Car Tracking Project Report

Rishabh Biyani
Saimouli Katragadda

May 16, 2017

List of Figures

1 Pipeline

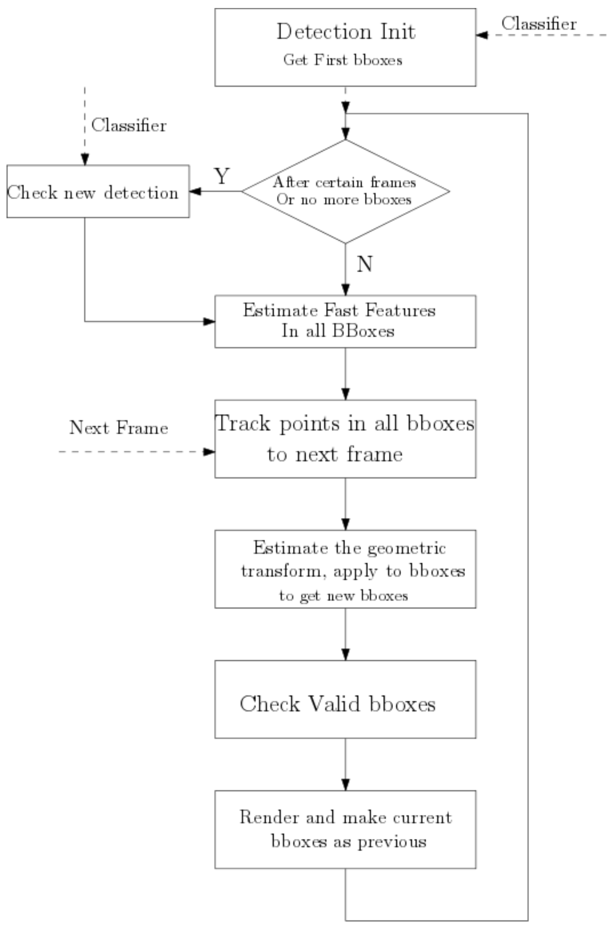

Figure 1: Pipeline used for Car Tracking

This project an attempt was made to detect and track cars as viewed from a car for a self-driving car application. Pipeline can summarized shortly as shown in figure 1. The very first step is to train a classifier. A cascade classifier was used along with HOG features to get a Car-Detector. The task initially is to detect cars in a particular frame and then track it in the next frames until tracking leads to reasonable results or the detector should be run again to detect new possible cars that might have entered. Thus, in the first step we get initial detections and bounding boxes corresponding to these detections. This actually initializes our tracking algorithm. Then we estimate **FAST** features in each of these bounding boxes which are passed to the point tracker, that tracks these points to the next frame. Using the correspondence of these points between the current and the previous frame we compute a affine transformation and apply the transform to the corners of the bounding box to obtain the bounding box in the new frame. This transformation might not be correct depending on whether the car is really visible in the next frame - this might lead to false transformation on bounding boxes that doesn't represent the car. We filter such bounding boxes based on aspect ratio, distance threshold in the next step. Finally, we render this boxes on the current frame and make bounding boxes detected in the current frame as precursor for the next frame. The following sections describe each of the components in the pipeline briefly.

2 Classification

Cascade Classifier along with HOG features was used for classification. There were quite a few issues with training the Cascade Classifier in R2016a. One of the reasons being that function required inputs in a specific format. **getPositiveInstances.m** is the script which constructs the required struct of positive instances as required by the function **trainCascadeObjectDetector**. Also, even after several trials classifier did not produce acceptable results for detection. The false positives were plenty. To allay this problem, the amount of positive and negative samples were increased. Positive samples were increased by taking its mirror image (making it twice the size). Negative samples were increased by taking its mirror image, water image and mirror image of its water image, thereby increasing negative samples by three times. **flipPositiveSamples.m** and **Increasenegativesamples.m** are two scripts to do this. After this operation, only the 'far' and 'middle' folders were used as positive instances. This gave good enough detection for the simple video.

3 Detection

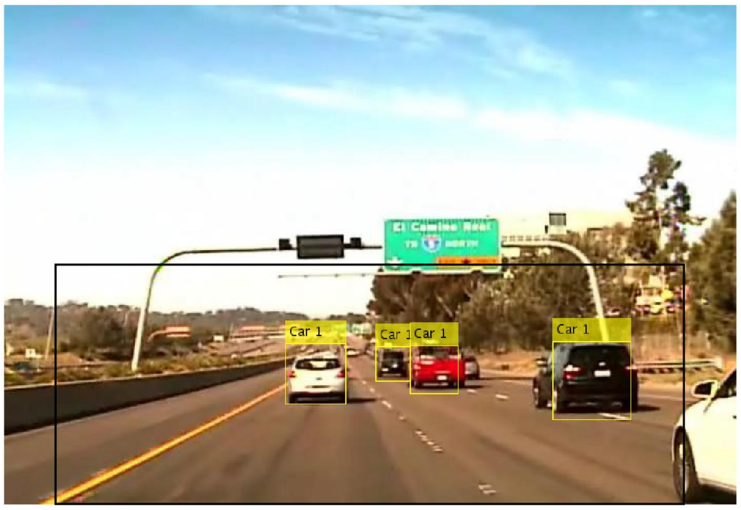

Figure 2: Detection carried out in the region specified by the black rectangle to minimize false positives

