비주얼 컴퓨팅

HW1

실험 보고서

Thresholding
&
Histogram Equalization

2019204014
정보융합학부
오승준

목차

Ⅰ. 실험 개요

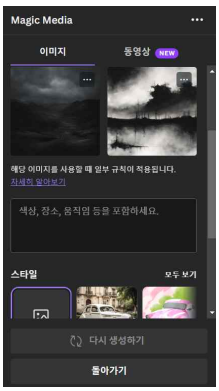
1.1 실험 개요

본 보고서에서 다루는 실험은 총 두 가지로, Image Processing Technique 중 Thresholding과 Histogram Equalization을 각각 이미지에 적용할 때의 변화 양상을 관찰하고, Kernel-Based Local Solution과 Global Solution의 차이를 비교 분석한다.

1.2 데이터 수집 및 선정

Thresholding은 모든 픽셀들의 값을 이진화하여 객체와 배경을 구분하거나 특정 밝기 범위의 픽셀을 강조하기 위해 사용되며, Histogram Equalization은 이미지의 히스토그램을 평탄화하여 이미지의 대비를 높이기 위해 사용된다. 따라서, 실험에 사용된 이미지는 이러한 Image Processing Method의 효과를 쉽게 관찰하기 위해 대비가 낮거나 배경과 객체를 구분하기가 어려운 것들로 후보를 선정하였고, 이 중에서 Local - Global Solution의 뚜렷한 차이를 관찰하기 위해 특정 Region의 Brightness가 주변보다 유난히 높거나 낮은 것을 최종 선택하였다.

이러한 조건들을 만족하는 이미지를 직접 촬영하거나 인터넷에서 직접 찾는 것은 생각보다 쉽지 않았다. 따라서 프롬프트 기반 AI 이미지 생성 모델을 활용하여 실험에 필요한 이미지를 생성하였다. 이때 사용한 서비스는 Canva의 'Magic Media'이며, 이를 이용하여 최종적으로 600x600 Size의 아래와 같은 두 이미지를 구하였다.



Magic Media


역광, 풍경, 그림자,
부분 밝기가 다른


어두운, 저대비의, 풍경,
흐린, 자연

Ⅱ. [실험 1] Thresholding

2.1 Thresholding

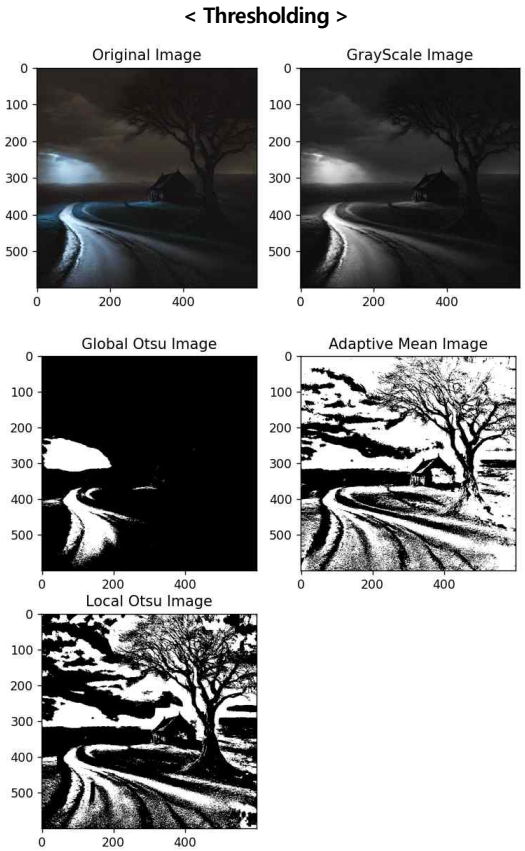
본 실험에서는 세 종류의 Thresholding Method를 동일한 이미지에 적용하고 결과를 비교한다.
실험에서 사용된 Method는 Global Otsu Thresholding, Adaptive Mean Thresholding, Local Otsu Thresholding 이며, Global Otsu Thresholding은 이미지 히스토그램을 토대로 배경과 객체 간 분산이 최대가 되는 '단일 임계값'을 찾은 뒤 전체 이미지에 대해 이 값을 기준으로 모든 Pixel의 value를 0 또는 255로 조정한다.
Adaptive Mean Thresholding은 '주변 Pixel의 평균값'을 활용하여 각 Pixel의 임계값을 다르게 결정하며, Local Otsu Thresholding은 '주변 Pixel들 사이의 히스토그램'을 활용하여 주변 Pixel들 간의 임계값을 결정한다.

