# 영상처리 n주차 과제

학번: 20xxxxxxx

이름: XXX

반드시 .pdf 파일로 저장하여 제출해주세요

1. **과제 내용**

**목표 :graythresh 내장 함수를 사용하지 않고 threshold 함수를 구현**

**Between과 within을 구현해야됨**

**Within 코드**

```
    [x,y] = size(img);
    list_k = [];
    for k= 1:256
        until_k = img(img<=k-1);
        if isempty(until_k)
            until_k=[0,0,0];
        end
        from_k = img(img>k-1);
        if isempty(from_k)
            from_k=[0,0,0];
        end
        uk =var(double(until_k));
        fk = var(double(from_k));
        list_k(k) = uk+fk;
    end
    real_k = find(list_k==min(list_k));
if strcmp(type, 'within')
    % Within variance
    % Fill here
    until_k = img<=real_k;
    mask_img = img>real_k;
    within_res = double(img).*double(mask_img);
    imshow(double(img)-within_res);
```

**우선 x,y에 이미지에 대한 가로,세로 길이를 저장합니다.**

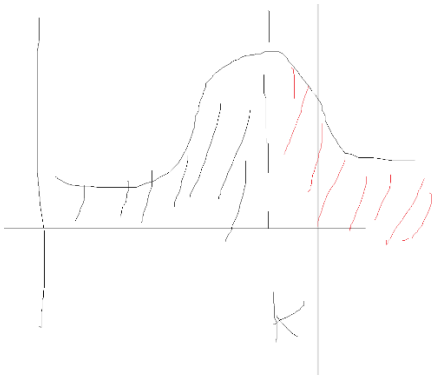
**그후 list_k를 만드는데 이것은 각 픽셀이 가지는 variance의 값을 리스트로 저장하기 위함입니다.**

**0~255가 픽셀의 범위지지만 메트랩은 1부터 인덱스가 시작하므로 1부터 256까지로했습니다. 실**

**절적인 값저장은 -1을해서 0~255까지임. 이제 포문을 분석해보면 img(img<=k-1)부분이 나오는데 이것은 특정 픽셀값보다 적은 곳에 대해 이미지가 가지고있는 모든 픽셀들을 1차원배열로 늘려놓은 것입니다 즉 img(img<=4)면 [1,2,3,4,0,0,0,0,0,0,0,4] 이런식이되는 것입니다. 이것이 until_k이고 반대로 from_k는 k보다 1보다 큰 값인 나머지 전부 입니다. 그러나 배열이 빈경우 연산에 문제가 생길수 있으므로 0,0,0을 넣어줘서 분산 연산에 지장이 없게 합니다.**

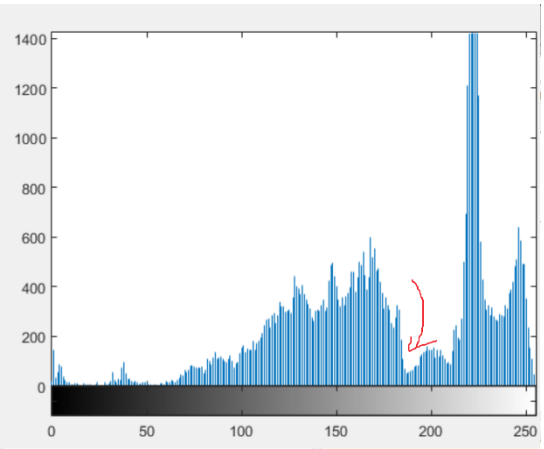
```
uk =var(double(until_k));
fk = var(double(from_k));
```

**이연산은 위의 픽셀 분포에 대해 이 픽셀들의 분산을 나타내는 것입니다.**




**그림처럼 K를 기준으로 나뉩니다.**

**이 k값을 기준으로 나눠진 분산을** `list_k(k) = uk+fk;` **에 추가합니다.**
**for루프가 모두 마친후 가장 이상적인 분포를 갖게하는 픽셀밧을 찾아야하므로,**
`real_k = find(list_k==min(list_k));` 분산의 합의 크기가 가장 적은 즉




**저 부분을 찾는 과정입니다.**

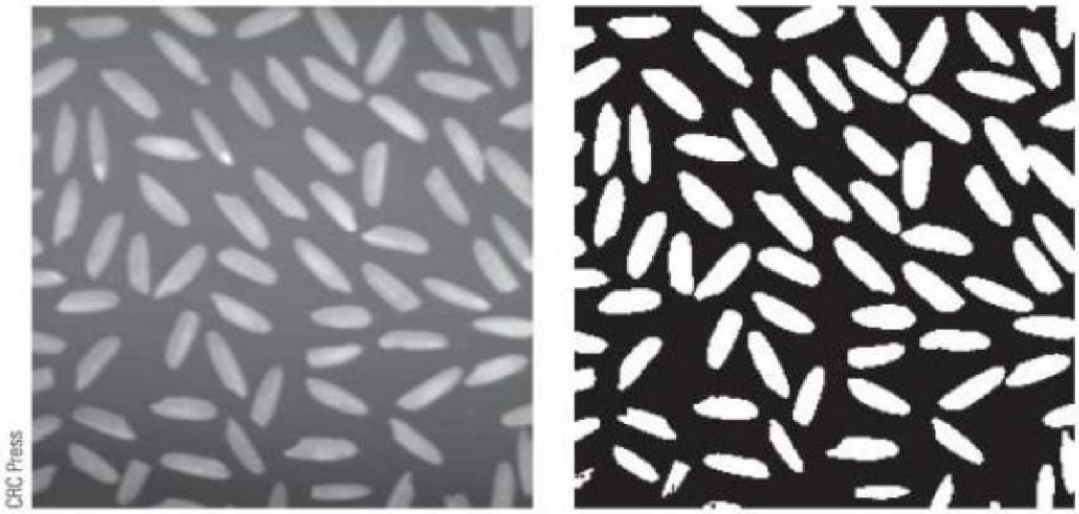
이제 within부분입니다.

