초급 영상처리

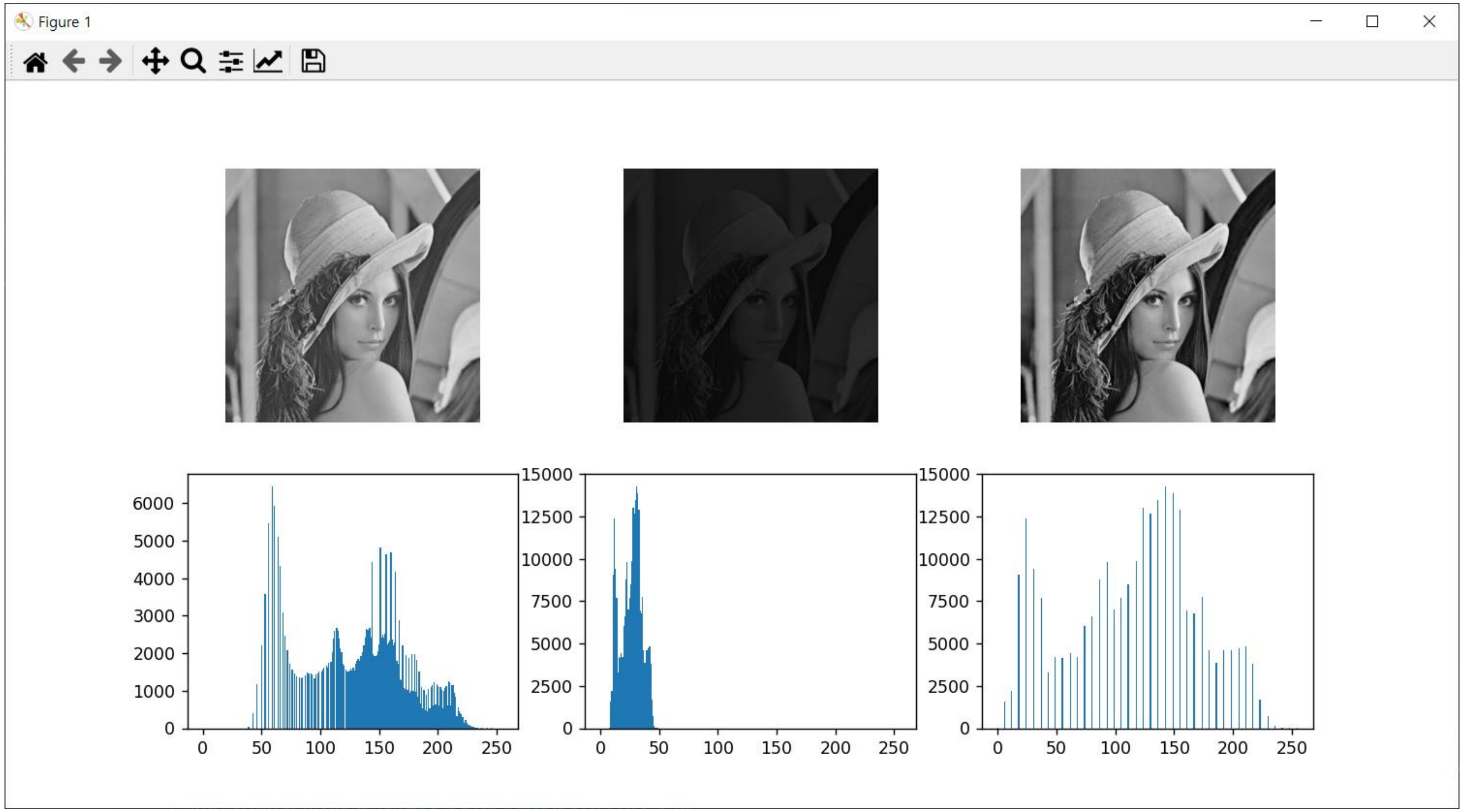
(나만의 Opencv 구현하기)

박화종

INDEX

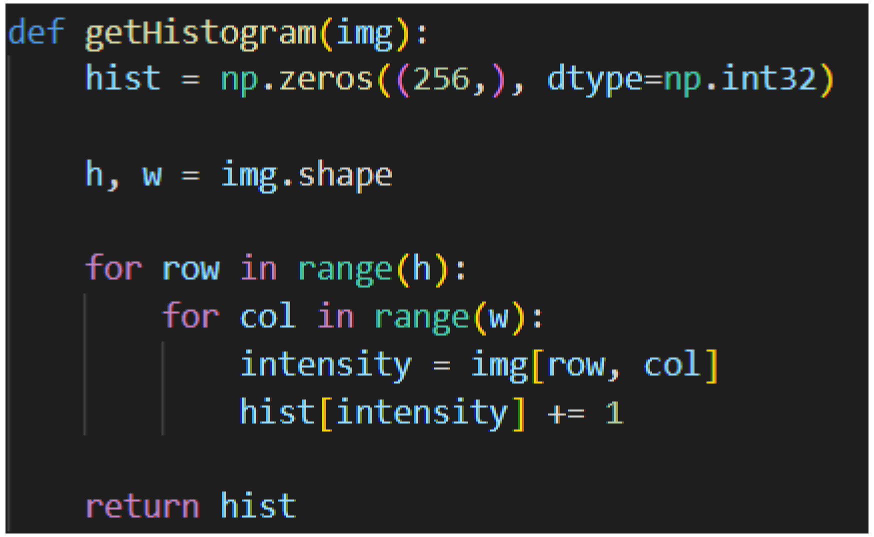
저번 주 과제(IP2_test1)

- 히스토그램 그리기 & stretching

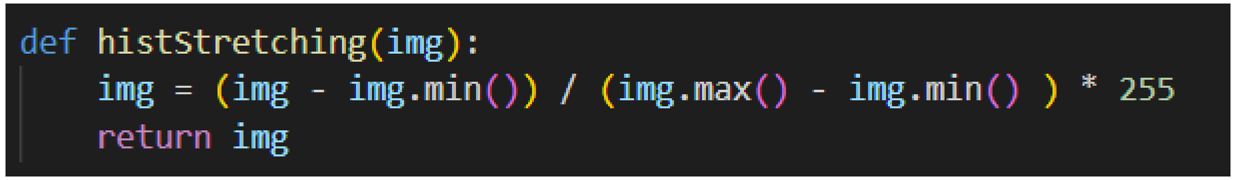


저번 주 과제(IP2_test1)

