

## ■ Homework #1: 2021학년 1학기

교과목명	디지털영상처리II	수업주차	4주차
이름	황진주	학번	20193148
<div> <div>■ 주제</div> <p>자신이 가진 사진을 하나 선택하여, 수업 시간에 배운 다양한 에지 검출기를 활용하여 에지를 검출한다. 자신이 원하는 에지 검출 결과를 얻기 위해 조절한 파라미터나 함수의 적용 내용을 분석하여 작성한다.</p> </div>			
<div> <div>결과 1) 에지 검출 파이썬 코드</div> <div> <div> <pre> from skimage import filters, feature, img_as_float, exposure, img_as_ubyte from skimage.io import imread from skimage.color import rgb2gray # 이미지 그리기 import matplotlib.pyplot as pylab # 컨볼루션 from scipy import signal, ndimage import numpy as np import cv2  # 히스토그램 평활화 def equalize_func(im):     return exposure.equalize_hist(im) # 콘트라스트 스트레칭 def contrast_str_func(im):     from_, to_ = np.percentile(im, (2, 98))     return exposure.rescale_intensity(im, in_range=(from_, to_))  ### 소벨이미지와 라플라시안 이미지  im = rgb2gray(imread("../images/3.jpg")) im = contrast_str_func(im)  # 소벨 커널 생성 ker_x = [ [-1, 0, 1], [-2, 0, 2], [-1, 0, 1]] ker_y = [ [-1, -2, -1], [0, 0, 0], [1, 2, 1]]  im_x = signal.convolve2d(im, ker_x, mode="same") im_y = signal.convolve2d(im, ker_y, mode="same")  # magnitude im_mag = np.sqrt( (im_x)**2 + (im_y)**2 ) # direction im_dir = np.clip(np.arctan( im_y / im_x ), 0, 1)  # 라플라시안 커널 : 등방성 ker_lp = [[0, -2, 0], [-1, -8, -1], [0, -2, 0]]  im_lp = signal.convolve2d(im, ker_lp, mode="same")  pylab.figure(figsize=(30, 30)) pylab.subplot(2,2,1), pylab.imshow(im, cmap='gray') </pre> </div> <div> <pre> im = rgb2gray(imread("../images/3.jpg")) im = equalize_func(im)  # LoG # 숫자조정해서 가우시안 값을 바꿀 수 있다. im_g = filters.gaussian(im, 2) im_l = filters.laplace(im_g)  # T값도 내 맘대로 조정 가능하다 edge = zero_cross(im_l, T=np.max(im_l)*0.001)  # DoG im_dog = filters.difference_of_gaussians(im, 2.0) edge2 = zero_cross(im_dog, T=np.max(im_dog)*0.001)  # fft height, width = im.shape dft = cv2.dft(np.float32(edge2), flags=cv2.DFT_COMPLEX_OUTPUT) dft_shift = np.fft.fftshift(dft) im_fft = 20*np.log(cv2.magnitude(dft_shift[:, :, 0], dft_shift[:, :, 1]))  from skimage.morphology import disk  blur = filters.median(edge, disk(1))  pylab.figure(figsize=(30, 30)) pylab.subplot(2,2,1), pylab.imshow(im, cmap='gray') pylab.subplot(2,2,2), pylab.imshow(im_l, cmap='gray') pylab.subplot(2,2,3), pylab.imshow(edge, cmap='gray') pylab.subplot(2,2,4), pylab.imshow(edge2, cmap='gray') # pylab.subplot(2,2,4), pylab.imshow(im_fft, cmap='gray') # pylab.subplot(2,2,4), pylab.imshow(blur, cmap='gray')  ### Edge Detector  im = rgb2gray(imread("../images/3.jpg")) im = equalize_func(im)  im = filters.median(im) edges = filters.sobel(im) # pylab.figure(figsize=(30,20)) </pre> </div> </div> </div>			

```

pylab.subplot(2,2,2), pylab.imshow(im_mag, cmap='gray')
pylab.subplot(2,2,3), pylab.imshow(im_dir, cmap='gray')
# pylab.subplot(2,2,4), pylab.imshow(im_fft, cmap='gray')
pylab.subplot(2,2,4), pylab.imshow(im_lp, cmap='gray')

