

**과목명 | 디지털영상처리**

**실습 | 7**

**담당교수 | 정진우**

**학과 | 컴퓨터공학과**

**학년 | 3**

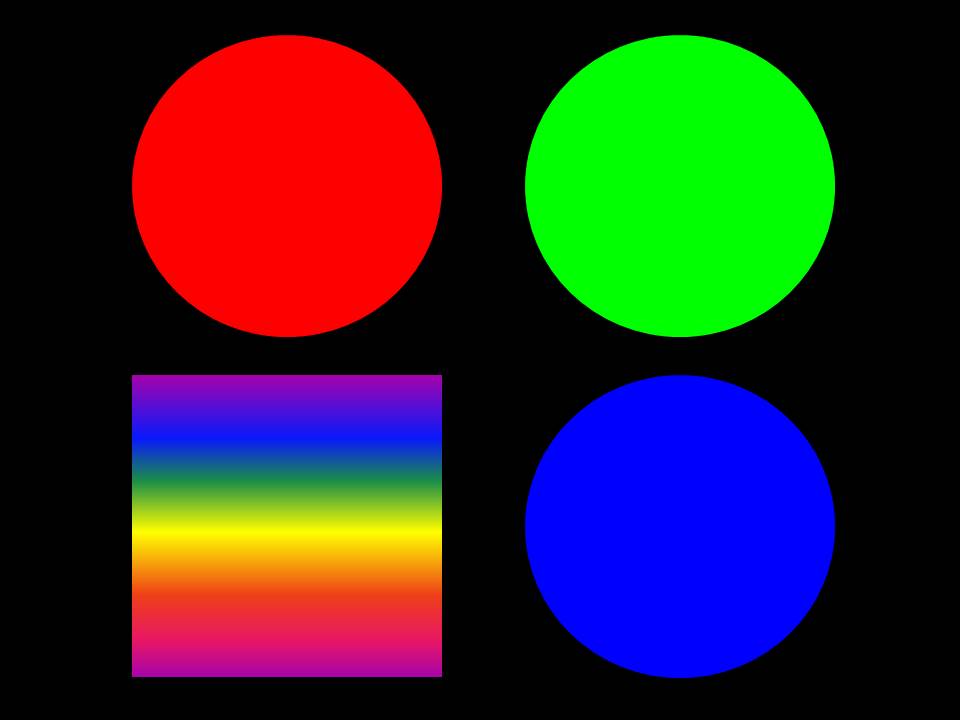
**학번 | 2010112406**

**제출일자 | 2016. 5. 12**

**이름 | 박도령**

**제출일 | 2016. 3. 31**

**실습1 에 사용한 이미지)**

** **

**Lenna.PNG stuff\_color.jpg**

**실습 1)**

♋ **동작 순서 및 원리**

**1. Convert an image to a gray scale image.(Input : any image, stuff\_color.jpg)**

**Opencv에서 지원하는 함수 cvtColor를 사용하여 이미지들을 그레이스케일 이미지로 각각 변환한다.**

**[img1 - Lenna.PNG, img2 - stuff\_color.jpg]**

//Converts 2 images into grayscale images

cvtColor(img1, img1, CV\_BGR2GRAY);

cvtColor(img2, img2, CV\_BGR2GRAY);

**2. Extract Sobel edge - E1 Image.**

**Opencv에서 지원하는 함수 Sobel을 사용하여 각각 x좌표의 Sobel 필터가 적용된 이미자와 y좌표의 Sobel 필터가 적용된 이미지를 구한다. 이렇게 구한 두 이미지를 더하면 Sobel Edge 이미지를 구할 수 있다. 코드는 다음과 같다.**

//-----------------------Image1 - E1-----------------------

Sobel(img1, sobelx, CV\_8U, 1, 0);//Gets Image filtered by sobel x

Sobel(img1, sobely, CV\_8U, 0, 1);//Gets Image filtered by sobel y

add(abs(sobelx), abs(sobely), img1\_E1);//Adds 2 images to get result

**3. Extract Laplacian edge - E2 Image**

**Opencv에서 지원하는 함수 Laplacian을 사용하여 Laplacian 필터를 적용한 뒤 함수 convertScaleAbs를 사용하여 unsigned int 매트릭스로 변환 시킨다. 이 과정을 거치면 Laplacian Edge 이미지를 얻을 수 있다. 코드는 다음과 같다.**