- 히스토그램 그리기 & stretching



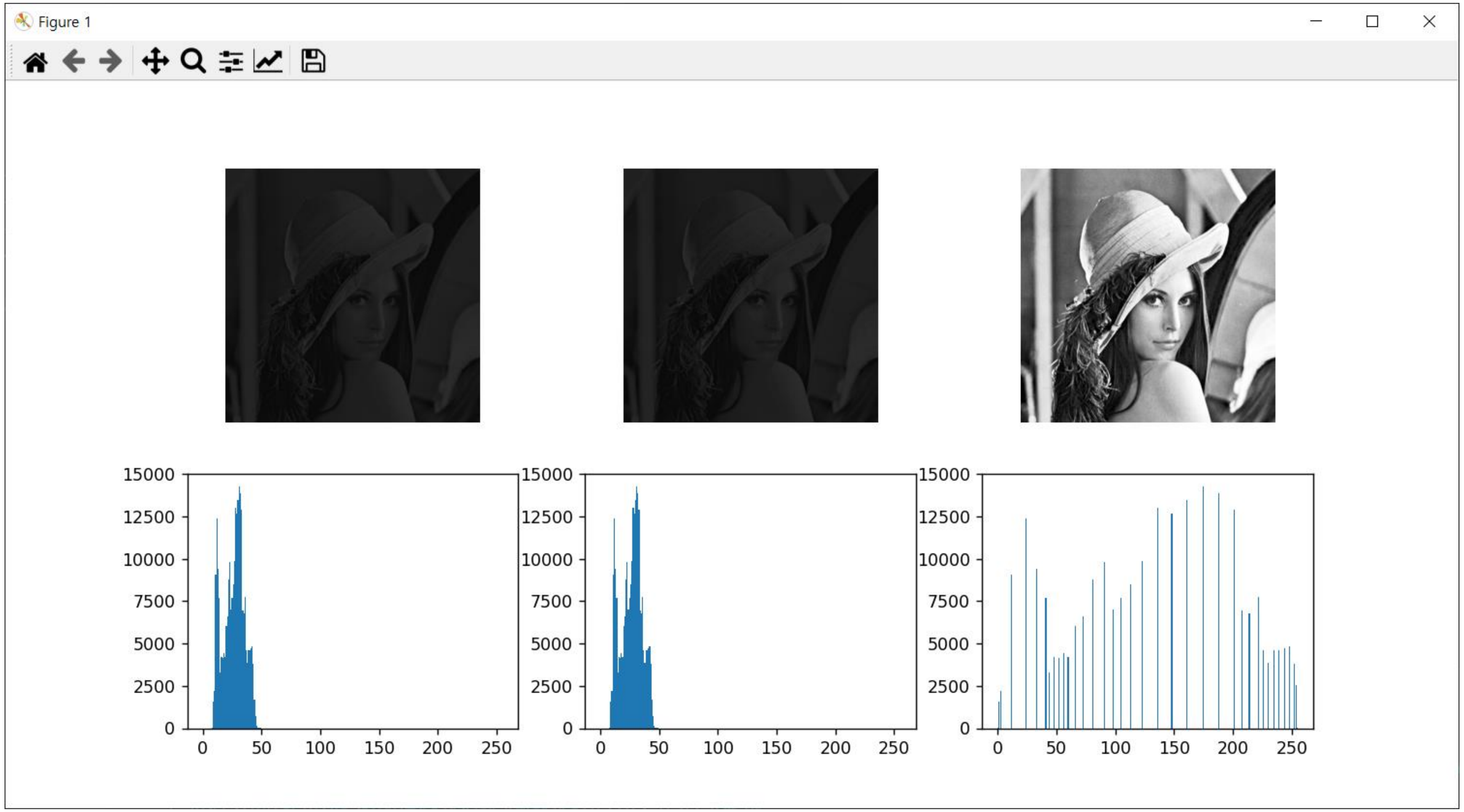
```python
def getHistogram(img):
    hist = np.zeros((256,), dtype=np.int32)

    h, w = img.shape

    for row in range(h):
        for col in range(w):
            intensity = img[row, col]
            hist[intensity] += 1

    return hist
```



```python
def histStretching(img):
    img = (img - img.min()) / (img.max() - img.min() ) * 255
    return img
```

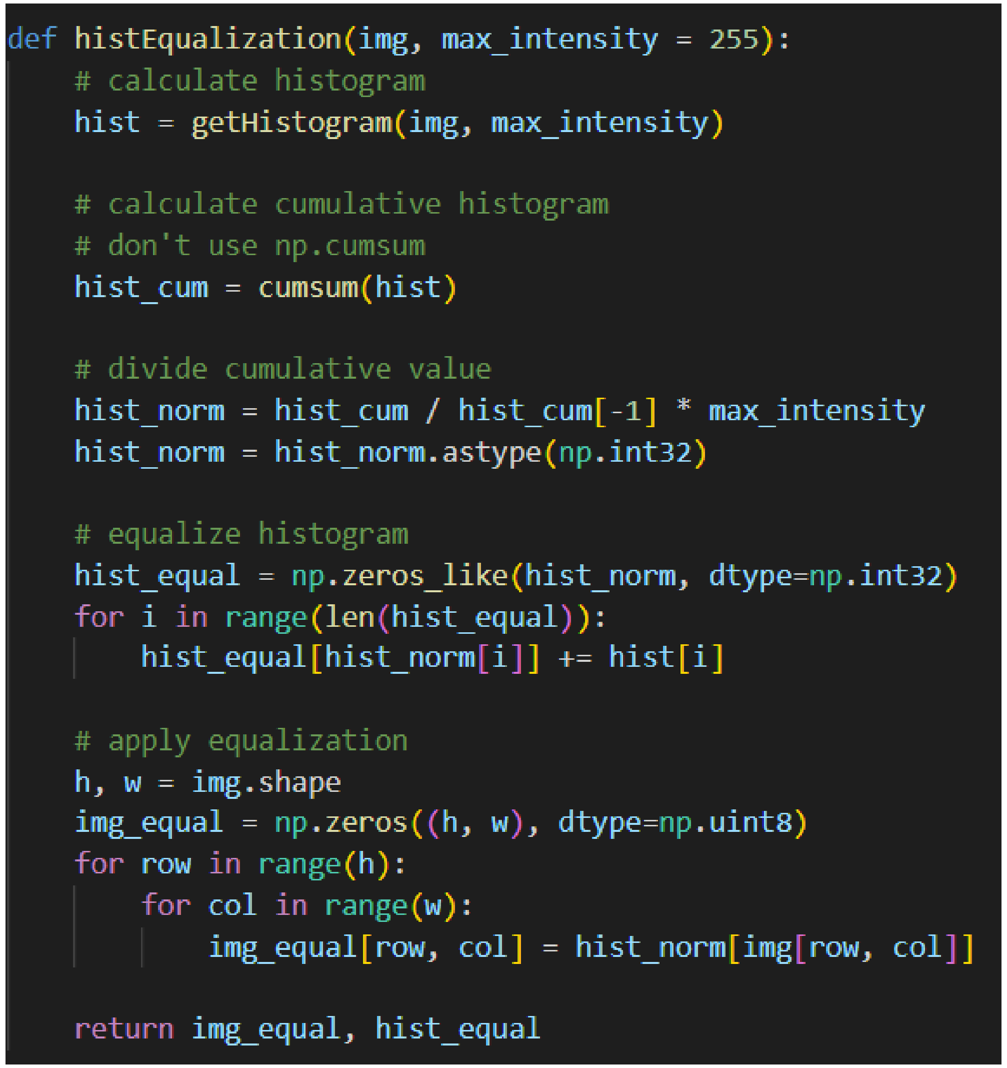
저번 주 과제(IP2_test2)

- Histogram equalization

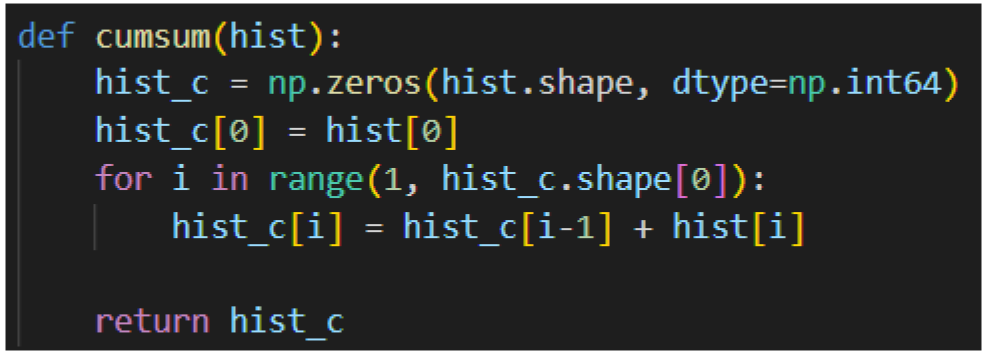


저번 주 과제(IP2_test2)

