Vehicle Detection And Tracking

The goals / steps of this project are the following:

- Perform a Histogram of Oriented Gradients (HOG) feature extraction on a labeled training set of images and train a classifier Linear SVM classifier
- Optionally, you can also apply a color transform and append binned color features, as well as histograms of color, to your HOG feature vector.
- Note: for those first two steps don't forget to normalize your features and randomize a selection for training and testing.
- Implement a sliding-window technique and use your trained classifier to search for vehicles in images.
- Run your pipeline on a video stream (start with the test_video.mp4 and later implement on full project_video.mp4) and create a heat map of recurring detections frame by frame to reject outliers and follow detected vehicles.
- Estimate a bounding box for vehicles detected.

Feature Extraction

Features are needed to train a classifier and make predictions on the test or real-world images. The project required to build classifier which would be able to answer if there is a car in image or not. Three features have been chosen(histogram of oriented gradients, color histogram and binned color(for shape and color features). The combination of these three features provided enough information for accurate image classification.

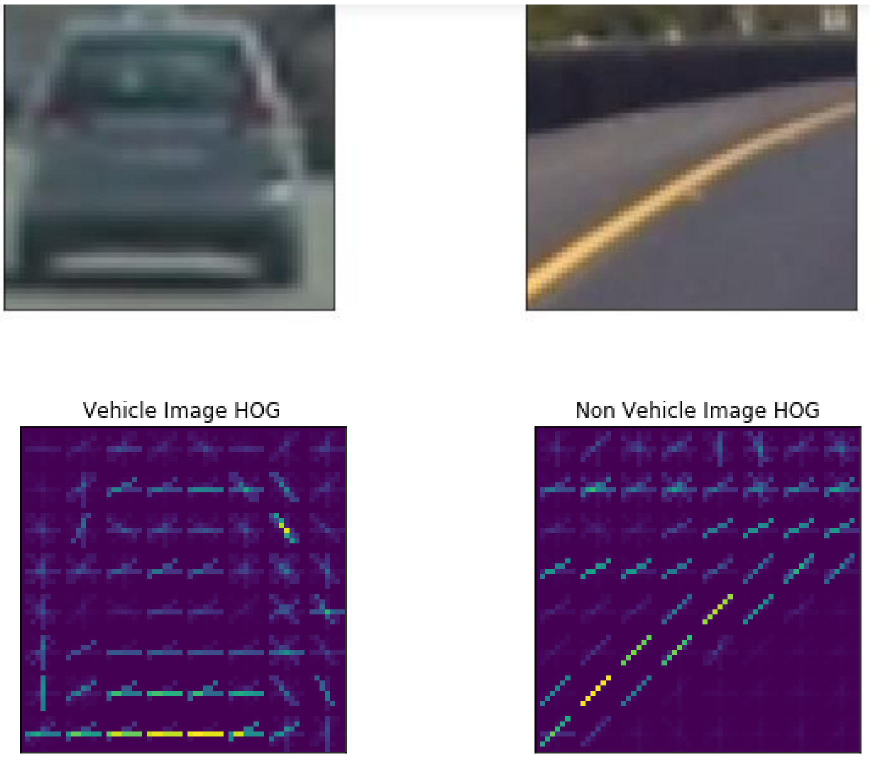
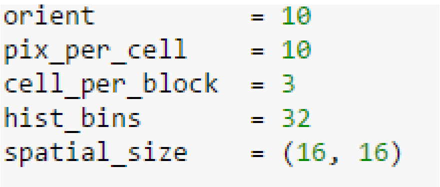


Image from training data set(upper images) and their HOG representation

Training The Classifier

Extracted features(HOG of grayscale ,color histogram and binned color) from training data. were normalized with “StandardScaler” and used to train Linear Support Vector Classifier. Normalization process ensures that a classifier's behavior isn't dominated by just a subset of the features, and that the training process is as efficient as possible. To get better generalization simple augmentation technique was used. Both non-car and car images were flipped around horizontal axis. With provided features, classifier obtained a test accuracy of 0.9899%. Final parameters for feature extraction are:

