Udacity 3D-Camera Final Project

Brandon Marlowe

2019-10-10

FP.1 : Match 3D Objects

The `matchBoundingBoxes` function iterates through all points found in the `matches` vector, and determines if the point in question exists in the previous and current frames. If this is true, the count of the occurrence is updated within a temporary vector named `listOfMatches`. Otherwise, the loop continues on to the next point in the `matches` vector. Once all points in the `matches` vector have been compared, another `for` loop is used to iterate through all points in the tempoary vector, and store the best points in the `bbMatches` map.

FP.2 : Compute Lidar-based TTC

The time to collision (TTC) was calculated using the following formula:

```
TTC = (minXCurr * dT) / (averageXPrev - averageXCurr);
```

The `minXCurr` is distance of the closest lidar point from the preceding vehicle. The `averageXCurr` and `averageXPrev` are the average distances of lidar points in the current and preceding frames, respectively.

Originally, I did attempt to use the formulas mentioned in previous lessons, however, the results were supbar.

FP.3 : Associate Keypoint Correspondences with Bounding Boxes

The keypoint matches are clustered and associated with bounding boxes in the `clusterKptMatchesWithROI` function. A `for` loop iterates through all points in the `kptMatches` vector, checks if the point exists in the given bounding box, and if so, stores it in a vector. Next, the average distance of all points found in the bounding box is calcuated. Finally, another `for` loop again iterates through

all the points in the `kptMatches` vector, checks if it's again within the bounding box, and if the current point falls within a distance threshold. If so, the point is stored in that bounding box's keypoint matches.

FP.4 : Compute Camera-based TTC

The formula used in the `computeTTCCamera` function was adapted from the previous lessons. The TTC was calculated by iterating through all points in the `kptMatches` vector, and each of those points to all other points in the same vector using an inner `for` loop.

The formula used to calculate the TTC is as follows:

```
TTC = -dT / (1 - medDistRatio);
```

The median distance was used to remove any outlier influence.

Performance Evaluation

In some cases, the TTC Lidar estimation very obviously inaccurate, and this occurred when the number of points detected by the detector and descriptor combination detected very few points. Having a small number of points to base measurements off would explain poor accuracy in timing estimation. The happened when the Harris detector was used in combination with other descriptor type.

A spreadsheet containing the final results can be found here:

`report/Brandon_Marlowe_Final_3D_Camera_Project.csv`.