//-----------------------Image1 - E2-----------------------

Laplacian(img1, img1\_E2, CV\_16S, 3, 1, 0, BORDER\_DEFAULT);//Adjusts Laplacian edge filter to image1

convertScaleAbs(img1\_E2, img1\_E2);//Converts image into CV\_8U

**4. Extract Canny edge - E3 Image.**

**Opencv에서 지원하는 함수 Canny를 사용하여 한번에 Canny Edge 이미지를 얻을 수 있다. 코드는 다음과 같다.**

Canny(img1, img1\_E3, 50, 200, 3);//Adjusts Canny edge filter to image1

**5. Extract Laplacian of Gaussian - E4 Image.**

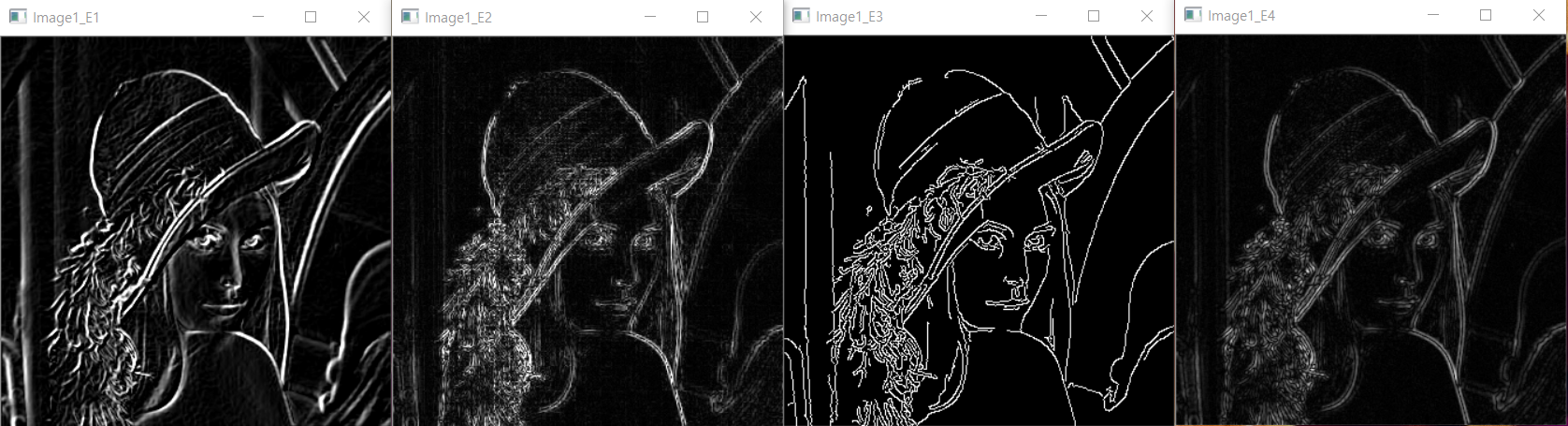
**우선 이미지를 Gaussian Blur 처리를 한 후에 해당 이미지를 대상으로 Laplacian 필터를 적용한다. 그 뒤에 마찬가지로 함수 convertScaleAbs를 사용하여 unsigned int 매트릭스로 변환시키면 Laplacian of Gaussian Edge 이미지를 얻을 수 있다.**

GaussianBlur(img1, gaussian\_blurred1, Size(3, 3), 0, 0, BORDER\_DEFAULT);//Adjusts Gaussian blur filter to image1

Laplacian(gaussian\_blurred1, img1\_E4, CV\_16S, 3, 1, 0, BORDER\_DEFAULT);//Adjusts Laplacian edge filter

convertScaleAbs(img1\_E4, img1\_E4);//Converts image into CV\_8U

**♋ 결과 이미지 (Lenna.PNG)**



**비교: 기본적으로 보았을 때 Canny Edge 이미지가 가장 노이즈가 없는 것을 확인할 수 있었다. 그리고 E2의 경우는 원래 이미지를 Laplacian Edge 이미지로 바꾼 경우라면 E4의 경우는 원래 이미지를 Gaussian Blur 처리 한 뒤 Laplacian Edge 이미지로 바꾼 것이다. 이미지를 비교해 보았을 때, E4의 경우가 E2의 경우보다 노이즈가 적다는 것을 확인할 수 있었다.**

