영상처리

- Pixel-wise operation-

제출일자	2021.3.22
분 반	01
이 름	강인한
학 번	201701969

1	구현 코드 및 설명

```
def my_normalize_hist(hist, pixel_num):
    normalized_hist= [hist[i] / pixel_num for i in range(len(hist))]
    return normalized_hist
```

이 함수는 픽셀의 개수인 256개로 나눠주는 작업으로 간단히 for 문으로 index에 접근하여 pixel_num 으로 나눠준다.

```
def my_PDF2CDF(pdf):
    cdf= []
    for i in range(len(pdf)):
        if i==0 :
            cdf.append(pdf[i])
        else :
            cdf.append(pdf[i]+ cdf[i-1])
    return cdf
```

함수 my_PDF2CDF 는 PDF의 값을 쌓는 과정으로 cdf의 마지막엔 무조건 1이 들어가게 된다.

```
def my_denormalize(normalized, gray_level):
    denormalized = []
    for i in range(len(normalized)):
        denormalized.append(normalized[i] * gray_level)
    denormalized = numpy.array(denormalized)
    return denormalized
```

함수 my_denormalized 는 cdf의 값에 256을 곱해주는 작업이다.

```python
def my_calcHist_equalization(denormalized, hist):
    hist_equal = []
    for i in range(len(hist)):
        sum=0
        for j in range(256):
            if denormalized[j]==i:
                sum= sum + hist[j]
        hist_equal.append(sum)

    return hist_equal
```

my_calcHist_equalization 함수는 평활화를 해주는 작업으로 인자로 받은 denormalized 함수의 값이 0부터 255의 값을 갖는 j를 모두 찾아 개수를 쌓아 hist_equal에 추가한다.

```python
def my_equal_img(src, output_gray_level):

    h, w = src.shape[:2]
    dst = np.zeros((h,w), dtype = np.uint8)
    for row in range(h):
        for col in range(w):
            dst[row,col]= output_gray_level[src[row,col]]
    return dst
```

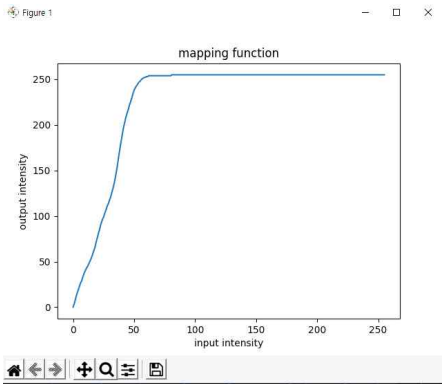
my_equal_img 함수는 기존의 이미지인 src를 평활화된 dst로 바꾸는 작업이다. dst의 틀을 만들고 이중 for문을 통해 src의 값을 인자로 갖는 output_gray_lever을 dst에 저장한다.

```python
# show mapping function
#####################################################################
# TODO                                                              #
# plt.plot(???, ???)완성                                            #
# plt.plot(y축, x축)                                                #
#####################################################################
plt.plot(output_gray_level, )
plt.title('mapping function')
plt.xlabel('input intensity')
plt.ylabel('output intensity')
plt.show()
```