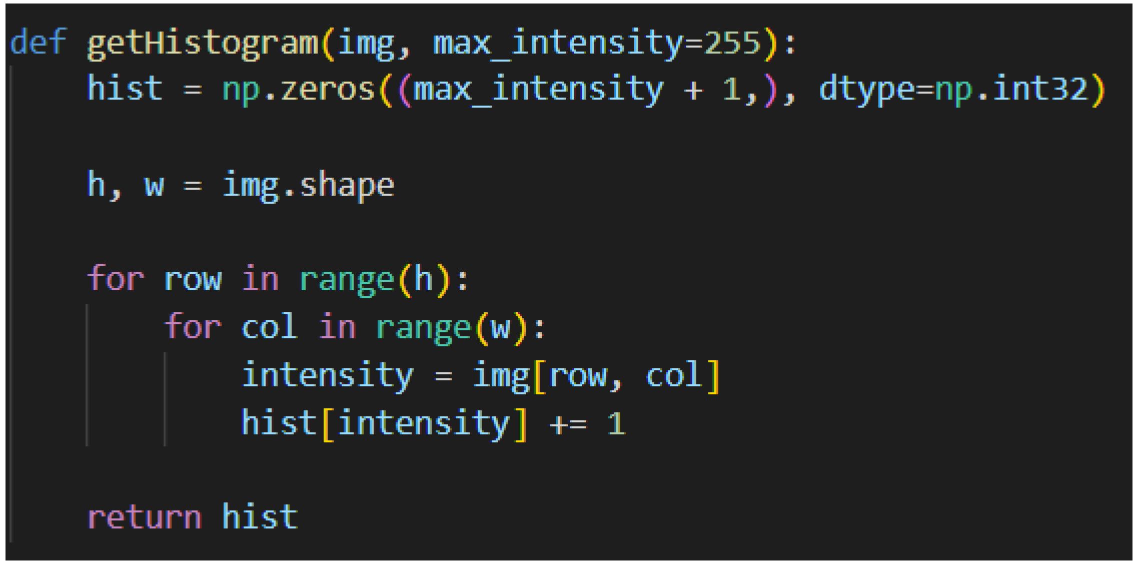
- Histogram equalization



```python
def histEqualization(img, max_intensity = 255):
    # calculate histogram
    hist = getHistogram(img, max_intensity)

    # calculate cumulative histogram
    # don't use np.cumsum
    hist_cum = cumsum(hist)

    # divide cumulative value
    hist_norm = hist_cum / hist_cum[-1] * max_intensity
    hist_norm = hist_norm.astype(np.int32)

    # equalize histogram
    hist_equal = np.zeros_like(hist_norm, dtype=np.int32)
    for i in range(len(hist_equal)):
        hist_equal[hist_norm[i]] += hist[i]

    # apply equalization
    h, w = img.shape
    img_equal = np.zeros((h, w), dtype=np.uint8)
    for row in range(h):
        for col in range(w):
            img_equal[row, col] = hist_norm[img[row, col]]

    return img_equal, hist_equal
```



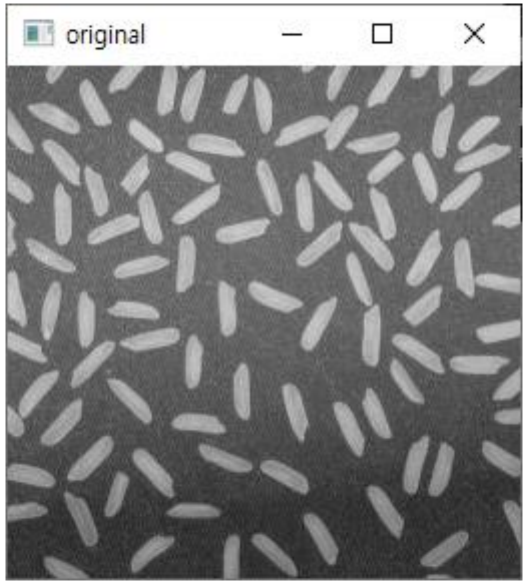
```python
def cumsum(hist):
    hist_c = np.zeros(hist.shape, dtype=np.int64)
    hist_c[0] = hist[0]
    for i in range(1, hist_c.shape[0]):
        hist_c[i] = hist_c[i-1] + hist[i]

    return hist_c
```



```python
def getHistogram(img, max_intensity=255):
    hist = np.zeros((max_intensity + 1,), dtype=np.int32)

    h, w = img.shape

    for row in range(h):
        for col in range(w):
            intensity = img[row, col]
            hist[intensity] += 1

    return hist
```

