과제 2: Local Feature Matching

김의찬
국민대학교 전자공학부
Uichan8@naver.com

1. 서론

이미지에서 특징 점을 뽑는 get_interest_points 함수를 구현하여 특징점인 점들의 좌표를 계산하고 그 좌표 들에서 semi SIFT 기법을 이용하여 get_features 를 구현하여, 기술자 즉 그 점들의 특징을 뽑아내고, 그 점들을 match_features 로 얼마나 점들이 유사한지 가려 두사진에서 같은 점을 찾아줍니다.

2. 과제 수행 내용

2.1 get_interest_points

과제에 제가 2 개를 구현했는데, 두개의 차이점은 **IxIx, IxIy, IyIy 를 구할 때 2 계도 미분을 넣었느냐, 그냥 제곱을 해서 넣었느냐** 그 두가지의 차이입니다. 원래라면 제곱을 해서 넣는게 맞는데 성능이 너무 안 나와서 **그냥 2 계도 미분을 해서 넣었습니다.** 2 계도 미분한 것은 noise 에 민감하고, edge 에도 민감해서 그러한 점들을 쳐내는 작업이 필요한데, adaptive Non-maximal suppression(과제 추가점수부분)으로 선별을 했습니다. 참고로, 2 계도 미분에서 테일러 급수로 근사해서 제곱을 구하는 거라 나머지는 다 똑같습니다.

대략적인 구현 과정은 다음과 같습니다,

1. 이미지를 가우시안 필터링해 noise 를 잡습니다
2. 이미지를 편미분 합니다
3. IxIx, IxIy, IyIy 를 구해줍니다.
4. Det, trace 를 구합니다
5. Cornerness score 을 계산합니다.
6. 임계값 이하는 버립니다.
7. 지역적 max 를 뽑습니다.
8. adaptive Non-maximal suppression 로 특징점 같은 것 만 뽑아줍니다.
9. 파라미터를 실험적으로 넣어줍니다.

6,7,8 과정에서 adaptive Non maximal suppression 적용하고 임계값을 버려줘도 되긴 한데 이미지 전체에서 돌릴라면, 시간이 오래 걸립니다. 그래서 다음과 같은 과정을 거쳤습니다.

feature_width 는모호해서 안썼습니다.

2.2 get_features

1. 이미지에 가우시안 필터를 적용합니다.
2. 이미지와 좌표를 반으로 줄입니다.
이는 16x16 window 에 32x32 짜리 사진이 들어가게 합니다.
3. 이미지를 편미분 합니다.
4. 이미지 그레디언트의 방향과 크기를 구합니다
5. 주어진 좌표를 기준으로 16*16 으로 방향과 크기를 자릅니다
6. 4*4 윈도우로 16 등분한 뒤 각각을 8 개의 히스토그램으로 표현합니다.
7. 8 개짜리 히스토그램 16 개를 이어 붙입니다.
8. 정규화 하고, 0.2 넘어가는 것을 0.2 로 고정하고 다시 정규화해줍니다.
9. 벡터들을 리턴해줍니다.

2.3 match_features

NNDR 을 이미지 1 안에서 구해서, 거리가 가까운 집단들도 안 넣는게 좋지만 그냥 중복되는 인덱스들을 제외 하는걸로 처리했습니다.

1. 두 이미지 벡터 사이의 모든 유클리드 거리를 구합니다
2. image1 을 기준으로 최소값, 2 번째 최소값을 구하고 NNDR 을 구해줍니다.
3. 최소값과 NNDR 을 가지고 match 인지 아닌지를 결정해줍니다.
4. 한 점에서 여러개의 점들이 매칭 될 경우 결과에서 빼줍니다.
5.confidence 는 -nndr 로 결정합니다. 거리가 작고 nndr 도 작아야 둘의 정합도가 높은 것 입니다. 하지만 둘은 다른 물리량을 가지기 때문에, 단순 가중합으로 표현하기는 힘듭니다. 그래서 실험적으로 nndr 을 음수로 만들어 넣어줍니다.
6. 실험적으로 파라미터를 넣어줍니다.

3. 실험 결과 및 분석

사진 1 의 결과는 다음과 같습니다.

