지 매개 변수 설정
APERTURE = 29
FREE_PARAMETER = 0.04

detector_response = cv2.cornerHarris(image_gray,
                                     BLOCK_SIZE,
                                     APERTURE,
                                     FREE_PARAMETER)

print(detector_response)

THRESHOLD = 0.02
image_read[detector_response > THRESHOLD *
           detector_response.max()] = [255,255,255]

image_gray = cv2.cvtColor(image_read, cv2.COLOR_BGR2GRAY)
image_show(image_gray)
```

RESULT

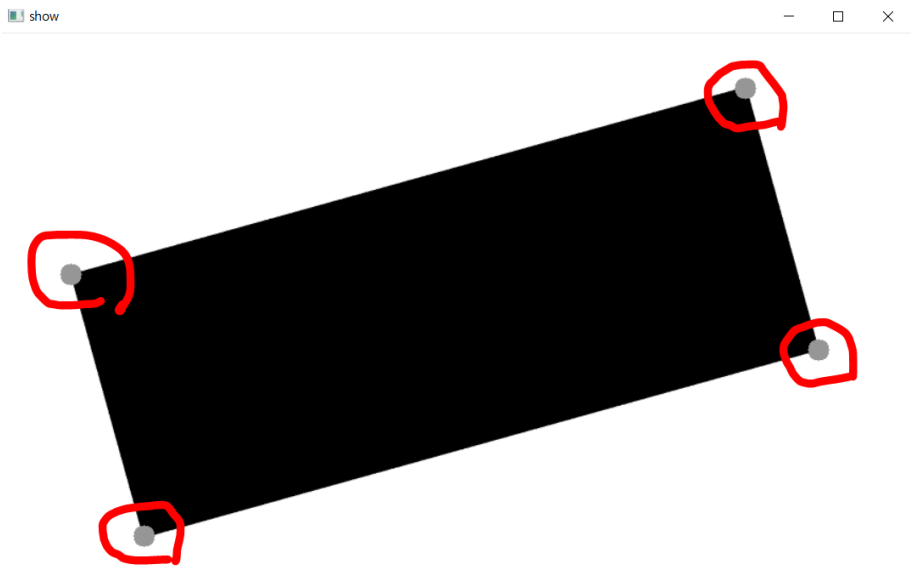


exp_08. 모서리 감지 (goodFeatruesToTrack)

CODE

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

# 이미지 경로
image_path = "./22.12.06_d45_image/data/test1.jpg"
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
MINIMUM_DISTANCE = 25

# 모서리 감지
corners = cv2.goodFeaturesToTrack(
    image_gray, CORNERS_TO_DETECT, MINIMUM_QUALITY_SCORE, MINIMUM_DISTANCE)
# print(corners)

for corner in corners:
    x, y = corner[0]
    cv2.circle(image_read, (int(x), int(y)), 10, (0,255,0), -1)

image_gray_temp = cv2.cvtColor(image_read, cv2.COLOR_BGR2GRAY)
image_show(image_gray_temp)
```

RESULT



exp_09. 이미지 사이즈 변경 (resize)

CODE

```python
import cv2
from utils import image_show

# 이미지 경로
image_path = "./22.12.06_d45_image/data/cat.jpg"
image_gray = cv2.imread(image_path, cv2.IMREAD_GRAYSCALE)  # 그레이 이미지 변경

image_10_by_10 = cv2.resize(image_gray, (10,10))
image_10_by_10.flatten()  # 이미지 데이터를 1차원 벡터로 변환
image_show(image_10_by_10)
```

RESULT

exp_10. 이미지 사이즈 변경 (resize)

CODE

```python
import cv2
from utils import image_show

# 이미지 경로
image_path = "./22.12.06_d45_image/data/cat.jpg"
image = cv2.imread(image_path)

# 10 x 10 변환
image_color_10_by_10 = cv2.resize(image, (10,10))
image_color_10_by_10.flatten()
image_show(image_color_10_by_10)

# 225 x 255 변환
image_color_225_by_255 = cv2.resize(image, (225,255))
image_color_225_by_255.flatten()
image_show(image_color_225_by_255)
```

RESULT

10 X 10

225 X 255

exp_11. 이미지 인코딩

CODE

```python
### 평균색 특성 인코딩
import cv2
import numpy as np
# from utils import image_show

# 이미지 경로
image_path = "./22.12.06_d45_image/data/cat.jpg"
image = cv2.imread(image_path)
channels = cv2.mean(image)
print("Channels: ", channels)

observation = np.array([(channels[2], channels[1], channels[0])])
print(observation)
```

