Final Project 3D Object Tracking

Overview

Methodology

The project consisted of implementing and analyzing time-to-collision (TTC) for both lidar and camera sensors. An algorithm for statistical removal of lidar sensor data outliers was developed for a more stable estimate of distance by lidar. Similar approaches was attempted for camera sensor data in order to improve estimates with mixed results. A cursory analysis of algorithms and results are presented.

Application Interface

The project template that was provided was re-structured to allow ease of running analyses from the command line in single or (compiled in batch) flows. Example:

```
dan@ros:~/SFND_3D_Object_Tracking/build$ ./3D_object_tracking
    incomplete arguments given.
    usage:
    ./3D_object_tracking -d <DETECTOR_TYPE> -m <MATCHER_TYPE> -x <DESCRIPTOR_TYPE>
-s <SELECTOR_TYPE> \
        [-v] [-o1] [-o2] [-o3]
    where required argument types are:
      DETECTOR_TYPE:  SHITOMASI, HARRIS, FAST, BRISK, ORB, AKAZE, SIFT
      MATCHER_TYPE:  MAT_BF, MAT_FLANN
      DESCRIPTOR_TYPE: BRISK, BRIEF, ORB, FREAK, AKAZE, SIFT
      SELECTOR_TYPE:  SEL_NN, SEL_KNN
    optional arguments:
      -v: visualize results
      -o1: remove bounding box outliers
      -o2: remove keypoint quality outliers
      -o3: remove keypoint distance outliers
      -b: run compiled in batch tests (output stats.csv)
    Example:
       ./3D_object_tracking -d SHITOMASI -m MAT_BF -x BRISK -s SEL_NN
```

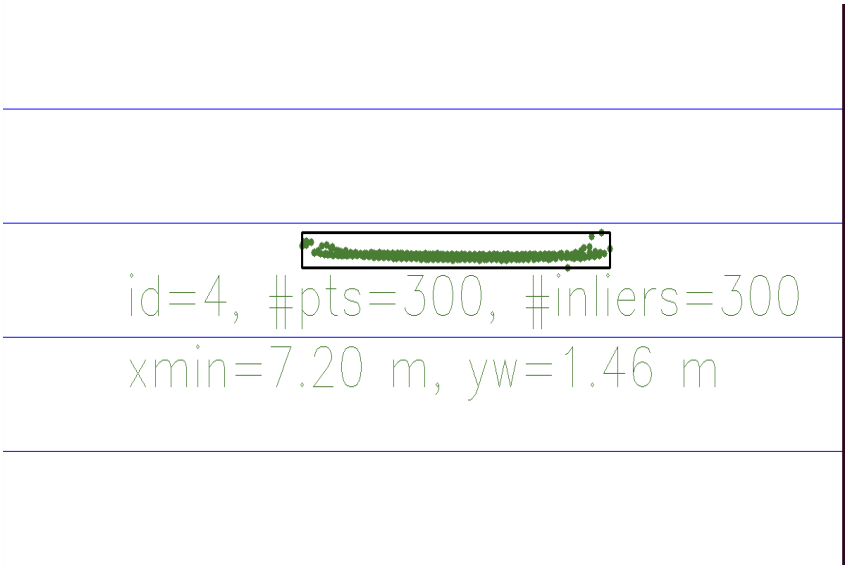
Task FP.1 : Match 3D Objects

Per suggestion, a multimap was used as a pseudo sparse matrix to map the matches to one or more bounding boxes in the previous and current frames by selecting the bounding box with the maximum number of matches. For each keypoint, determine the zero or more bounding boxes it is within for both previous and current frames. Loop over the keys of the multimap (rows) and determine the maximum number of entries for each corresponding box (col).

Code is found in camFusion_Student.cpp:matchBoundingBoxes

Task FP.2 : Compute Lidar-based TTC

This task required computing the time-to-collision estimate based on successive lidar data, while accounting for outlier points in the data. The following image, from lidar points processed from frame #11, show outliers as well as rounding in the spatial points at the edges of the ROI. Occasionally,points both behind the main grouping would appear.



In order to minimize the effect of outliers on lidar distance estimates, a few approaches were considered:

- Use median value of distances
- Use mean value of 'outlier-corrected' data.
- Combination of above.