Initially, a region of interest was chosen to run the detector on. This is illustrated in figure 2. Then, some easy false detections were removed based on max_area, min_area of the bounding box rectangle. To further make the detection robust, the Detection was carried out by taking support of the current frame and the next frame. This means that the detector was run on both the current and the next frame and the bounding boxes obtained in each of these frames were compared. The comparison was done based on the distance between the centroid of the two bounding-boxes(whether each bbox in current frame has a counterpart in the next frame) and ratio of the corresponding bounding boxes. The threshold values for this were tuned by observation across few frames. These operations are performed in **filter_bbox.m**

Another aspect of detection comes when we want to check whether there are any new detections in between tracking. For this part we run the detector on the current frame and compare the obtained bounding boxes with the existing bounding boxes(obtained from point tracking). To confirm that we have a new detection we again check the distance between the centroids of the bounding boxes and also their overlap ratio using the inbuilt function **bboxOverlapRatio**. Again, by tuning the thresholds, new detections if any are captured appropriately. The detector is constrained to run after every **N** frames. This checking for new detection happens in **check_for_new_detection.m**

4 Tracking and Geometric Transform

pointTracker is the inbuilt function based on KLT tracking algorithm used to track the features within the bounding box to the next frame. **estimateGeometricTransform** is another inbuilt function used to estimate the affine transformation using the point correspondences and transform the bounding box to the next frame.

5 Checking the Validity of Bounding Boxes

Again, the transformed bounding boxes were checked for its validity using properties such as the aspect ratio of the transformed bounding box and the ratio between the areas of the current bounding boxes to the previous bounding boxes. Only, Bounding boxes passing these checks are kept and are passed for rendering.

6 Rendering

To show the successful tracking the bounding boxes corresponding to the same car across frames has the same colour. Also, they are labelled with a number. Every time a new tracking is obtained we initialize it with a different colour and a global car counter number. Currently, even if a tracking is lost in a frame and the same car is tracked again in subsequent frames we render it with different colour and number. This is certainly a bug and will need to be addressed.

7 Conclusion

Car Tracking performance was fair. In the output video, one or two frames from a total of 300 frames there were false detections which even the filtering techniques mentioned were not able to isolate. Also, the tracker looses the car for a couple of frames and then gets its

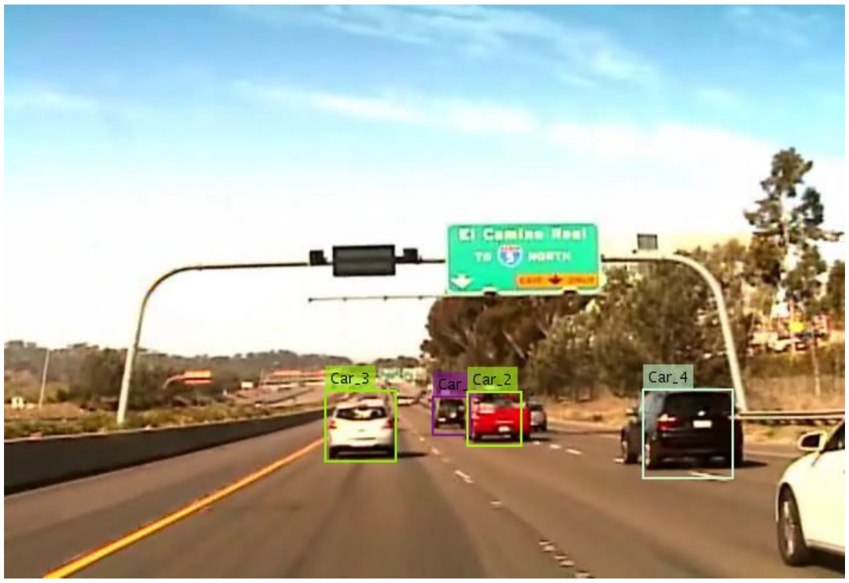

Figure 3: One of the output frames from the simple video

back because of the detector. There were few instances when more than two bounding boxes were detected on the same car. At this point we only take the outer bounding box. But, this is sort of a hack. The author certainly believes that a better training and a better detector will produce better results.