Vehicle-Detection

Vehicle Detection Project

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.

Vehicle-Detection

In this project, I use histograms of oriented gradients (HOGs) and color features, along with a linear support vector machine classifier, in order to detect vehicles in a road video.

Histogram of Oriented Gradients (HOG)

The code for this step can be found in function:get_hog_features of detect_function.py.

I started by reading in all the `vehicle` and `non-vehicle` images.

I then explored different color spaces and different `skimage.hog()` parameters (`orientations`, `pixels_per_cell`, and `cells_per_block`). I grabbed random images from each of the two classes and displayed them to get a feel for what the `skimage.hog()` output looks like. In this project, `YCrCb` color space and HOG parameters of `orientations=9`, `pixels_per_cell=8` and `cells_per_block=2` are used to solve problem.

Sliding Window Search

In this project, the sliding windows search is used to classify the cars from the image. In order to lighten the computation load, searchable area of a frame is restricted to under 400 on y-axis. And, I searched on 4 scales using YCrCb 3-channel HOG features plus spatially binned color and histograms of color in the feature vector, which provided a nice result.

Video Implementation

My video is "project_vid_output_with_lanes.mp4".

I recorded the positions of positive detections in each frame of the video. From the positive detections I created a heatmap and then thresholded that map to identify vehicle positions. I then used `scipy.ndimage.measurements.label()` to identify individual blobs in the heatmap. I then assumed each blob corresponded to a vehicle. I constructed bounding boxes to cover the area of each blob detected.

Discussion

It is not a perfect project for a vehicle detections. Because, the deep CNN can provide a better performance on accuracy and speed.
The furthure improvement is using YOLO,SSD or RCNN to make a better classifer.