RESULT

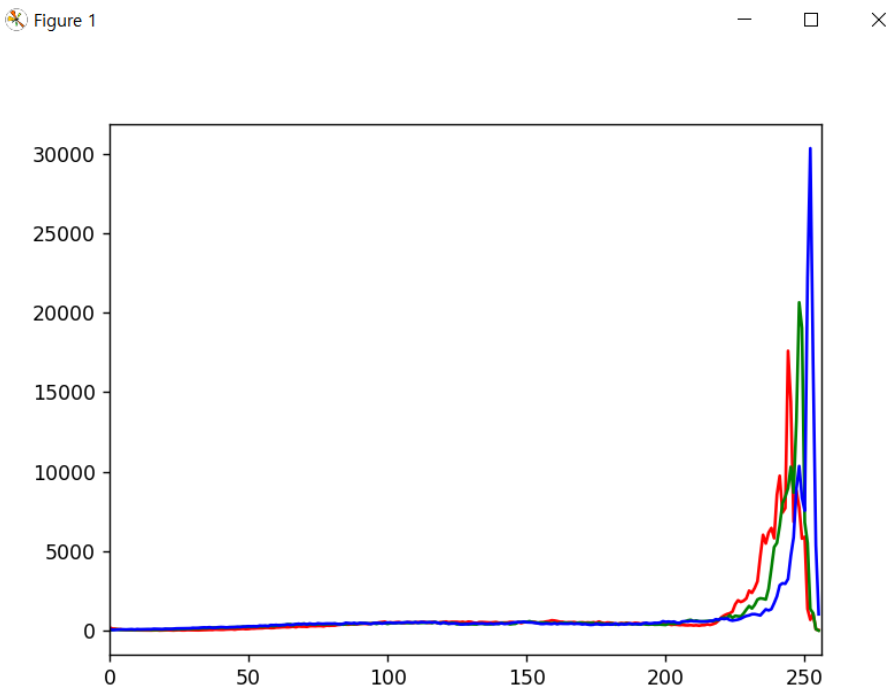
```
Channels:  (205.8482456140351, 205.4207560568087, 207.63455722639935, 0.0)
[[207.63455723 205.42075606 205.84824561]]
PS C:\Users\user\Documents\Microsoft-AI-School>
```

exp_12. 이미지 인코딩 (calcHist)

CODE

```python
### calcHist
import cv2
import numpy as np
import matplotlib.pyplot as plt
from utils import image_show

# 이미지 경로
image_path = "./22.12.06_d45_image/data/cat.jpg"
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 체널에 대해 히스토그램을 계산
for i, channel in enumerate(colors):
    # calcHist([이미지], [체널 인덱스], [마스크 없으므로 None], [히스토그램 크기])
    histogram = cv2.calcHist([image_rgb], [i], None, [256], [0,256])
    plt.plot(histogram, color= channel)
    plt.xlim([0, 256])
plt.show()
```

RESULT



Base_01. 이미지 선명화

CODE

```python
### 기본적인 이미지 처리 기술을 이용한 이미지 선명화
import cv2
import numpy as np

image_path = "./22.12.06_d45_image/data/car.jpg"
img = cv2.imread(image_path, 0)
img_resize = cv2.resize(img, (320,240))

print(img.shape)

blurred_1 = np.hstack([
    cv2.blur(img_resize, (3,3)),
    cv2.blur(img_resize, (7,7)),
    cv2.blur(img_resize, (11,11))
])

cv2.imshow("show", blurred_1)
cv2.waitKey(0)
```

RESULT

Base_02. 가우시안 필터 (GaussianBlur)

CODE

```python
### 가우시안 필터
import cv2
import numpy as np
from utils import image_show

image_path = "./22.12.06_d45_image/data/car.jpg"
img = cv2.imread(image_path, 0)

image_resize = cv2.resize(img, (320,240))

Gaussian_bluured_1 = np.hstack([
    cv2.GaussianBlur(image_resize, (3,3), 0),
    cv2.GaussianBlur(image_resize, (7,7), 0),
    cv2.GaussianBlur(image_resize, (11,11), 0),
])
image_show(Gaussian_bluured_1)

image_name = "./22.12.06_d45_image/data/gaussian_blur.png"
cv2.imwrite(image_name, Gaussian_bluured_1)
```

RESULT

Base_03. 이미지 선명화 (filter2D)