A first attempt to be able to identify and remove outliers was based on pcl::StatisticalOutlierRemovalpcl::PointXYZ. After some encouraging results, this approach was further refined to implement the algorithm directly into the function:

camFusion_Student.cpp:findInliers()

The algorithm involves two steps:

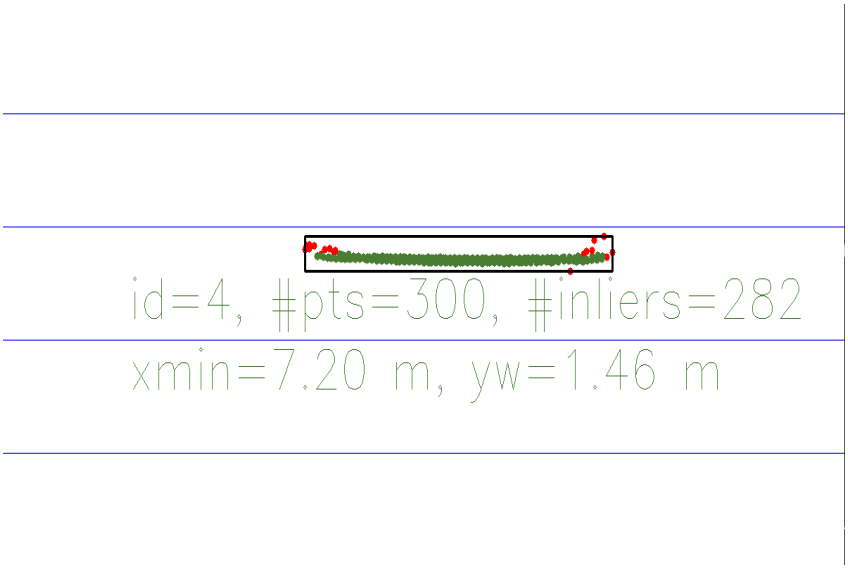
1. compute the mean and standard deviation of the distances from each point in the cloud to their 'nn' nearest neighbors.

2. mark the points in the cloud (pt) in which the mean distance is greater than a threshold value defined as:

```
threshold = pt.mean + std_mult*pt.sigma
```

In addition, the outlier detection ignores the height dimension for distance calculation (essentially squashing the points vertically). The following image shows the same lidar points from frame #11, but with the outliers colored in read, using nn = 10, and std_mult = 1 (show3DObjects() was modified to provide stats and

annotation for assessing the algorthim). The outliers are then ommitted from inclusion in TTC averaging. This option is set by '-o1' on the command line for the application.



In analyses, both average and median lidar distance estimates were evaluated. In the final analysis, the average (with the above mentioned outlier filtering with values of nn=10, and std_mult=1.0) was used as it provided slightly better performance.

Task FP.3 : Associate Keypoint Correspondences with Bounding Boxes

For each keypoint match, we associate it to any bounding box region of interest in the current frame. Because the following TTC estimate is based on the ratio of change between between different keypoints of all the keypoints (O(n^2)), looked at filtering out keypoint mismatches - or keypoints that changed more than could be expected between frames.

Filtering Questionable Keypoint Matches

To filter poor keypoint matches, used a mean-standard deviation approach again to eliminate keypoints whose changes were too large. This filtering algorithm also involved two steps:

1. compute the mean and standard deviation of the distance change of a keypoint matches between frames

2. eliminate those keypoints that are greater than a maximum threshold:

```
max_threshold = mean + std_mult*std_dev
```

In the subsequent analyses, std_mult=1.0 is used in kpt filtering. This option is set by '-o2' on the command line.

Note that this approach didn't seem to improve or degrade TTC performance, but did improve computational performance as it tended to filter out between 5-15% of keypoint matches (which might also be done by

randomly dropping matches as well). This approach requires further investigation, and was omitted in the final analyses.

Task FP.4 : Compute Camera-based TTC

The camera-based TTC calculation algorithm is based on the constant velocity model, using the ratio of the relative pixel-image distance changes between subsequent keypoints (which factors out intrinsic camera parameters, etc).

```
TTC = -delta_t/(1.0 - medianDistRatio)
```

During analysis, medianDistRatio can be <= 1.0, which causes discontinuities as well negative TTC estimates, reformulated to max TTC of 120 min:

```
double ttc(const double delta_t, const double distRatio) {
    const double ttc_max = 120;   // use 120 [min] as 'inf' for ttc calc
    const double drMin = 1.0 + delta_t/120;
    if (distRatio > drMin) {
        return -delta_t/(1.0 - distRatio);
    } else {
        return 120;
    }
}
```

