Udacity Sensor Fusion - Midterm Project - Camera Based 2D Feature Tracking

Deniz Bardakci

February 2024

1 MP.0 Mid-Term Report

Provide a Writeup / README that includes all the rubric points and how you addressed each one. You can submit your writeup as markdown or pdf. This document serves to fullfill this requirement.

2 MP.1 Data Buffer Optimization

Implement a vector for dataBuffer objects whose size does not exceed a limit (e.g. 2 elements). This can be achieved by pushing in new elements on one end and removing elements on the other end.

- Using a ring buffer with a fixed size would optimize memory usage. It efficiently manages images by deleting the oldest ones from the beginning of the vector, while ensuring the latest images are always stored at the end. The idea has been implemented in MidTermProject_Camera_Student.cpp file line 105-111.

3 MP.2 Keypoint Detection

Implement detectors HARRIS, FAST, BRISK, ORB, AKAZE, and SIFT and make them selectable by setting a string accordingly.

- The mentioned detectors have been divided into 3 functions.
- ShiTomasi detector has been defined as detKeypointsShiThomasi in matching2D_students.cpp line 13. It's corresponding decleration in matching2D.hpp
- Harris detector has been defined as detKeypointsHarris in matching2D_students.cpp line 54. It's corresponding decleration in matching2D.hpp
- Other detectors FAST, BRISK, ORB, AKAZE, and SIFT habe been defined as detKeypointsModern in matching2D_students.cpp line 135. Their corresponding decleration in matching2D.hpp
- Calling these functions have been encapsulated by a user defined callDetector function in MidTermProject_Camera_Student.cpp line 22. The corresponding function has been called in same cpp file's line 124.

4 MP.3 Keypoint Removal

Remove all keypoints outside of a pre-defined rectangle and only use the keypoints within the rectangle for further processing.

- Corresponding removal operation has been implemented in MidTermProject_Camera_Student.cpp line 140.

5 MP.4 Keypoint Descriptors

Implement descriptors BRIEF, ORB, FREAK, AKAZE and SIFT and make them selectable by setting a string accordingly.

- Corresponding descriptors are defined in matching2D_student.cpp as descKeypoints function in line 196.

6 MP.5 Descriptor Matching

Implement FLANN matching as well as k-nearest neighbor selection. Both methods must be selectable using the respective strings in the main function.

- FLANN matching has been implemented in matchDescriptors function line 299 of matching2D_student.cpp.
- K-Nearest neighbour has been implemented in matchDescriptors function line 323 matching2D_student.cpp.

7 MP.6 Descriptor Distance Ratio

Use the K-Nearest-Neighbor matching to implement the descriptor distance ratio test, which looks at the ratio of best vs. second-best match to decide whether to keep an associated pair of keypoints.

- The descriptor ratio test logic has been implemented in matchDescriptors function line 272 matching2D_student.cpp.

8 MP7 Performance Evaluation 1

Following tables evaluates the performance of the **detectors**. Count the number of keypoints on the preceding vehicle for all 10 images and take note of the distribution of their neighborhood size.

- The number of keypoints detected by detectors have been shared as Table: 1.
- If the number of Region of Interest (ROI) points have been selected as evaluation criteria BRISK, FAST, and AKAZE have been performed as top 3 detectors.
- The resulting plot can be seen as Fig.1
- Fig.2, Fig.3, Fig.4, Fig.5, Fig.6, Fig.7, and Fig.8 show the results of different corner detectors.

Table 1: KeyPoint Detection Results

	Total Number of KeyPoints / Number of Region of Interest Keypoints						
	Methods						
	SHITOMASI	HARRIS	FAST	BRISK	ORB	AKAZE	SIFT
Image 0	1370/125	339/51	1824/149	2757/264	500/92	1351/166	1438/138
Image 1	1301/118	286/41	1832/152	2777/282	500/102	1327/157	1371/132
Image 2	1361/123	349/63	1810/150	2741/282	500/106	1311/161	1380/124
Image 3	1358/120	356/58	1817/155	2735/277	500/113	1351/155	1335/137
Image 4	1333/120	521/85	1793/149	2757/297	500/109	1360/163	1305/134
Image 5	1284/113	2611/322	1796/149	2695/279	500/125	1347/164	1370/140
Image 6	1322/114	200/38	1788/156	2715/289	500/130	1363/173	1396/137
Image 7	1366/123	806/136	1695/150	2628/272	500/129	1331/175	1382/148
Image 8	1389/111	572/96	1749/138	2639/266	500/127	1357/177	1463/159
Image 9	1339/112	1471/194	1770/143	2672/254	500/128	1331/179	1422/137