2.2 구현

실험에서 사용된 Global Otsu Thresholding과 Adaptive Mean Thresholding은 Opencv의 Threshold 함수와 Adaptive Threshold 함수 + mean_c 옵션을 사용했으며, 이때 함수들의 min, max val은 0과 255, Kernel size는 적절하게 이미지가 구별되는 51, c는 2로 설정하였다.
Local Otsu Thresholding의 경우는 따로 함수가 존재하지 않아 Kernel size만큼 픽셀들을 건너뛰면서 Kernel size로 분할된 영역마다 임계값을 구하도록 구현하였다. 마찬가지로 Kernel size는 51로 설정하였으며, 이미지의 테두리 Pixel들에 대해서도 동일한 처리를 적용하기 위해 Zero Padding을 하려 했으나 이미지 자체가 어둡기에 만약 Zero Padding을 진행한다면 Zero 값의 영향으로 테두리 부근에서의 임계값이 제대로 결정되지 못할 가능성을 염두에 두어 BORDER_DEFAULT 옵션을 사용해 주변 Pixel들과 유사한 값들로 Padding하였다.

2.3 실험 결과

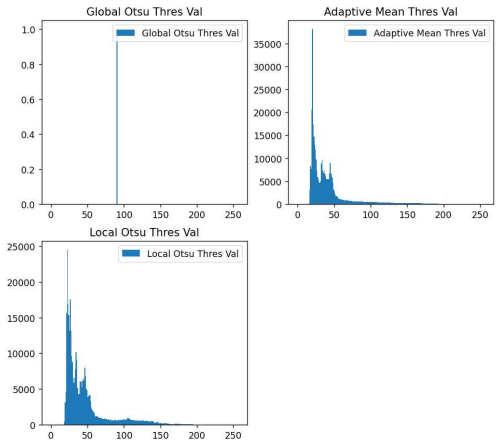
① 실험 1-1



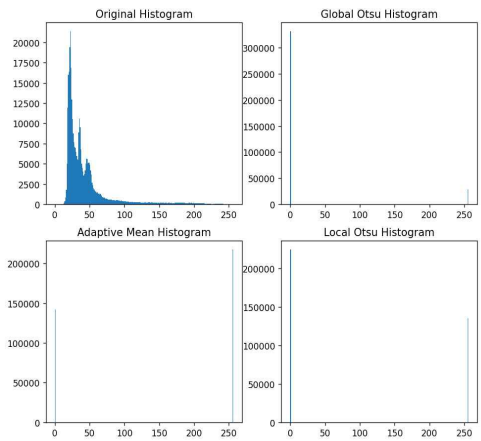
실험에 사용된 이미지는 대체로 어두운 배경에 일부 밝은 빛이 존재하는 그림이다. 따라서, 이미지의 특정 영역은 밝고 특정 영역은 어두운 패턴이 나타나는데, 이러한 영향 때문인지 이미지 전체에 대해 단일 임계값을 사용하는 Global Otsu Thresholding 방식의 경우 이미지가 담고 있는 정보 손실이 크게 나타난 것을 확인할 수 있다.
그에 반해 지역적으로 임계값을 설정하는 Adaptive Mean Thresholding이나 Local Otsu Thresholding 방식의 경우는 상대적으로 이미지의 정보를 잘 담고 있는 것을 확인할 수 있는데, 이는 Kernel Based Solution으로 주변 Pixel들을 기준으로 임계값을 설정했기 때문이다.
추가적으로는 동일한 Kernel Size를 갖더라도 개별 Pixel마다 임계값을 갖는 Adaptive Mean Thresholding 방식보다 인근 Pixel들이 모두 동일한 임계값을 갖는 Local Otsu Threshold 방식이 보다 Edge Detection과 Noise에 강건한 것을 확인할 수 있다.