CODE

```python
import cv2
import numpy as np
import matplotlib.pyplot as plt
from utils import image_show

image_path = "./22.12.06_d45_image/data/car.jpg"
image = cv2.imread(image_path)

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

sharpen_img = cv2.filter2D(image, -1, filter)
cv2.imshow("Original Image", image)
cv2.waitKey(0)
image_show(sharpen_img)
```

RESULT

Original

Filter

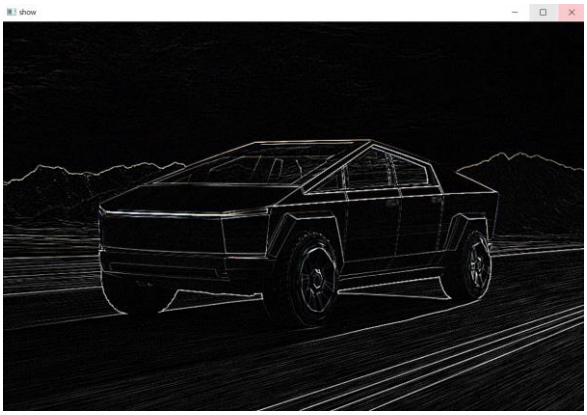
Base_04. 멕시칸 모자 필터

CODE

```python
### Mexican hat filter
import cv2
import numpy as np
from utils import image_show

image_path = "./22.12.06_d45_image/data/car.jpg"
image = cv2.imread(image_path)

# Mexican hat filter (3x3)
filter = np.array([[-1,-1,-1],[-1,8,-1],[-1,-1,-1]])
mexican_hat_image_3_by_3 = cv2.filter2D(image, -1, filter)
image_show(mexican_hat_image_3_by_3)
cv2.imwrite("./22.12.06_d45_image/data/mexican_hot_3x3.jpg",
            mexican_hat_image_3_by_3)

# Mexican hat filter (5x5)
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]])
mexican_hat_image_5_by_5 = cv2.filter2D(image, -1, filter)
image_show(mexican_hat_image_5_by_5)
cv2.imwrite("./22.12.06_d45_image/data/mexican_hot_5x5.jpg",
            mexican_hat_image_5_by_5)
```

