FP.6 Performance Evaluation 2

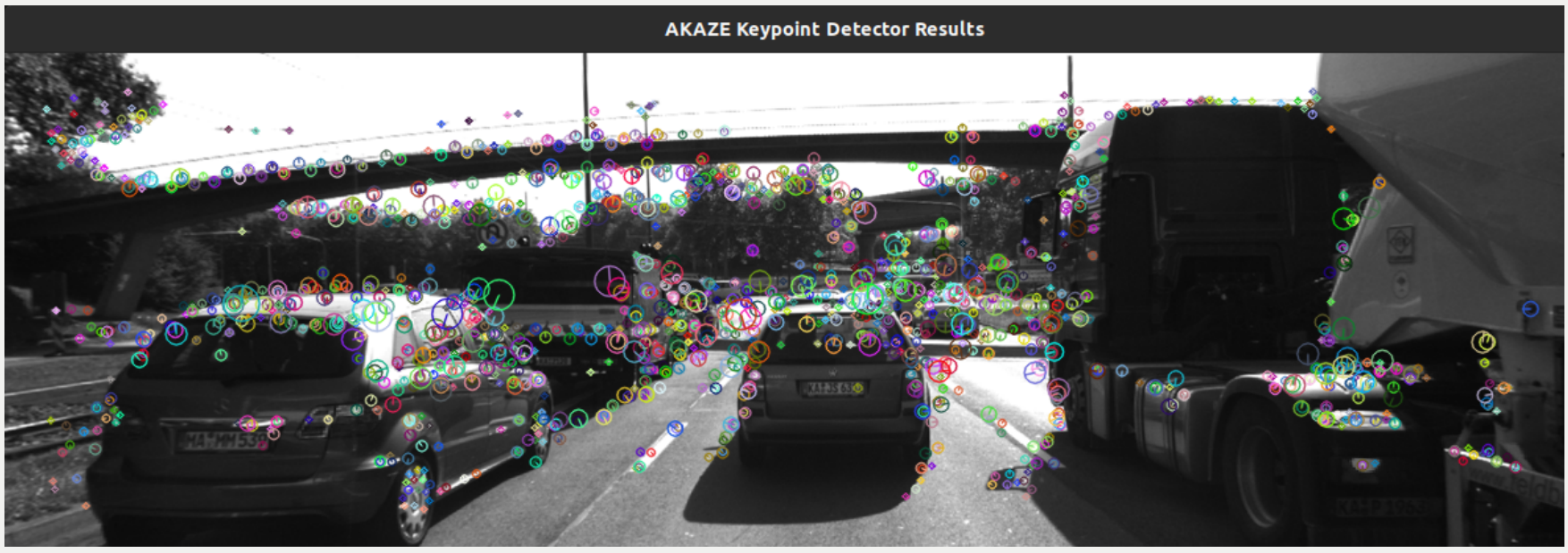
Run several detector / descriptor combinations and look at the differences in TTC estimation. Find out which methods perform best and also include several examples where camera-based TTC estimation is way off. As with Lidar, describe your observations again and also look into potential reasons.

To see all the collected data (full table) see `Camera-TTC-data.ods` file at the base of project directory.

Based on the results below:

* `SHITOMASI` and `AKAZE` keypoint detectors when combine with other descriptors give consistent and smooth (without outlier) TTC predictions that are similar to the ones from Lidar-TTC computation.
* If want to chose one option as best or use it in a system that would be the combination of `AKAZE-SIFT` (has more distance ratio elements for calculating TTC; hence, a bit more robust) **or** `AKAZE-AKAZE` keypoint detector-descriptor.
    * Since, it has high number of matched kpts (results in more robust calculations) and overall resulted in more coherent estimations even when combined with other descriptors.
    * Moreover, the detected keypoints have more variation in size (as shown below) which indicates that it is more robust to variation in scale which is exactly the scenario happening here.

All Keypoints (with AKAZE kpt-detector):



Matched Keypoints used to Match Bounding-Boxes:

As can be seen, `AKAZE` results in more keypoints which are also more diverse and spread on the vehicle ahead. On the other hand we have some bad estimates which you can see in figures below; in all of them we have small number of keypoints with less diversity and not spread on the vehicle ahead which lead to noisy TTC computation (more explanations and observations below).

Some Observations:

***** `AKAZE` descriptor is only compatible with its own keypoint detector.
***** Combination of `SIFT` keypoint detector and `ORB` descriptor results in out of memory error.
***** In some frames, not able to calculate camera-based TTC due to median-distance-ratio being too close to 1 (means no substantial distance change registered between two subsequent frames; the formula is: `camera-TTC = -dt / (1 - medianDistRatio)`)
***** As can be seen, the **camera-based TTC heavily depends on the quantity and the quality of the detected keypoints (especially keypoint detectors that are robust to scale)** and less depends on the descriptor. This suggest that in scenarios that we see poor performance and jumps (up and downs) in TTC computation, it is due to the fact that the detected keypoints are not sufficiently spread on the vehicle ahead which leads to having points close to each other and distance calculation can get noisy (specially if number of detected & matched keypoints are low); here are some examples:

HARRIS-BRIEF (Image 3-4):

FAST-BRISK (Image 4-5):

ORB-SIFT (Image 3-4):

Correlations Table:

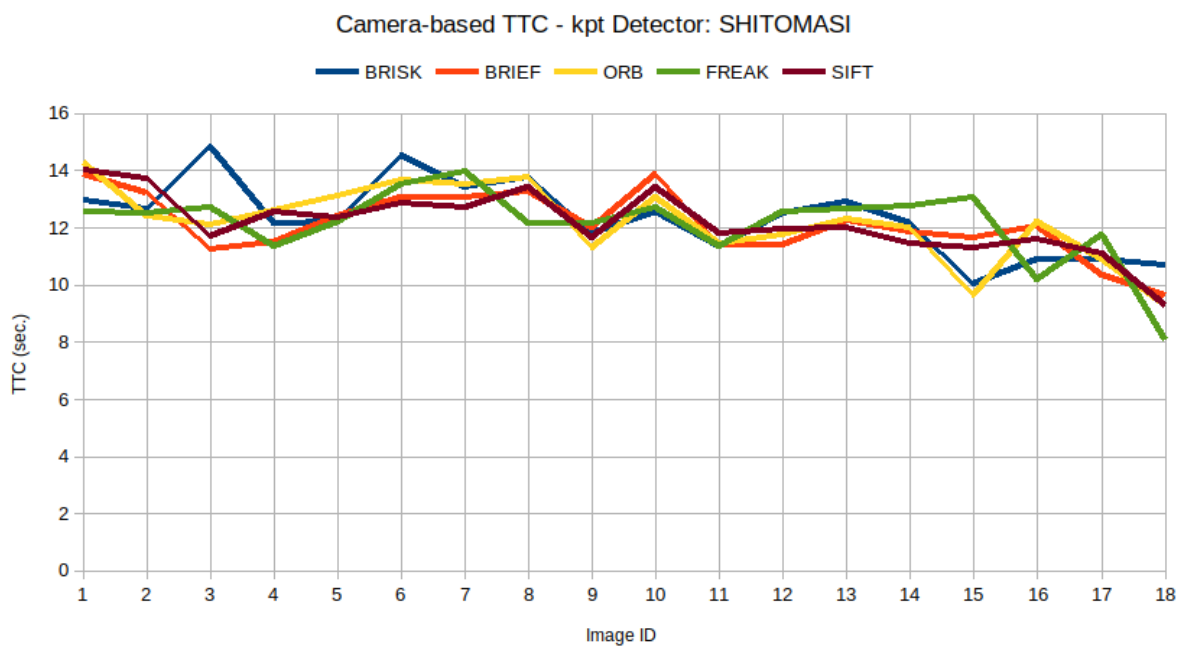
As can be seen in the table below, the first one shows correlation between `TTC` , `meanDistChange` (i.e.: how internal distance between keypoints in one frame changes in the next frame) and `distRatioVecSize` (i.e.: number of distance ratio elements used to calculate the TTC)