② 실험 1-2

< 임계값 분포 >



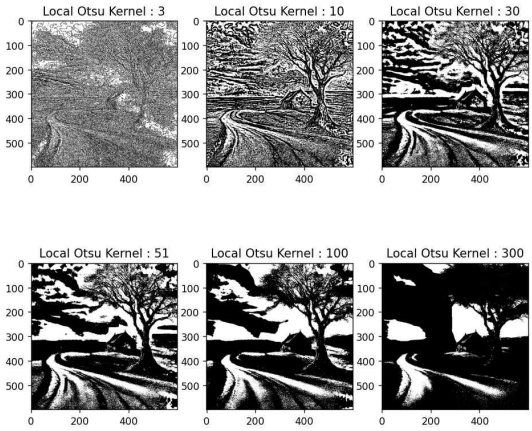
< Pixel Value 분포 >



Threshold 방법에 따른 임계값과 Pixel Value 분포를 보면 Global Otsu Thresholding은 단일 임계값을, 나머지 두 Thresholding은 다양한 임계값을 갖는 것을 알 수 있다. 특히 지역적인 임계값은 Local Otsu Thresholding의 경우가 Adaptive Mean Thresholding의 경우보다 다채롭게 결정된 것을 알 수 있으며, 흑백 Pixel 분포를 참고하면 Adaptive Mean Thresholding의 경우는 나머지 Otsu Thresholding보다 대체적으로 흰색으로 Mapping된 Pixel의 수가 많다는 특징이 있다.

③ 실험 1-3

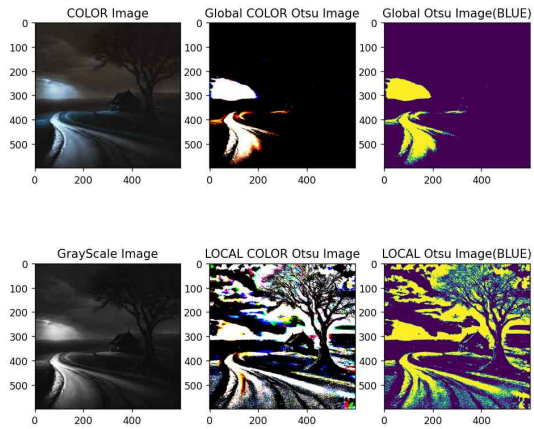
< Kernel Size에 따른 Local Otsu Thresholding >



Local Otsu Thresholding에서 Kernel Size가 작을수록 Noise가 증가하고 Kernel Size 만큼의 주변 Pixel 간 값 차이가 크지 않아 Kernel Size 만큼의 동일한 값들로 채워진 Pixel block들이 자주 관찰된다. 반대로 Kernel Size가 커질수록 점점 Global Otsu Thresholding과 유사한 결과가 나타난다. 따라서 Kernel Based Thresholding에는 적절한 Kernel Size가 중요하다는 것을 확인할 수 있다.

④ 실험 1-4

< RGB Channel에서의 Local Otsu Thresholding >



GrayScale에서의 Thresholding뿐만 아니라 RGB Channel로 split 후 각각의 Color Channel 별로 Thresholding을 적용한 뒤 이를 merge 혹은 특정 Channel만을 따로 보더라도 Global Otsu Thresholding보다는 Local Otsu Thresholding에서가 배경과 객체의 구분이 잘 일어나는 것을 확인할 수 있다.

Ⅲ. [실험 2] Histogram Equalization

3.1 Histogram Equalization

본 실험에서는 Global Histogram Equalization과 Adaptive Histogram Equalization을 동일한 이미지에 적용하고 결과를 비교한다.

Histogram Equalization은 이미지의 픽셀 값 분포를 균등하게 만들어 일부 영역에서 약간의 정보 손실을 감안하더라도 이미지의 대비를 향상시키는 방법이며, Global Histogram Equalization의 경우는 전체 이미지에 대해 균일화를 적용하고 Adaptive Histogram Equalization의 경우는 이미지의 블록(Kernel)에 대해 균일화를 적용한다는 특징이 있다.

