

**과목명 | 디지털영상처리**

**실습 | 5**

**담당교수 | 정진우**

**학과 | 컴퓨터공학과**

**학년 | 3**

**학번 | 2010112406**

**제출일자 | 2016. 4. 21**

**이름 | 박도령**

**제출일 | 2016. 3. 31**

**실습에 사용한 이미지)**

****

**chiltern\_hill.jpg**

**실습 1)**

♋ **동작 순서 및 원리**

**1. Convert an Image to a gray scale image.(Input : any image)**

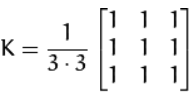
**Opencv에서 제공하는 함수 cvtColor를 이용하여 원래 이미지를 grayscale 이미지로 변환한다.**

**//Converts image into grayscale image.**

**cvtColor(img, img, CV\_BGR2GRAY);**

**2. Make a blurred image using the gray scale image.**

**Blurred image를 만들기 위해서는 다음과 같이 마스크를 씌운다. 주변 픽셀과 가운데 픽셀의 값을 모두 더한 뒤 9로 나눈 픽셀 값을 가운데 픽셀에 새로 적용하는 것이다.**



**이는 모든 원소가 1/9인 마스크와 같다. 따라서 마스크를 코드로 구현하면 다음과 같다.**

**//Mask used for blurring**

**Mat blurring\_mask = (Mat\_<float>(3, 3) << 0.11, 0.11, 0.11,**

**0.11, 0.11, 0.11,**

**0.11, 0.11, 0.11);**

**이렇게 구현한 마스크를 적용하기 위해서는 Opencv에서 제공하는 2Dfilter 함수를 사용한다. 코드로 구현하면 다음과 같다.**

**//Adjusts blurring mask made before to original image and gets blurred image.**

**filter2D(img, img\_blurred, img.depth(), blurring\_mask);**

****

**chiltern\_hill\_gray.jpg**

****

**Image\_blurred.jpg**

**3. Subtract the blurred image from original image.**

**두 이미지를 빼기 위해서 Opencv에 제공하는 함수 subtract를 사용한다 원래 이미지에서 blurring 처리 된 이미지를 뺀다.**

**//Gets result image of original image - blurred image.**

**subtract(img, img\_blurred, img\_subtracted);**

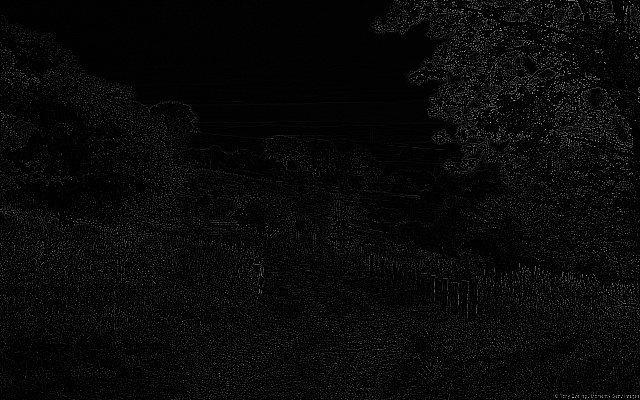
**4. Add the original image with the image from above step.**

**그 뒤에 Opencv에서 제공하는 함수 add를 사용하여 해당 이미지에 원본 이미지를 더한다.**

**//Gets result image of substracted image + original image.(It would be sharpen image.)**

**add(img, img\_subtracted, img\_readded);**

**3번 및 4번 과정의 결과 이미지는 다음과 같다.**

****

**Image\_subtracted.jpg**

****

**Image\_sharpened.jpg**

**원본 이미지와 비교해 보았을 때 4번 과정의 결과 이미지가 훨씬 더 선명해진 것을 확인할 수 있다. 실습1의 내용을 행렬 마스크들로 나타내면 과정을 조금 더 쉽게 이해할 수 있다.**

**- ( with ) + = Image Sharpened**

**Original Image - Original Mask adjusted by Blurring Mask + Original Image = Image Sharpened**

**원본 이미지에서 이미지를 Blurring 하게 되면 주변 Edge의 밝기가 증가하게 되어 물체의 경계가 흐릿해진다. 그러므로,원본 이미지에서 원본 이미지를 Blurring 한 결과를 빼게 되면 Edge 부분의 픽셀 값만 0 이상의 값을 가진 영상을 가지게 된다. 따라서 이 이미지를 원본 이미지에 더하면 Edge 부분이 더 밝아지는 효과를 줄 수 있기 때문에 밝은 부분과 어두운 부분의 경계를 조금 더 명확하게 볼 수 있게 된다. 이는 한마디로 이미지가 더 선명하게 보여지게 된다는 것이다.**