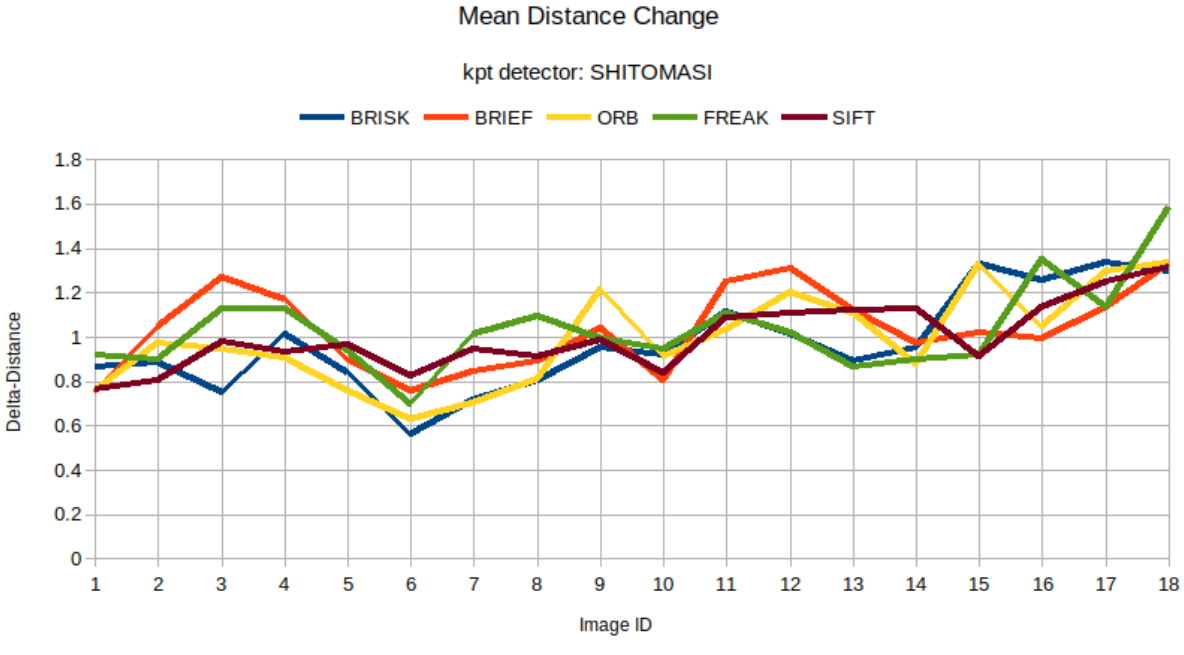
As it was expected, we have strong negative correlation between TTC and `meanDistChange` which suggest that as `meanDistChange` increase, `TTC` decreases. We also this that where we have poor performance (noisy TTC estimation) we have weaker correlations. Finally, we don't see any obvious correlation between TTC and the remaining metrics.

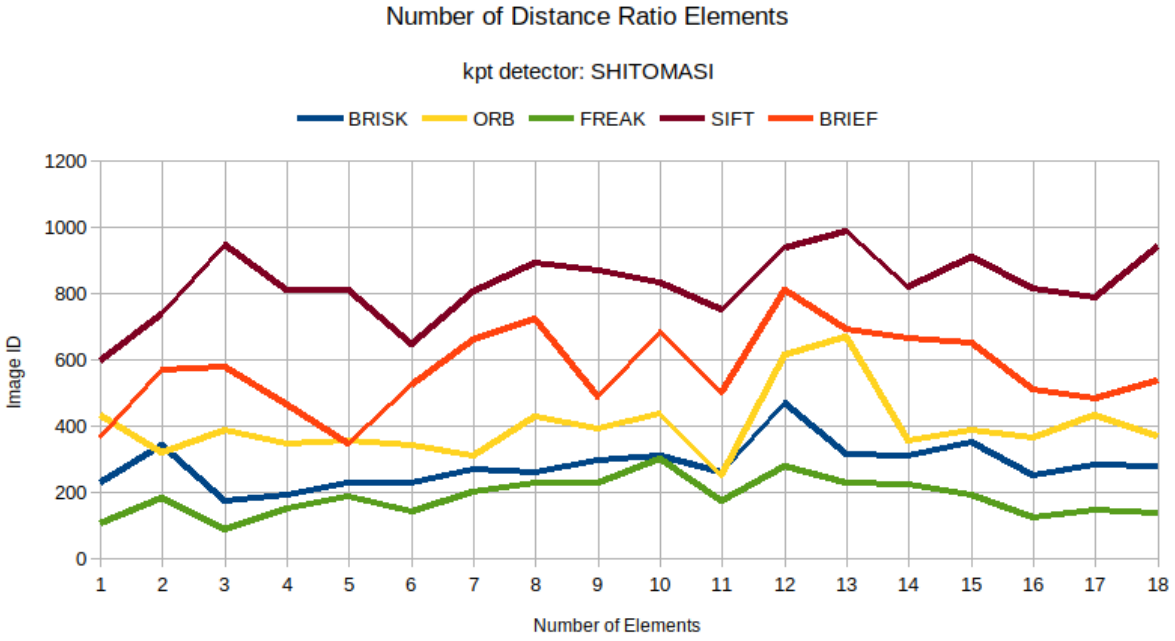
Summary of Results:

detectorType	Correlation (TTC vs meanDistChange)	Correlation (TTC vs stddevDistChange)	Correlation (TTC vs distRatioVecSize)
SHITOMASI	-0.8842	-0.1640	-0.0583
HARRIS	-0.7610	-0.2711	-0.1713
SIFT	**-0.9443**	0.1710	0.0473
FAST	-0.2566	0.0089	-0.1519
BRISK	-0.8354	-0.1557	-0.2861
ORB	-0.4501	0.3070	0.3097
AKAZE	**-0.9340**	-0.3710	-0.1215