**♋ 결과 이미지 (stuff\_color.jpg)**



**실습2 에 사용한 이미지)**

**- 원본 이미지**

**[Lenna.PNG, stuff\_color.jpg]**

**- 실습1의 결과 이미지들**

**[Image1\_E1.jpg, Image1\_E2.jpg, Image1\_E3.jpg, Image1\_E4.jpg]**

**[Image2\_E1.jpg, Image2\_E2.jpg, Image2\_E3.jpg, Image2\_E4.jpg]**

**실습 2)**

♋ **동작 순서 및 원리**

**사전 준비 : 실습1의 결과 이미지들을 로드와 동시에 한 벡터에 관리하기 위해서 반복문을 이용하여 한 번의 실행마다 하나의 파일의 이름을 만든다. 이름을 생성한 직후 해당 이름의 파일을 로드하고 벡터에 push한다. 모든 파일이 로드에 성공하면 실습1 에서 생성한 8개의 이미지 파일을 벡터에 저장할 수 있다. 이 실습에서는 해당 벡터를 edge라는 변수로 정의하였으며, 0~3 인덱스는 img1의 (E1~E4), 4~7 인덱스는 img2의 (E1~E4) 를 나타낸다.**

**1. Split the RGB image to R image, G image, B image.(Input : Same image as above)**

**Opencv 함수에서 지원하는 split 함수를 사용하여 분리된 Blue, Green, Red 채널을 저장하고 있는 벡터를 얻을 수 있다. 인덱스 0은 Blue channel, 1은 Green 채널, 그리고 2는 Red 채널이다. 이미지를 합성하기 전에 항상 이 동작이 이루어진다.**

split(img1, img1\_channels);//Splits image 1

**2. Add R image with E1 : R’ image.**

**add 함수를 사용하여 Sobel Edge 이미지인 E1과 분리한 Red channel 을 합친다.**

//Red channel + Sobel edge image1 = img\_merged

add(edges[0], img1\_channels[2], img\_merged);

**3. Composite R’ Image with G and B image : R’GB image.**

**Red Channel 을 합친 이미지로 대체한 뒤, merge 함수를 사용하여 해당 벡터를 하나의 이미지로 합친다. 결과 이미지는 벡터에 저장한다.**

img1\_channels.pop\_back();

img1\_channels.emplace\_back(img\_merged);//Replaces Red channel with img\_merged

merge(img1\_channels, results[0]);//Merges RGB channels and stores it

**4. Add G image with E2 : G’ image.**

**add 함수를 사용하여 Laplacian Edge 이미지인 E2와 분리한 Green channel 을 합친다.**

//Green channel + Laplacian edge image 1 = img\_merged

add(edges[1], img1\_channels[1], img\_merged);

**5. Composite G’ image with R and B image : RG’B image.**

**Green channel 을 합친 이미지로 대체한 뒤, merge 함수를 사용하여 해당 벡터를 하나의 이미지로 합친다. 결과 이미지는 벡터에 저장한다.**

img1\_channels[1] = img\_merged;//Replaces Green channel with img\_merged

merge(img1\_channels, results[1]);//Merges RGB channels and stores it

**6. Add B image with E3 : B’ image.**

**add 함수를 사용하여 Canny Edge 이미지인 E3와 분리한 Blue channel 을 합친다.**

split(img1, img1\_channels);//Splits image 1

**7. Composite B’ image with R and G image : RGB’ image**

**Blue channel 을 합친 이미지로 대체한 뒤, merge 함수를 사용하여 해당 벡터를 하나의 이미지로 합친다. 결과 이미지는 벡터에 저장한다.**

//Blue channel + Canny edge image 1 = img\_merged

add(edges[2], img1\_channels[0], img\_merged);

img1\_channels[0] = img\_merged;//Replaces Blue channel with img\_merged

**8. Add R image with E4 : R’’ image.**

**add 함수를 사용하여 Laplacian of Gaussian Edge 이미지와 분리한 Red channel을 합친 뒤 Red channel 을 합친 이미지로 대체한다.**

//Red channel + Laplacian of Gaussian edge image 1 = img\_merged

add(edges[3], img1\_channels[2], img\_merged);

img1\_channels[2] = img\_merged;//Replaces Red channel with img\_merged

**9. Add G image with E4 : G’’ image.**

**add 함수를 사용하여 Laplacian of Gaussian Edge 이미지와 분리한 Green channel을 합친 뒤 Green channel 을 합친 이미지로 대체한다.**

//Green channel + Laplacian of Gaussian edge image 2 = img\_merged

add(edges[3], img1\_channels[1], img\_merged);

