01 Matlab Code

1. Scale Transformation Function

설명 : 스케일 변환 함수

```
% Scaling transformation function
function y = Scale(x, max_value, min_value)
y = (255/(max_value-min_value)*(x-min_value));
end
```

2. Equalize Transformation Function

설명 : 평활화 변환 함수

```
%Equalize Transform Fuction, x is histogram value
function y = ETF(x)
m = size(x);
sum = 0;
for i = 1:m
    sum = sum+x(i);
    y(i,1) = sum;
end
end
```

3. Image Equalizing Fuction

설명 : 이미지가 ETF 에 의해 평활화 되도록 도와주는 함수

```
% x1 is original image and x2 is ETF
function y = EF(x1, x2)
[m, n] = size(x1);
for i = 1:m
    for j = 1:n
        y(i, j) = x2(x1(i, j)+1);
    end
end
```

4. Main Code

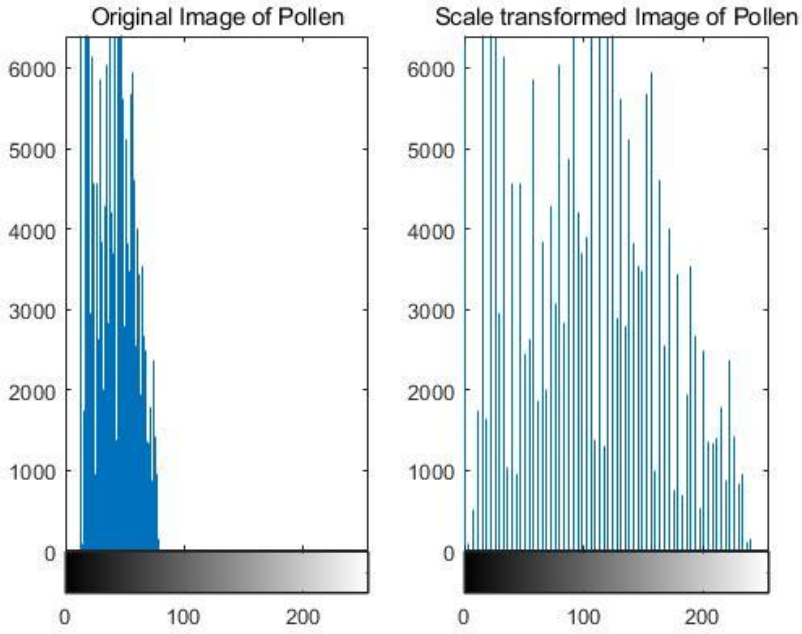
```
%%% Robot Vision%%%
%%% Dept. of Electronic Engineering
%%% 201314651 Lee Wonjai

IM_Pollen = imread('C:\Users\user\OneDrive\¹ÙÅÁ È-¸é\4ÇÐ³â
2ÇÐ±â\·Î°¿°ñÀü\Original
Images\dipum_images_ch03\Fig0308(a)(pollen).tif'); % read the targeted
image
IM_Moon = imread('C:\Users\user\OneDrive\¹ÙÅÁ È-¸é\4ÇÐ³â
2ÇÐ±â\·Î°¿°ñÀü\Original Images\dipum_images_ch03\Fig0310(a)(Moon
Phobos).tif');


P_Pollen = imhist(IM_Pollen);
P_Moon = imhist(IM_Moon);

%Find Max and Min value of Pollen Image
Min1 = min(find(P_Pollen))-1; %also can be done using min and max value of
IM_Pollen
Max1 = max(find(P_Pollen))-1; %the reason of -1 is that array in matlab
start to calculate the index of array as 1

%Find Max and Min value of Moon Image
Min2 = min(find(P_Moon))-1;
Max2 = max(find(P_Moon))-1;

IMScale_Pollen = Scale(IM_Pollen, Max1, Min1);
IMScale_Moon = Scale(IM_Moon, Max2, Min2);

PScale_Pollen = imhist(IMScale_Pollen);
PScale_Moon = imhist(IMScale_Moon);

[m1, n1] = size(IM_Pollen);
[m2, n2] = size(IM_Moon);

ETF_Pollen = ETF(P_Pollen)./(m1.*n1).*255;
ETF_Moon = ETF(P_Moon)./(m2.*n2).*255;

EF_Pollen = uint8(EF(IM_Pollen, ETF_Pollen));
EF_Moon = uint8(EF(IM_Moon, ETF_Moon));

CEF_Pollen = histeq(IM_Pollen, 256);
CEF_Moon = histeq(IM_Moon, 256);

% Compare the histogram of Original Image and scale transformed Image
figure(1);
subplot(1,2,1);
imhist(IM_Pollen)
title('Original Image of Pollen')
subplot(1,2,2);
imhist(IMScale_Pollen)
title('Scale transformed Image of Pollen')
```

```
figure(2);
subplot(1,2,1);
imhist(IM_Moon)
title('Original Image of the Moon')
subplot(1,2,2);
imhist(IMScale_Moon)
title('Scale transformed Image of the Moon')

% Compare the Original Image and scale transformed Image
figure(3);
subplot(1,2,1);
imshow(IM_Pollen)
title('Original Image of Pollen')
subplot(1,2,2);
imshow(IMScale_Pollen)
title('Scale transformed Image of Pollen')

figure(4)
subplot(1,2,1);
imshow(IM_Moon)
title('Original Image of the Moon')
subplot(1,2,2);
imshow(IMScale_Moon)
title('Scale transformed Image of the Moon')

% Compare the histogram of Original Image and equalized Image
figure(5);
subplot(1,2,1);
imhist(IM_Pollen)
title('Original Image of Pollen')
subplot(1,2,2);
imhist(EF_Pollen)
title('Equalized Image of Pollen')

figure(6);
subplot(1,2,1);
imhist(IM_Moon)
title('Original Image of the Moon')
subplot(1,2,2);
imhist(EF_Moon)
title('Equalized Image of the Moon')


% Compare the Original Image and equalized Image
figure(7);
subplot(1,2,1);
imshow(IM_Pollen)
title('Original Image of Pollen')
subplot(1,2,2);
imshow(EF_Pollen)
title('Equalized Image of Pollen')

figure(8);
subplot(1,2,1);
imshow(IM_Moon)
title('Original Image of the Moon')
```