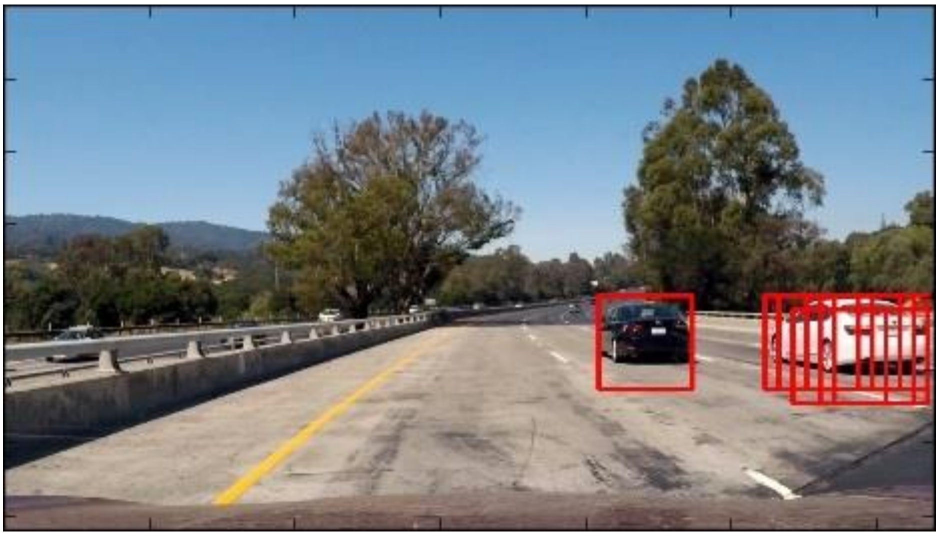


```
orient          = 10
pix_per_cell    = 10
cell_per_block  = 3
hist_bins       = 32
spatial_size    = (16, 16)
```

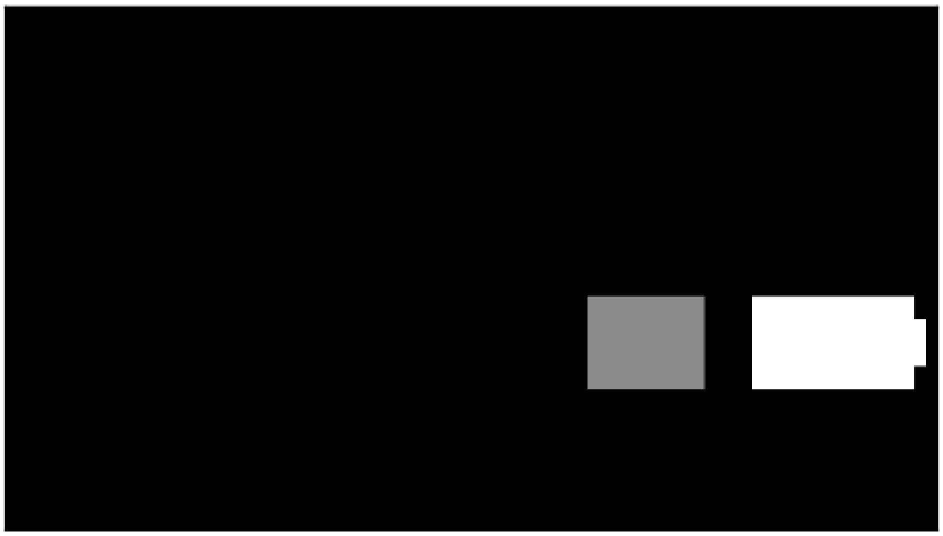
Final values of hyper-parameters have been selected by trial and error method. With those values I have best results. All images before feature extraction were converted to LUV color space. I choose this one because it gave me in my tests best results.

Sliding Window And Classifier Test

To increase performance HOG features have been extracted only once and only from fixed sized region of interest. Than array of features have been subsampled for each sliding window. Sliding windows have different size for vehicles which are near or far from our car. In order to reduce jitter I used simple low-pass filter on the new and previous car boxes coordinates and size. This makes car boundaries more stable and smooth. And finally every second from video was skipped. I assumed that there won’t be very fast moving objects.



Detected vehicles with drawn boxes



Applied heat map threshold method to remove false positives

Discussion

- Algorithm may fail in case of bad light or weather condition (fog)
- The algorithm failed if detected cars overlap. In such case algorithm treat both cars as one object. One solution is to implement something like memory. So, even if cars overlap the algorithm remembers last recorded coordinates of a car and estimates future position.
- The pipeline is too slow for real-time processing (about 8 fps). For high speed object detection one can use YOLO technique("you only looked once") which uses deep neural networks and it works in even 150 fps.
- To eliminate false positive on areas out of road, and improve classification one can use deep neural networks instead of SVM. Larger data set and well-designed deep convolutional neural net could give more accurate and stable results.