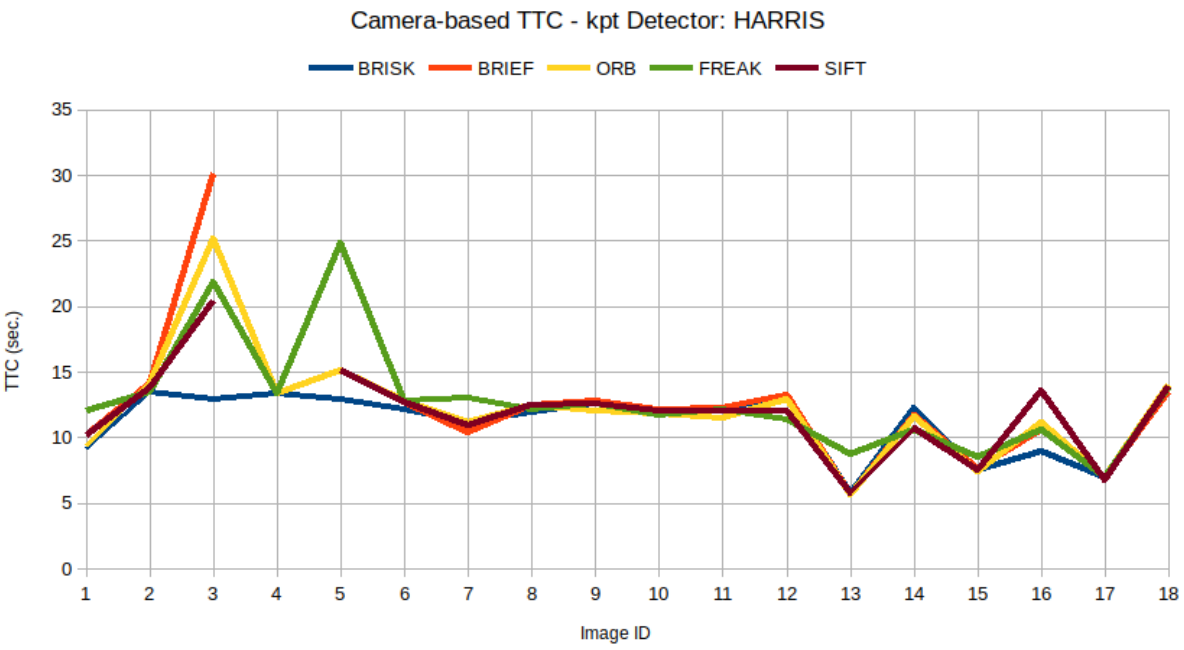
SHITOMASI (kpt detector)

