HW 04 – REPORT

소속 : 정보컴퓨터공학부

학번 : 2 0 2 0 5 5 5 3 6

이름 : 민 예 진

1. 서론

1) 실습 목표

파노라마 이미지를 위해 SIFT로 이미지의 feature를 추출하고 RANSAC를 통해 적절한 match들을 추출한다. 서로 다른 이미지를 합치기 위해 Homography matrix를 구한 후 직접 파노라마 이미지를 완성하는 것을 목표로 한다.

2) 이론적 배경 기술

① SIFT (Scale-Invariant Keypoints)

intensity에 영향을 받지 않는 descriptor를 구하는 과정이다.

i. feature 주위에 16 x 16 크기의 window를 구한다.

ii. 각 픽셀의 edge orientation을 계산한다.

iii. gradient magnitude의 threshold를 통해 weak edge를 제거한다.

iv. edge orientation당 edge gradient의 값을 히스토그램으로 나타낸다.

② RANSAC (RANdom SAmple Consensus)

데이터셋에서 노이즈를 제거하고 모델을 예측하는 알고리즘이다.

i. sample (가설을 만들 수 있는 최소한의 점의 개수) s개를 랜덤하게 고른다

ii. sample을 통해 모델을 fit한다.

iii. inlier의 개수를 센다. (inlier threshold)

iv. N번 반복한다. (number of rounds)

v. inlier가 가장 많은 모델을 고른다.

③ Panorama

사진을 여러 장 나누어 촬영한 뒤 이를 합쳐 360도 방향의 모든 경치를 한 장에 담은 사진

RANSAC을 통해 사진 간의 homography matrix를 구한 후 첫 번째 사진에 두 번째 사진을 overlap하는 방식을 통해 파노라마 사진을 만들 수 있다.

2. 본론

1) SIFT Keypoint Matching

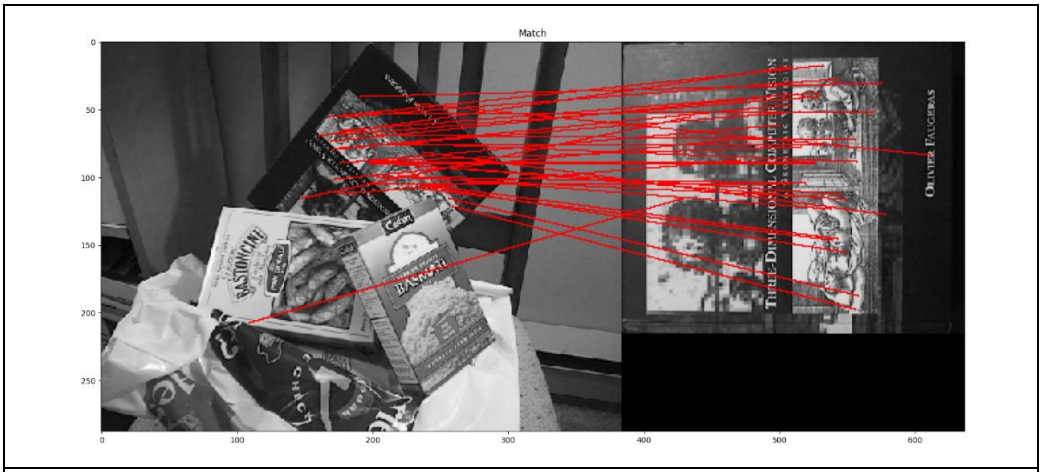
① FindBestMatches

```
matched_pairs = []
for i in range(y1):
    for j in range(y2):
        temp[j] = math.acos(np.dot(descriptors1[i], descriptors2[j]))
    compare = sorted(range(len(temp)), key = lambda k : temp[k])
    if (temp[compare[0]] / temp[compare[1]]) < threshold:
        matched_pairs.append([i, compare[0]])
```