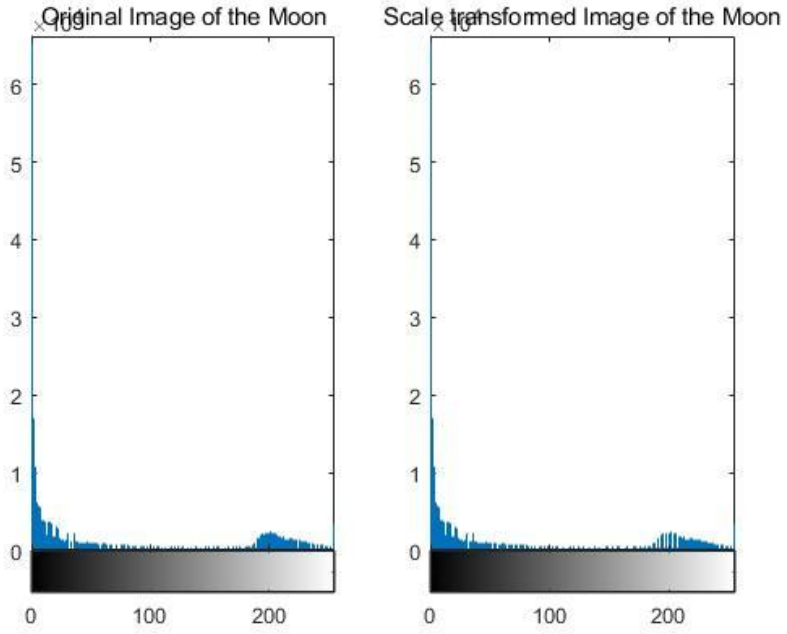
```
subplot(1,2,2);
imshow(EF_Moon)
title('Equalized Image of the Moon')

%Compare whether equalization that I made is correct or not
figure(9)
subplot(1,2,1);
imhist(EF_Pollen)
title('Equalization Fuction I made')
subplot(1,2,2);
imhist(CEF_Pollen)
title('Equalization Fuction')

figure(10)
subplot(1,2,1);
imhist(EF_Moon)
title('Equalization Fuction I made')
subplot(1,2,2);
imhist(CEF_Moon)
title('Equalization Fuction')

figure(11)
subplot(1,2,1);
imshow(EF_Pollen)
title('Equalization Fuction I made')
subplot(1,2,2);
imshow(CEF_Pollen)
title('Equalization Fuction')

figure(12)
subplot(1,2,1);
imshow(EF_Moon)
title('Equalization Fuction I made')
subplot(1,2,2);
imshow(CEF_Moon)
title('Equalization Fuction')
```

02 Result 1

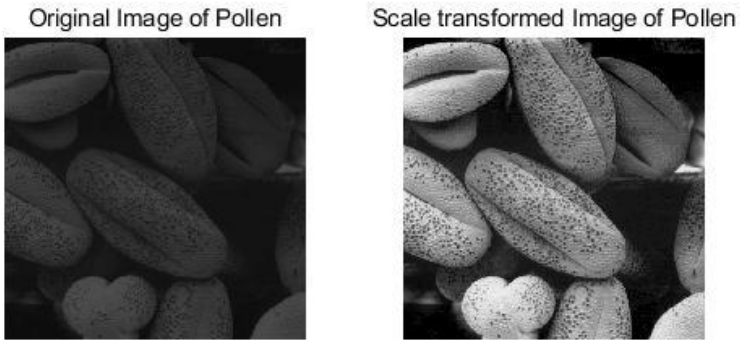
1. Histogram of Original and Scaled Image of Pollen



