FP.1 Match 3D Objects

In **matchBoundingBoxes** fn, we go through each keypoint match and find out in which bounding box in the previous and current image is the keypoint in. The bounding boxes which contains the keypoints can be paired to each other as the keypoints they contain are already paired.

FP.2 Compute Lidar-based TTC

In **computeTTCLidar** fn, find the average x distance of all lidar points in the boundingbox in the current and previous image. Check which point has x coordinate is less than 0.75*avgX. Use the below formula to calculate TTC

TTC = (minXCurr *dT)/ (avgPrevX -avgCurrX)

FP.3 Associate Keypoint Correspondences with Bounding Boxes

In clusterKpMatchesWithROI, look through each match and check if the train point within the match is contained within the boundingbox. If it is contained within the bounding box push the match into posKPMatchBBox and find the avg distance between the point of the match in the current image (train_pt) and the point in the previous image (query_pt). If it within the thresshold, add it to the boundingbox.

FP.4 Compute Camera-based TTC

In computeTTCCamera, loop through all keypoints and find the distance ratios for all keypoints between curr and prev frame. Find med dist ratio and then calculate TTC using the following formula

TTC = -dt/(1 - medDistRatio)

FP.5 Performance Evaluation 1

When using detectors that provide only few key points or descripts which provide only few matches, the TTC from the Lidar and TTC from Camera do not match. This can be seen when using ORB and HARRISOn detectors which produce a small detector count when compared to the other detectors.

Having more keypoints result in more matches and better estimates.

FP.6 Performance Evaluation 2

Check file FP.6 "Performance Evaluation 2.pdf" contains tabulated comparisons