컴퓨터그래픽스 6주차 과제 보고서
201300995 이상건 물리학과

구현 내용
가우시안 피라미드와 Different of Gaussian을 통해 keypoints를 추출하는 내용이다.

이유(구현 방법)

시작을 어떻게 해야 할지 너무 막막해서 주석에 써있는 것부터 차례대로 작성하기로 했었다.

```
print('make gaussian pyr')
# Gaussian을 계산
# ksize = 2 * int(4 * sigma + 0.5) + 1
for j in range(6):
    #Gaussian Pyramids를 만들어주세요.
    #예제에서는 한 Level(Octave)에 6개의 Gaussian Image가 저장됩니다.
    ksize = 2 * int(4 * lv1sigma[i] + 0.5) + 1
    lv1py[:,:,i] = cv2.GaussianBlur(doubled, (ksize, ksize), lv1sigma[i])

    ksize = 2 * int(4 * lv2sigma[i] + 0.5) + 1
    lv2py[:,:,i] = cv2.GaussianBlur(normal,(ksize,ksize),lv2sigma[i])

    ksize = 2 * int(4 * lv3sigma[i] + 0.5) + 1
    lv3py[:,:,i] = cv2.GaussianBlur(half,(ksize,ksize),lv3sigma[i])

    ksize = 2 * int(4 * lv4sigma[i] + 0.5) + 1
    lv4py[:,:,i] = cv2.GaussianBlur(quarter,(ksize,ksize),lv4sigma[i])
```

가우스 피라미드 만드는 부분이다. lv1py엔 두배크기 이미지의 가우시안 피라미드가 배열 순서대로 있고, lv2py엔 원본크기, lv3py엔 0.5배, lv4py엔 0.25배 크기의 이미지에 대한 가우스 피라미드가 있다. 각각의 배열 수에 대해 정의한 시그마값을 넣어서 가우시안블러 처리를 한다.

```
# DoG를 계산
for j in range(5):
    #Difference of Gaussian Image pyramids 를 구해주세요.
    DoGlv1[:,:,i] = cv2.subtract(lv1py[:,:,i],lv1py[:,:,i+1])
    DoGlv2[:,:,i] = cv2.subtract(lv2py[:,:,i],lv2py[:,:,i+1])
    DoGlv3[:,:,i] = cv2.subtract(lv3py[:,:,i],lv3py[:,:,i+1])
    DoGlv4[:,:,i] = cv2.subtract(lv4py[:,:,i],lv4py[:,:,i+1])
```

different of gaussian 부분이다. 아까 lv1py ...에서 저장했던 가우시안블러 처리한 값들을 자기 index와 그 앞 index간의 차를 빼서 저장한다.

```
    # Extrema의 위치 계산
    print('find extrema')
    extPy1 = get_extrema(DoGIv1, extPy1)
    extPy2 = get_extrema(DoGIv2, extPy2)
    extPy3 = get_extrema(DoGIv3, extPy3)
    extPy4 = get_extrema(DoGIv4, extPy4)
```

```
for i in range(1, 4):
    for j in range(1, DoG.shape[0]-1):
        for k in range(1, DoG.shape[1]-1):
            # 최대값 혹은 최소값인 지점을 extrema로 구해주세요.
            DoG1localMax = np.max(DoG[j-1:j+2,k-1:k+2,i-1])
            DoG1localMin = np.min(DoG[j-1:j+2,k-1:k+2,i-1])
            DoG2localMax = np.max(DoG[j-1:j+2,k-1:k+2,i])
            DoG2localMin = np.min(DoG[j-1:j+2,k-1:k+2,i])
            DoG3localMax = np.max(DoG[j-1:j+2,k-1:k+2,i+1])
            DoG3localMin = np.min(DoG[j-1:j+2,k-1:k+2,i+1])
            # print("i: {}, j: {}, k: {} ".format(i,j,k))
            # print(DoG1localMax)
            allLocalMax = max(DoG1localMax,DoG2localMax,DoG3localMax)
            allLocalMin = min(DoG1localMin,DoG2localMin,DoG3localMin)
            # print(allLocalMax)
            # print(allLocalMin)
            if ((allLocalMax == DoG[j][k][i]) or (allLocalMin == DoG[j][k][i])):
```

다음은 특징값 추출 장면이다. 우선 threshold 하기 전 코드를 보면 해당 index의 앞index, 뒤 index 모두 합쳐 최소 또는 최대 값이 해당 index의 중간이 되야 후보가 될 수 있다는 조건을 작성한 장면이다. 각각의 최대, 최소를 모두 비교해 전체에서 최대, 최소를 구하고 그게 해당 index의 중앙값인지 아닌지 확인하는 방법으로 했다.

```
H = [[d2Ddx2,d2Ddxy,d2Ddxs],[d2Ddxy,d2Ddy2,d2Ddxy],[d2Ddxs,d2Ddys,d2Dds2]]
dD = np.transpose([dDdx,dDdy,dDds])

xhat = np.linalg.lstsq(np.dot(-1,H), dD, rcond=-1)[0]
target = DoG[j,k,i]
Dxhat = target + 0.5 * np.dot(dD.transpose(), xhat)

# Thresholding를 수행해주세요. ( 적절한 위치만 ext 배열에 저장해주세요, )
if(np.abs(Dxhat) < thresh or np.min(np.abs(xhat)) > 0.5):
    continue

Hpart = np.array([[d2Ddx2,d2Ddxy],[d2Ddxy,d2Ddy2]])
traceHpartsquare = np.trace(Hpart) ** 2
detHpart = np.linalg.det(Hpart)
rc = ((r + 1) ** 2)/r
if (detHpart<0 or (traceHpartsquare/detHpart) > rc):
    continue
ext[j,k,i-1] = 1
```