RESULT

3 x 3



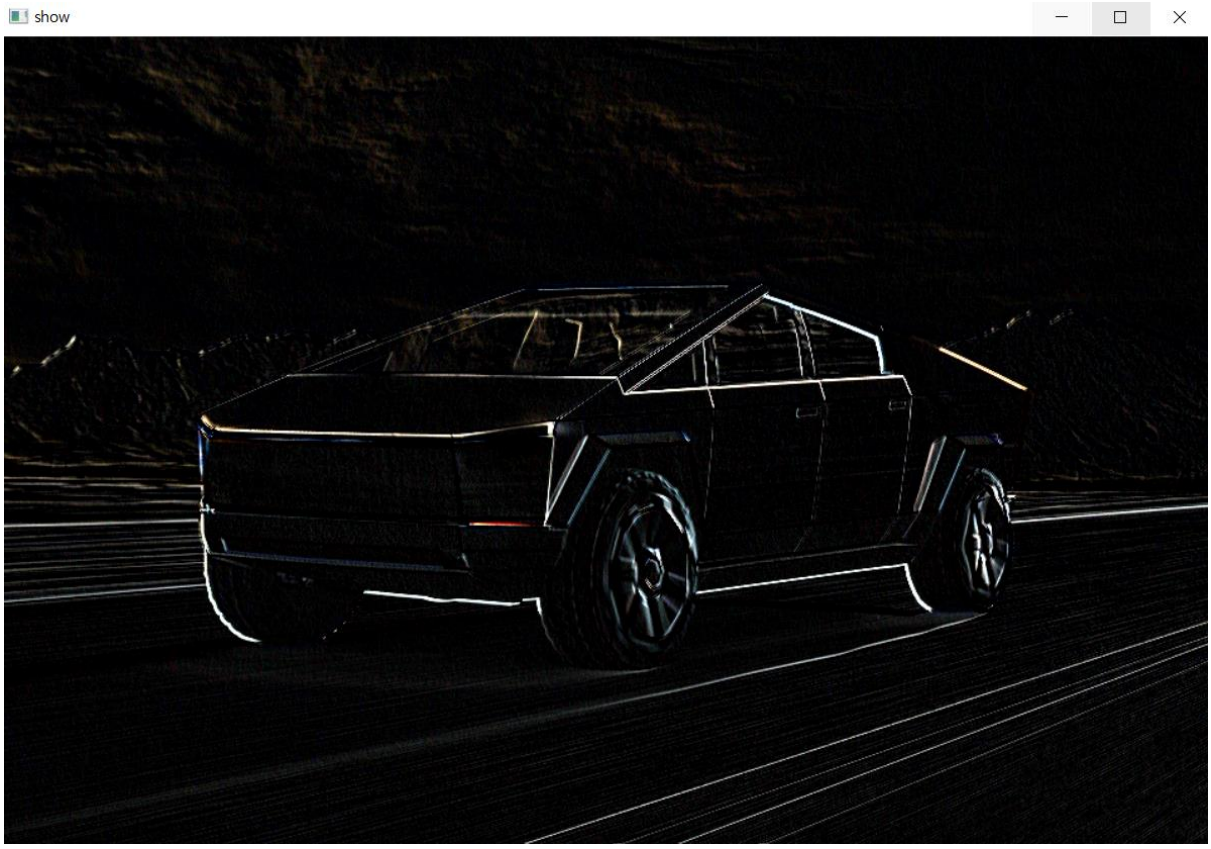
5 x 5

Base_05. 커스텀 필터 (Custom filter)

CODE

```python
### Custom Filter
import cv2
import numpy as np
from utils import image_show

image_path = "./22.12.06_d45_image/data/car.jpg"
image = cv2.imread(image_path)

filter = np.array(([3, -2, -3],[-4, 8, -6], [5, -1, -0]))
custom_image_filter = cv2.filter2D(image, -1, filter)

image_show(custom_image_filter)
```

RESULT



Base_06. 세피아 필터

CODE

```python
### 세피아 효과 필터
import cv2
import numpy as np
import matplotlib.pyplot as plt
from utils import image_show

image_path = "./22.12.06_d45_image/data/car.jpg"
image = cv2.imread(image_path)

filter = np.array([[0.272, 0.534, 0.131],
                   [0.349, 0.686, 0.168],
                   [0.393, 0.769, 0.189]])

sepia_img = cv2.transform(image, filter)
image_show(sepia_img)
```

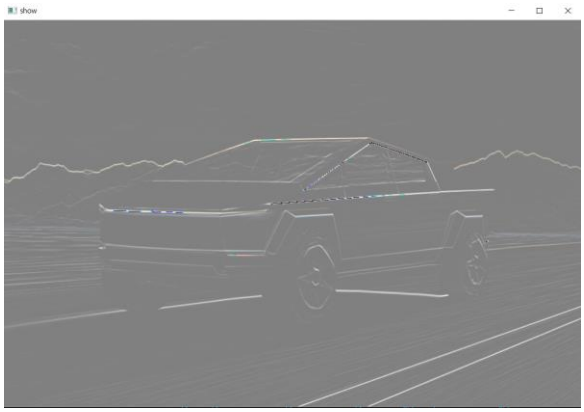
RESULT

Base_07. 엠보싱 효과 (Embossing)

CODE

```python
### 엠보싱 효과
import cv2
import numpy as np
import matplotlib.pyplot as plt
from utils import image_show

image_path = "./22.12.06_d45_image/data/car.jpg"
image = cv2.imread(image_path)

filter1 = np.array([[0, 1, 0], [0, 0, 0], [0, -1, 0]])
emboss_img1 = cv2.filter2D(image, -1, filter1)
emboss_img1 += 128
image_show(emboss_img1)

filter2 = np.array([[-1, -1, 0], [-1, 0, 1], [0, 1, 1]])
emboss_img2 = cv2.filter2D(image, -1, filter2)
emboss_img2 += 128
image_show(emboss_img2)
```