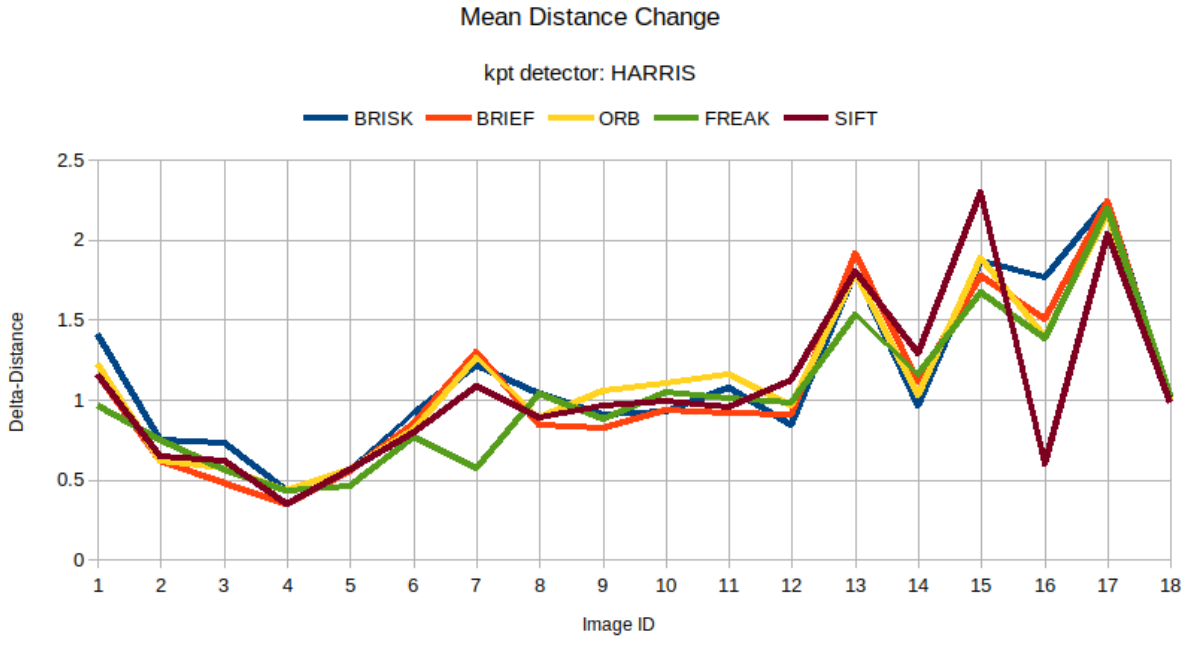


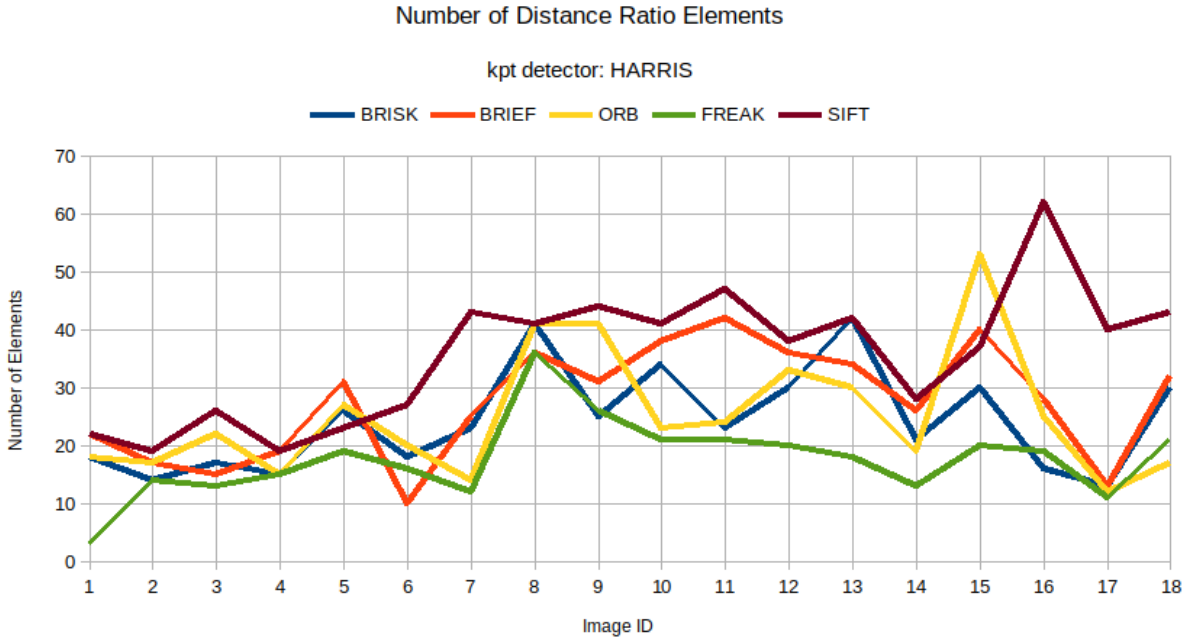




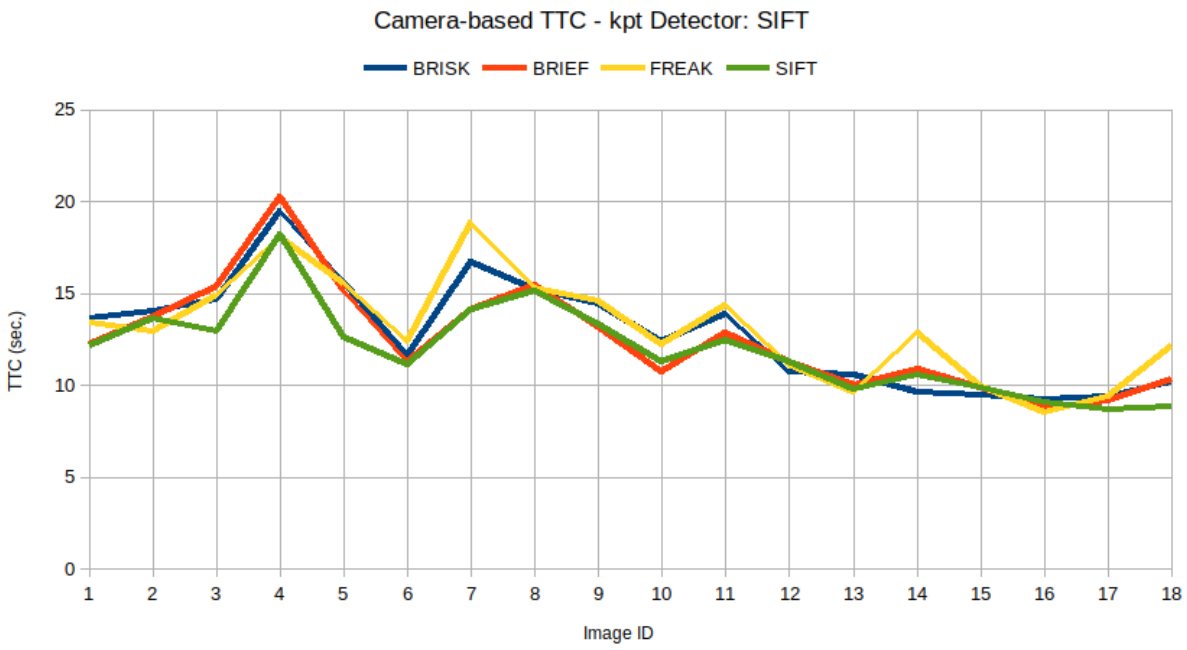
HARRIS (kpt detector)