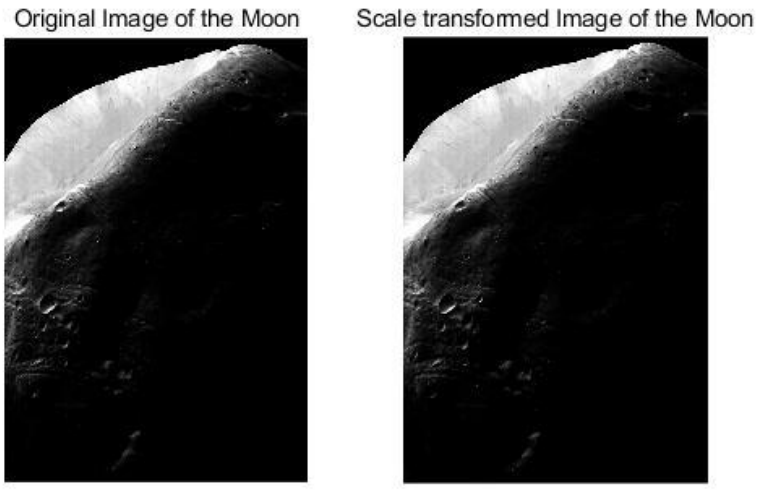
2. Histogram of Original and Scaled Image of the Moon



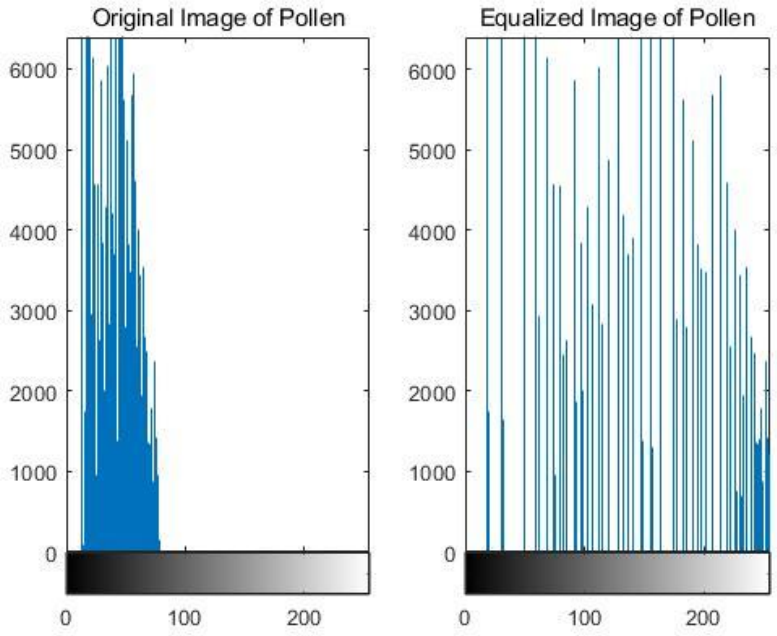
3. Original and Scaled Image of Pollen



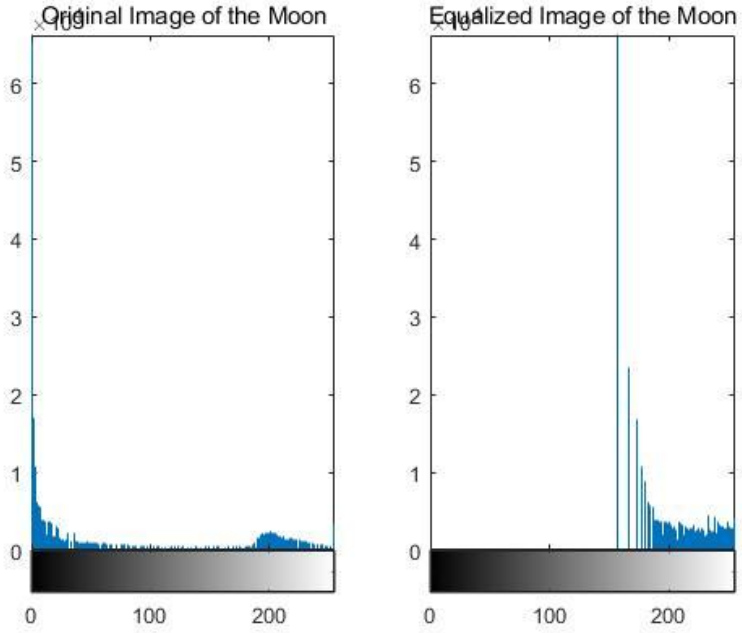
4. Original and Scaled Image of the Moon



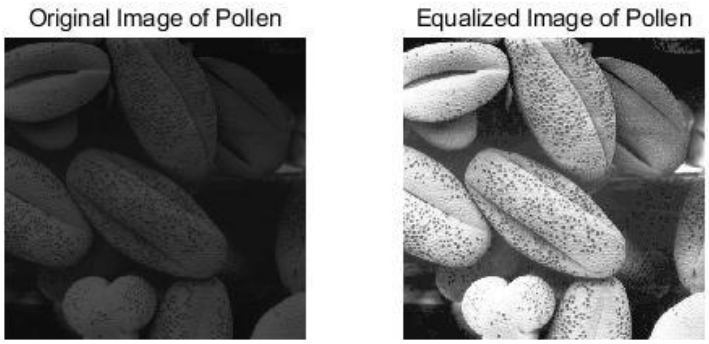
5. Histogram of Original and Equalized Image of Pollen