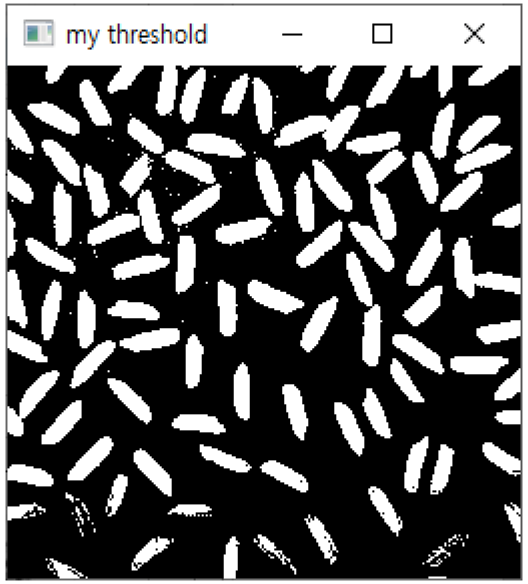
Binarization

- Binarization란?
  - 0 or 1 과 같이 2개 중 하나로 분류하는 것



Original

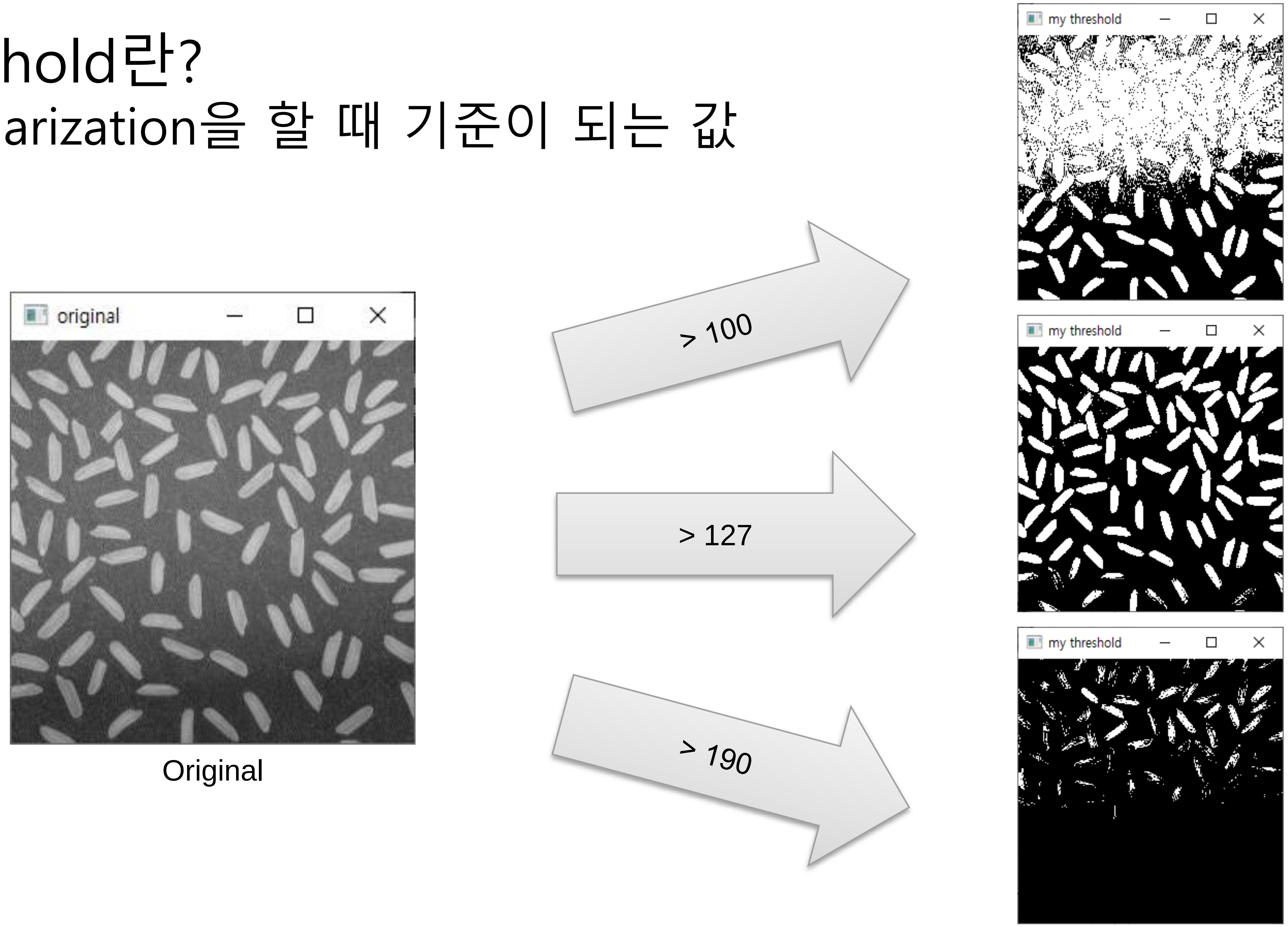
127보다 큰 값은 255
127 이하의 값은 0



Binarization

Threshold

- Threshold란?
  - Binarization을 할 때 기준이 되는 값



Original

Threshold

- OTSU's method란?
  - 클래스 내 분산을 최소화 하는 threshold값을 찾아 줌

$$T_{opt} = \mathrm{argmin}_{1 \le k \le L-1}(\sigma_\omega^2(k))$$

$$\sigma_\omega^2(t) = \omega_0(t)\sigma_0^2(t) + \omega_1(t)\sigma_1^2(t)$$

L : max intensity(256)
σ^2 : 분산
ω : 가중치(확률)

1) 히스토그램 계산

2) $T = k(k \ge 1)$에서 클래스 분리를 위한 확률 및 평균 계산

$$C_1(k) = \sum_{i=0}^{k-1} N_i, \qquad C_2(k) = \sum_{i=k}^{L-1} N_i = N - C_1(k)$$

$$\boldsymbol{\omega_1(k) = \frac{C_1(k)}{N}}, \qquad \boldsymbol{\omega_2(k) = \frac{C_2(k)}{N} = 1 - \omega_1(k)}$$

$$\mu_{T1}(k) = \sum_{i=0}^{k-1} i \cdot N_i, \quad \mu_{T2}(k) = \sum_{i=k}^{L-1} i \cdot N_i, \quad \mu_T = \sum_{i=0}^{L-1} i \cdot N_i$$

$$\boldsymbol{\mu_1(k) = \frac{\mu_{T1}(k)}{C_1(k)}}, \qquad \boldsymbol{\mu_2(k) = \frac{\mu_{T2}(k)}{C_2(k)} = \frac{\mu_T - \mu_{T1}(k)}{N - C_1(k)}}$$

(N: 전체 픽셀 수, i: 밝기값)

3) $T = k(k \ge 1)$ 에서 클래스 분리를 위한 분산 σ_W^2 계산

$$\sigma_1^2(k) = \sum_{n=0}^{k-1}[n - \mu_1(k)]^2 \frac{N_n}{C_1(k)}, \quad \sigma_2^2(k) = \sum_{n=k}^{L-1}[n - \mu_2(k)]^2 \frac{N_n}{C_2(k)}$$

$$\boldsymbol{\sigma_W^2(k) = \omega_1(k)\sigma_1^2(k) + \omega_2(k)\sigma_2^2(k)}$$