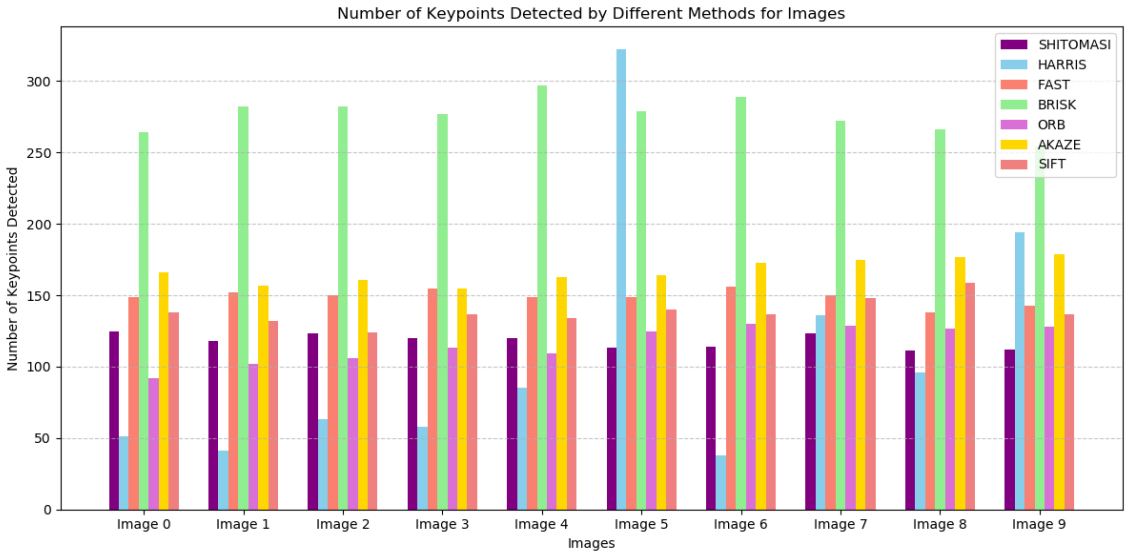

Figure 1: Number of Keypoints detected by detectors

Figure 2: Shi-Tomasi Corner Detector Results

Figure 3: HARRIS Corner Detector Results

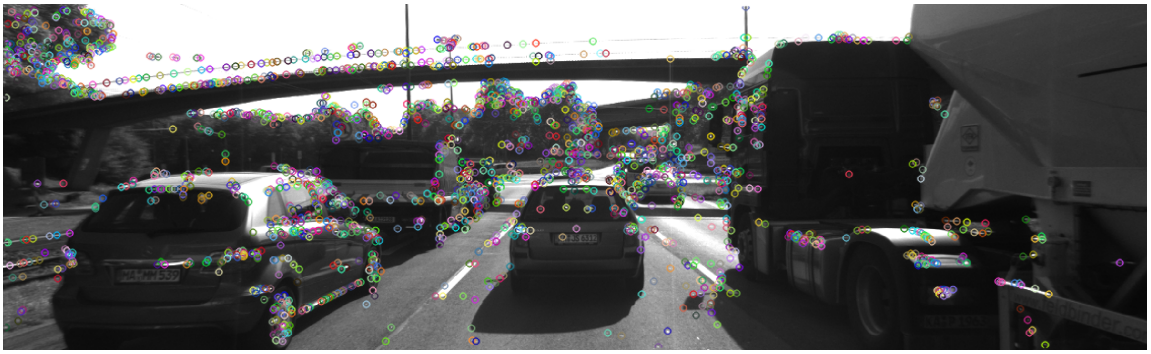

Figure 4: FAST Corner Detector Results

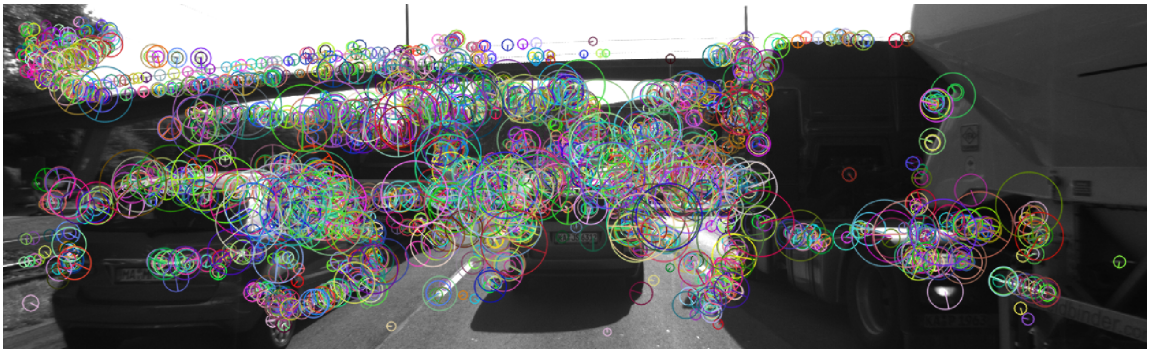

Figure 5: BRISK Corner Detector Results

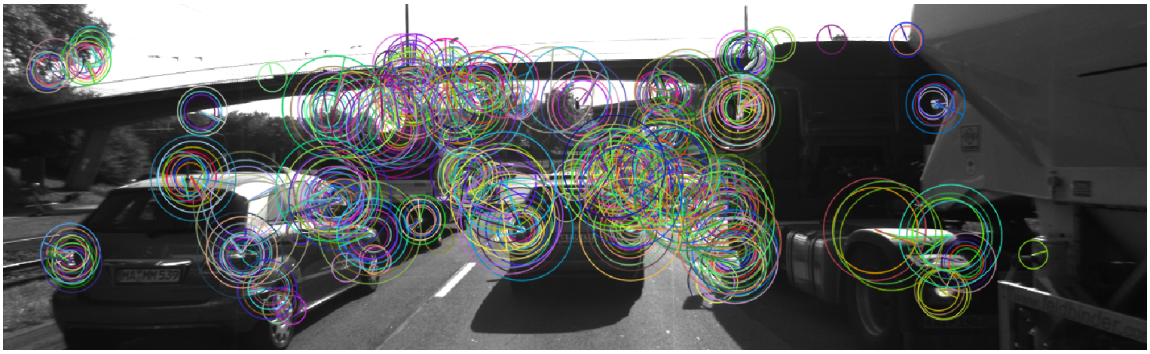

Figure 6: ORB Corner Detector Results

Figure 7: AKAZE Corner Detector Results

Figure 8: SIFT Corner Detector Results

9 MP.8 Performance Evaluation 2

Count the number of matched keypoints for all 10 images using all possible combinations of detectors and descriptors. In the matching step, the BF approach is used with the descriptor distance ratio set to 0.8.

- MAT_BF has been used for matching process.
- The average number of keypoints detected by detectors with combination of different descriptors have been shared as Table: 2.
- According to analysis BRISK detector together with BRIEF descriptor has been performed better than other candidates. Sadly, some of the candidates could not perform analysis due to Known Issue which is raised by OpenCV(4.1.0).

Table 2: Average Matched Keypoints with different descriptors

	Descriptors					
Detectors	BRISK	BRIEF	ORB	FREAK	AKAZE	SIFT
SHITOMASI	85	105	101	86	N/A	N/A
HARRIS	15	19	18	16	N/A	N/A
FAST	100	122	120	98	N/A	N/A
BRISK	175	191	169	170	N/A	N/A
ORB	83	62	85	47	N/A	N/A
AKAZE	135	142	132	132	132	N/A
SIFT	66	79	N/A	66	N/A	N/A

10 MP.9 Performance Evaluation 3

