**OSP: Lec9-Feature Matching and Fitting**

1871098 손수민

1. Image Stitching using Affine Transform

Case1) Mx=b

본 예제는 두 개의 이미지에 대해 SIFT descriptor를 적용하고, NN을 사용하여 feature matching을 수행한 후 cross-checking & ratio-based thresholding을 사용해 정제한 특징 매칭점으로 affine transform estimation을 수행하여 이미지를 stitching하는 예제이다. 이미지를 붙이는 코드는 lab2의 코드를 사용하였으며 feature matching은 그 전 강의의 코드를 사용하였다.

함수 euclidDistance, nearnestNeighbor, findPair는 그 전 강의에서 SIFT.cpp에서 쓴 함수를 가져왔고, cal\_affine, blend\_stitiching은 lec2에서 쓴 stitiching.cpp에서 그대로 가져왔다.

우선 이미지 input1, input2를 불러와 저장하고 이를 회색이미지로 변환해준다. 그 후 resize를 이용하여 이미지 크기를 반으로 줄여준다.

텍스트이(가) 표시된 사진

자동 생성된 설명

그 후 SiftFeatureDetector를 이용해 detector를 선언해주고 SiftDescriptorExtrator를 이용해 extractor도 선언해준다.

텍스트이(가) 표시된 사진

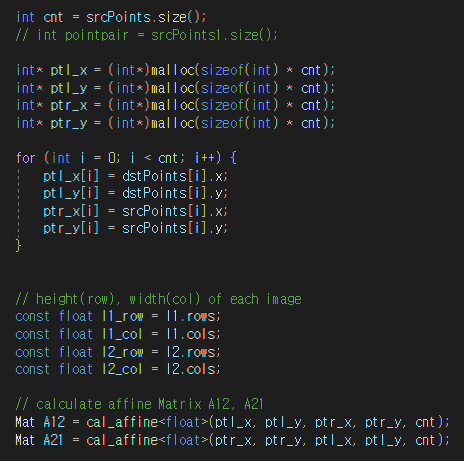
자동 생성된 설명

Input1과 input2의 keypoints들을 detect와 compute를 통해 찾아주고 crosscheck와 ratio\_threshold를 true로 설정해 findPairs를 통해 feature matching을 해준다.

텍스트, 스크린샷, 모니터, 화면이(가) 표시된 사진

자동 생성된 설명

그 후 ptl\_x, ptl\_y, ptr\_x, ptr\_y에 각각의 dstPoints, srcPoint를 저장해준뒤 이를 다시 cal\_affine 함수에 넣어 A12, A21 affine matrix를 구해준다.



그 후에는 stitching.cpp에서 구현한대로 bound\_u, bound\_b, bound\_l, bound\_r를 구해 l\_f도 구하고 이를 blend\_stitiching 함수에 넣으면 stitching 이미지가 반환된다.

텍스트이(가) 표시된 사진

자동 생성된 설명

결과값은 다음과 같이 반환된다.

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트, 산, 자연이(가) 표시된 사진

자동 생성된 설명

텍스트, 산, 하늘, 실외이(가) 표시된 사진

자동 생성된 설명

Case2) Mx=b + RANSAC

본 예제는 앞의 예제와 똑같은 과정을 거치되, affine transform estimation에서 Mx=b + RANSAC을 추가하여 진행하는 코드이다. 마찬가지로 이미지를 붙이는 코드는 lab2의 코드를 사용하였으며 feature matching은 그 전 강의의 코드를 사용하였다.

함수 euclidDistance, nearnestNeighbor, findPair는 그 전 강의에서 SIFT.cpp에서 쓴 함수를 가져왔고, cal\_affine, blend\_stitiching은 lec2에서 쓴 stitiching.cpp에서 그대로 가져왔다.

앞의 부분은 동일하므로 RANSAC이 추가된 부분을 중심으로 보고서를 작성하였다.

K=3, s=10, sigma=3으로 세팅하여 진행하였다. (threshold=sigma^2)

cnt는 앞에서 SIFT를 통해 구한 특징점들의 개수이다.

텍스트이(가) 표시된 사진

자동 생성된 설명

그 다음으로 S번 다음과 같은 과정을 반복하게 된다. 먼저 SIFT를 통해 구한 점들에서 k개를 뽑기 위해 rand()함수를 이용하고 뽑은 인덱스들은 겹치지 않도록 한다.

그 후 인덱스에 해당하는 point들을 src,dst에 저장해준다.

텍스트이(가) 표시된 사진

자동 생성된 설명

그렇게 뽑은 src,dst들로 cal\_affine함수에 넣어 affine transformation T를 구해준다.

그 다음 모든 점들에 대해 |Tp − p|^2 < sigma^2 를 만족하는 개수를 세어 score에 넣어준다.

이 위의 과정을 총 s번 반복해준다.

텍스트이(가) 표시된 사진

자동 생성된 설명

이 과정에서 제일 높은 score는 maxscore에 저장이 되고, 그에 해당하는 inliner 점들은 best\_src, best\_dst에 저장이 된다. 마지막으로 다시 이것으로 affine transform을 구해주면 RANSAC과정이 들어간 affine transform estimation 과정이 끝난다.

그 후로는 마찬가지로 image stitching 과정이 들어가면 된다.

텍스트이(가) 표시된 사진

자동 생성된 설명

앞의 과정에 대한 결과값이다.

텍스트, 산, 스크린샷이(가) 표시된 사진

자동 생성된 설명

1. Line Fitting using Hough Transform

Explain how to estimate the line segments in the Hough transform.

Canny edge detector를 실행하고 Hough transform(HoughLines, HoughLinesP)를 사용해 line fitting result를 그려보았다. 관련 함수는 공식 문서를 참고하였다.

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트, 건물이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

테이블이(가) 표시된 사진

자동 생성된 설명

텍스트, 실외, 건물, 빨간색이(가) 표시된 사진

자동 생성된 설명