Some of the resulting NxN distance calculated used in estimating TTC had value ranges of 0.5 to 1.5. A fairly large range with lots of values in the tails. So added an optional post filtering step to filter out extreme values. This filtering algorithm also involved two steps:

1. compute the mean and standard deviation of the distances

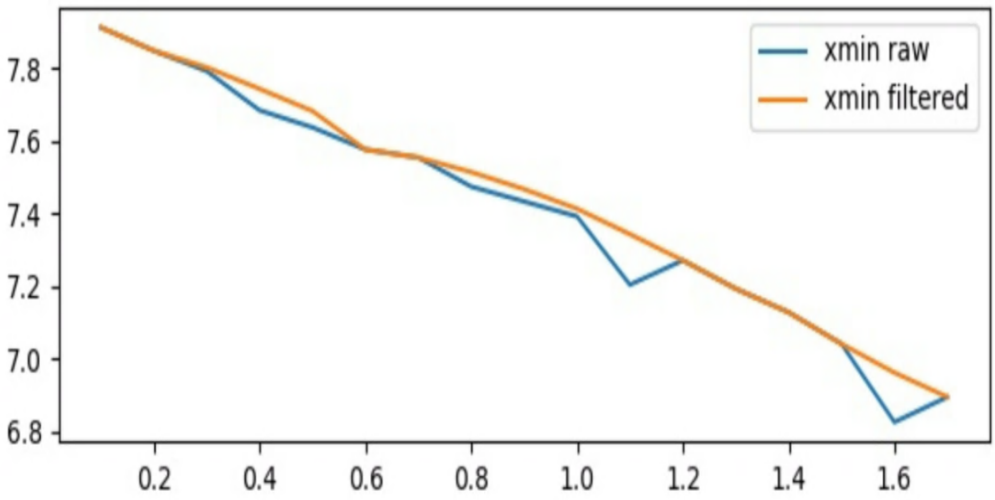
2. eliminate those distances that are less than the minimum or greater than the maximum threshold:

```
min_thr = max(0.0, mean - std_mult*std_dev);
max_threshold = mean + std_mult*std_dev
```

This allowed comparison of both the average and median distance of the ratios to compute TTC. By observation, the average distance didn't perform as well as the median. In the following analyses, the median ratio of ratio changes was chosen to compute the TTC, as per work in previous class exercises.

FP.5 : Performance Evaluation 1 - Lidar TTC

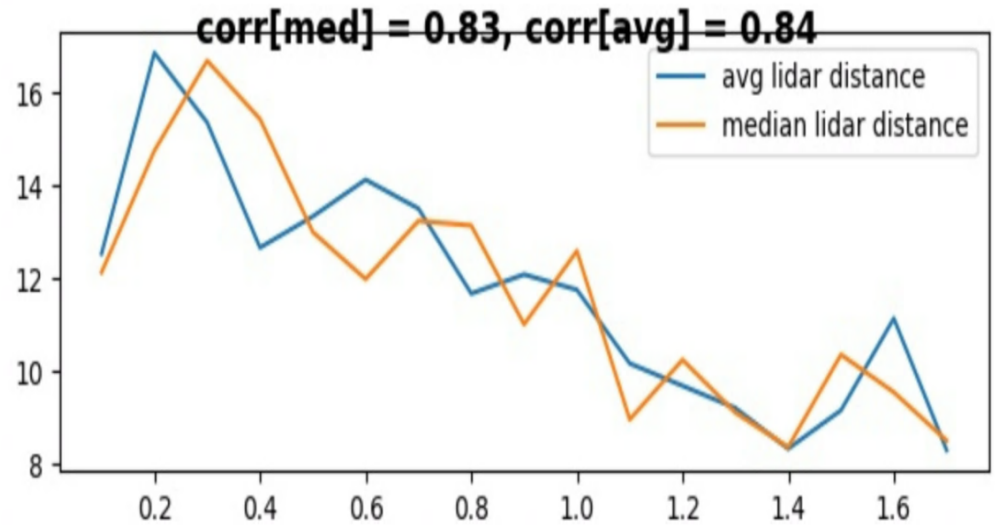
As a reference, gathered and analyzed the lidar sensor data for the estimate of the minimum x distance to the vehicle. Trace "xmin raw" represents the distance to the first lidar point, "xmin filtered" represents the distance to the first lidar point for the dataset with outliers removed. The characteristics of filtering the lidar data were shown previously in the section on bounding box selection.