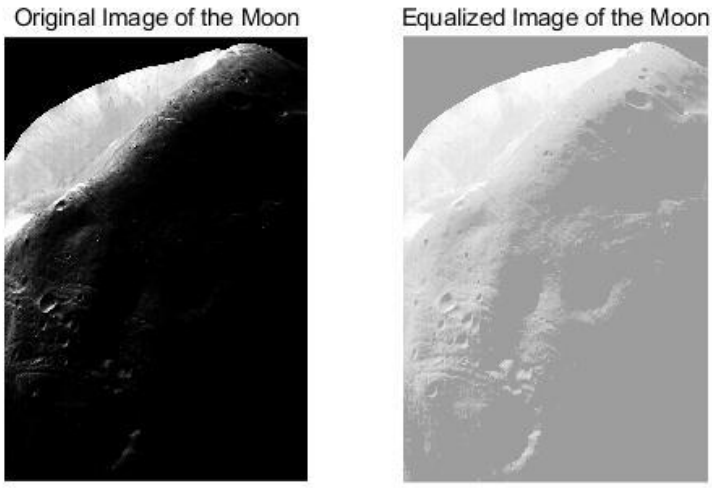
6. Histogram of Original and Equalized Image of the Moon



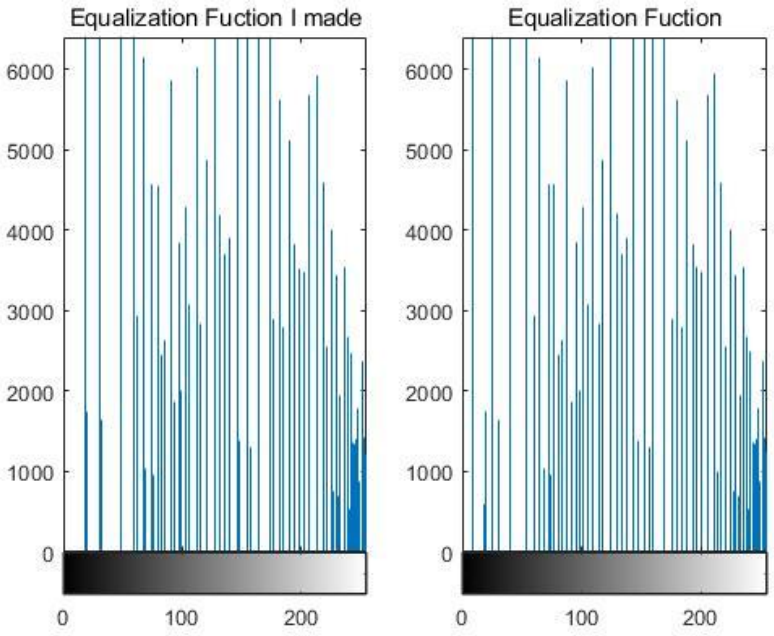
7. Original and Equalized Image of Pollen



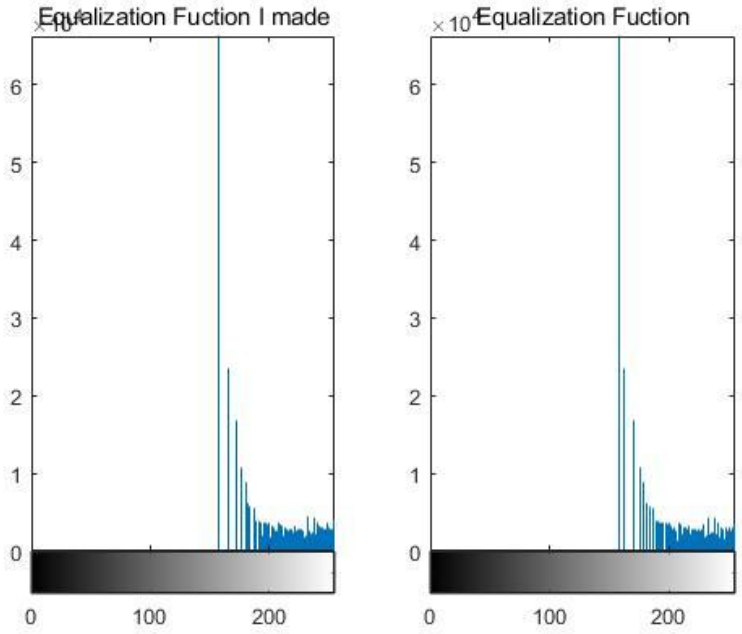
8. Original and Equalized Image of the Moon