3.2 구현

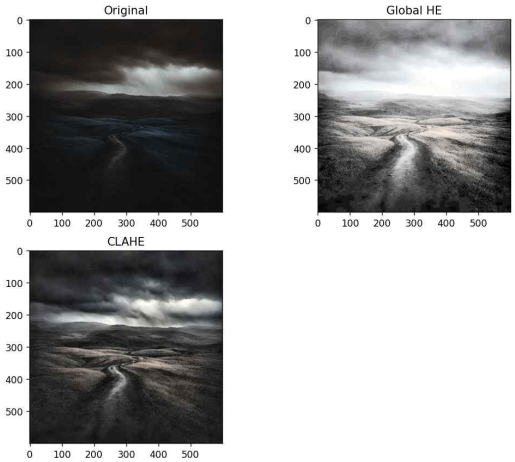
실험에서 사용된 Global Histogram Equalization은 Opencv의 equalizeHist 함수를 사용했으며, Adaptive Histogram Equalization은 Opencv의 createCLAHE 함수를 사용하되, 함수의 clipLimit은 5.0, tileGridSize는 (10, 10)로 설정하였다. CLAHE는 Adaptive Histogram Equalization의 한 종류로써 극단적인 균일화를 제한하여 이미지의 노이즈를 감소시키도록 설계되었다. 함수의 파라미터 중 clipLimit은 Histogram의 각 Bin에서 허용할 수 있는 최대 대비 값을 의미하고, tileGridSize은 이미지를 나눌 블록의 크기를 의미한다.

본 실험에서는 컬러 이미지에 대해 Histogram Equalization을 적용하기 위해 이미지를 R, G, B Channel로 나누고 각 채널에서의 최고, 최저값을 기준으로 새롭게 Pixel 값을 Mapping 한 뒤 Histogram Equalization을 적용하고 다시 Merge하는 방식과 YCrCb color space로 이미지를 변환한 다음 밝기에 해당하는 Y Channel (Y = 0.299×R + 0.587×G + 0.114×B)에 대해서만 Histogram Equalization을 적용한 뒤 다시 Merge 후 RGB로 변환하는 방식 중 후자를 선택하여 Histogram Equalization을 통한 컬러 이미지의 대비 향상을 확인하였다.

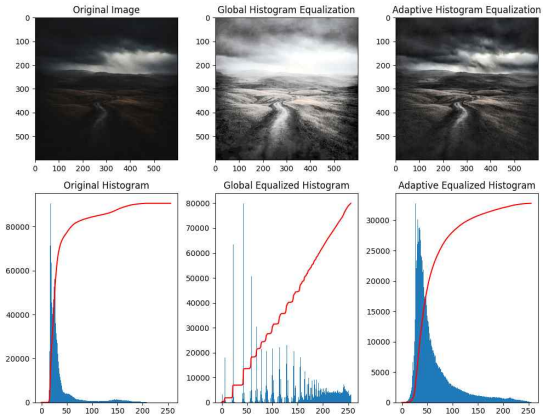
3.3 실험 결과

① 실험 2-1

< Histogram Equalization >



< Histogram Equalization with PDF-CDF>

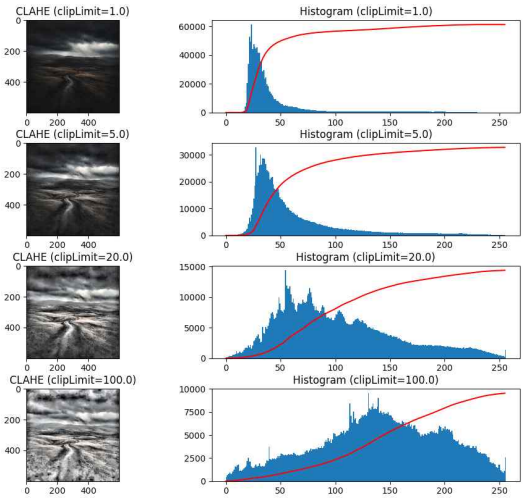


실험에 사용된 이미지는 대체로 어두운 그림이다. 따라서, 실험 1에서의 그림과 마찬가지로 이미지의 특정 영역은 밝고 특정 영역은 어두운 패턴이 나타나는데, Global Histogram Equalization의 경우는 Original Image에 존재했던 흰 구름에 대한 부분이 사라지고 해당 영역이 더 넓어져 버리는 등의 변형 이외에도 색이 밝아지면서 뒤에 존재하던 산의 형체가 흐릿해지고 바닥의 색이 변하는 이미지 왜곡 문제가 발생했다.