If using raw data, would expect to see exaggerated estimates of TTC in which the outliers introduced noise in the series, for example at t=1.1[s] and t=1.6[s].

Visual analysis of the "xmin filtered" time series would imply that the relative velocity of the vehicle is constant over the time frame, and hence the TTC estimate should be similarly linearly decreasing with time. So for an evaluation metric, the Pearson correlation coefficient is used to compare the time series for the TTC estimate and the "xmin filtered" data set.

Using the filtered lidar sensor data, TTC estimates are shown below for both the average and median of the lidar sensor data.



The differences between the average and median lidar sensor distances isn't significantly different wrt correlation, but the median value appears to a bit higher volatility - potential implications for downstream control systems.

FP.6 : Performance Evaluation 2 - Camera TTC

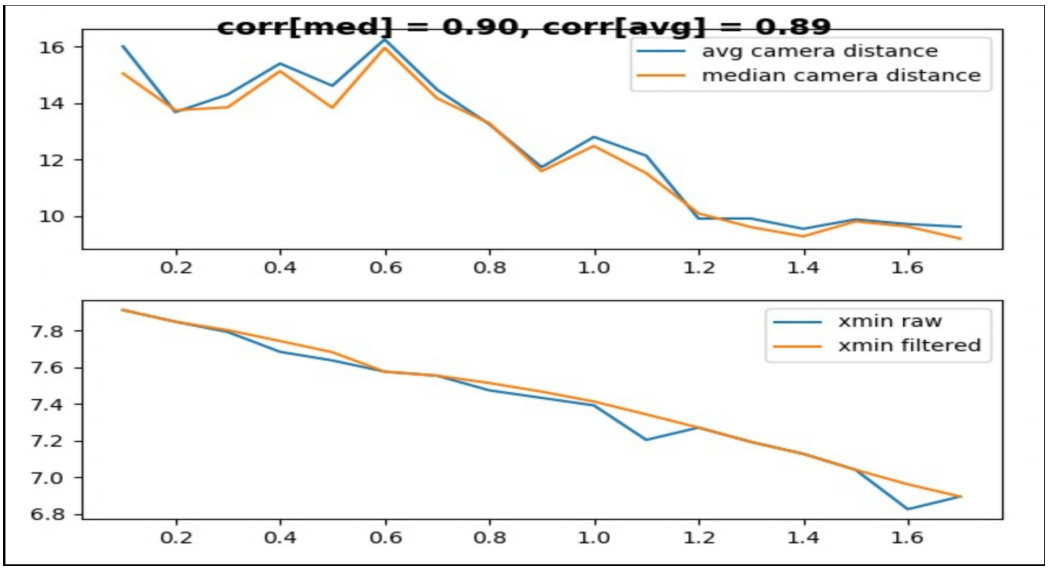
In sorting thru the analysis results (Appendix A), summaries were generated that incorporated the Pearson correlation coefficient in order to help rank results (Appendix B). The correlation measured was between the TTC estimate and "xmin filtered" as a proxy for the estimate in the presence of constant velocity (which appears to be the case by the linear relationship of "xmin filtered" to time). Sorting on the correlation

coefficient, and visually corroborating the results, found that AKAZE feature detectors performed the best/most reliably, and with a variety of descriptors. Following AKAZE, some combinations of SIFT and FAST also did well.