img1\_channels[1] - img\_merged;//Replaces Green channel with img\_merged

**10. Composite B image with R’’ and G’’ image : R’’G’’B image.**

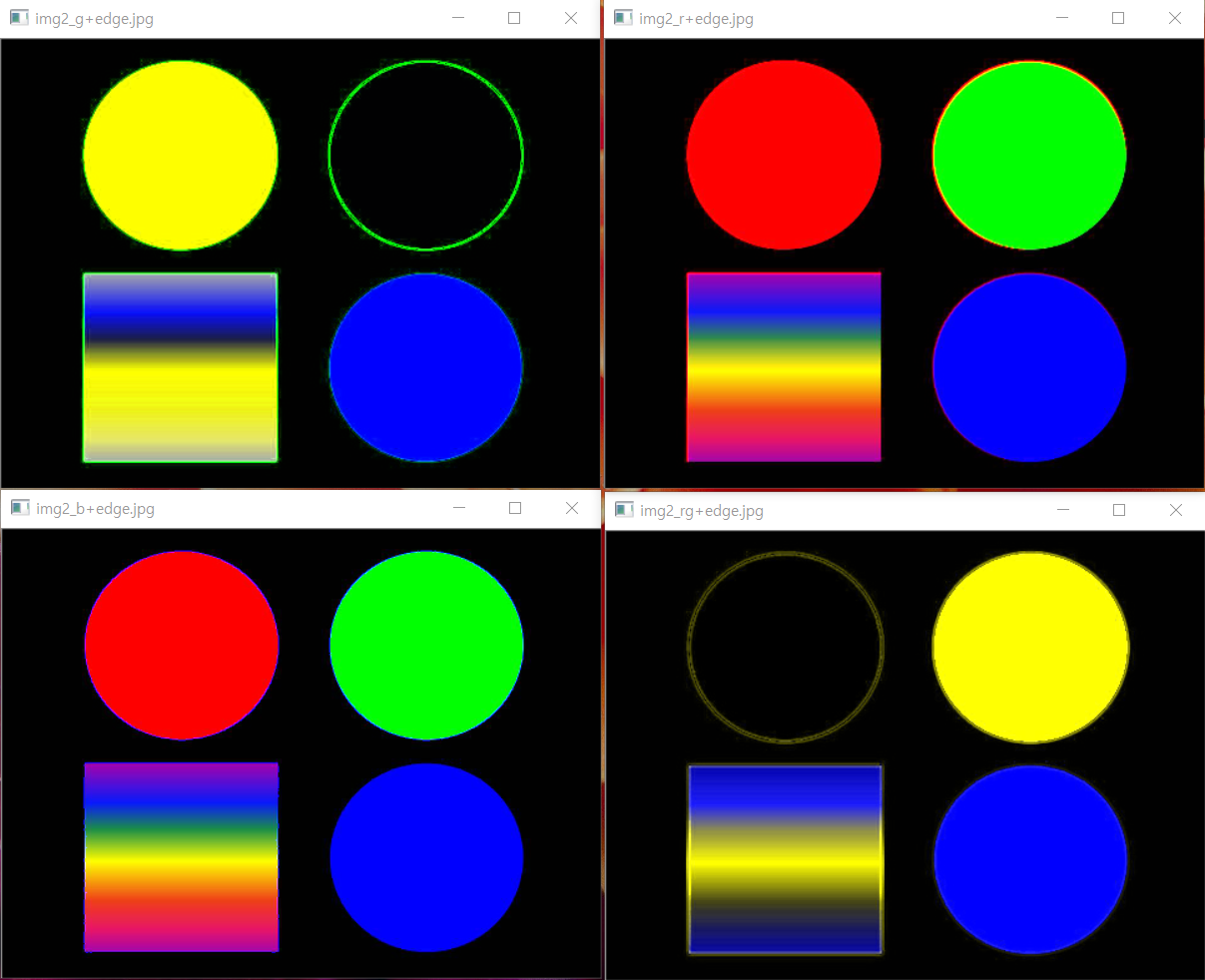
**해당 벡터를 하나의 이미지로 합친다.**

merge(img1\_channels, results[3]);//Merges RGB channels and stores it

**♋ 결과 이미지 (Lenna.PNG)**



**♋ 결과 이미지 (stuff\_color.jpg)**



**비교 : 기본적으로 Edge가 높은 픽셀 값을 갖고 있으므로 한 채널과 더할 경우 해당 채널의 Edge 부분이 높은 픽셀 값을 갖게 된다. 따라서 이를 나머지 채널들과 합성할 경우 해당 채널의 색깔을 가진 Edge가 합친 이미지에 드러나게 될 것이다. 따라서 R’GB 이미지의 경우 빨강 Edge, RG’B의 경우 초록 Edge, 그리고 RGB’의 경우 파란 Edge를 가지게 될 것이다. 결과 이미지를 보면 각 색깔의 Edge가 도드라지게 나타난 것을 확인할 수 있다. 마지막의 경우 초록과 빨강이 합쳐지는 경우 노란색이 되므로 노란색 edge가 드러나게 될 것이다. 4번째 이미지에서 확인할 수 있다.**