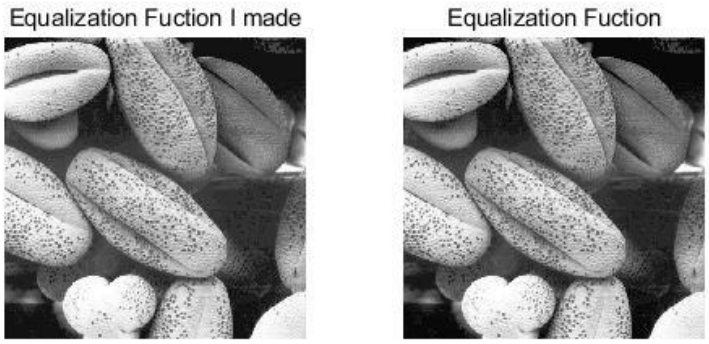
9. Check the Function that I made whether right or not (Pollen)



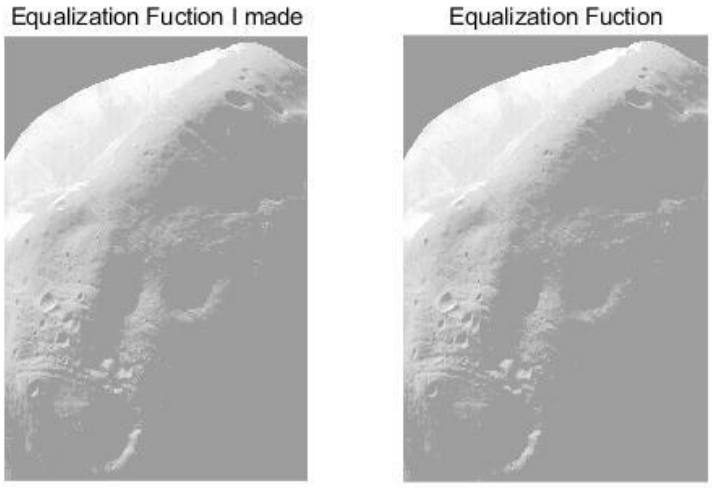
10. Check the Function that I made whether is right or not (Moon)



11. Check the Function that I made whether right or not (Pollen)



12. Check the Function that I made whether right or not (Moon)



02 심화학습(python 구현)

```
import cv2
import numpy as np
import random
from matplotlib import pyplot as plt

#Read the Images
Im_Pollen = cv2.imread('Fig0308(a)(pollen).tif',cv2.IMREAD_UNCHANGED)
Im_Moon = cv2.imread('Fig0310(a)(Moon Phobos).tif',cv2.IMREAD_UNCHANGED)


#Make Histogram Matrix
Hist_Pollen = cv2.calcHist([Im_Pollen],[0],None,[256],[0,256])
Hist_Moon = cv2.calcHist([Im_Moon],[0],None,[256],[0,256])


#Scale Transformation Function
def Scale(Matrix1):
    Min = np.amin(Matrix1)
    Max = np.amax(Matrix1)
    M1, N1 = Matrix1.shape
    A1 = np.zeros((M1,N1))
    A1 = (255/(Max-Min)*(Matrix1-Min))
    return A1
```

```
#Equalize Transform Fuction
def ETF(Histogram1):
    M2, N2 = Histogram1.shape
    Sum = 0
    A2 = np.zeros((M2,N2))
    for i in range(0,M2):
        Sum = Sum + Histogram1[i]
        A2[i] = Sum
    return A2

#Function that helps to make image equilized
def EF(Orig_Image, ETF1):
    M3, N3 = Orig_Image.shape
    A3 = np.zeros((M3,N3))
    for j in range(M3):
        for k in range(N3):
            A3[j, k] = ETF1[Orig_Image[j, k]]
    return A3

#Scale Transformation
Sim_Pollen = np.uint8(Scale(Im_Pollen))
Sim_Moon = np.uint8(Scale(Im_Moon))

#Histogram of Scaled Images
Hist_SPollen = cv2.calcHist([Sim_Pollen],[0],None,[256],[0,256])
Hist_SMoon = cv2.calcHist([Sim_Moon],[0],None,[256],[0,256])
```