	avgkpts	avgmats	avglidcorr	medlidcorr	avgcamcorr	medcamcorr
/AKAZE_BRIEF_MAT_BF_SEL_NN_stats.csv	1547	1561	0.84	0.83	0.92	0.91
/AKAZE_ORB_MAT_BF_SEL_KNN_stats.csv	1547	1561	0.84	0.83	0.88	0.91
/AKAZE_ORB_MAT_FLANN_SEL_KNN_stats.csv	1547	1561	0.84	0.83	0.88	0.91
/AKAZE_BRISK_MAT_BF_SEL_NN_stats.csv	1547	1561	0.84	0.83	0.84	0.91
/AKAZE_SIFT_MAT_BF_SEL_NN_stats.csv	1547	1561	0.84	0.83	0.92	0.90
/AKAZE_FREAK_MAT_BF_SEL_KNN_stats.csv	1547	1561	0.84	0.83	0.91	0.90
/AKAZE_FREAK_MAT_FLANN_SEL_KNN_stats.csv	1547	1561	0.84	0.83	0.91	0.90
/AKAZE_SIFT_MAT_FLANN_SEL_KNN_stats.csv	1547	1561	0.84	0.83	0.90	0.90
/AKAZE_BRIEF_MAT_BF_SEL_KNN_stats.csv	1547	1561	0.84	0.83	0.89	0.90
/AKAZE_BRIEF_MAT_FLANN_SEL_KNN_stats.csv	1547	1561	0.84	0.83	0.89	0.90
/AKAZE_SIFT_MAT_BF_SEL_KNN_stats.csv	1547	1561	0.84	0.83	0.89	0.90
/AKAZE_AKAZE_MAT_BF_SEL_NN_stats.csv	1547	1561	0.84	0.83	0.88	0.90
/AKAZE_ORB_MAT_BF_SEL_NN_stats.csv	1547	1561	0.84	0.83	0.90	0.89
/AKAZE_BRISK_MAT_BF_SEL_KNN_stats.csv	1547	1561	0.84	0.83	0.88	0.89
/AKAZE_BRISK_MAT_FLANN_SEL_KNN_stats.csv	1547	1561	0.84	0.83	0.88	0.89
/AKAZE_FREAK_MAT_BF_SEL_NN_stats.csv	1547	1561	0.84	0.83	0.85	0.89
/AKAZE_BRIEF_MAT_FLANN_SEL_NN_stats.csv	1547	1561	0.84	0.83	0.48	0.89
/AKAZE_AKAZE_MAT_BF_SEL_KNN_stats.csv	1547	1561	0.84	0.83	0.76	0.88
/AKAZE_AKAZE_MAT_FLANN_SEL_KNN_stats.csv	1547	1561	0.84	0.83	0.76	0.88
/AKAZE_ORB_MAT_FLANN_SEL_NN_stats.csv	1547	1561	0.84	0.83	0.16	0.88
/FAST_FREAK_MAT_BF_SEL_KNN_stats.csv	5270	5563	0.84	0.83	0.65	0.86
/FAST_FREAK_MAT_FLANN_SEL_KNN_stats.csv	5270	5563	0.84	0.83	0.65	0.86
/FAST_SIFT_MAT_BF_SEL_KNN_stats.csv	5270	5563	0.84	0.83	0.81	0.82
/BRISK_FREAK_MAT_BF_SEL_NN_stats.csv	2979	3135	0.84	0.83	0.66	0.82
/SIFT_FREAK_MAT_BF_SEL_NN_stats.csv	1973	2094	0.84	0.83	0.37	0.82
/FAST_SIFT_MAT_FLANN_SEL_KNN_stats.csv	5270	5563	0.84	0.83	0.82	0.81
/SIFT_BRIEF_MAT_FLANN_SEL_NN_stats.csv	1973	2094	0.84	0.83	0.65	0.81
/BRISK_ORB_MAT_BF_SEL_NN_stats.csv	2979	3135	0.84	0.83	0.56	0.81
/SIFT_BRISK_MAT_BF_SEL_NN_stats.csv	1973	2094	0.84	0.83	0.48	0.81
/SIFT_SIFT_MAT_BF_SEL_NN_stats.csv	1973	2094	0.84	0.83	0.16	0.81
/FAST_BRIEF_MAT_BF_SEL_NN_stats.csv	5270	5563	0.84	0.83	0.78	0.80
/SHITOMASI_SIFT_MAT_BF_SEL_KNN_stats.csv	1876	1919	0.84	0.83	0.75	0.80
/SHITOMASI_BRISK_MAT_BF_SEL_KNN_stats.csv	1876	1919	0.84	0.83	0.74	0.80
/SHITOMASI_BRISK_MAT_FLANN_SEL_KNN_stats	1876	1919	0.84	0.83	0.74	0.80
/SIFT_FREAK_MAT_BF_SEL_KNN_stats.csv	1973	2094	0.84	0.83	0.68	0.80
/SIFT_FREAK_MAT_FLANN_SEL_KNN_stats.csv	1973	2094	0.84	0.83	0.68	0.80
/SHITOMASI_SIFT_MAT_FLANN_SEL_KNN_stats.c	1876	1919	0.84	0.83	0.50	0.80

