컴퓨터그래픽스 5주차 과제 보고서
201300995 이상건 물리학과

구현 내용
가우시안 필터, DoG, sobel 이용해서 엣지 검출하기다.

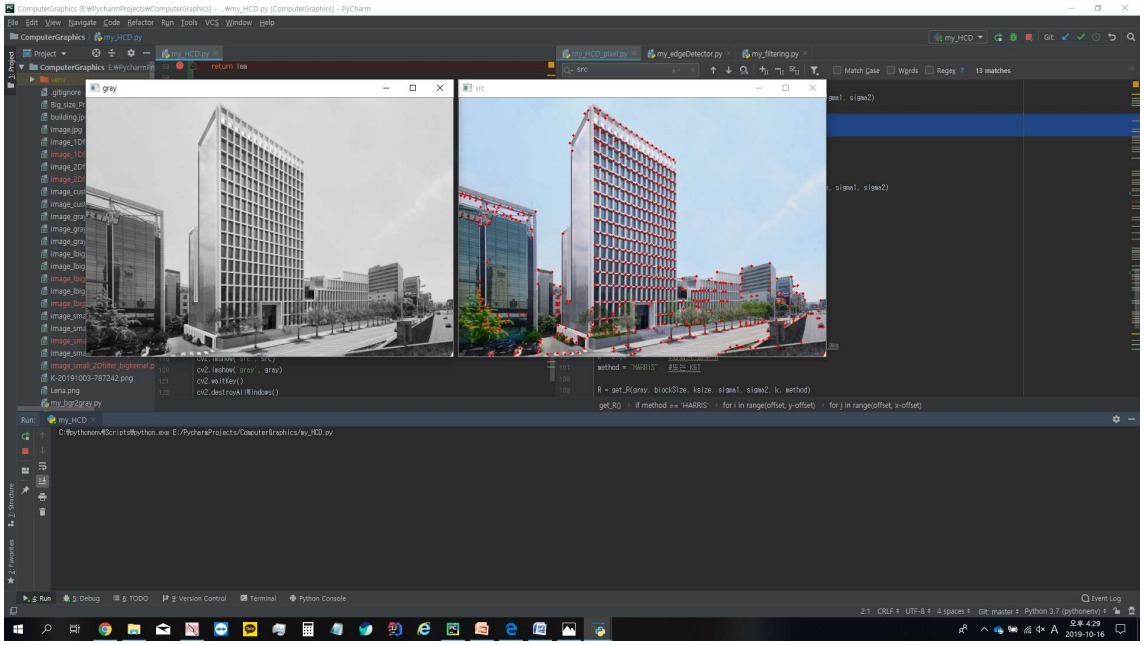
이유(구현 방법)
우선 기본으로 주어진 my_HCD_pixel.py 코드를 많이 참고하였다(사실 for문 빼고 다 배껴씀).
코드를 보자.
우선 my_HCD 함수 부분을 보자.

```
    lam = np.zeros((y,x,2))
    R = np.zeros((y,x))
    # harris 방법
    if method == "HARRIS":
        for i in range(offset, y-offset):
            for j in range(offset, x-offset):
                #Harris 방법으로 R를 계산하세요.
                lam[i,j] = get_eig_custom(src,(i,j),blockSize,ksize,sigma1,sigma2)
                det = lam[i,j,0] * lam[i,j,1]
                tr = lam[i,j,0] + lam[i,j,1]
                R[i,j] = det - k * (tr ** 2)

    # # Kanade & Tomasi 방법
    elif method == "K&T":
        for i in range(offset, y-offset):
            for j in range(offset, x-offset):
                lam[i, j] = get_eig_custom(src, (i, j), blockSize, ksize, sigma1, sigma2)
                R[i, j] = np.min(lam[i,j])
    #Kanade & Tomasi 방법으로 R를 계산하세요.

    return R
```

pixel 코드와 거의 비슷하다. 엣지를 얻는 함수부분 이름만 달라졌다.
lam는 각 좌표에 대한 eigen value가 들어있다. R함수는 이 eigen value의 결과값을 계산한 결과다. 계산하는 방법은 harris와 k&t 방식에 따라 달라진다.
그러면 핵심은 내가 만든 get_eig_custom 함수에 있다. 이 함수를 보자.

```python
def get_eig_custom(src, target, blockSize, ksize, sigma1, sigma2):
    y,x = target[0], target[1]
    offset = blockSize // 2


    roi = src[y-offset:y+offset+1, x-offset:x+offset+1]

    #DoG
    #gradX = my_DoG(roi, ksize, sigma1, gx=1, boundary = 2)
    #gradY = my_DoG(roi, ksize, sigma1, gx=0, boundary = 2)

    #Sobel
    gradX = cv2.Sobel(roi, cv2.CV_32F, dx = 1, dy = 0, ksize = ksize)
    gradY = cv2.Sobel(roi, cv2.CV_32F, dx = 0, dy = 1, ksize = ksize)

    gaus = cv2.GaussianBlur(roi,(blockSize,blockSize),sigma2)

    M = np.zeros((2,2))

    M[0,0] = np.sum(np.multiply(np.multiply(gradX,gradX),gaus))
    M[0,1] = np.sum(np.multiply(np.multiply(gradX,gradY),gaus))
    M[1,0] = M[0,1]
    M[1,1] = np.sum(np.multiply(np.multiply(gradY,gradY),gaus))

    lam = np.linalg.eigvals(M)

    return lam
```

함수를 보면 pixel.py 파일에 있던 이중 for문이 사라지고 matrix의 곱이 나와있다. M을 담는 행렬의 0,0은 gradX를 2번 원소곱에 가우스 곱 한것의 합, 0,1과 1,0은 gradX,gradY에 대하여, 1,1은 graY를 2번 곱한것에 대하여 나타난 것에 착안하여 코드를 작성했다. 뭘 이해했는지 쓰겠다.
우선 맨 위 y,x랑 target으로 되어있는 곳. 이 곳은 어느 좌표에 대해 가우스필터 쓰고 엣지 검출하고.. 할 지를 나타낸다. offset은 행렬의 특성상 0,0 이 중간값이면 안되므로 적용한거다. gradX와 gradY는 각각 X축의 그래디언트, Y축의 그래디언트를 나타낸다. 즉 이 부분의 변화가 얼마나 있는지를 나타난다. 해당 roi 값에 미분을 한번 한 거다. gaus는 내장 함수로 구했다.
최종적으로는 람다를 반환하는데, my_HCD 함수에서 이 람다1과 람다2 값으로 det과 tr을 계산하기 때문에 반환하는 것이다.



결과 또한 잘 나온다.

느낀 점
내가 공부를 안하는 것 같다.

과제 난이도
어렵다.