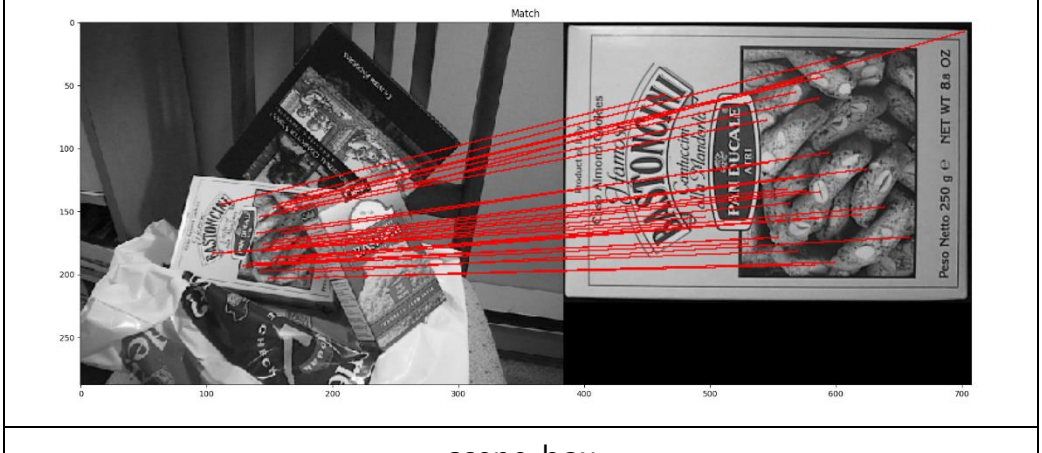
FindBestMatches 함수를 이용하여 image1과 image2의 descriptor를 매칭한다.

descriptor가 이루는 각도를 이용하여 적절한 match를 찾는다.

실행결과



scene-book



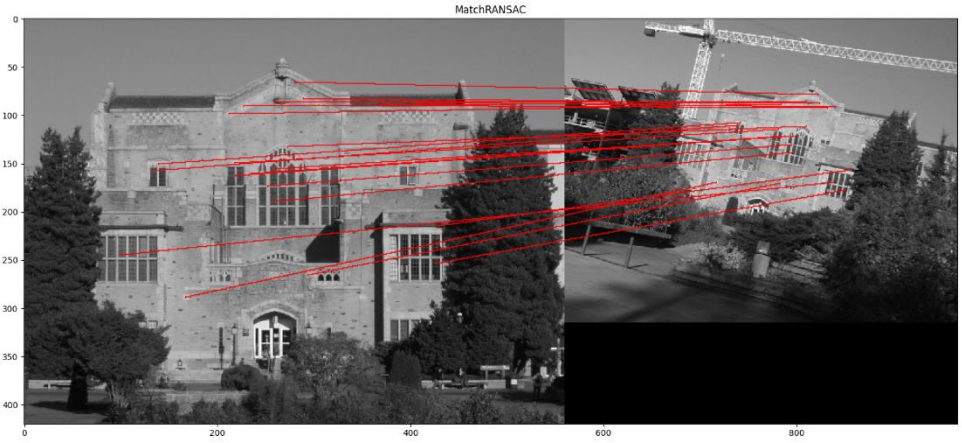
scene-box

② RANSACFilter

RANSACFilter 함수를 이용하여 image1과 image2의 descriptor를 매칭한다.

descriptor의 orientation 차이를 30도를 기준으로 scale의 비율을 0.5를 기준으로 하여 inlier를 결정한다.

실행결과



library-library2

2) Panorama

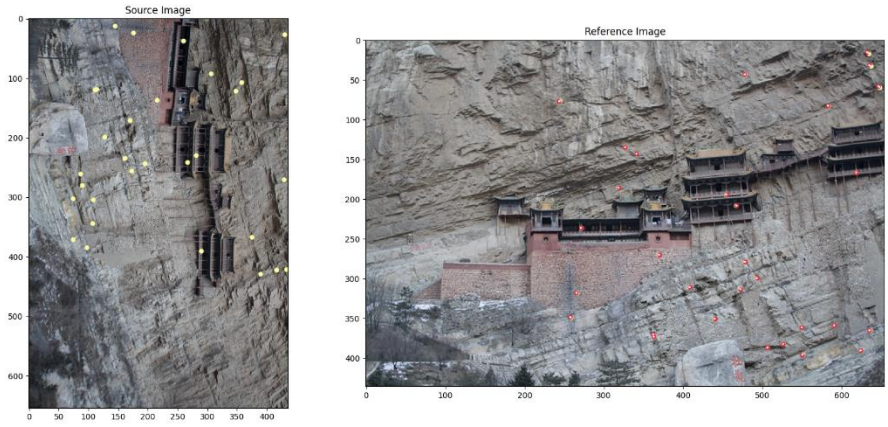
① KeypointProjection

```
xy_points_temp = np.c_[xy_points, [1] * len(xy_points)]
xy_points_out = h @ xy_points_temp.T
xy_points_out = xy_points_out.T

for i, point in enumerate(xy_points_out):
    div = point[2]
    if div == 0.0:
        div = 1e10

    xy_points_out[i] /= div
```

KeypointProjection 함수를 통해 Homography matrix를 이용하여 이미지의 점을 projection한다.