```

### LoG와 DoG로 에지 검출하기

```

# 가우시안 필터 가져오기
from skimage import filters

```

```

def zero_cross(image, T=0):
    zimg = np.zeros(image.shape)
    # 커널이 넘치지 않게 하려고 -1만큼 까지만 돌게됨
    for i in range(0, image.shape[0]-1):
        for j in range(0, image.shape[1]-1):
            # if문 끝에 ₩는 라인 엔터치려고 넣은거임

            # 오른쪽
            if image[i][j]*image[i+1][j] < 0 ₩
            and np.abs( image[i+1][j] - image[i][j]) > T :
                zimg[i,j] = 1

            # 위상단
            elif image[i][j]*image[i+1][j+1] < 0 ₩
            and np.abs( image[i+1][j+1] - image[i][j]) > T :
                zimg[i,j] = 1

            # 위
            elif image[i][j]*image[i][j+1] < 0 ₩
            and np.abs( image[i][j+1] - image[i][j]) > T :
                zimg[i,j] = 1

    return zimg

```

```

pylab.subplot(121),          pylab.imshow(im,          cmap='gray'),
pylab.title('original')
pylab.subplot(122),          pylab.imshow(edges,          cmap='gray'),
pylab.title('original')

```

### Canny

```

im = rgb2gray(imread('../images/3.jpg'))
im = contrast_str_func(im)

```

```

im = ndimage.gaussian_filter(im, 2)
edges = feature.canny(im, sigma=2)

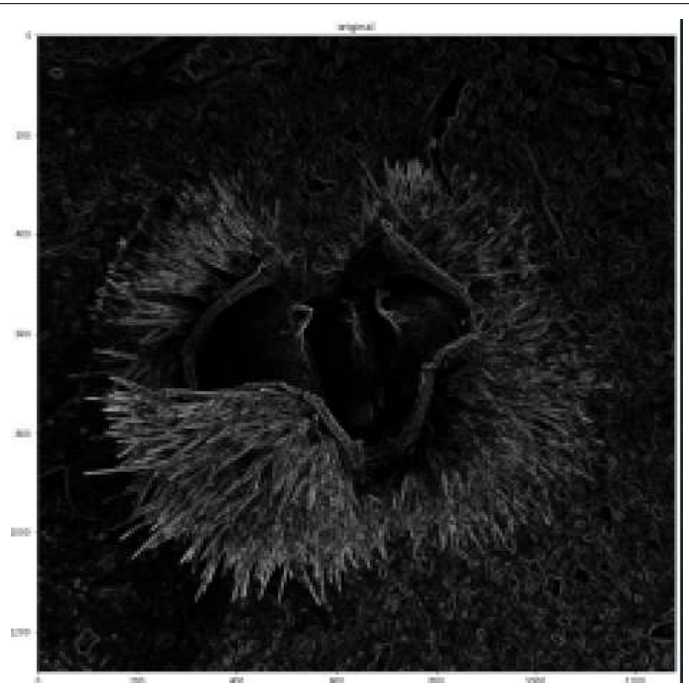
```

```

pylab.figure(figsize=(30,20))
pylab.subplot(121),          pylab.imshow(im,          cmap='gray'),
pylab.title('original')
pylab.subplot(122),          pylab.imshow(edges,          cmap='gray'),
pylab.title('original')

```

## 결과 2) 검출에 활용된 원본 사진과 결과



## 결과 3) 원하는 결과를 얻기 위한 방법 서술


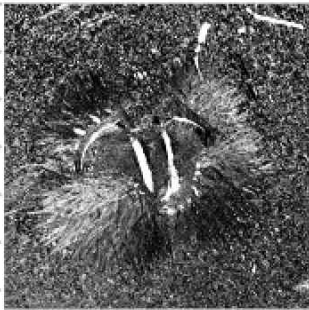
사람이 직접하기 어려운 주제로 에지를 검출해내고 싶었다. 그래서 지난 추석에 딴 밤송이의 에지를 검출해보고자 여러 기법을 도입하였다.

에지 검출 전 이미지 대비에 대한 조정을 하여 더욱 원활한 검출이 될 수 있도록 하였다.

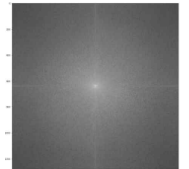
전처리 : 히스토그램 평활화 or 콘트라스트 스트레칭

## >> 소벨

- 전처리 : 콘트라스트 스트레칭
- 커널 x :  $\text{ker\_x} = \begin{bmatrix} -1, & 0, & 1 \\ -2, & 0, & 2 \\ -1, & 0, & 1 \end{bmatrix}$
- 커널 y :  $\text{ker\_y} = \begin{bmatrix} -1, & -2, & -1 \\ 0, & 0, & 0 \\ 1, & 2, & 1 \end{bmatrix}$