그러나 Adaptive Histogram Equalization의 경우는 밝은 부분에서도 어두운 부분에서도 큰 이미지 왜곡 없이 자연스럽게 이미지의 밝기가 증가한 것을 확인할 수 있다. Histogram의 개형 역시 Global HE에 비해 Original과 훨씬 비슷하지만 자연스럽게 좌우로 퍼진 모습을 보여준다.

② 실험 2-2

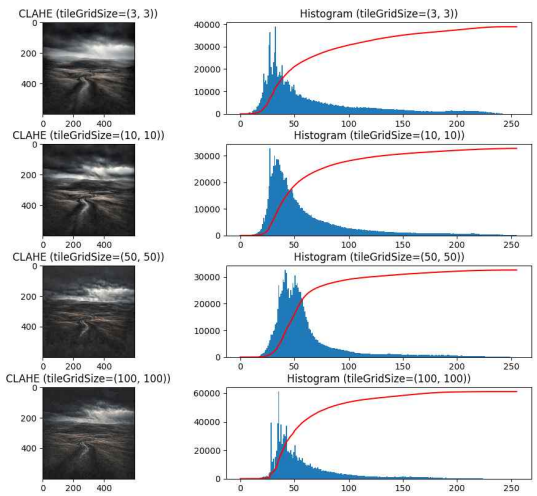
< clipLimit 값 변화에 따른 CLAHE 결과 >



Histogram의 각 Bin에서 허용할 수 있는 최대 대비 값을 의미하는 ClipLimit의 값을 변화시킨 결과를 통해 ClipLimit의 값이 커질수록 Noise에 대한 제어가 적절히 일어나지 않아 밝은 Pixel의 수가 늘어나면서 이미지의 왜곡은 더욱 심해지고 Histogram은 정규분포 형태에 가까워지는 모습을 관찰할 수 있었다.

③ 실험 2-3

< tileGridSize 값 변화에 따른 CLAHE 결과 >



이미지를 나눌 블록의 크기를 의미하는 tileGridSize의 값을 변화시킨 결과를 통해 tileGridSize의 크기가 너무 작은 경우는 약간의 Blur 효과가, 크기가 너무 큰 경우는 대비 변화가 크지 않은 것을 확인할 수 있었다.

IV. 결과 정리 및 고찰

두 실험 결과를 통해 이미지의 Pixel 값 분포는 고르지만 전체적으로 그 값들이 어두운 경우와 같이 특별한 Case가 아니라면, 일반적으로 Thresholding을 적용하거나 Histogram Equalization을 적용하는 경우 모두 Image Pixel 전체에 대한 Global한 Solution을 사용하는 것보다는 Local한 Solution을 사용하는 것이 만족스러운 결과를 얻을 가능성이 높다는 것은 실험을 통해 확인할 수 있었다. 다만, 추가적으로 진행한 실험들에서도 알 수 있듯이 Local한 Solution에서도 Kernel Size 등의 적절한 Hyperparameter가 제공되지 않는다면 얼마든지 Image를 왜곡시키는 현상이 나타날 수 있기에, Image Processing 분야에서도 No Free Lunch Theory는 적용된다는 것을 새삼 느낄 수 있었다. 또한, 이렇게 Hyperparameter에 민감한 Image Processing Method가 많은 만큼 이미지의 대비 정도, 노이즈 정도 등을 정량화하는 방법에 대한 후속 연구는 컴퓨터 비전 분야에 있어 꼭 필요한 연구라는 생각을 해 보았다.