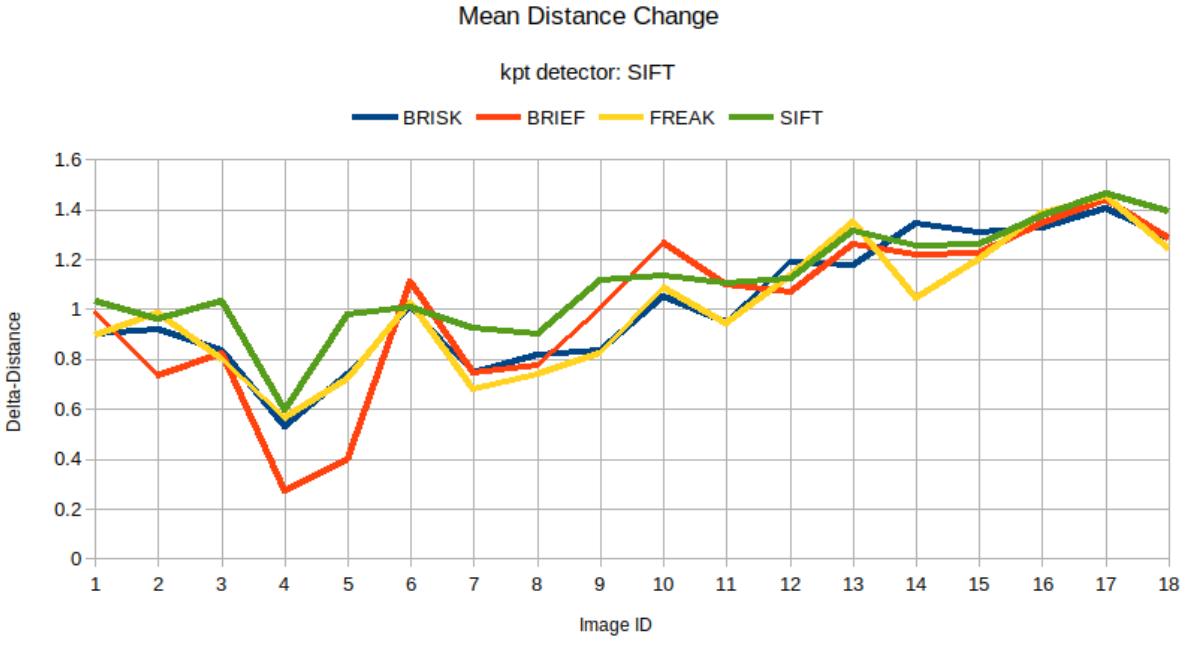


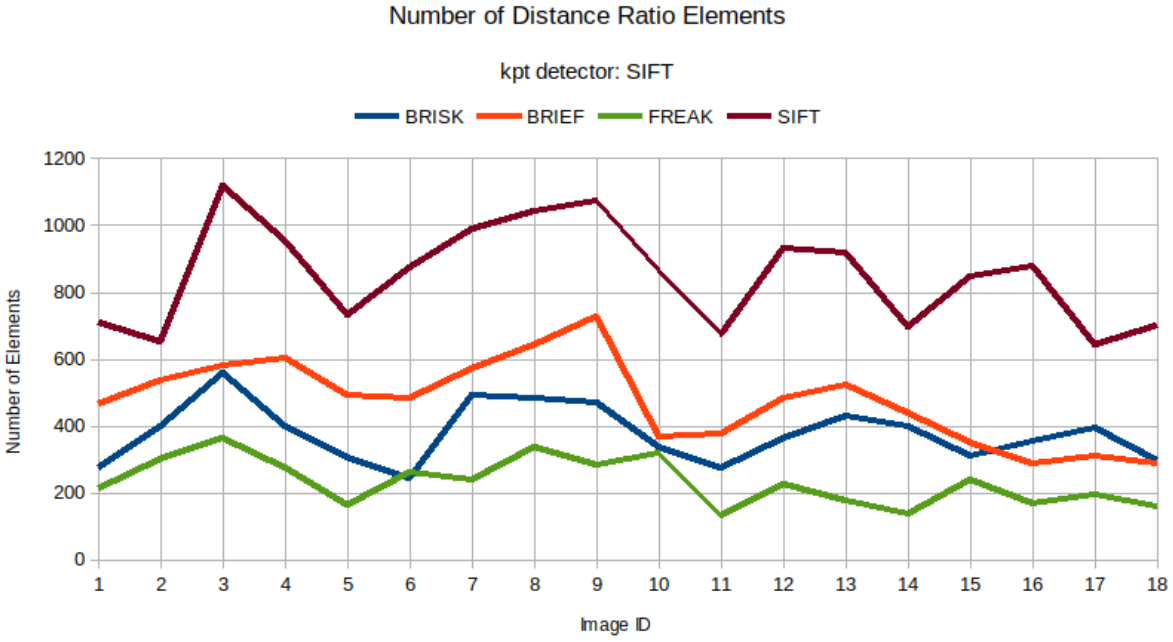




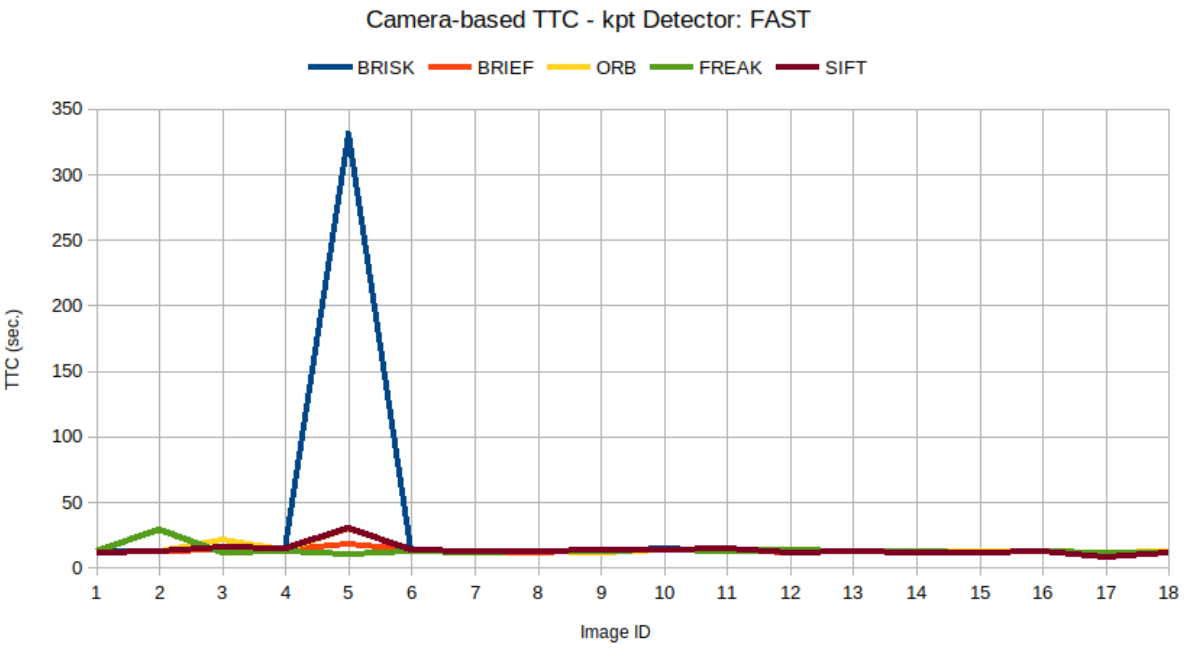
SIFT (kpt detector)