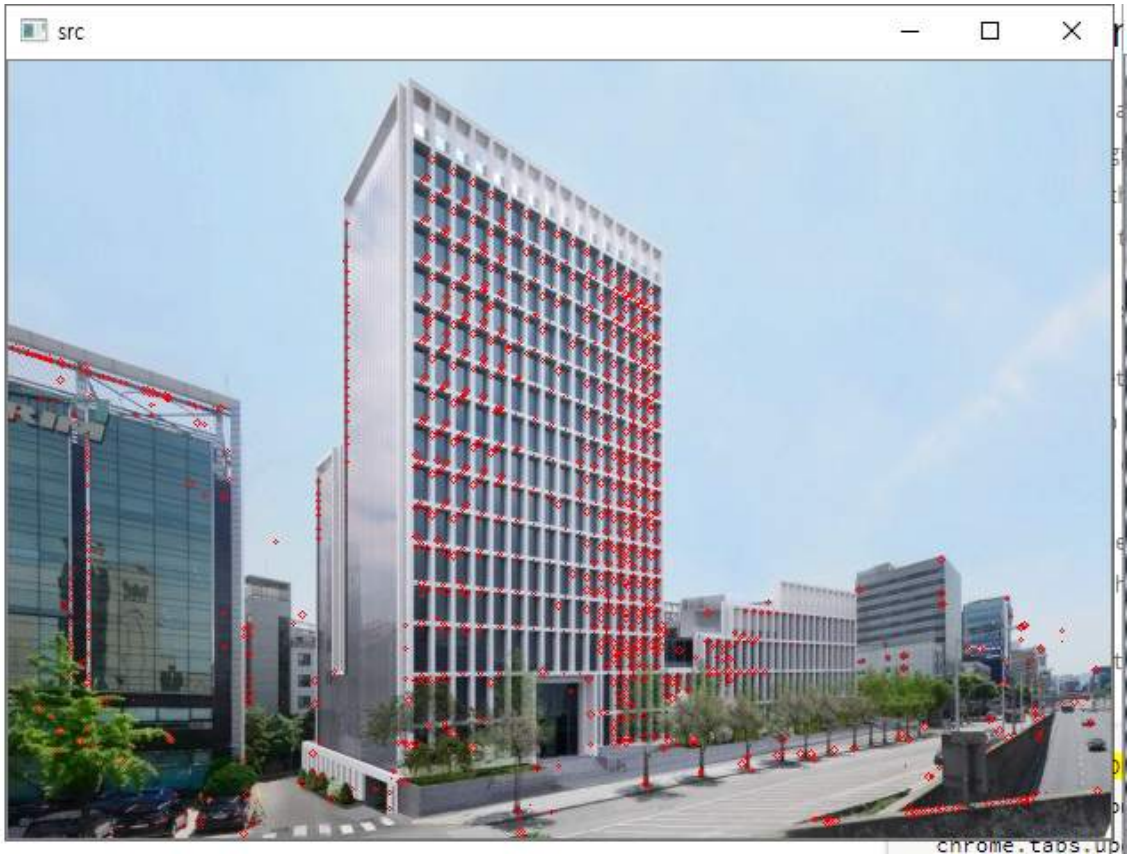
위에서 추출한 후보를 feature에 넣을지 안넣을지 걸러내는 코드다.
H, dD, xhat, Dxhat을 정의에 따라 수행한다. threshold 방식은 if문에 continue를 주어 걸러내도록 진행했다. 모든 조건을 충족했으면, 해당 좌표값을 1로 두어 저장한다.