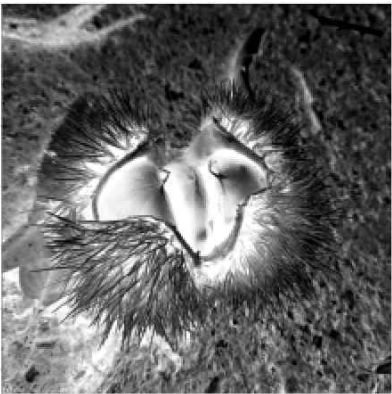
magnitude	direction
	

- 분석 : magnitude
  - > 제곱값 변경 시 색상대비가 낮아지기만 하여 2가 최적이었다.
  - > 밤송이에 대해서만 에지가 나타나 원하는 결과에 부합하였다.
  - > 다만, 방송이 껍데기는 잘 표현이 되었지만 밤 자체는 잘 드러나지 않아 아쉬웠다.
- 분석 : direction
  - > 밤의 껍질 밤송이에 대해서 뚜렷하게 표현되지만 주변 노이즈가 심하였다.
  - > 노이즈 제거를 위해 가우시안 블러를 적용했지만 형태가 오히려 무너지게 되었다.
  - > 혹여나 주파수 변환 영상을 보면 방법을 찾을 수 있을 수 있으나 싶어 확인해보았지만 뚜렷한 특징은 보이지 않았다.



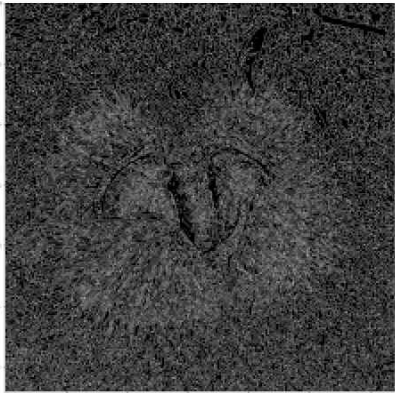
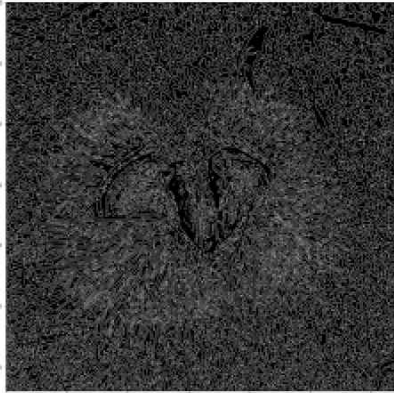
## >> 라플라시안

- 전처리 : 콘트라스트 스트레칭
- 커널 x :  $\text{ker\_x} = \begin{bmatrix} 0, & -2, & 0 \\ -1, & -8, & -1 \\ 0, & -2, & 0 \end{bmatrix}$

라플라시안	분석
	<ul style="list-style-type: none"> <li>&gt; 커널 값을 변경하면 이미지의 색상 대비가 커지거나, 전체적인 밝기만 변경될 뿐이었다.</li> <li>&gt; 밤송이만의 엣지가 따지는 것이 아니라 아쉬웠다.</li> <li>&gt; 엣지를 따기보다는 이미지 대비를 주는 듯 하였다.</li> </ul>

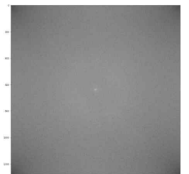
## >> LoG, DoG

- 전처리 : 히스토그램 평활화, 가우시안블러(2), 라플라시안
- edge :  $T = \text{np.max(im\_l)} * 0.001$

LoG	DoG
	

### - 분석 : LoG

- > 블러값을 낮추었을 때 밤 자체의 깊이가 명확히 보였다.
- > edge값이 0인 경우에 비해 밤송이와 배경의 대비가 더 뚜렷하였다.




### - 분석 : DoG

- > 값 변경은 LoG와 동일한 속성을 보였다.
- > 노이즈 때문에 fft를 적용하였는데, 전체적으로 밝은 분위기를 띄었다.
- > 평균 필터를 사용하고자 하였지만, 소금후추 노이즈에 효과적인 필터이므로 가우시안 필터에 적합한 필터링에 대한 학습이 필요하다.


## >> Edge Detector

- 전처리 : 히스토그램 평활화, 평균 필터, 소벨

Edge Detector	분석
	<ul style="list-style-type: none"> <li>&gt; 현재까지 제일 깔끔하게 에지 검출이 되었다.</li> <li>&gt; 기본 제공 필터들만 이용했기 때문에 값조정 과정이 없다.</li> </ul>

## >> 캐니

- 전처리 : 콘트라스트 스트레칭, 가우시안 필터(2), 캐니(시그마=2)

캐니	분석
	<p>&gt; 밤송이 특성상 세밀하게 표현되어야 하므로 높은 시그마 값을 가지게 되면, 밤송이가 표현되지 않았다.</p> <p>&gt; 기본 제공 필터들만 이용했기 때문에 값조정 과정이 없다.</p>