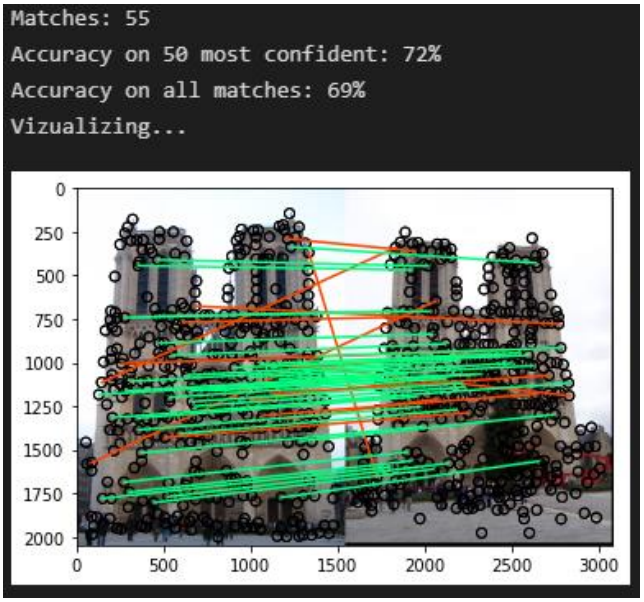
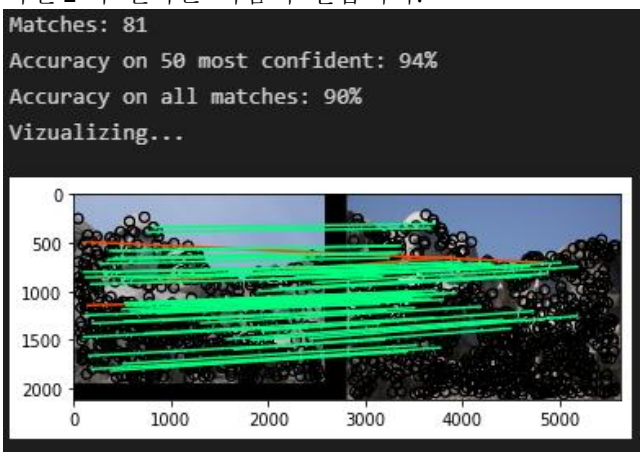
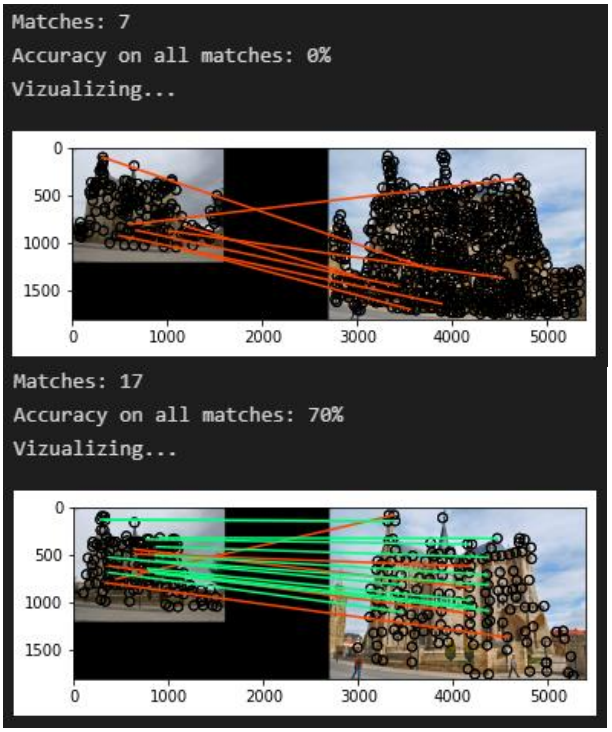


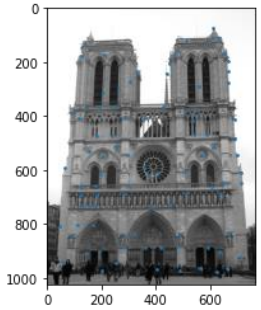
사진 2 의 결과는 다음과 같습니다.



3 번째 사진 같은 경우 그냥 생성했던 조건들로 돌리면 안 좋은 결과가 나옵니다. 사진을 봤을 때 성의 크기차이가 많이 나는 것을 확인 할 수 있는데 **기술자를 생성할 때 성의 대략적인 크기를 반영**해서 넣어주면, 살짝 성능이 좋아지는 것을 확인 할 수 있습니다. Cheat point 기준으로 돌린 것인데, 제가 크기를 고려하여 기술자를 뽑게 만들고, 특징점을 뽑을 때 도 크기에 맞게 잘 뽑게 하고, 적당한 코너 점을 내주도록 했으면 3 번째도 잘 matching 이 됐을 것 입니다. (get_features scale 25)



Matches 가 너무 안 잡혀서 점들을 많이 나오도록 파라미터를 수정했습니다. 적절하게 파라미터를 조정하면 어느정도는 특징점이 잡히긴 잡힙니다.



기술자 추출에서 밝기는 상관없지만 크기나 회전에 대해서 전혀 고려하지 않았기 때문에 Matches 가 잘 안 잡힙니다.

크기랑 각도가 비슷한 두 이미지는 잘 매칭 됩니다. 3 번째 match 의 파라미터를 조정하면, match 되는 수 자체는 줄지만, 정확도를 높일 수 있습니다.

4. 결론

첫번째 특징점 함수는 기능은 하나 성능이 너무 안 좋습니다.

두번째 기술자 함수는 목적에 맞게 특징들을 어느정도 뽑아내는 것 같습니다.

세번째 match 는 목표에 맞게 점들을 비교하여 줍니다.

참고문헌

[1] 수업내용, 과제에서 주어진 링크들만 활용하였습니다.