찾은 k를 기준으로 k까지의 값과 k이후의 값으로 나뉩니다. 그러나

스레셔홀드 특성상

$$\text{A pixel becomes } \begin{cases} \text{white if its gray level is} > T, \\ \text{black if its gray level is} \leq T. \end{cases}$$

```
>> r=imread('rice.tif');
>> imshow(r),figure,imshow(r>110)
```




객체를 나타내야하는 부분을 찾는 것이 주 목표이므로 k보다 큰 것을 마스크로 사용하였습니다.

Between입니다.

Within과 다르게 솔직히 어떻게 해야할지 감이 잘안잡혀서 그냥 식대로 구현했습니다.

Between

```
else
    % Between variance
    % Fill here
    q1t = zeros(size(img));
    for q1i = 1:real_k
        q1t= q1t+ q1i*double(img==q1i).*double(img);
    end

    q2t = zeros(size(img));
    for q2i = real_k+1:255
        q2t= q2t+q2i*double((img==q2i).*double(img));
    end
    q1 = double(img<=real_k).*double(img);
    q2 = double(img>real_k).*double(img);
    m1 = q1t./q1;
    m1(isnan(m1))=0;

    m2 = q2t./q2;
    m2(isnan(m2))=0;

    mg = q1.*m1+q2.*m2;
    Between = uint8(q1.*((m1-mg)/(x*y)).^2+ q2.*((m2-mg)/(x*y)).^2);
    imshow(Between);

end
```

$$q_1(k)=\sum_{i=0}^{k}p(i) \qquad q_2(k)=\sum_{i=k+1}^{L-1}p(i)$$

$$m_1(k)=\frac{\sum_{i=0}^{k}ip(i)}{\sum_{i=0}^{k}p(i)}=\frac{1}{q_1(k)}\sum_{i=0}^{k}ip(i) \qquad m_2(k)=\frac{\sum_{i=k}^{L-1}ip(i)}{\sum_{i=k+1}^{L-1}p(i)}=\frac{1}{q_2(k)}\sum_{i=k}^{L-1}ip(i)$$

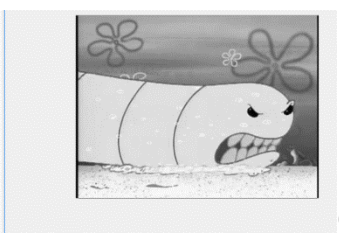
$$\sigma_1^2(k)=\frac{1}{q_1(k)}\sum_{i=0}^{k}[i-m_1(k)]^2p(i) \qquad \sigma_2^2(k)=\frac{1}{q_2(k)}\sum_{i=k+1}^{L-1}[i-m_2(k)]^2p(i)$$

$$=\frac{1}{q_1(k)}\sum_{i=0}^{k}i^2p_i-m_1^2(k) \qquad =\frac{1}{q_2(k)}\sum_{i=k+1}^{L-1}i^2p(i)-m_2^2(k)$$

Pi는 하나의 히스토그램이 가지는 즉 이미지 내에 특정 픽셀이 가지는 값입니다.

우선 q1+q2 = 1

```
0
0
0
0
0
0
0
0
0
0
imshow(q1+q2);
imshow(uint8(q1+q2));
```



여기서 1은 원래의 이미지 값을 의미합니다

즉 q1은 k까지의 이미지가 가진 히스토그램 즉 k만큼의 이미지를 가지고있는 것이고

q2는 k+1부터 255까지의 픽셀을 가진 이미지의 조각입니다. 이것들을 가지고 공식대로 연산을 하는데, k로 인해 나눠진 시그마를 구할 때, 점 나눔을 합니다.

mg는 m1q1+m2q2이므로 공식대로 해줍니다.

그리고 마지막 Between을 구할때

$$\sigma_B^2(k) = q_1(k)[m_1(k) - m_G]^2 + q_2(k)[m_2(k) - m_G]^2$$

이 공식을 사용합니다.

그리고 m을 구할 때 하나 빼먹은 것이 있었는데 일부러 1/MN을 안해주고 있었습니다.

Between을 구할 때 제곱부분 괄호안의 연산을 할 때 같이 묶어서 나누어 줍니다. 그리고 q1 q2를 곱하여 더하면 between에 대한 분산으 완성이 됩니다.

1. **느낀 점**

**조금 개념이 공식과 이미지의 차원이 다차원인 점을 고려하면, 조금 이해가 어려웠고,**

**Pi,var 등등이 이미지 내에서 어떻게 작용하는지, 즉 무엇을 의미하는지에 대해 설명이 더 있엇으면 좋겠습니다.**

**within**

**Between**