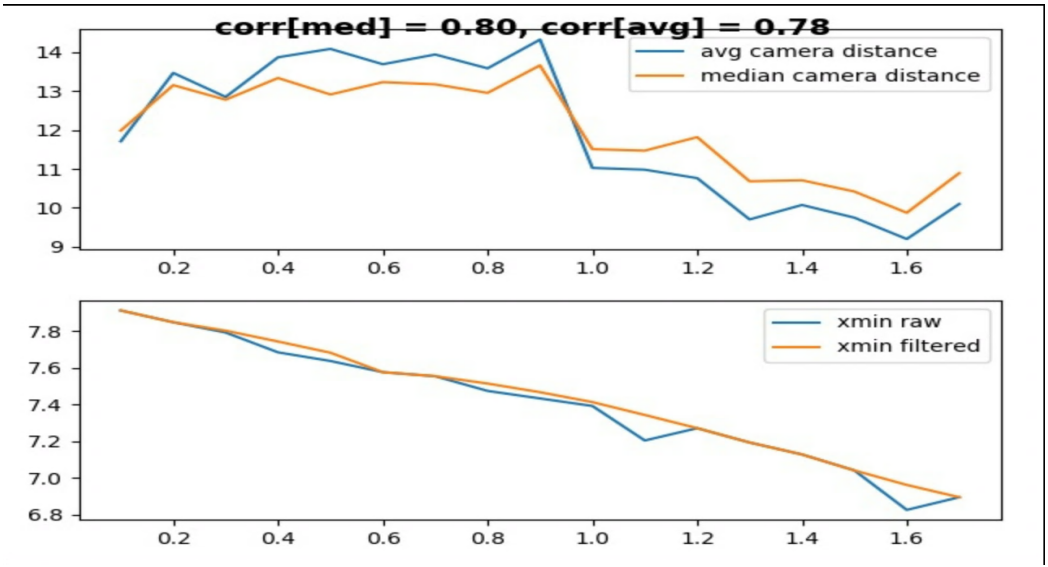
The columns *avglidcorr* and *medlidcorr* correspond to the correlation between "xmin filtered" and TTC estimate using either average and median lidar distances. Similarly, columns *avgcamcorr* and *medcamcorr* correspond to the correlation between the TTC estimate and "xmin filtered" and the TTC estimate using either the average or median distance ratio estimate.

A representative example is shown below for AKAZE detector and descriptors and MAT_BF and SEL_NN. Note that the estimate based on the average ratio was comparable to the median - indicating the the distance ratios were fairly uniform in distribution.



An average performing TTC estimates would look more along the lines of the below, for SIFT/BRIEF/MAT_BF/SEL_KNN combination. The distance ratios would be off, but mostly tracking between the

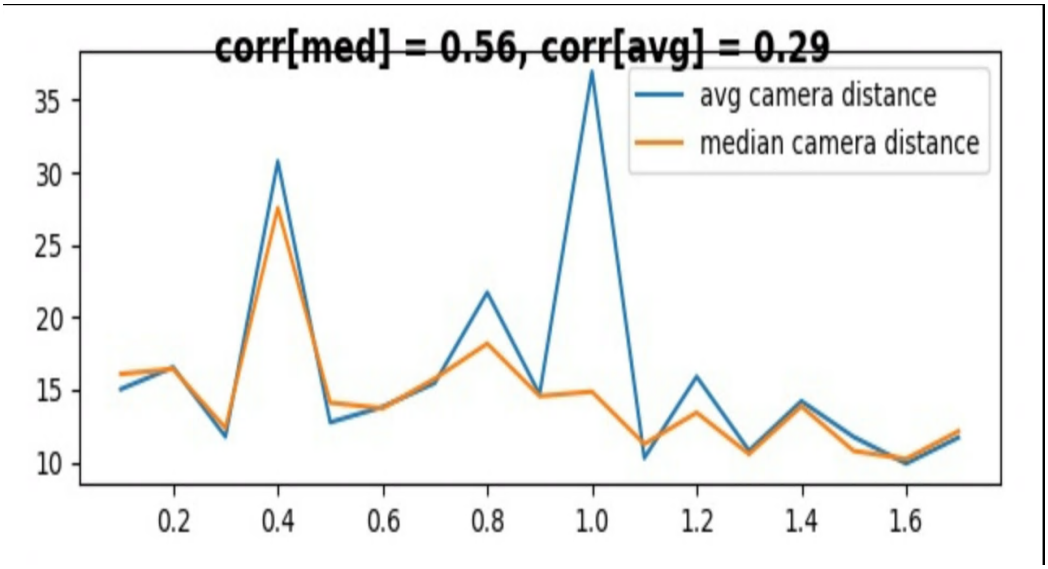
average and median values.