점을 projection하기 위해 각 점마다 마지막 원소로 1을 추가한 후 Homography matrix와 곱한다. 이때, 마지막 원소를 1로 만들기 위하여 마지막 원소로 나누어야 하는데 그 값이 0일 경우 1e10로 나누어 준다.

실행결과



Hanging1-Hanging2

② RANSACHomography

```
eigvals, eigvecs = np.linalg.eig(matrix_a.T @ matrix_a)
v = eigvecs[:, np.argmin(eigvals)]
h_temp = v.reshape((3, 3))
h_temp /= h_temp[-1, -1]
```

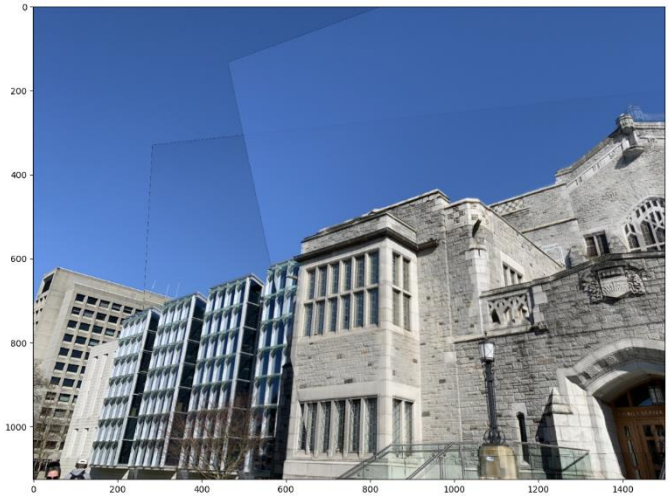
RANSACHomography 함수를 통해 RANSAC을 이용하여 Homography matrix를 직접 구한다.

eigenvalue의 최솟값에 해당하는 eigenvector를 구하고 이를 Homography matrix의 값으로 사용한다.

실행결과

fountain40 (num_iter=100, tol=5, ratio_thres=0.9)

garden034 (num_iter=400, tol=5, ratio_thres=0.9)



irving_out365 (num_iter=250, tol=5, ratio_thres=0.9)

3. 결론

토의 및 결론 (1페이지)

SIFT, RANSAC 방법을 통해 descriptor를 얻고 적절한 descriptor를 추출한 후 Homography matrix를 구하여 파노라마 이미지를 완성하는 시간을 가졌다.

SIFT 방식 이외의 또 다른 이미지의 특징점을 찾는 방법을 알아보자.

SURF(Speede Up Robust Features)

SIFT보다 더 빠르고 강력하게 특징들을 검출할 수 있는 알고리즘

박스 필터를 사용하여 연산자를 빠르게 계산함으로써, 추적이나 객체 인식 같이 실시간 응용을 가능하게 한다.

[특징 추출]

$I_\Sigma(\mathbf{x}) = \sum_{i=0}^{i\leqslant x}\sum_{j=0}^{j\leqslant y} I(i,j)$	$\mathcal{H}(\mathbf{x},\sigma) = \begin{bmatrix} L_{xx}(\mathbf{x},\sigma) & L_{xy}(\mathbf{x},\sigma) \\ L_{xy}(\mathbf{x},\sigma) & L_{yy}(\mathbf{x},\sigma) \end{bmatrix}$
적분 영상	헤시안 행렬

행렬 연산 전에 계산량을 줄이기 위해 적분 영상을 추출하고 수식을 이용하여 헤시안 행렬을 계산한다. 또한, 커널의 크기를 업스케일링 하는 방식으로 스케일 공간을 분석한다.

- 적분 영상 : 원점과 x에 의해 형성된 직사각형 영역 내의 입력 영상 I에 있는 모든 픽셀의 합
- 헤시안 행렬 : L_xx(X, 시그마)는 점 x에서의 영상 I와 2차 미분한 가우시안 커널을 컨볼루션한 것

[특징 기술]

1. 특징점 주변 원 영역의 정보를 기반으로 reproducible한 방향을 고정한다.
2. 선택된 방향에 맞춰 정렬된 정사각형 영역을 구성하고 해당 영역에서 SURF descriptor를 추출한다.