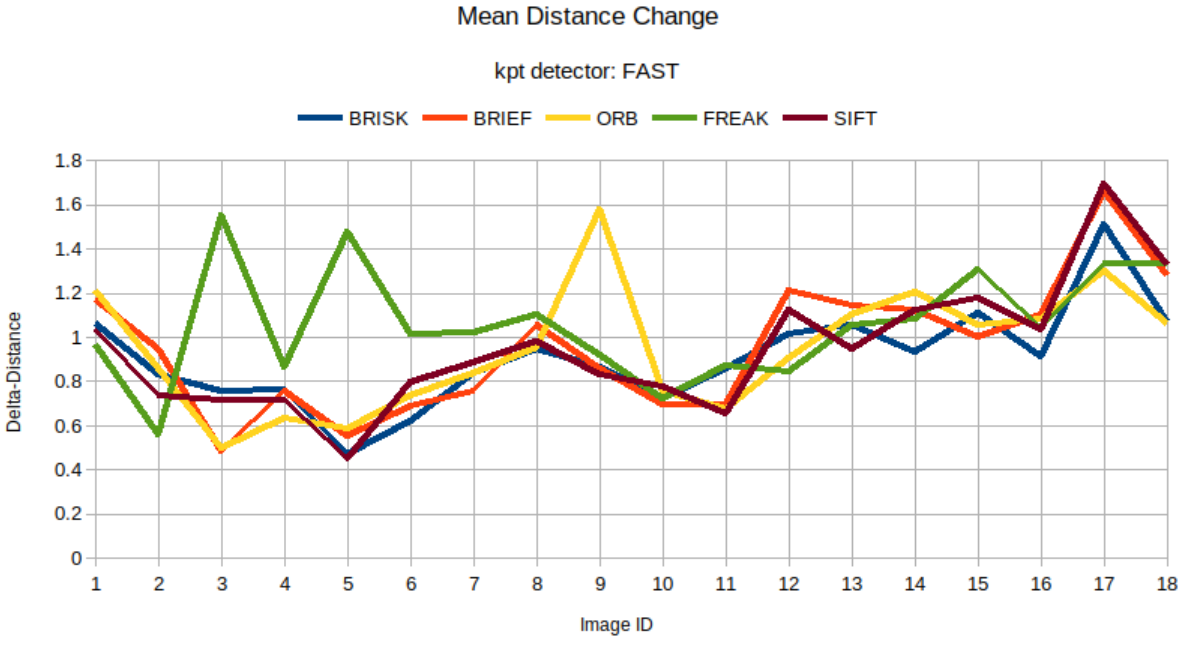


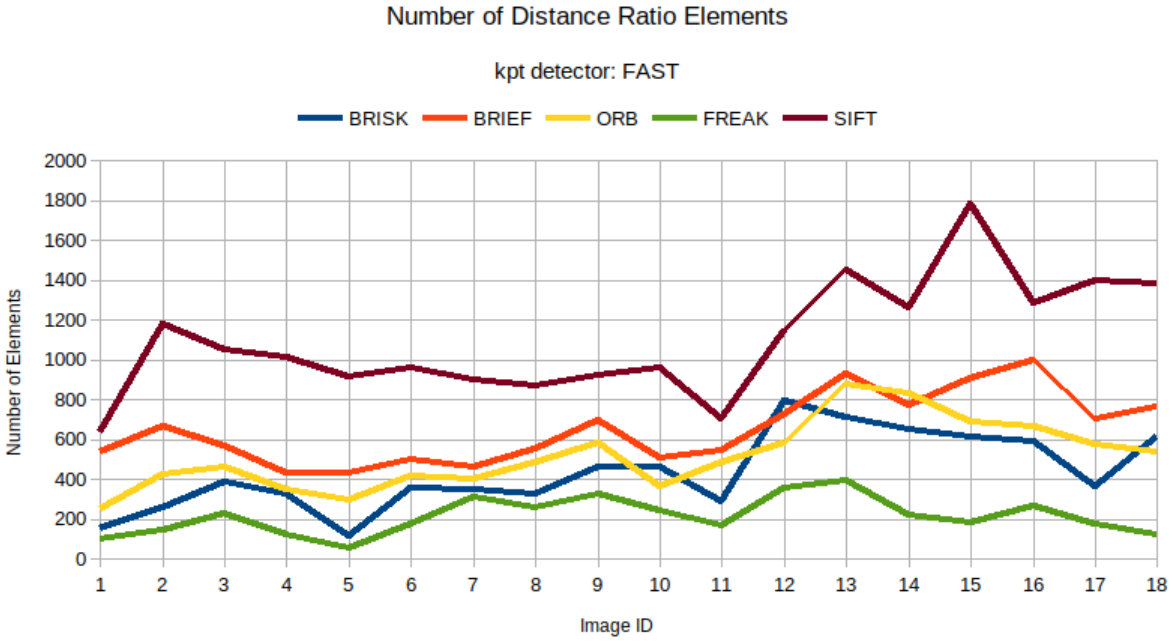




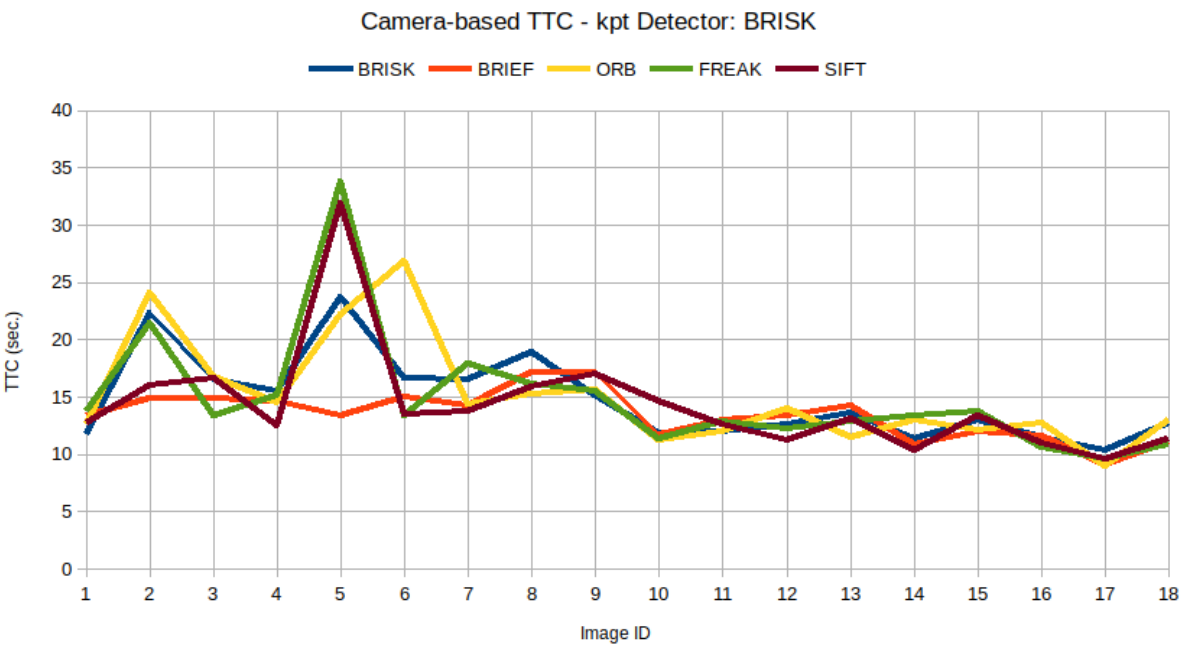
FAST (kpt detector)

