Midterm Project Performance Evaluation

Task 1: Data Buffer Optimization

- Initially, first image is added in data buffer. The second image is added in the buffer consecutively
- Data buffer size is checked against predefined **dataBufferSize = 2**. As the third image is to be pushed, the older image (in this context the first image) will be replaced
- This continues iteratively ensuring two images are present in data buffer at a given time

Task 2: Key point Detection

- Based on the user input detector type string is used to select appropriate detector functions
- Predefined prototypes are used to add the functions for the detectors as follows (matching2D_Student.cpp)

  1. detKeypointsShiTomasi → SHI Tomasi
  2. detKeypointsHarris → Harris
  3. detKeypointsModern → FAST, BRISK, ORB, AKAZE, and SIFT.

- String selector is used as **string detectorType = "FAST"; // <-- User input for selection of detector**

Task 3: Key point Removal

- Key points are taken from the predefined rectangular box to reduce noise and save computation power
- **float Cx = 535, Cy = 180, W = 180, H = 150; // Taking co-ordinates for bounding box of preceding vehicle**
- Selection Criteria used → **(keypoints[i].pt.x > Cx && keypoints[i].pt.x < Cx+W) && (keypoints[i].pt.y > Cy && keypoints[i].pt.y < Cy+H)**

Task 4: Key point Descriptors

- Based on the user input Descriptor type string is used to select appropriate Descriptors in function **descKeypoints** (matching2D_Student.cpp)
- OpenCV build-in descriptors (BRIEF, ORB, FREAK, AKAZE and SIFT) are used with default parameter setting
- String selector is used as **string descriptorType = "BRIEF"; //BRISK, BRIEF, ORB, FREAK, AKAZE, SIFT**
- When SIFT descriptor is used, string descriptorType = " **DES_HOG**"; // DES_BINARY, DES_HOG is used to convert the image appropriately for matching keypoints (Done while doing performance evaluation reporting)

Task 5: Descriptor Matching

- Descriptor matcher is selected based on string user input as MAT_BF OR MAT_FLANN
- For performance evaluation, MAT_BF is used
- Although, while using MAT_FLANN following conversion check is used

```
if ((descSource.type() != CV_32F) || (descRef.type() != CV_32F) )
    {
     descSource.convertTo(descSource, CV_32F);
     descRef.convertTo(descRef, CV_32F);
    }
```

Task 6: Descriptor Distance Ratio

- For descriptor matching, nearest neighbor (NN) and K nearest neighbors (KNN; default k=2) are used
- For KNN matching, match points are filtered using descriptor distance ratio as below
- double minDescDistRatio = 0.8; (Predefined check)

```
for (auto it = knn_matches.begin(); it != knn_matches.end(); ++it) {
      if ((*it)[0].distance < minDescDistRatio * (*it)[1].distance) {
        matches.push_back((*it)[0]);
      }
    }
```

Task 7: Frame Analysis

DetectorFrame	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	Total Keypoints/Execition time (ms)
SHI Tomasi	1370	1301	1361	1358	1333	1284	1322	1366	1389	1339	13423
	29.34	19.36	21.49	20.1	19.22	19.27	20.28	19.59	18.79	19.66	207.1
Harris	115	98	113	121	160	383	85	210	171	281	1737
	21.63	18.81	18.08	17.01	19.82	39.21	18.62	21.05	26	24.91	225.14
Fast	1824	1832	1810	1817	1793	1796	1788	1695	1749	1770	17874
	1.07	1.01	1.03	0.97	1.01	1.01	0.95	1.05	1.02	0.98	10.1
ORB	500	500	500	500	500	500	500	500	500	500	5000
	26.5	9.68	8.79	9.29	8.49	8.96	9.25	8.79	8.42	9.4	107.57
ANKAZE	1351	1327	1311	1351	1360	1347	1363	1331	1357	1331	13429
	132.26	116.213	113.34	113.029	113.61	120.62	123.562	123.31	135.164	134.27	1225.378
SIFT	1438	1371	1380	1335	1305	1370	1396	1382	1463	1422	13862
	199.12	188.685	201.012	201.72	184.64	177.86	179.6	176.71	211.06	183.72	1904.127

Task 8: Matched Key points

Descriptor\Detector	Harris	SHITOMASI	FAST	BRISK	ORB	AKAZE	SIFT
BRISK	137	350	344	308	349	342	185
BRIEF	164	410	396	312	272	359	221
ORB	157	395	394	307	336	326	Out Of Memory
FREAK	140	342	328	322	242	330	188
AKAZE	x	x	x	x	x	366	x
SIFT	157	403	380	319	368	355	324

Task 9: Execution time

Descriptor\Detector	Harris	SHITOMASI	FAST	BRISK	ORB	AKAZE	SIFT
BRISK	355.5	355.53	3427.5	700.5	344.19	441.3	489.9
BRIEF	249.946	200.798	22.1816	4415.06	94.667	1131.7	1694.68
ORB	240.969	202.362	53.3015	444.3	134.387	2321.26	x
FREAK	627.854	584.66	442.482	1516.3	600.203	1884.2	x
AKAZE	x	x	x	x	x	2087.37	x
SIFT	484.871	439.949	307.328	4173.11	668.025	1452.36	2438.63

Best Combinations

1. FAST + BRIEF
2. FAST + ORB
3. ORB + BRIEF