FP.1

I Implemented the function matchBoundingBoxes. It works by making a map of maps, that counts the connections between boxes in the previous frame, and the current frame. The map is structured as prevFrameId → nextFrameId → counter. I later go over the connections and take the one that has the highest counter, and consider that a match.

FP.2

I compute the TTC by taking two indicative points. One from the previous LidarPoints and one from the CurrentLidar points. The way I decide on an indicative point is by first finding the standard deviation of X values, and choosing the point with the lowest X value that is at-most 0.67 standard deviations away from the mean. The 0.67 threshold was experimentally chosen.

FP.3

I iterate over all the matches and see if they fall inside the bounding box both in the current and the next frame. If this is true I append them to a list. I then go over the list and find the mean distance between keypoints, and remove all entries which have a distance over the mean.

FP.4

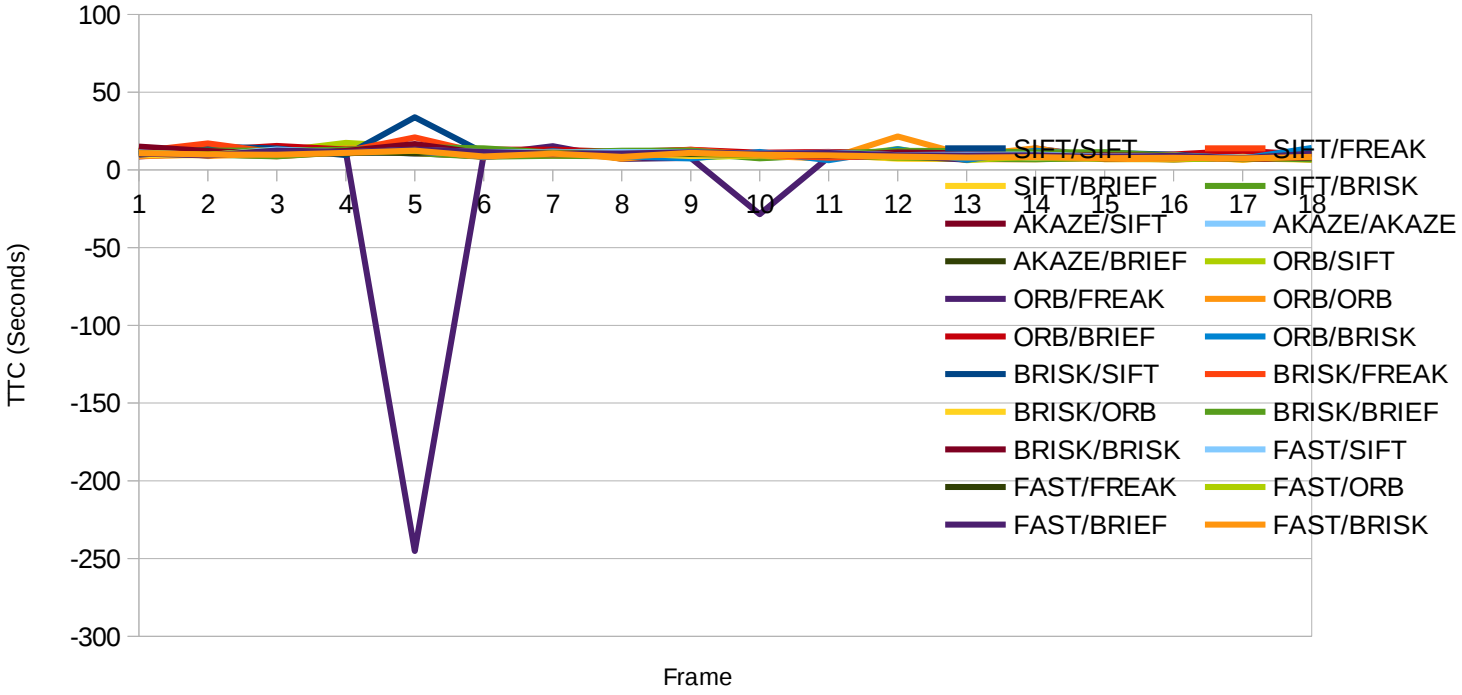
I compute the TTC by first computing the difference between all keypoints in the previous frame, and all keypoints in the current frame. I then take the ratios between the difference pairs in the previous and current pair. I remove all pairs which have a ratio of 1, because that cannot happen unless the distance to the car in front did not change. I then use the median of all remaining ratios to calculate the TTC.

FP.5.1

TTC Lidar 12.9436 seconds Velocity 0.619998 8.025 7.963 10
TTC Lidar 11.8851 seconds Velocity 0.669999 7.963 7.896 10
TTC Lidar 19.2586 seconds Velocity 0.409999 7.896 7.855 10
TTC Lidar 14.5463 seconds Velocity 0.539999 7.855 7.801 10
TTC Lidar 13.222 seconds Velocity 0.59 7.801 7.742 10
TTC Lidar 10.9042 seconds Velocity 0.710001 7.742 7.671 10
TTC Lidar 14.4736 seconds Velocity 0.53 7.671 7.618 10
TTC Lidar 14.3736 seconds Velocity 0.53 7.618 7.565 10
TTC Lidar 14.2736 seconds Velocity 0.53 7.565 7.512 10
TTC Lidar 13.1789 seconds Velocity 0.570002 7.512 7.455 10
TTC Lidar 10.65 seconds Velocity 0.699997 7.455 7.385 10
TTC Lidar 11.1893 seconds Velocity 0.660005 7.385 7.319 10
TTC Lidar 8.71313 seconds Velocity 0.839996 7.319 7.235 10
TTC Lidar 10.7985 seconds Velocity 0.669999 7.235 7.168 10
TTC Lidar 8.23905 seconds Velocity 0.870004 7.168 7.081 10
TTC Lidar 9.19614 seconds Velocity 0.769997 7.081 7.004 10
TTC Lidar 11.2967 seconds Velocity 0.620003 7.004 6.942 10
TTC Lidar 7.97932 seconds Velocity 0.869999 6.942 6.855 10

Some anomalous results in the lidar TTC are results 3 and 7 because the TTC seems to jump up then down. Looking at the overhead view of those TTC readings we notice that they are more noisy and have more variance in the x values. When we have more noise, and higher X variance, the value chosen to represent that cluster after filtering will tend to be lower. So when we have noisy then non-noisy frames the distance between the two frames will oscillate causing the velocity and TTC estimates to also oscillate.

FP.5.2



The source for the graph can be found in graph.pdf. Overall all the detectors/descriptors performed similarly. Some combinations such as BRISK/SIFT had some extreme values for some frames. I think this happens because of low keypoint counts which makes small amounts errors in keypoint matching have big effects on the TTC. The combination that has the most issues is ORB/FREAK because it has negative values.
Because we don't have any ground truth values its hard to say which combination performed the best. I would choose ORB/SIFT because it seems to match the Lidar results the best.