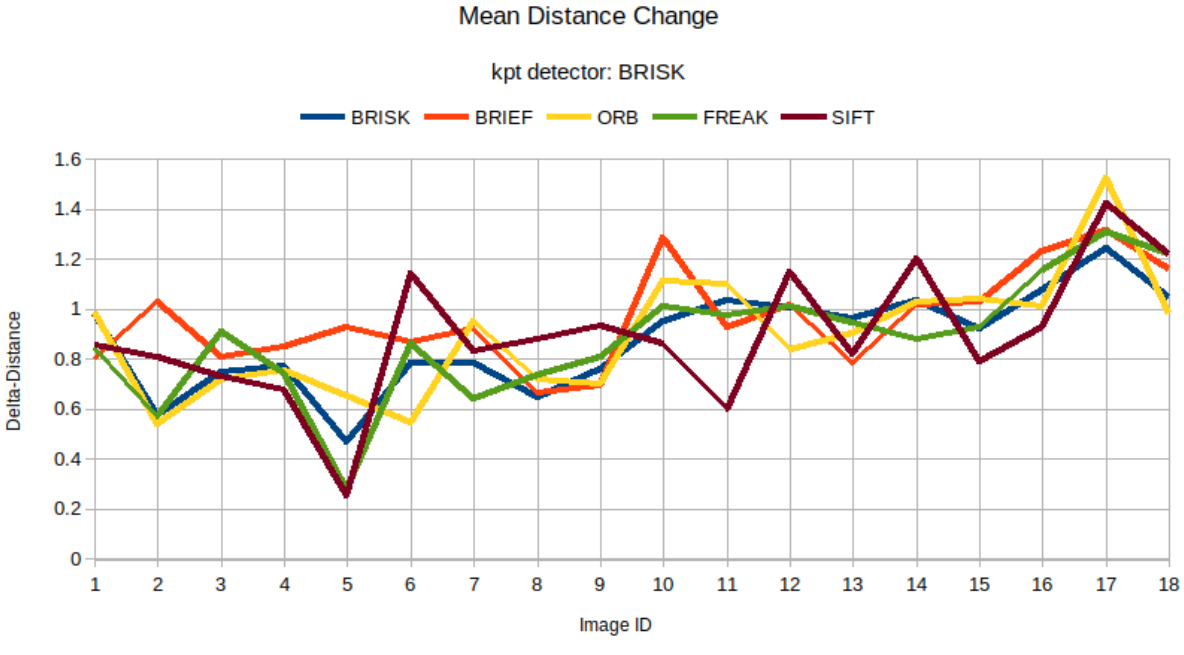


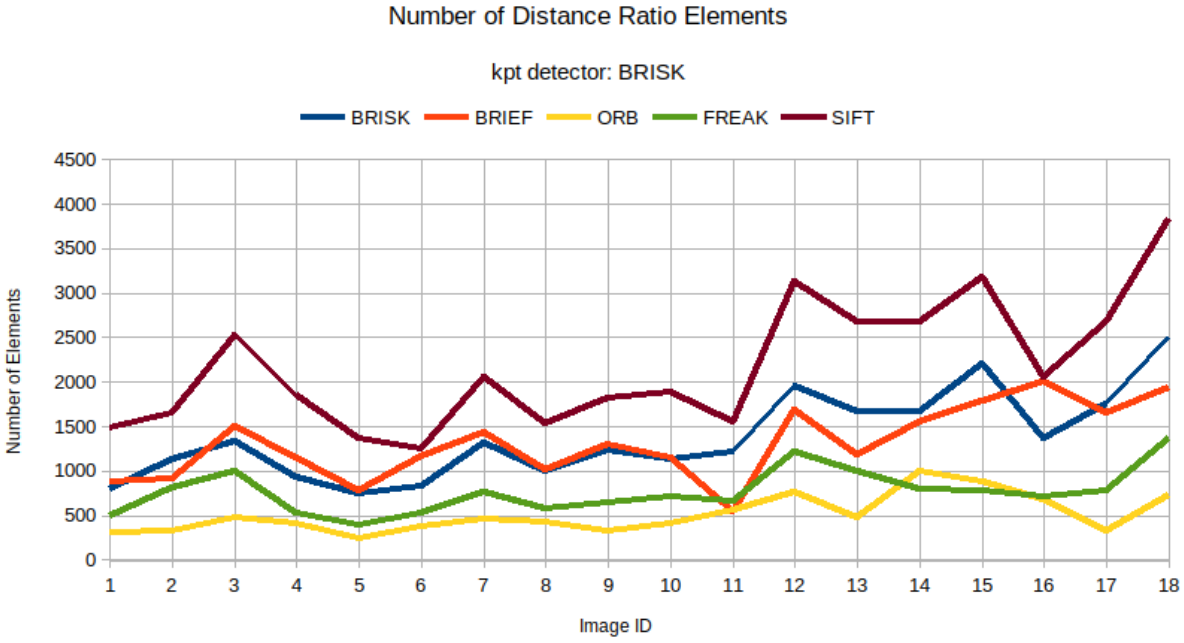




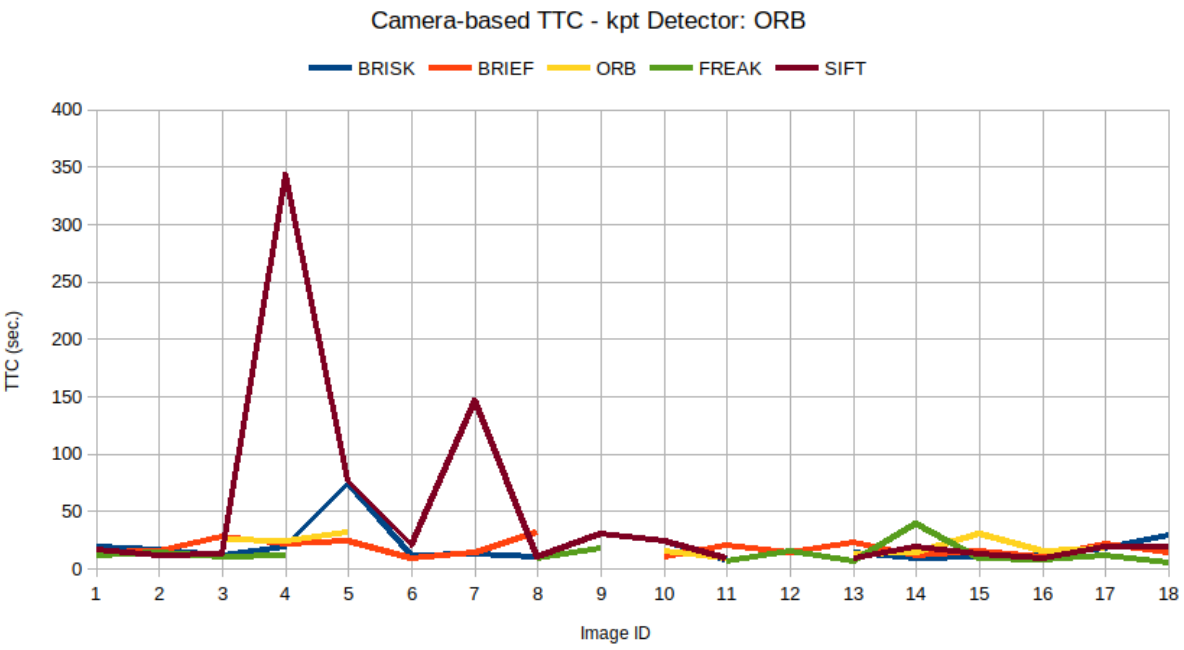
BRISK (kpt detector)