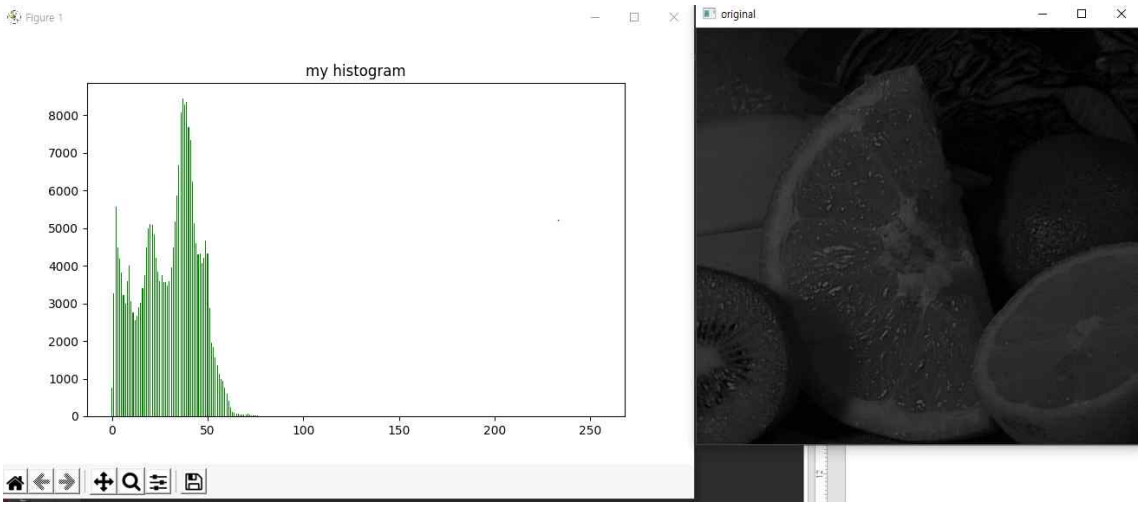
mapping function을 만들기 위한 작업으로 y축에는 output_gray_level의 값을 넣어주고 x축은 빈칸으로 한다.

2	결과 화면 및 이미지

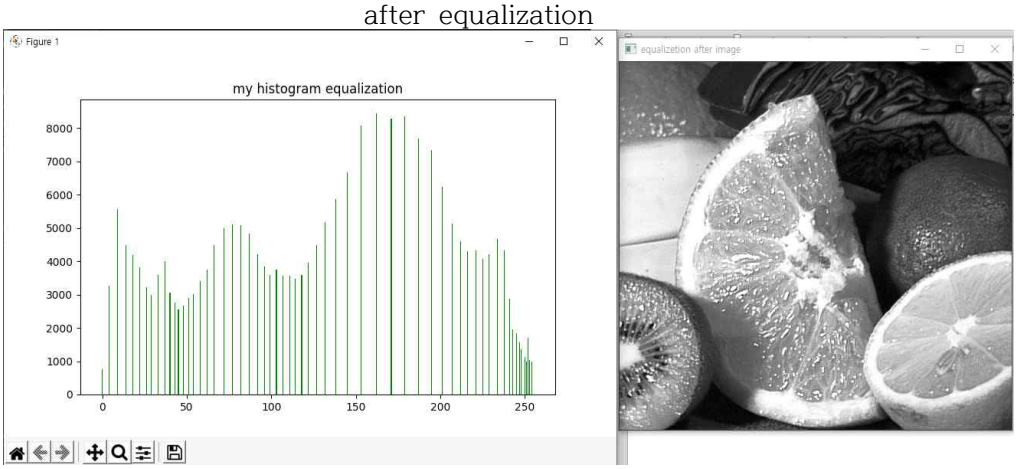
mapping function



original



after equalization



3	느낀 점 및 과제 난이도

과제에 대해 느낀 점으로는 출력에 대해서 이전에는 단순히 결과 화면이었는데 출력이 이미지로 그리고 그래프로 나오니 흥미로웠습니다.

과제의 난이도는 간단히 곱하고 pdf를 적재하고 등등은 쉬웠습니다. 하지만 python에 익숙하지 않아 자료 형을 다루는 것에 있어서 조금 어려운 부분이 있었습니다. 또 src를 통해 dst를 만드는 함수인 my_equal_img에서 뭘 어떻게 할지 몰라 조금 어려웠습니다. 또한 평활화를 하는 my_calcHist_equalization 함수에서 denormalized값과 hist값 을 통해 hist_equal의 값을 만들어 내는 것을 생각하기가 어려웠습니다. 이 부분에서 시간이 많이 소요돼 과제 총시간은 7~8시간 정도 걸린 것 같습니다.