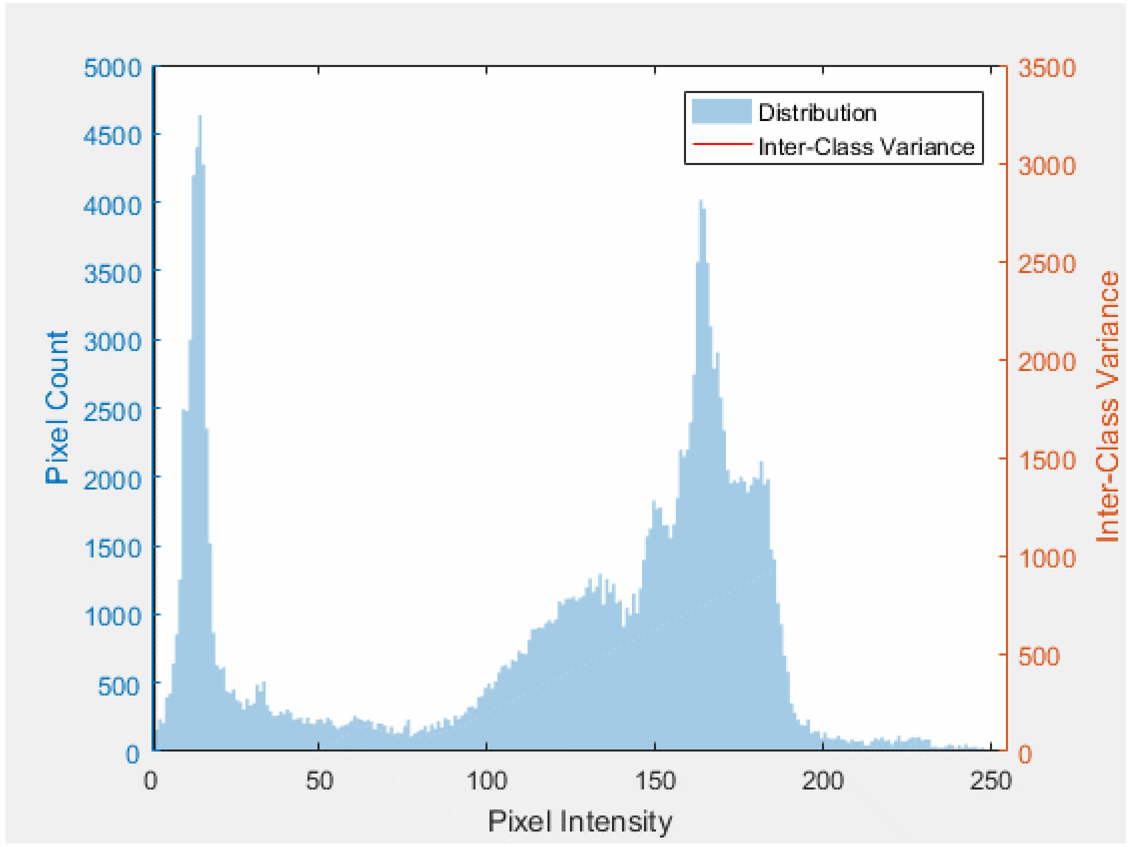
4) 모든 레벨에 대해 반복하여 최적 임계값 선택

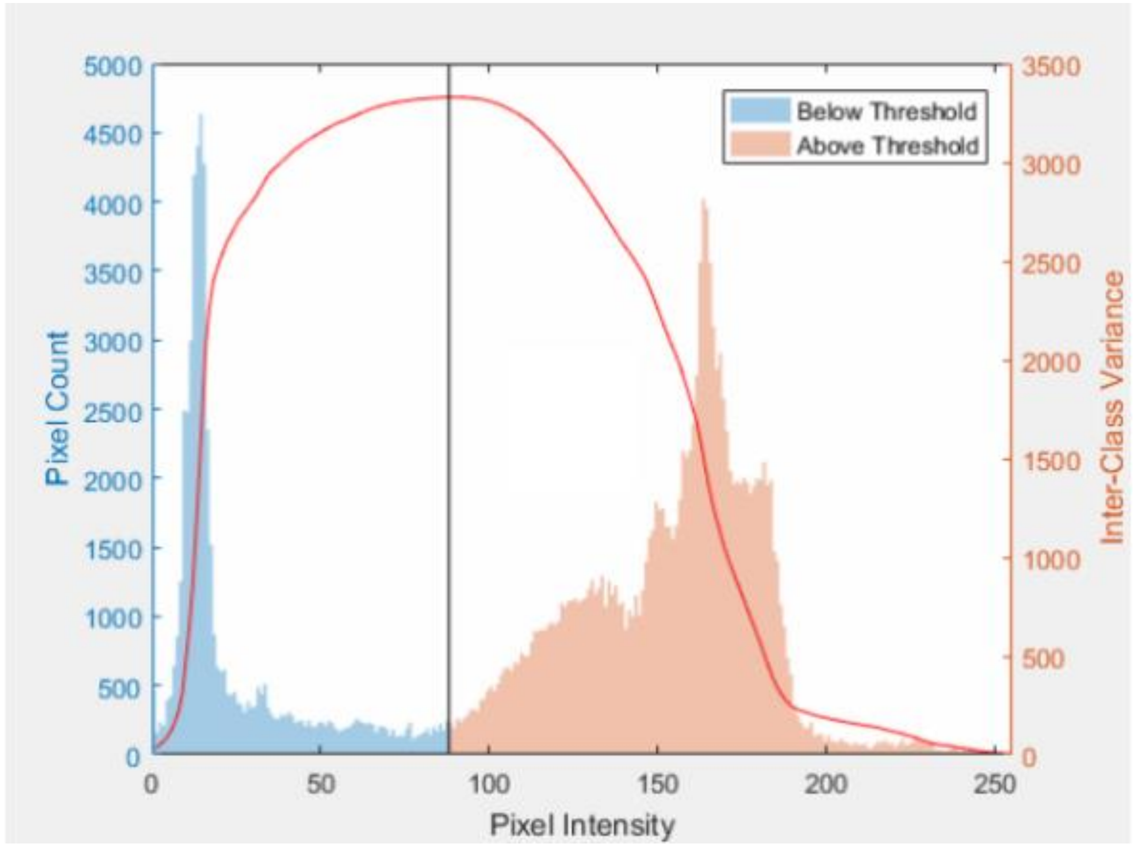
$$\boldsymbol{T_{opt} = \mathrm{argmin}_{1 \le k \le L-1}\left(\sigma_W^2(k)\right)}$$

설명 : https://en.wikipedia.org/wiki/Otsu%27s_method
설명(kr) : https://swkdn.tistory.com/entry/5-%EA%B7%B8%EB%A0%88%EC%9D%B4%EB%A0%88%EB%B2%A8-%EC%9E%84%EA%B3%84%ED%99%94-%EA%B7%B8%EB%A0%88%EC%9D%B4%EB%A0%88%EB%B2%A8-%EC%9E%84%EA%B3%84%ED%99%94-%EB%B0%8F-Ostus-%EB%B0%A9%EB%B2%95

Threshold

- OTSU's method란?
  - 클래스 내 분산을 최소화 하는 threshold값을 찾아 줌

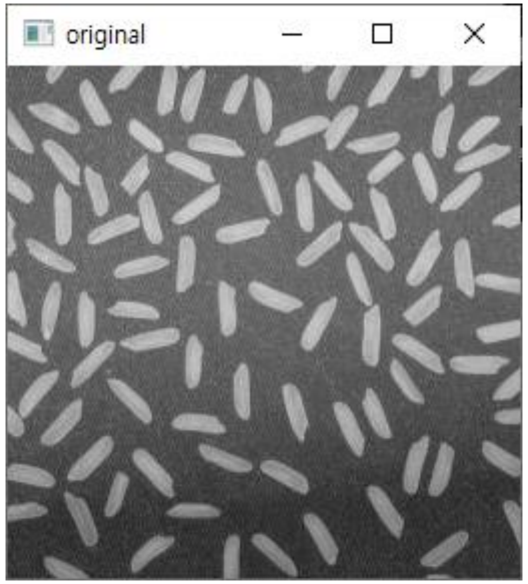




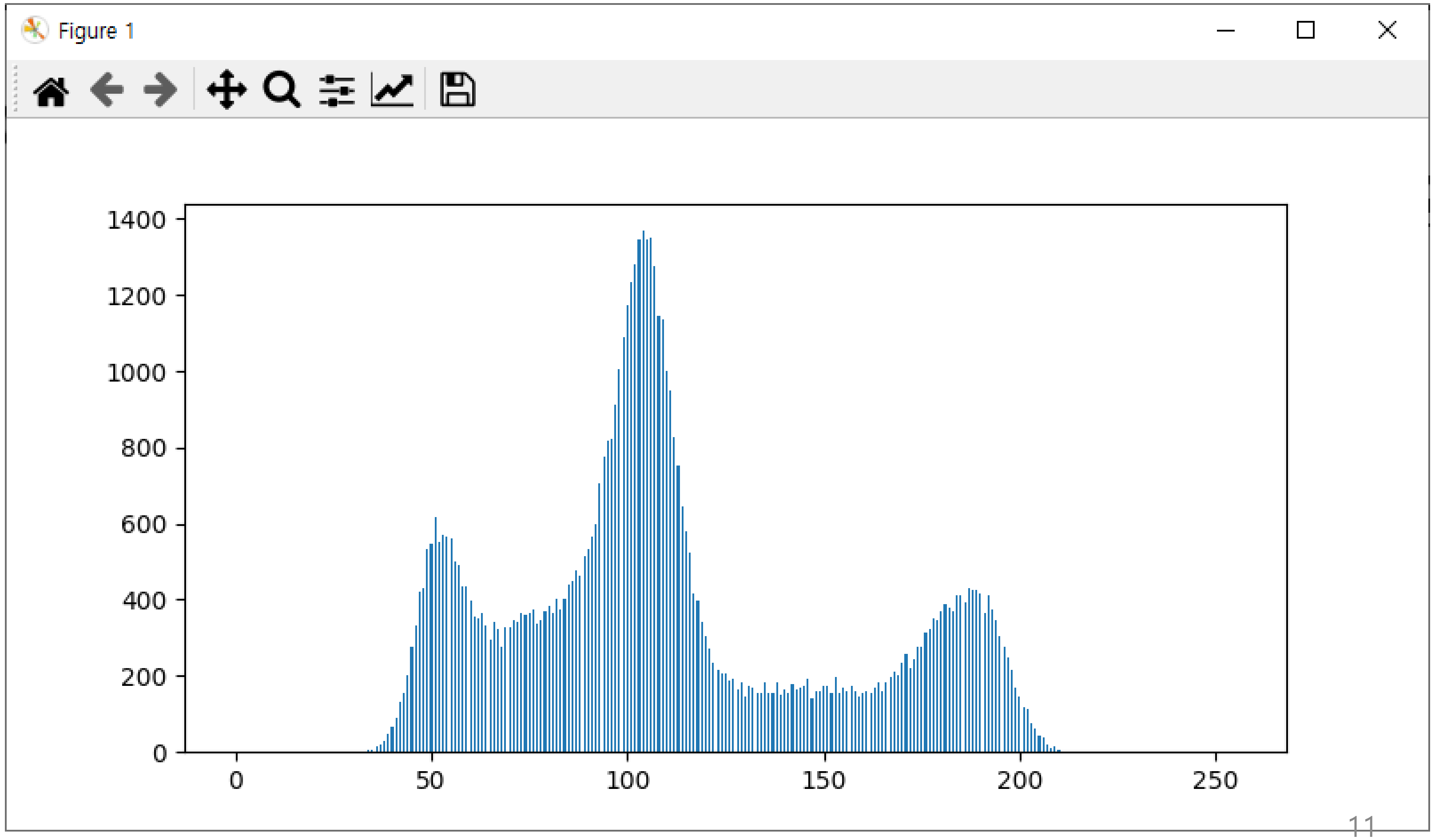
https://en.wikipedia.org/wiki/Otsu%27s_method