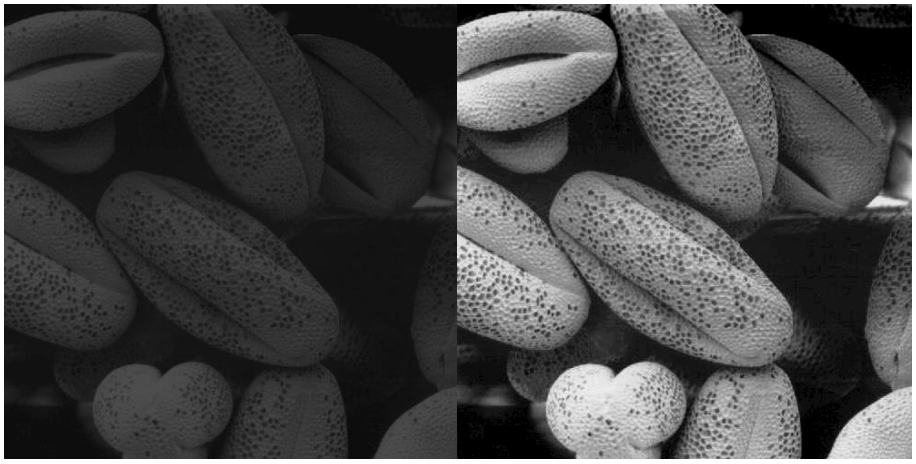
```
#Make Equalize Transform Function
M4, N4 = Im_Pollen.shape
M5, N5 = Im_Moon.shape
ETF_Pollen = ETF(Hist_Pollen)*255/(M4*N4)
ETF_Moon = ETF(Hist_Moon)*255/(M5*N5)

#Equalize the Images
EF_Pollen = np.uint8(EF(Im_Pollen, ETF_Pollen))
EF_Moon = np.uint8(EF(Im_Moon, ETF_Moon))


#Histogram of Equalized Images
Hist_EFPollen = cv2.calcHist([EF_Pollen],[0],None,[256],[0,256])
Hist_EFMoon = cv2.calcHist([EF_Moon],[0],None,[256],[0,256])

#Show all Images
cv2.imshow('Pollen1', Im_Pollen)
cv2.imshow('Pollen2', SIm_Pollen)
cv2.imshow('Epollen', EF_Pollen)

cv2.imshow('Moon1', Im_Moon)
cv2.imshow('Moon2', SIm_Moon)
cv2.imshow('EMoon', EF_Moon)
```

```
#Store all Images
cv2.imwrite('Pollen.tif', Im_Pollen)
cv2.imwrite('SPollen.tif', SIm_Pollen)
cv2.imwrite('EFpollen.tif', EF_Pollen)

cv2.imwrite('Moon.tif', Im_Moon)
cv2.imwrite('SMoon.tif', SIm_Moon)
cv2.imwrite('EFMoon.tif', EF_Moon)


#Show Images and Their Histogram
plt.figure(1)
plt.subplot(221),plt.imshow(Im_Pollen,'gray'),plt.title('Pollen')
plt.subplot(222),plt.imshow(Im_Moon, 'gray'),plt.title('Moon')
plt.subplot(223),plt.plot(Hist_Pollen)
plt.subplot(224),plt.plot(Hist_Moon)
plt.xlim([0,256])
plt.show()

#Compare the Histogram of Orginal Images and Scaled Images
plt.figure(2)
plt.subplot(221),plt.plot(Hist_Pollen),plt.title('Pollen')
plt.subplot(222),plt.plot(Hist_Moon),plt.title('Moon')
plt.subplot(223),plt.plot(Hist_SPollen)
plt.subplot(224),plt.plot(Hist_SMoon)
plt.xlim([0,256])
plt.show()
```

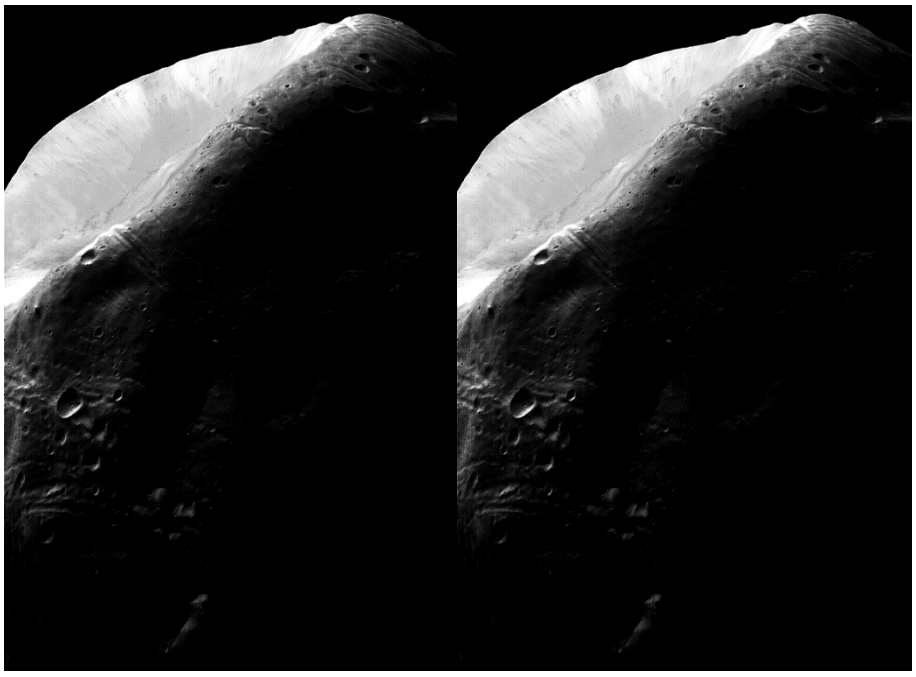
```
#compare the Histogram of Original Images and Equalized Images
plt.figure(3)
plt.subplot(221),plt.plot(Hist_Pollen),plt.title('Pollen')
plt.subplot(222),plt.plot(Hist_Moon),plt.title('Moon')
plt.subplot(223),plt.plot(Hist_EFPollen)
plt.subplot(224),plt.plot(Hist_EFMoon)
plt.xlim([0,256])
plt.show()


cv2.waitKey(0)
cv2.destroyAllWindows()
```

