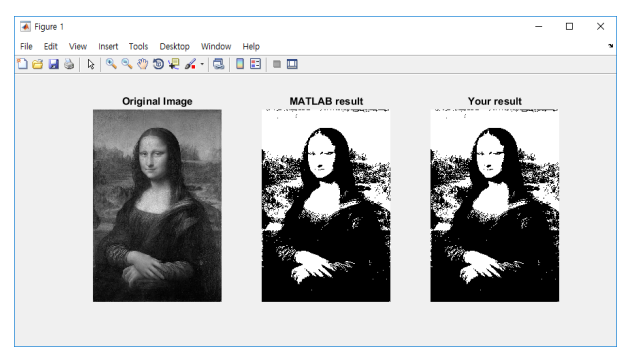
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **과목명** | **영상신호처리** | **학기** | **2023 - 1** | **담당교수** | **장주용** |
| **학과** | **전자통신공학과** | **학번** | **2018707072** | **이름** | **이승룡** |
| **과제 3 - Otsu’s method** | | | | | |

**1. 과제설명**

Implement the Otsu’s method in “otsu.m”. When you run a script file (“hw3.m”), you should get the following figures. Do not use the thresholding related function provided by Matlab (e.g. graythresh())



**2. 이론**

Otsu 알고리즘은 이미지 분할을 위한 optimal threshold을 결정하기 위해 널리 사용되는 방법이다. 해당 알고리즘은 이미지의 픽셀 강도의 between-class variance을 최대화하여 segmentation 을 위한 최적의 threshold를 계산하는데, 과정은 다음과 같다.

- 이미지에서 픽셀 강도의 히스토그램을 계산하고 합이 1이 되도록 정규화한다.

- 정규화된 히스토그램을 이용해 누적 분포 함수(CDF)를 구한다. 여기서 CDF는 픽셀이 특정

값보다 작거나 같은 값을 가질 확률을 나타낸다. 이후 픽셀들의 평균값을 구한다.

- 이후 가능한 모든 임계값(0 ~ 255)을 반복하고 각 임계값에 대한 between-class variance을

계산한다. between-class variance은 임계값이 픽셀 강도를 두 클래스로 얼마나 잘 분리하는지

나타낸다. σBtween(t)=w1(t)w2(t)(μ1(t)−μ2(t))^2

여기서 t는 임계값이고 w1(t)및 w2(t)는 임계값 t로 분할된 두 클래스의 확률이다. μ1(t)및 μ2(t)

는 두 클래스의 평균 픽셀 강도 값이다.

- 임계값을 0-255까지 반복하여 계산하면 max between-class variance에 해당하는 임계값이 선

택된다.

**3. 소스코드 설명**

|  |
| --- |
| % Compute the histogram  counts = imhist(img);  probs = counts / numel(img);  % Compute the cumulative distribution function  cdf = cumsum(probs); |

imhist()를 사용해서 입력 이미지의 히스토그램을 계산하고 numel()을 통해 구한 총 픽셀수로 나눠주어 히스토그램을 nomalize한다. 이후 cumsum()을통해 cumulative distribution function(cdf)를 계산한다.

|  |
| --- |
| max\_var = 0;  th = 0;  % Loop (0 to 255)  for i = 1:256  % Compute the probabilities  class1\_p = cdf(i);  class2\_p = 1 - class1\_p; |

between class variance의 최댓값을 저장할 변수와, 최적의 threshold를 저장할 변수를 초기화한다. threshold가 가능한 값들 0 - 255를 루프 범위로 설정한다. 지정한 threshold(i 값)을 이용해 그룹1이 나올 확률과, 그룹2가 나올 확률을 계산한다.

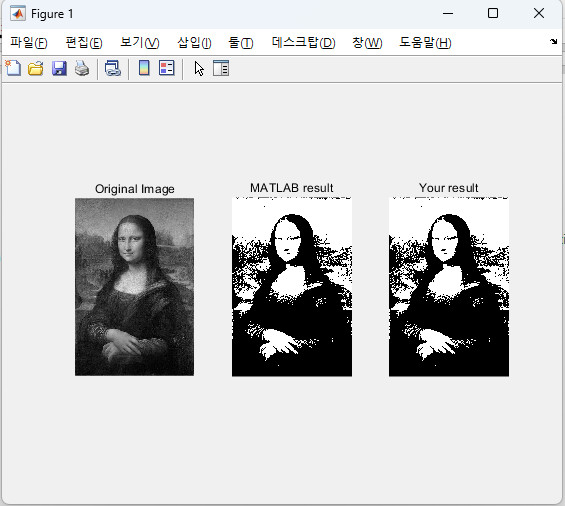
|  |
| --- |
| if class1\_p > 0 && class2\_p > 0  % Compute the means  class1\_m = sum((0:i-1) .\* probs(1:i)') / class1\_p;  class2\_m = sum((i:255) .\* probs(i+1:end)') / class2\_p;  % Compute the between-class variance  var\_between = class1\_p \* class2\_p \* (class1\_m - class2\_m).^2; |

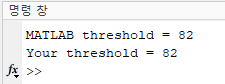
두 그룹의 평균들을 각각 구하고, between-class variance를 구한다.

|  |
| --- |
| % update the optimal threshold  if var\_between > max\_var  max\_var = var\_between;  th = i-1; |

만약 새로 계산한 between-class variance가 기존 값들의 최대인 max\_var보다 크면 threshold와 max\_var을 업데이트한다.

**4. 실행결과**





실행 결과를 확인해보면 matlab function과 동일하게 나온 것을 확인할 수 있다. 또한 명령창에 threshold 값이 82로 동일하게 출력되었다.

**5. 전체 소스코드**

|  |
| --- |
| function th = otsu(img)  % Compute the histogram  counts = imhist(img);  % Normalize the histogram  probs = counts / numel(img);  % Compute the cumulative distribution function  cdf = cumsum(probs);  % Initialize the maximum between-class variance and the optimal threshold value  max\_var = 0;  th = 0;  % Loop (0 to 255)  for i = 1:256  % Compute the probabilities  class1\_p = cdf(i);  class2\_p = 1 - class1\_p;  if class1\_p > 0 && class2\_p > 0  % Compute the means  class1\_m = sum((0:i-1) .\* probs(1:i)') / class1\_p;  class2\_m = sum((i:255) .\* probs(i+1:end)') / class2\_p;  % Compute the between-class variance  var\_between = class1\_p \* class2\_p \* (class1\_m - class2\_m).^2;  % update the optimal threshold  if var\_between > max\_var  max\_var = var\_between;  th = i-1;  end  end  end |