HARRIS, SHITOMASI, and ORB feature based-detector combinations were primarily poor performers. Although HARRIS detectors returned minimal features, and hence matches, SHITOMASI and ORB detectors identified sufficient number of features and still performed poorly indicating that that performance issues were not due to only feature count.

filename	avgkpts	avgmats	avglidcorr	medlidcorr	avgcamcorr	medcamcorr
/home/dan/SFND_3D_Object_Tracking/analysis/HARRIS_SIFT_MAT_BF_SEL_KNN_stats.csv	195	126	0.84	0.83	0.08	-0.00
/home/dan/SFND_3D_Object_Tracking/analysis/HARRIS_SIFT_MAT_BF_SEL_NN_stats.csv	195	126	0.84	0.83	0.08	-0.00
/home/dan/SFND_3D_Object_Tracking/analysis/HARRIS_SIFT_MAT_FLANN_SEL_KNN_stats.csv	195	126	0.84	0.83	0.08	-0.00
/home/dan/SFND_3D_Object_Tracking/analysis/ORB_SIFT_MAT_FLANN_SEL_NN_stats.csv	2000	2000	0.84	0.83	-0.11	-0.02
/home/dan/SFND_3D_Object_Tracking/analysis/HARRIS_AKAZE_MAT_BF_SEL_NN_stats.csv	195	126	0.84	0.83	0.06	-0.04
/home/dan/SFND_3D_Object_Tracking/analysis/HARRIS_ORB_MAT_FLANN_SEL_NN_stats.csv	195	126	0.84	0.83	-0.16	-0.04
/home/dan/SFND_3D_Object_Tracking/analysis/HARRIS_AKAZE_MAT_FLANN_SEL_NN_stats.csv	195	126	0.84	0.83	-0.09	0.09
/home/dan/SFND_3D_Object_Tracking/analysis/HARRIS_SIFT_MAT_FLANN_SEL_NN_stats.csv	195	126	0.84	0.83	0.18	0.11
/home/dan/SFND_3D_Object_Tracking/analysis/HARRIS_BRIEF_MAT_BF_SEL_KNN_stats.csv	195	126	0.84	0.83	-0.12	-0.14
/home/dan/SFND_3D_Object_Tracking/analysis/HARRIS_BRIEF_MAT_FLANN_SEL_KNN_stats.csv	195	126	0.84	0.83	-0.12	-0.14
/home/dan/SFND_3D_Object_Tracking/analysis/HARRIS_AKAZE_MAT_BF_SEL_KNN_stats.csv	195	126	0.84	0.83	-0.15	-0.16
/home/dan/SFND_3D_Object_Tracking/analysis/HARRIS_AKAZE_MAT_FLANN_SEL_KNN_stats.cs	195	126	0.84	0.83	-0.15	-0.16
/home/dan/SFND_3D_Object_Tracking/analysis/HARRIS_BRISK_MAT_BF_SEL_NN_stats.csv	195	126	0.84	0.83	-0.17	-0.18
/home/dan/SFND_3D_Object_Tracking/analysis/HARRIS_BRIEF_MAT_BF_SEL_NN_stats.csv	195	126	0.84	0.83	0.31	0.23
/home/dan/SFND_3D_Object_Tracking/analysis/HARRIS_BRIEF_MAT_FLANN_SEL_NN_stats.csv	195	126	0.84	0.83	0.10	0.25
/home/dan/SFND_3D_Object_Tracking/analysis/HARRIS_BRISK_MAT_FLANN_SEL_NN_stats.csv	195	126	0.84	0.83	0.36	0.34
/home/dan/SFND_3D_Object_Tracking/analysis/HARRIS_FREAK_MAT_FLANN_SEL_NN_stats.csv	195	126	0.84	0.83	0.42	0.41
/home/dan/SFND_3D_Object_Tracking/analysis/SHITOMASI_BRIEF_MAT_FLANN_SEL_NN_stats.	1876	1919	0.84	0.83	0.10	0.42
/home/dan/SFND_3D_Object_Tracking/analysis/ORB_SIFT_MAT_BF_SEL_KNN_stats.csv	2000	2000	0.84	0.83	0.30	0.43
/home/dan/SFND_3D_Object_Tracking/analysis/SHITOMASI_BRIEF_MAT_BF_SEL_NN_stats.csv	1876	1919	0.84	0.83	0.30	0.44
/home/dan/SFND_3D_Object_Tracking/analysis/BRISK_BRISK_MAT_FLANN_SEL_NN_stats.csv	2979	3135	0.84	0.83	-0.25	0.45
/home/dan/SFND_3D_Object_Tracking/analysis/ORB_FREAK_MAT_BF_SEL_KNN_stats.csv	2000	2000	0.84	0.83	0.38	0.45
/home/dan/SFND_3D_Object_Tracking/analysis/ORB_FREAK_MAT_FLANN_SEL_KNN_stats.csv	2000	2000	0.84	0.83	0.38	0.45
/home/dan/SFND_3D_Object_Tracking/analysis/ORB_SIFT_MAT_FLANN_SEL_KNN_stats.csv	2000	2000	0.84	0.83	0.38	0.45
/home/dan/SFND_3D_Object_Tracking/analysis/BRISK_SIFT_MAT_FLANN_SEL_NN_stats.csv	2979	3135	0.84	0.83	0.11	0.46
/home/dan/SFND_3D_Object_Tracking/analysis/ORB_FREAK_MAT_FLANN_SEL_NN_stats.csv	2000	2000	0.84	0.83	0.34	0.48
/home/dan/SFND_3D_Object_Tracking/analysis/AKAZE_AKAZE_MAT_FLANN_SEL_NN_stats.csv	1547	1561	0.84	0.83	0.18	0.49
/home/dan/SFND_3D_Object_Tracking/analysis/ORB_BRIEF_MAT_BF_SEL_KNN_stats.csv	2000	2000	0.84	0.83	0.72	0.49
/home/dan/SFND_3D_Object_Tracking/analysis/ORB_BRIEF_MAT_FLANN_SEL_KNN_stats.csv	2000	2000	0.84	0.83	0.72	0.49
/home/dan/SFND_3D_Object_Tracking/analysis/ORB_ORB_MAT_BF_SEL_KNN_stats.csv	2000	2000	0.84	0.83	0.72	0.49
/home/dan/SFND_3D_Object_Tracking/analysis/ORB_ORB_MAT_FLANN_SEL_KNN_stats.csv	2000	2000	0.84	0.83	0.72	0.49