Threshold

- OTSU's method란?
  - 클래스 내 분산을 최소화 하는 threshold값을 찾아 줌



rices.png



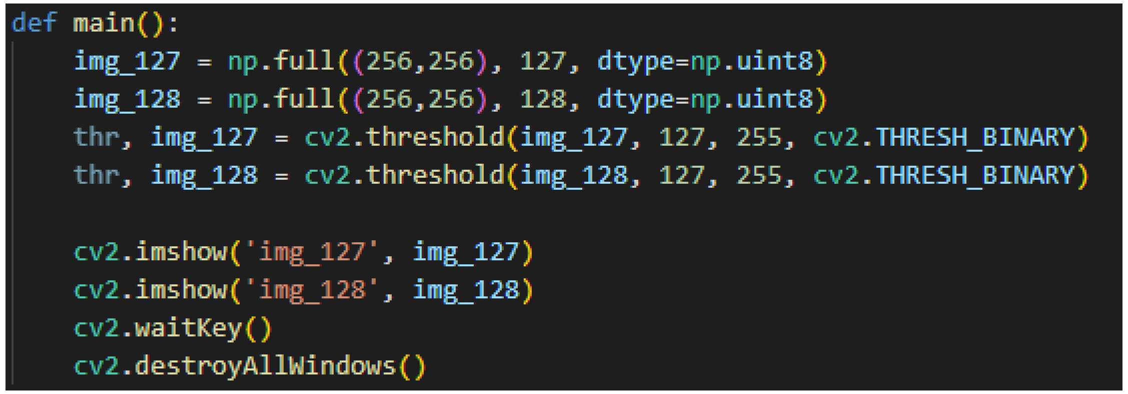
실습 및 과제

- Github : [Hwa-Jong/MyOpenCV: study Opencv (github.com)]

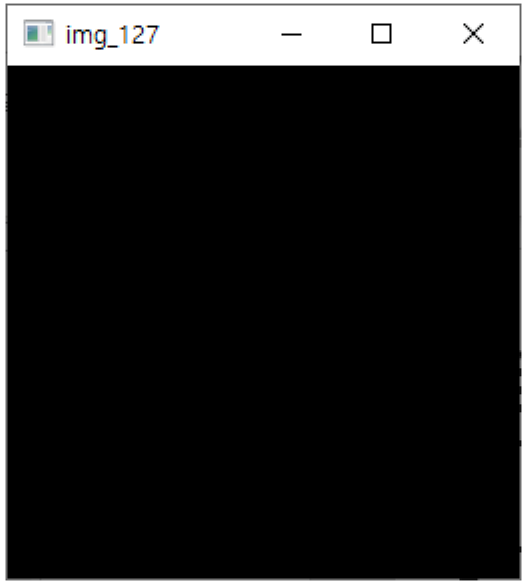
[OTSU 한글 설명]
https://swkdn.tistory.com/entry/5-%EA%B7%B8%EB%A0%88%EC%9D%B4%EB%A0%88%EB%B2%A8-%EC%9E%84%EA%B3%84%ED%99%94-%EA%B7%B8%EB%A0%88%EC%9D%B4%EB%A0%88%EB%B2%A8-%EC%9E%84%EA%B3%84%ED%99%94-%EB%B0%8F-Ostus-%EB%B0%A9%EB%B2%95

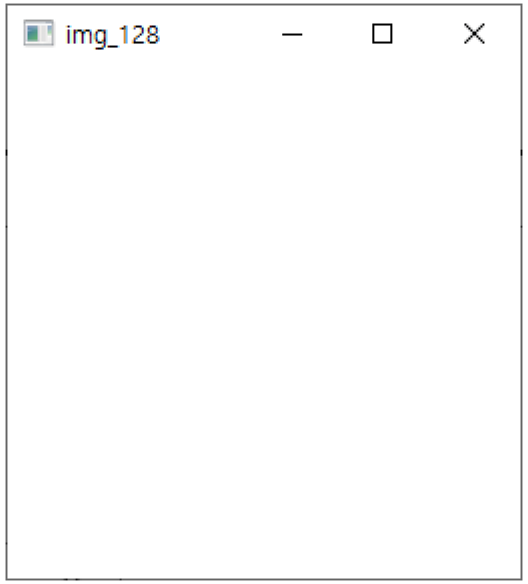
실습(IP3_1)

- Threshold 함수 사용해보기



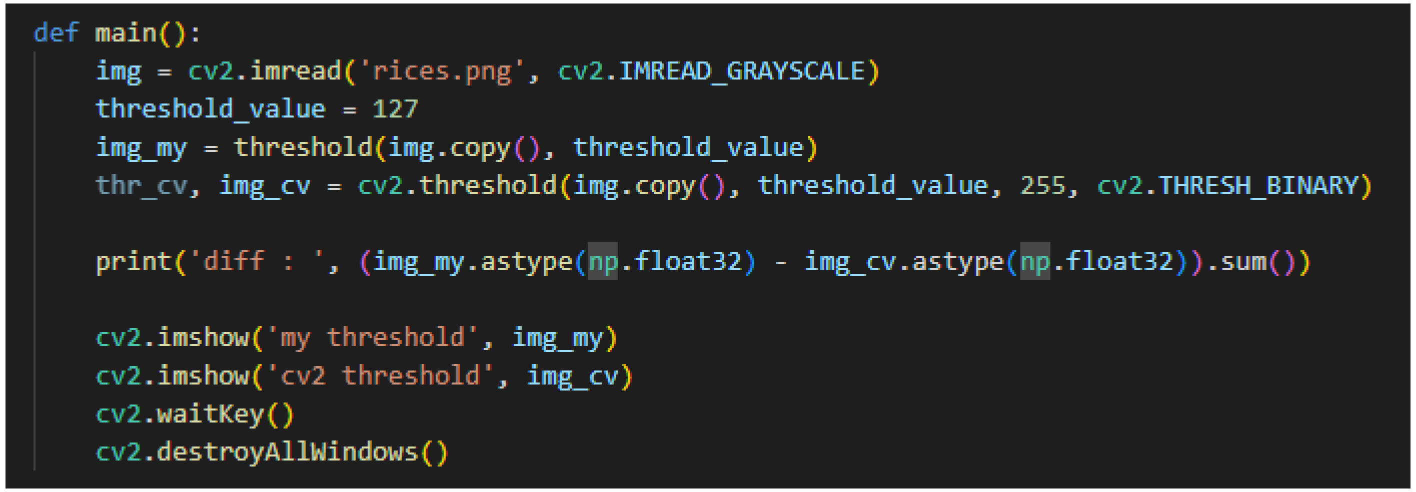
```python
def main():
    img_127 = np.full((256,256), 127, dtype=np.uint8)
    img_128 = np.full((256,256), 128, dtype=np.uint8)
    thr, img_127 = cv2.threshold(img_127, 127, 255, cv2.THRESH_BINARY)
    thr, img_128 = cv2.threshold(img_128, 127, 255, cv2.THRESH_BINARY)

    cv2.imshow('img_127', img_127)
    cv2.imshow('img_128', img_128)
    cv2.waitKey()
    cv2.destroyAllWindows()
```