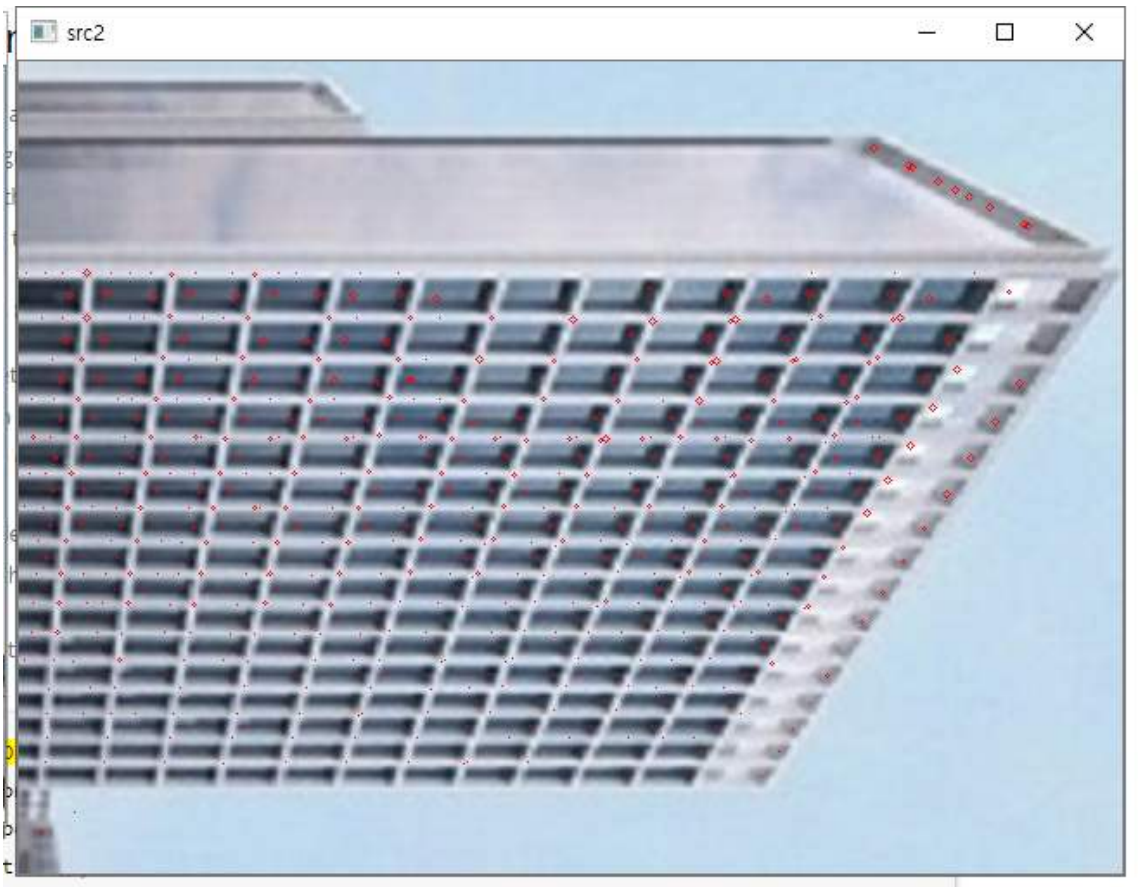
```
extr_sum = extPy1.sum() + extPy2.sum() + extPy3.sum() + extPy4.sum()
extr_sum = extr_sum.astype(np.int)
keypoints = np.zeros((extr_sum, 3))  # 원래는 3가지의 정보가 들어가나 과제에선 Y좌표, X 좌표, scale 세 가지의 값만 저장한다.
```

```
#값 저장
count = 0 #keypoints 수를 Count

for i in range(3):
    for j in range(doubled.shape[0]):
        for k in range(doubled.shape[1]):
            #Lv1
            #Keypoints 배열에 Keypoint의 정보를 저장하세요. 함수로 만들어서 수행하셔도 됩니다.
            if (extPy1[j,k,i] == 1):
                keypoints[count,0] = j * 0.5
                keypoints[count,1] = k * 0.5
                keypoints[count,2] = i
                count += 1
for i in range(3):
    for j in range(normal.shape[0]):
        for k in range(normal.shape[1]):
            #Lv2
            #Keypoints 배열에 Keypoint의 정보를 저장하세요.
            if (extPy2[j,k,i] == 1):
                keypoints[count,0] = j
                keypoints[count,1] = k
                keypoints[count,2] = i
                count += 1
for i in range(3):
    for j in range(half.shape[0]):
        for k in range(half.shape[1]):
            #Lv3
            #Keypoints 배열에 Keypoint의 정보를 저장하세요.
            if (extPy3[j,k,i] == 1):
                keypoints[count,0] = j * 2
                keypoints[count,1] = k * 2
                keypoints[count,2] = i
                count += 1
for i in range(3):
    for j in range(quarter.shape[0]):
        for k in range(quarter.shape[1]):
            #Lv4
            #Keypoints 배열에 Keypoint의 정보를 저장하세요.
            if (extPy4[j,k,i] == 1):
                keypoints[count,0] = j * 4
                keypoints[count,1] = k * 4
                keypoints[count,2] = i
                count += 1
```

위에서 저장한 좌표값을 keypoints 배열에 다시 저장한다. keypoints에는 위에서 추출한 좌표값 개수 만큼의 크기를 만들고 저장한다. 확대, 축소 비율을 고려하여 좌표값을 조정한다.


src


src2

결과 화면이다. 그럭저럭 잘 된 것 같다.

느낀 점
어렵다. 시간을 많이 잡아먹음.
실습이 이론 진도를 따라잡아 이해가 힘들었다. 근데 이건 시험기간의 휴강으로 해결될..까?

과제 난이도
어렵다.