Report

Step 1:
Using Harris to detect feature points, then store their coordinates.
For ‘uttower_left.JPG’, I get 289 points. (assume this is image1)
For ‘uttower_right.JPG’, I get 316 points. (assume this is image2)

Step 2:
Build a setDescriptor function to extract fixed size image patches. I choose 5*5 patches. And then change this 5*5 matrixes to 25*1 matrix. At the same time, I need to ensure every point of a patch I get within the image. If it over the boundary of image, I set the value of this point to be zero.

Step 3:
Use this setDescriptor function, traverse all feature points to get their descriptors, then store in a matrix.

	1	2	3	4	5	6	7	8	9	10	11	12
7	82	198	208	89	116	174	101	172	42	172	101	112
8	181	141	186	164	180	158	189	166	55	88	139	125
9	227	172	135	183	177	98	202	133	83	119	121	90
10	214	212	105	185	178	72	181	0	173	205	86	109
11	51	214	210	88	60	176	79	179	110	160	61	24
12	86	204	216	99	114	173	92	173	84	104	118	145
13	182	136	205	164	179	157	187	169	118	73	157	190
14	228	169	158	182	178	97	207	138	163	111	159	160
15	216	215	119	185	179	72	188	0	200	202	108	205
16	75	213	205	87	59	177	79	175	184	63	75	31
17	90	207	210	95	114	172	93	179	143	64	123	146
18	182	135	196	166	180	156	189	176	139	51	150	183
19	227	171	138	182	179	96	209	148	163	64	174	191
20	218	214	116	187	181	72	185	0	199	167	169	217
21	69	211	206	81	61	179	80	173	181	46	58	55
22	90	204	203	99	115	173	97	170	126	49	67	160
23	180	135	174	169	179	156	191	168	93	48	64	187
24	225	176	148	183	178	97	209	140	83	48	69	190
25	220	213	138	188	182	73	180	0	172	156	71	217

This table is a part of descriptors’ matrix of image1.

Step4:
Compute the normalized correlation. I find a formula to compute this.

$$dc(y_1, y_2) = \frac{y_1^T y_2}{|y_1||y_2|}$$

Then I get a 289*316 matrix to store the value of normalized correlation.

After that, for each point in image, I find the most match descriptor in image2 by using this normalized correlation matrix.

Step5:
In order to run RANSAC, I change the coordinates of corner of image1 and image2 to Homogeneous plane.

Step6:
Randomly choose 8 point pairs from corners of image1 and image2, to generate a fundamental matrix. This is a temporary F model. Then compute error function.

	1	2	3
1	-0.6976	-3.0591	1
2	0.2124	1.9846	1
3	0.4098	1.1513	1
4	-0.1604	-2.2696	1
5	0.1247	1.4912	1
6	-0.3358	2.7521	1
7	0.0809	-2.5108	1
8	0.3659	0.4605	1

8 point pairs of image1 (normalized)

	1	2	3
1	0.2173	-1.3703	1
2	-0.1523	-1.6988	1
3	0.2036	-3.0127	1
4	-0.6997	1.6270	1
5	0.1831	-1.7878	1
6	0.1831	1.3327	1
7	0.4226	2.1744	1
8	-0.3576	2.7356	1

8 point pairs of image2 (normalized)

	1	2	3
1	3.0337e-06	1.2273e-06	-0.0023
2	-2.1403e-07	-1.9137e-07	2.4439e-04
3	-0.0014	-4.8955e-04	1

Temporary best F

Then select all point pairs fit with model. If the error of a pair less than error that we get before, we treat it as inliner point. So we get outliners ratio e: $e = 1 - (\text{number of inliers})/(\text{total number}$

of points), s= 8 (I choose 8 point pairs), p=0.9. Then I can follow the formula to write my code to find the right F. N is number of samples. I assign a very large number to initialize it.

$$N = \log(1-p)/\log\left(1-(1-e)^{s}\right)$$

Step7:
Then I try to improve my code by using the normalized eight-point algorithm. First, I normalized original corner feature point in image1 and image2 to get a 3*289 array. Second, fundamental matrix has rank 2 by performing a SVD and then reconstructing with the two largest singular values. Third, denormalize F.

	1	2	3
1	3.0337e-06	1.2273e-06	-0.0023
2	-2.1403e-07	-1.9137e-07	2.4439e-04
3	-0.0014	-4.8955e-04	1

before denormalize

	1	2	3
1	3.0287e-06	1.2200e-06	-0.0023
2	-2.2000e-07	-2.0002e-07	2.4439e-04
3	-0.0014	-4.8955e-04	1.0000

After SVD

	1	2	3
1	1.8678e-10	7.5235e-11	-1.8286e-05
2	-1.3567e-11	-1.2336e-11	1.9432e-06
3	-1.0833e-05	-3.8513e-06	1.0147

After denormalize