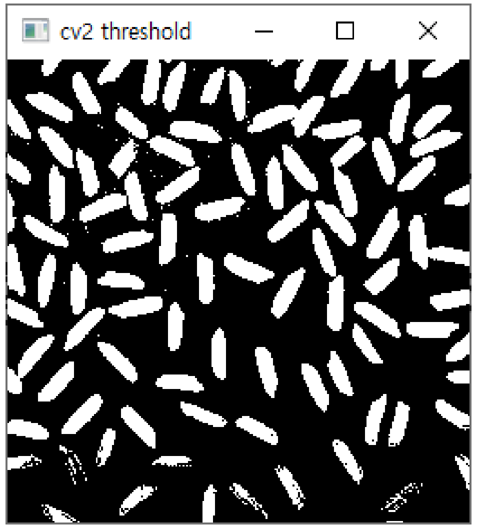


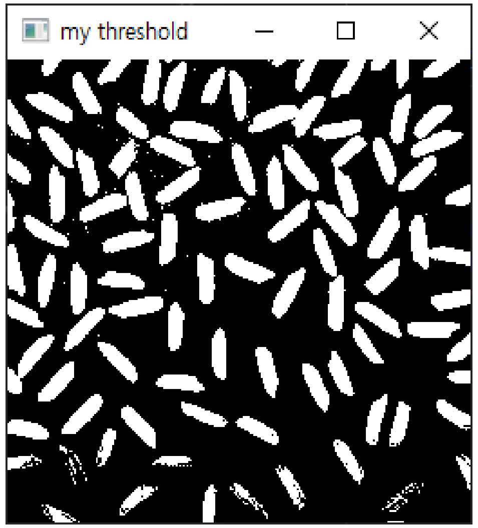
과제(IP3_test1)

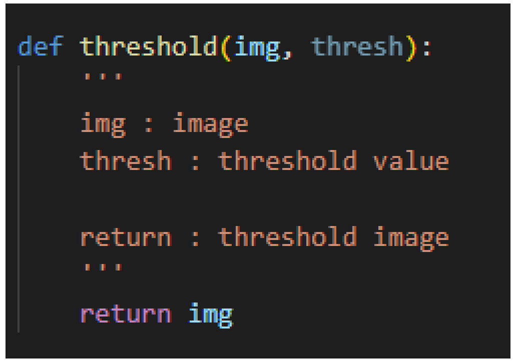
- Threshold 함수 직접 구현하기



```python
def main():
    img = cv2.imread('rices.png', cv2.IMREAD_GRAYSCALE)
    threshold_value = 127
    img_my = threshold(img.copy(), threshold_value)
    thr_cv, img_cv = cv2.threshold(img.copy(), threshold_value, 255, cv2.THRESH_BINARY)

    print('diff : ', (img_my.astype(np.float32) - img_cv.astype(np.float32)).sum())

    cv2.imshow('my threshold', img_my)
    cv2.imshow('cv2 threshold', img_cv)
    cv2.waitKey()
    cv2.destroyAllWindows()
```





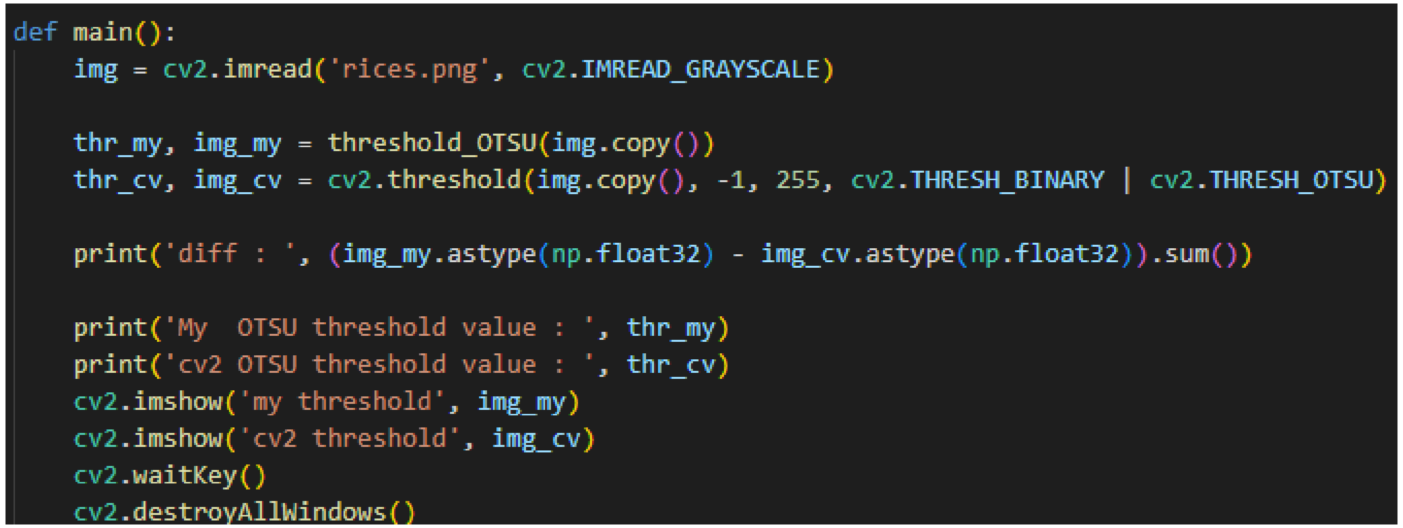


```python
def threshold(img, thresh):
    '''
    img : image
    thresh : threshold value

    return : threshold image
    '''
    return img
```

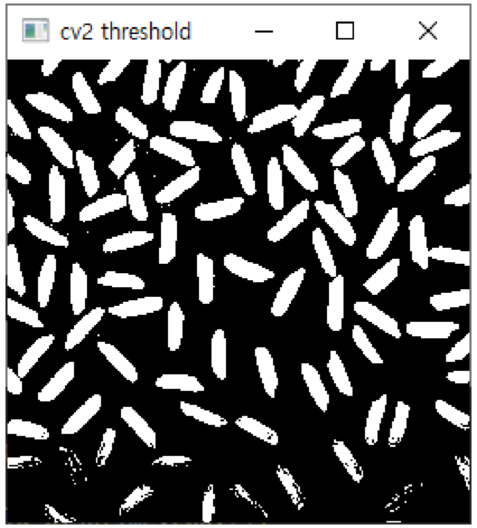
```
diff :  0.0
```

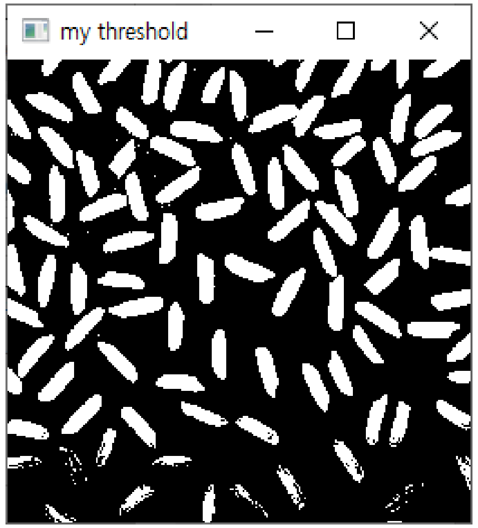
과제(IP3_test2)