RESULT



Filter 1

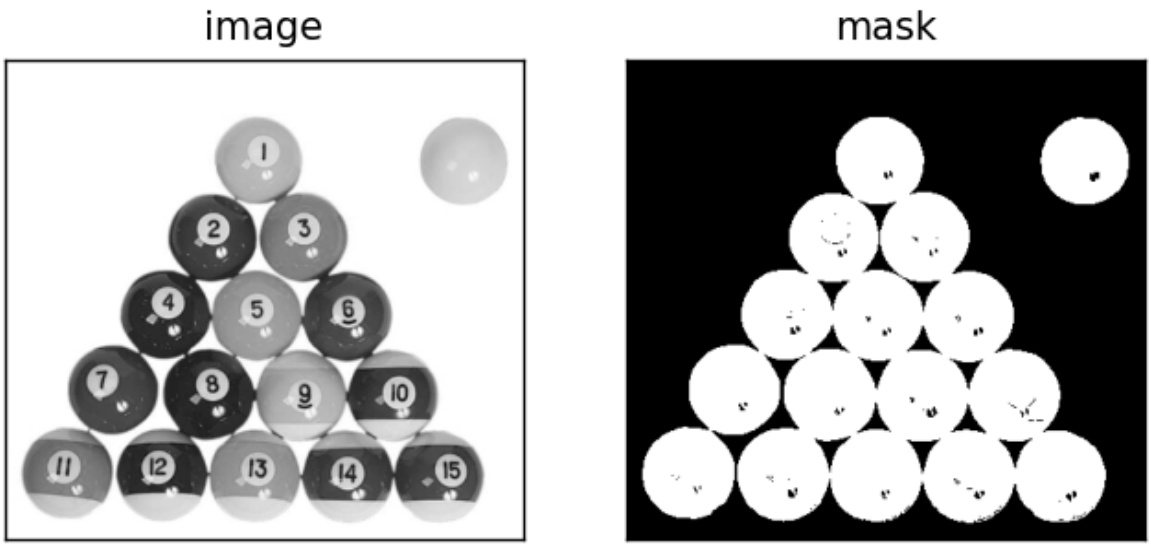
Filter 2

Base_08. 형태학적 변환 & 팽창 & 침식

CODE

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

image_path = "./22.12.06_d45_image/data/billiards.jpg"
image = cv2.imread(image_path, cv2.IMREAD_GRAYSCALE)  # gray

# 임계값 연산자의 출력을 마스크 라는 변수에 저장
# 230 이하: 흰색 처리 // 230 이상: 검은색 처리
_ , mask = cv2.threshold(image, 230, 255, cv2.THRESH_BINARY_INV)

titles =["image", "mask"]
images = [image, 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()
```

RESULT

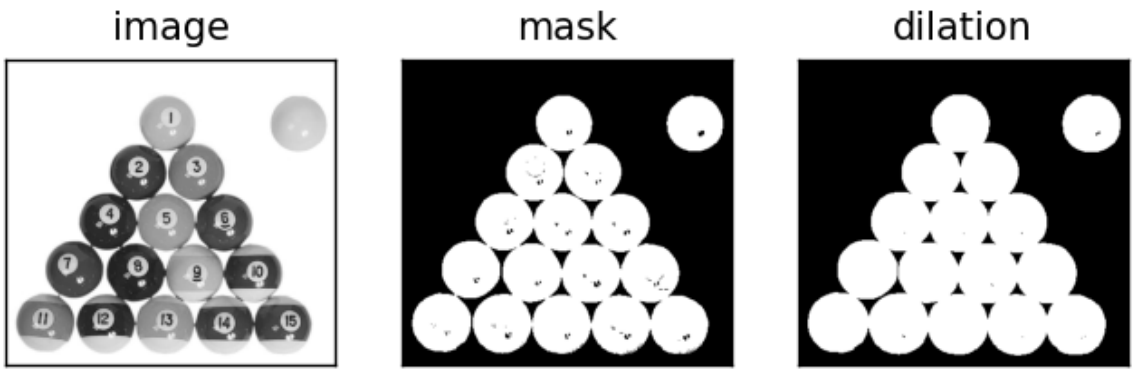


Base_09. 형태학적 변환 & 팽창 & 침식

CODE

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

image_path = "./22.12.06_d45_image/data/billiards.jpg"
image_gray = cv2.imread(image_path, cv2.IMREAD_GRAYSCALE)  # gray
_ , mask = cv2.threshold(image_gray, 230, 255, cv2.THRESH_BINARY_INV)

# 3x3 Kernel
kernel = np.ones((3,3), np.uint8)
print(kernel)

dilation = cv2.dilate(mask, kernel)

titles = ["image", "mask", "dilation"]
images = [image_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()
```

RESULT



Base_10. 형태학적 변환 & 팽창 & 침식

CODE

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

image_path = "./22.12.06_d45_image/data/billiards.jpg"
image_gray = cv2.imread(image_path, cv2.IMREAD_GRAYSCALE)  # gray
_ , mask = cv2.threshold(image_gray, 230, 255, cv2.THRESH_BINARY_INV)

# 3x3 Kernel
kernel = np.ones((3,3), np.uint8)
dilation = cv2.dilate(mask, kernel, iterations= 10)

titles = ["image", "mask", "dilation"]
images = [image_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()
```

RESULT