Log the time it takes for keypoint detection and descriptor extraction. The results must be entered into a spreadsheet and based on this data, the TOP3 detector / descriptor combinations must be recommended as the best choice for our purpose of detecting keypoints on vehicles.

- Table 3 summarises the keypoint detection/descriptor extraction durations by each detector/descriptor pairs for all images. The values shown in table are results of average performance durations of 10 images.
- The descriptor/detector durations are obtained by utilizing new namespace in matching2D.hpp, this namespace's variables are utilizied in order to record description and detection and matching process durations.
- Sadly, some of the candidates could not perform analysis due to Known Issue which is raised by OpenCV(4.1.0). Sadly almost all of the detectors have a version issue with AKAZE and SIFT descriptors due to OpenCV(4.1.0).
- Among the other Detectors, FAST performed enormously fast operations. TOP3 best performers can be listed as 1)FAST/BRIEF combination was the best performer for keypoint detection and descriptor extraction. 2)FAST/ORB 3)FAST/BRISK

Table 3: Average Durations [ms] of Keypoints detection - Detector/Descriptor Pairs

	Descriptors					
Detectors	BRISK	BRIEF	ORB	FREAK	AKAZE	SIFT
SHITOMASI	11.81/1.57	11.91/0.76	12.03/0.66	14.03/24.8	N/A	N/A
HARRIS	15.14/0.96	14.79/0.63	15.424/0.65	11.707/21.4	N/A	N/A
FAST	0.84/1.674	0.73/0.71	0.58/0.79	0.68/24.74	N/A	N/A
BRISK	40.40/2.92	38.93/0.71	36.16/7.00	39.5/33.65	N/A	N/A
ORB	9.48/1.28	6.2/0.43	8.04/9.25	11.71/29.94	N/A	N/A
AKAZE	164.91/1.61	160.28/6.85	164.92/2.35	140.29/45.85	147.69/164.53	N/A
SIFT	141.68/1.43	148.16/0.75	N/A	138.00/37.95	N/A	N/A