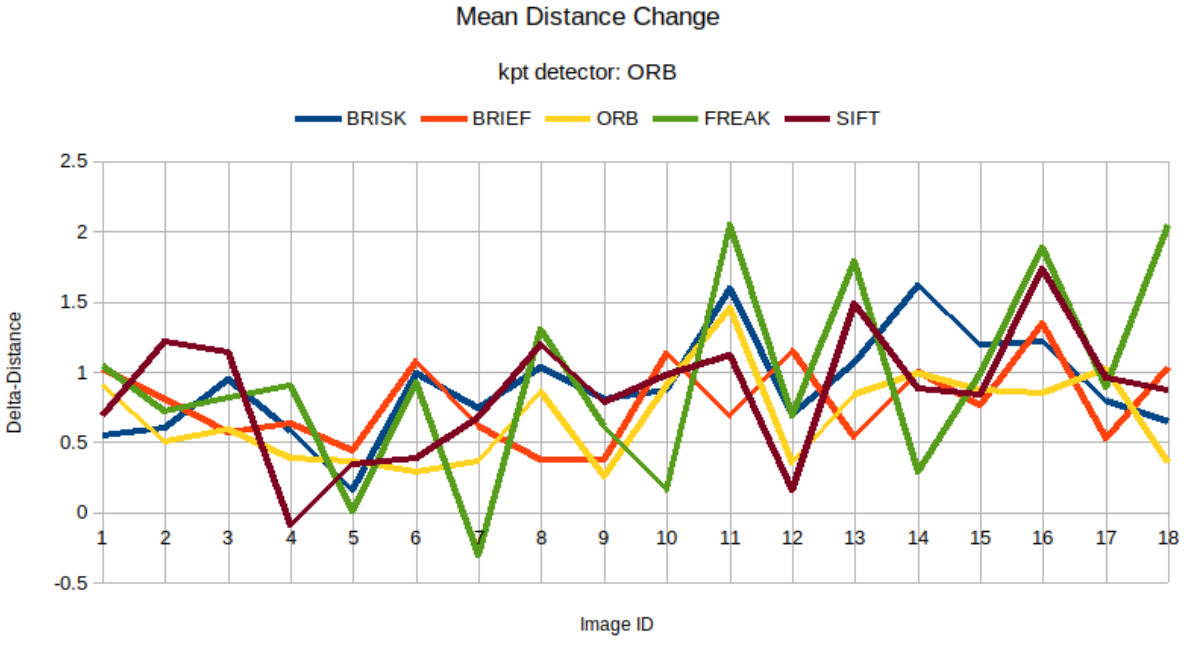


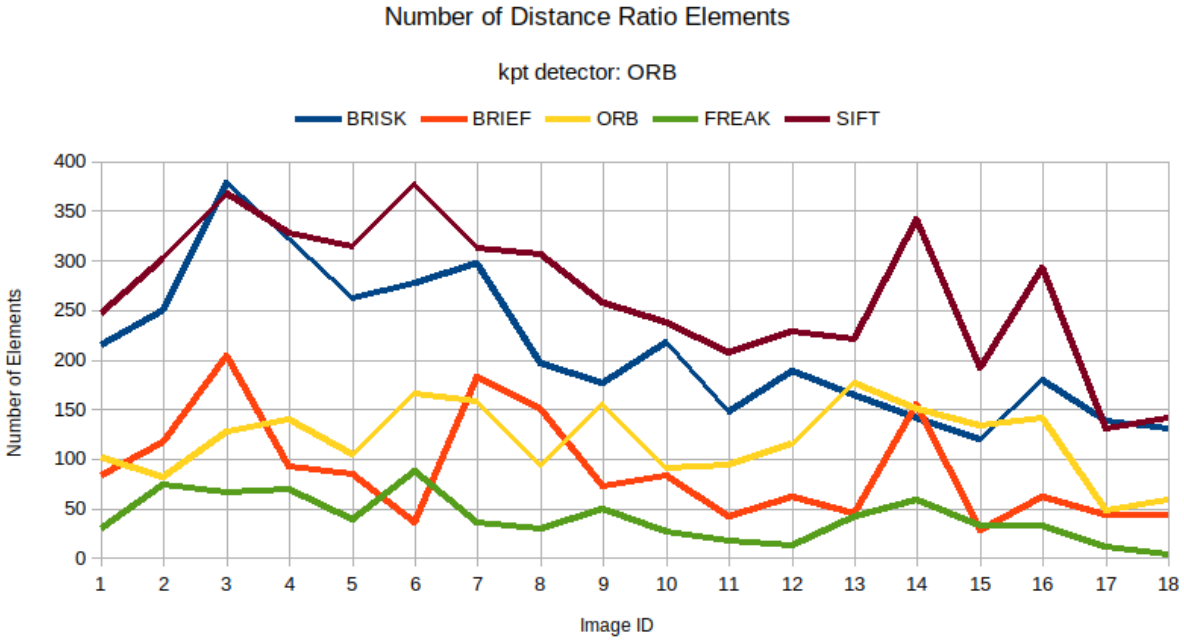




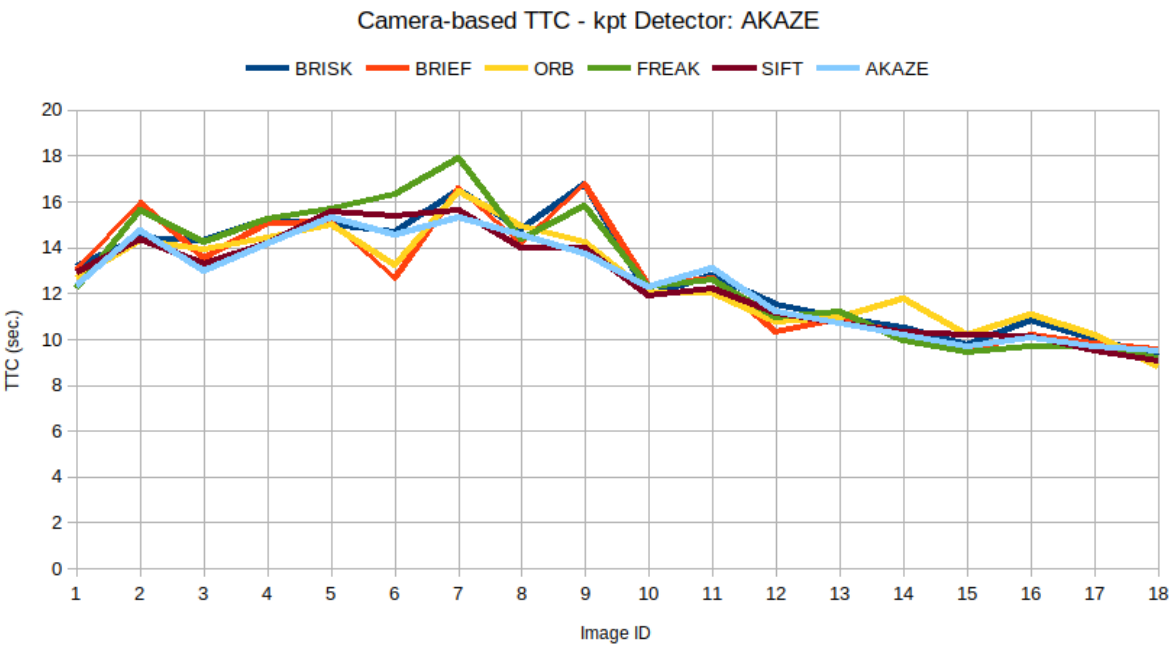
ORB (kpt detector)

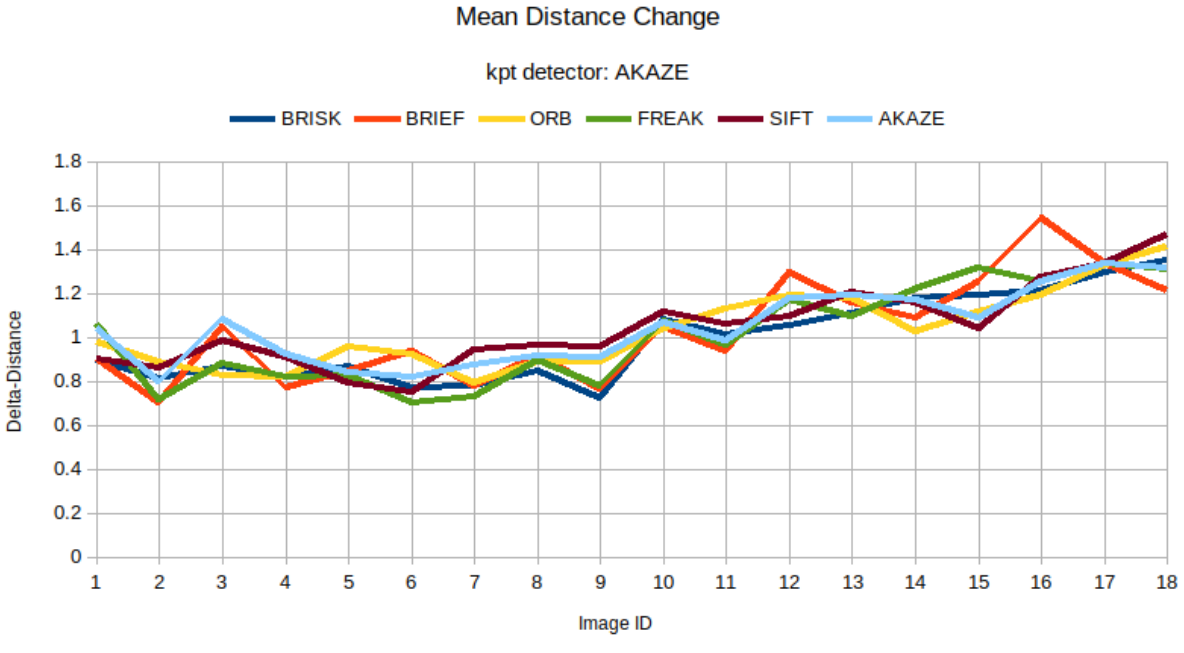


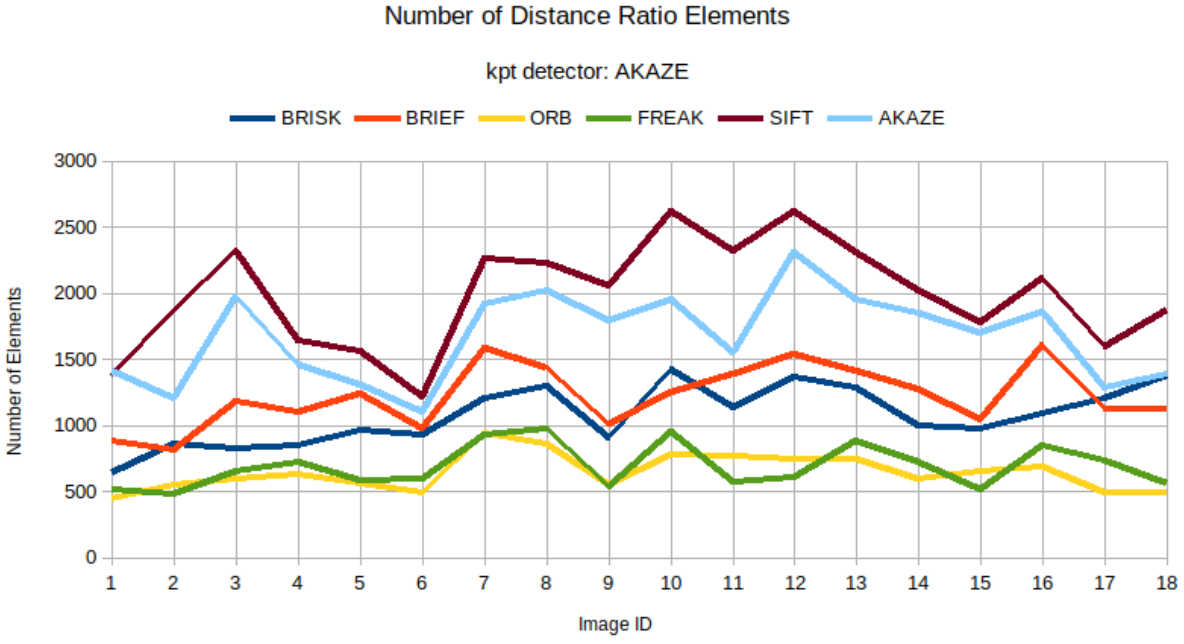




AKAZE (kpt detector)




Mean Distance Change
kpt detector: AKAZE
BRISK
BRIEF
ORB
FREAK
SIFT
AKAZE
Delta-Distance
Image ID


Number of Distance Ratio Elements
kpt detector: AKAZE
BRISK
BRIEF
ORB
FREAK
SIFT
AKAZE
Number of Elements
Image ID