Result 2

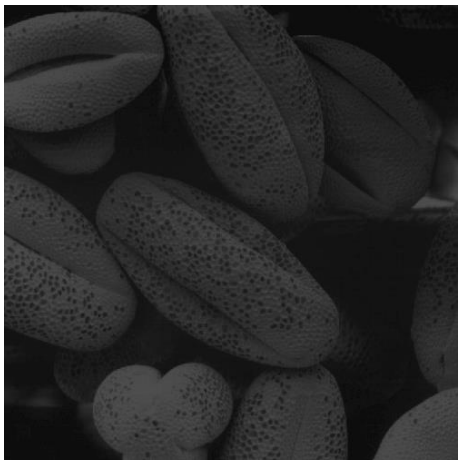
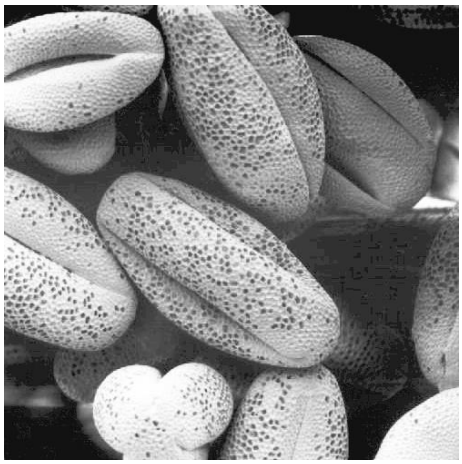
1. Original and Scaled Image of Pollen



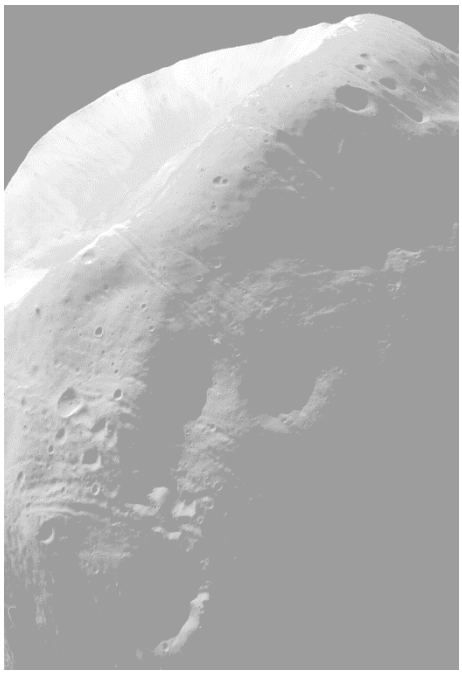
2. Original and Scaled Image of the Moon



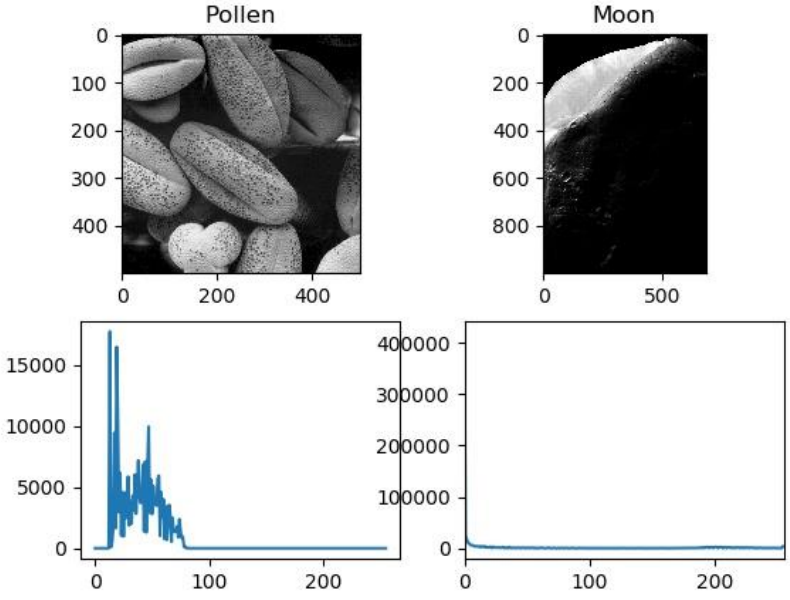
3. Original and Equalized Image of Pollen