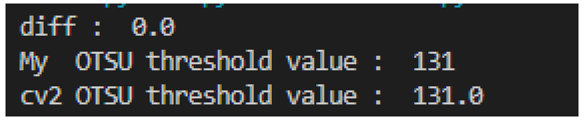
- OTSU's method 직접 구현하기



```
def main():
    img = cv2.imread('rices.png', cv2.IMREAD_GRAYSCALE)

    thr_my, img_my = threshold_OTSU(img.copy())
    thr_cv, img_cv = cv2.threshold(img.copy(), -1, 255, cv2.THRESH_BINARY | cv2.THRESH_OTSU)

    print('diff : ', (img_my.astype(np.float32) - img_cv.astype(np.float32)).sum())

    print('My  OTSU threshold value : ', thr_my)
    print('cv2 OTSU threshold value : ', thr_cv)
    cv2.imshow('my threshold', img_my)
    cv2.imshow('cv2 threshold', img_cv)
    cv2.waitKey()
    cv2.destroyAllWindows()
```



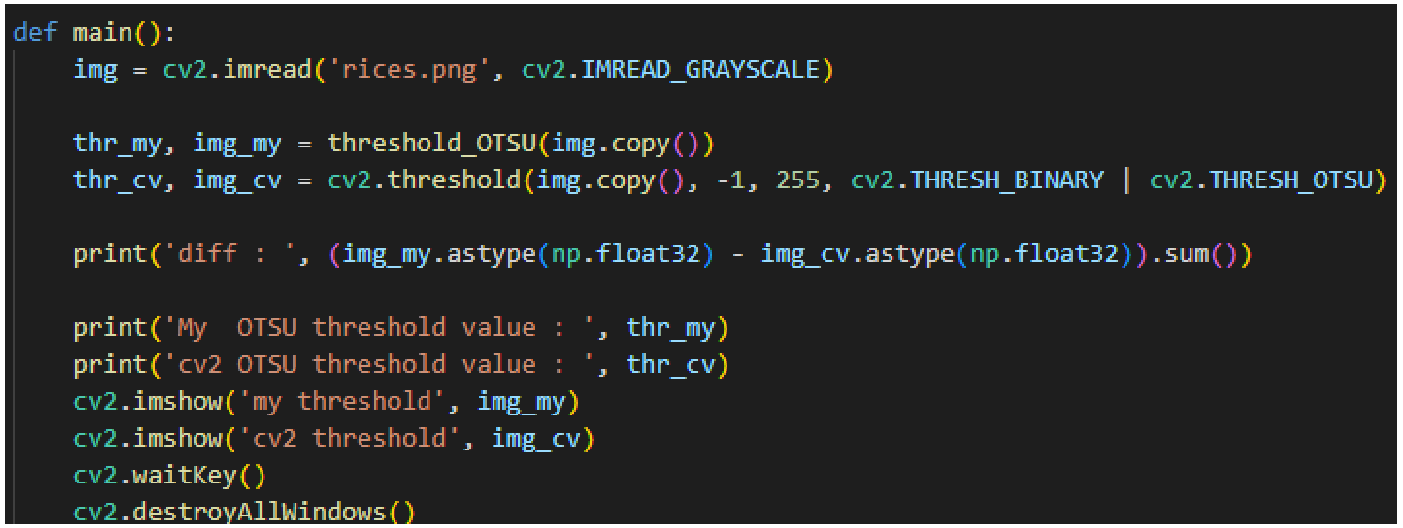




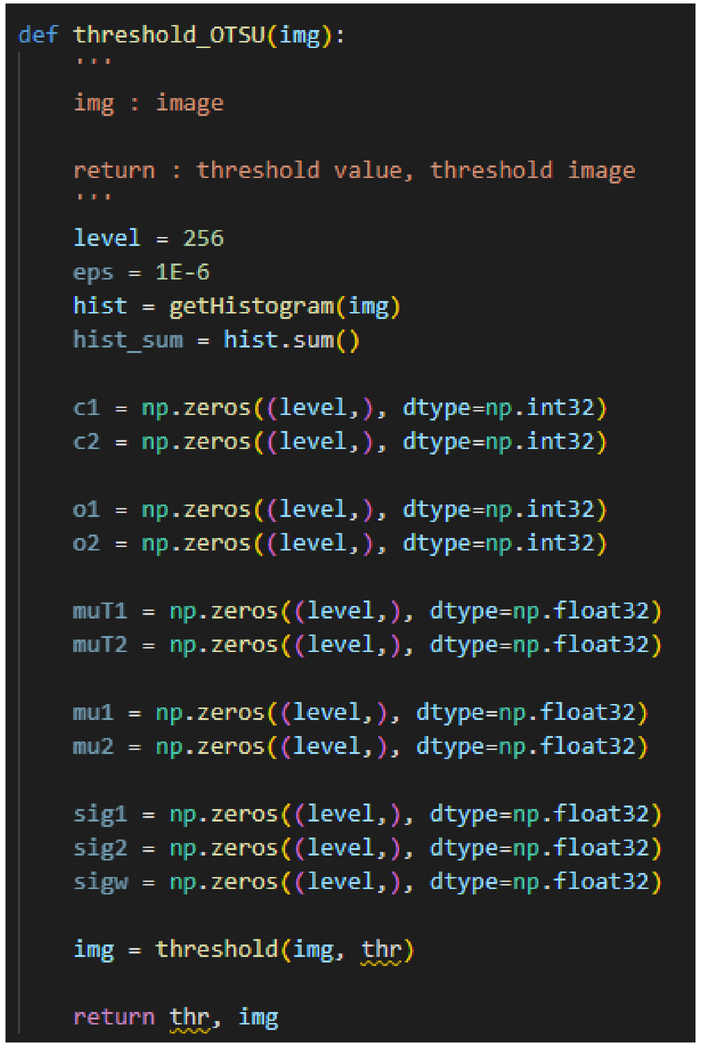
```
diff :  0.0
My  OTSU threshold value :  131
cv2 OTSU threshold value :  131.0
```

과제(IP3_test2)

- OTSU's method 직접 구현하기



```python
def main():
    img = cv2.imread('rices.png', cv2.IMREAD_GRAYSCALE)

    thr_my, img_my = threshold_OTSU(img.copy())
    thr_cv, img_cv = cv2.threshold(img.copy(), -1, 255, cv2.THRESH_BINARY | cv2.THRESH_OTSU)

    print('diff : ', (img_my.astype(np.float32) - img_cv.astype(np.float32)).sum())

    print('My  OTSU threshold value : ', thr_my)
    print('cv2 OTSU threshold value : ', thr_cv)
    cv2.imshow('my threshold', img_my)
    cv2.imshow('cv2 threshold', img_cv)
    cv2.waitKey()
    cv2.destroyAllWindows()
```



```python
def threshold_OTSU(img):
    '''
    img : image

    return : threshold value, threshold image
    '''
    level = 256
    eps = 1E-6
    hist = getHistogram(img)
    hist_sum = hist.sum()

    c1 = np.zeros((level,), dtype=np.int32)
    c2 = np.zeros((level,), dtype=np.int32)

    o1 = np.zeros((level,), dtype=np.int32)
    o2 = np.zeros((level,), dtype=np.int32)

    muT1 = np.zeros((level,), dtype=np.float32)
    muT2 = np.zeros((level,), dtype=np.float32)

    mu1 = np.zeros((level,), dtype=np.float32)
    mu2 = np.zeros((level,), dtype=np.float32)

    sig1 = np.zeros((level,), dtype=np.float32)
    sig2 = np.zeros((level,), dtype=np.float32)
    sigw = np.zeros((level,), dtype=np.float32)

    img = threshold(img, thr)

    return thr, img
```

QnA