Poor performing combinations exhibited results like those shown below. Even the use of the median distance ratio exhibited spikes in the TTC estimate - outside what might be reasonably expected (large acceleration/deceleration). In many combinations, the TTC estimate essentially spiked to the max allowed by the algorithm (120 [min]). The use of the average distance ratio also seems to be sensitive to the distribution of distances used in the TTC estimate calculation.



Discussion

Empirically, it appears that the better TTC camera estimates were derived from detectors that focused on the edge of the vehicle in the ROI as opposed to the center (license plate). This largely can be explained as an artifact of the algorithm for TTC with camera data based on the ratios of all distances between feature points. Feature points in the center of the ROI (license plate), might introduce smaller distances that would be more susceptible to pixel quantitization error. The next best performing combinations included FAST, BRISK, and SIFT exhibited a higher percentage of features detected at the outer edge of the vehicle as well.

There seemed to be limited correlation to the descriptor used. In the case of AKAZE, only AKAZE with SIFT descriptors performed outside of the best performing combinations.

Appendix A. Analysis Results By Frame

See stats_full.csv. Table headers are:

```
detector,descriptor,matcher,selector,rem_bb_out,rem_kpt_out,rem_kpt_post_out,keypo
ints,matchpts,proc_time,frame,ttcCamera,ttcLidar,medTtcCamera,medTtcLidar,xmin_raw
,width_raw,xmin_filt,width_filt,time,delta_t
```

The table embeds the analysis conditions, as well as results.

Appendix A. Analysis Summary

The analysis summary results table aggregates the frame data results for each combination of the TTC estimators. Included are average keypoints and match points of the frames, as well as correlation to "xmin filtered" between the TTC estimators on both the average and median values.

See stats_summary.csv. Table headers are:

```
filename,avgkpts,avgmats,avglidcorr,medlidcorr,avgcamcorr,medcamcorr
```