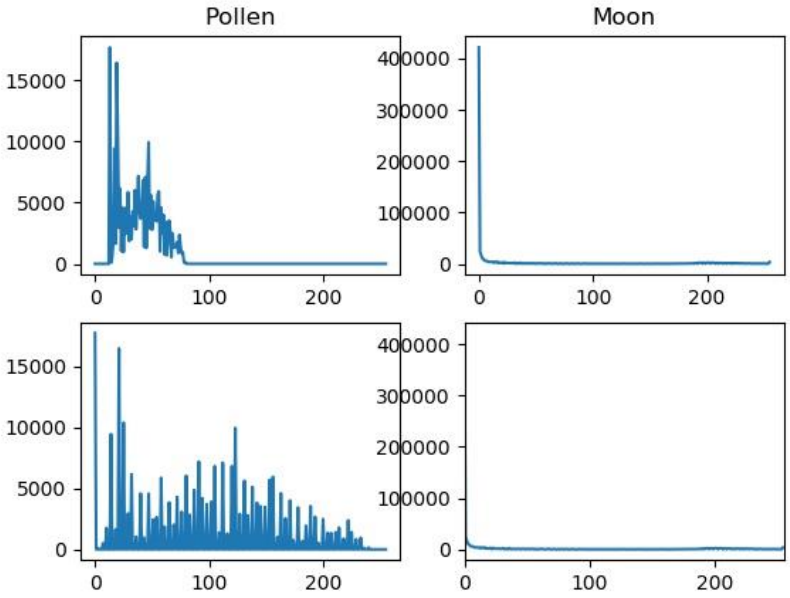
4. Original and Equalized Image of Moon

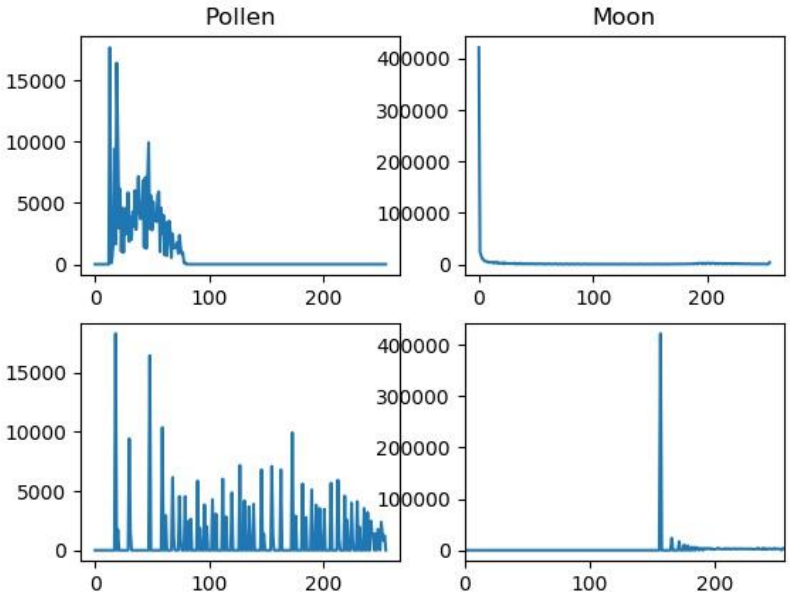
5. Histogram of the Originals and It's Images



6. Histogram of Original and Scaled Images



7. Histogram of Original and Equalized Images



Conclusion

1. Matlab

- Scale 처리의 경우 **Pollen은 좋은 이미지**가 되었고, 이를 히스토그램에서 확인할 수 있었다. 그러나 **Moon은 큰 차이가** 없었다. 그 이유는 히스토그램을 보면 기존의 Moon 이미지 픽셀 값 자체가 **넓게 분포**되어 있기 때문이고, 특히 **어두운 값(1에 가까운 값) 밝은 값(255에 가까운 값)에 집중 분포** 되어있기에 Scale 처리 효과가 미미하다.
- **Equalization은 이미지 전체 밝기를 전반적으로 밝게** 해준다. 따라서 Moon의 경우에도 Scale에서 미처 하지 못했던 기능을 수행한다. 다만 전반적 밝기가 밝아져, Contrast가 약해졌다. 그럼에도 불구하고 **어두운 부분의 이미지 판별**이 필요한 경우가 있어 **용이한 사진**이다.
- **Main Code에서 Scale 범위 찾을 때 find 함수 쓴 이유:** 단순히 히스토그램에서 0이 아닌 값의 시작과 끝을 찾는다는 의미를 주기 위해서이다. 그러나 굳이 find 쓸 필요 없이 이미지에 **min(min())** 또는 **max(max())**를 쓰면 컴퓨터적으로 계산이 더 쉽다.
- Equalization할 때에는 Scale과 다르게 함수 식을 세워 대입하는 것이 아니기에 **CDF**에 **이미지 값**을 직접 **for구문을 써서 대입**해준다.

2. Python

- 이번 python의 경우 library로 cv2, numpy, matplotlib를 썼다. 특히 **matplotlib**의 경우 매트랩과 비슷한 개발 환경을 제공해주는 라이브러리이기에 **두 영상 간의 변화를 비교하기에 좋다**.
- Matplotlib를 이용했기에 **plt** 함수를 이용하여, Matlab처럼 **figure, subplot을 만드는 것이 가능**해짐.
- **Matplotlib**은 주로 **데이터를 시각화** 할 때 많이 쓰인다.