**실습 2)**

**앞서 Opencv로 구현한 실습 1을 행렬로 표시해 보았다. 행렬에는 교환법칙이 성립하므로 우선 행렬 마스크로 표현한 다음 식도 똑같은 결과를 가져올 것이라고 가정해본다.**

**+ - ( with ) = Image Sharpened**

**Original Image + Original Image - Original Mask adjusted by Blurring Mask = Image Sharpened**

**따라서 더 간단히 나타내면 다음과 같을 것이다.**

**- ( with ) = Image Sharpened**

**Original Image \* 2 - Original Mask adjusted by Blurring Mask = Image Sharpened**

♋ **동작 순서 및 원리**

**1. Convert an image a gray scale image.(Input : any image)**

**Grayscale Image를 얻는 방법은 실습1과 동일하다.**

**2. Make a sharpened image using the gray scale image.**

**위의 식을 구현해보기 보기 위해 2개의 마스크를 준비한다. 하나는 기존에 사용하는 Blurring Mask이며, 나머지는 이미지를 2배고 곱하기 위한 마스크이다. 두 마스크를 코드로 구현하면 다음과 같다.**

**//Mask used for blurring**

**Mat blurring\_mask = (Mat\_<float>(3, 3) << 0.11, 0.11, 0.11,**

**0.11, 0.11, 0.11,**

**0.11, 0.11, 0.11);**

**//Mask used for adding two original images.**

**Mat original\_2times\_mask = (Mat\_<float>(3, 3) << 0, 0, 0,**

**0, 2, 0,**

**0, 0, 0);**

**두 마스크를 filter2D함수를 사용하여 원본 이미지에 적용하여 각각의 두 결과 이미지를 얻는다. 코드로 구현하면 다음과 같다.**

**//Gets result image of original image + original image**

**filter2D(img, img\_2times, img.depth(), original\_2times\_mask);**

**//Gets image blurred**

**filter2D(img, img\_blurred, img.depth(), blurring\_mask);**

** **

**Image\_2times\_multiplied.jpg Image\_blurred.jpg**

**Opencv에서 제공하는 함수 subtract를 사용하여 Image\_2times\_multiplied에 Image\_blurred를 뺀다.**

**//Gets result image of Original image + Original Image - blurred image**

**subtract(img\_2times, img\_blurred, img\_sharpened);**

**뺀 결과 이미지는 다음과 같다.**

****

**Image\_expected\_to\_be\_sharpened.jpg**

**이론대로라면 실습1의 이미지와 동일한 이미지가 나와야 하지만 전혀 다른 이미지가 나왔다. 그 이유를 생각해보면 다음과 같다.**

**2배 마스크를 적용한 이미지는 모든 픽셀의 값이 Clamping 효과에 의해 2배가 되지 않는다. 예를 들어 원본 이미지의 230의 밝기를 가진 픽셀에 2배 마스크를 적용하였을 때 이론대로라면 460의 밝기를 가진 픽셀이 되어야 하지만 밝기 값은 0~255사이에서만 존재하므로 Clamping 효과에 의해 255의 밝기를 가지게 된다. 따라서 원래는 픽셀 값 460에서 Blurred Image의 픽셀 값을 빼야 하지만, Clamping 처리된 픽셀 값 255에서 Blurred Image의 픽셀 값을 빼게 된다. 그러므로 실습1의 이미지보다 훨씬 더 어두운 픽셀 값을 가지는 결과를 갖게 된다. 결론적으로 존재하여야 할 많은 값들이 Clamping 효과에 의해 사라져 버리기 때문에 원래 의도한 이미지와 다른 이미지를 갖게 되었다고 할 수 있다.**