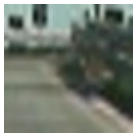
Vehicle Detection Project

1. Explain how (and identify where in your code) you extracted HOG features from the training images.

The code for this step is contained in the extract_features method present in utils.py file.

I started by reading in all the `vehicle` and `non-vehicle` images. Here is an example of one of each of the `vehicle` and `non-vehicle` classes:

Vehicle

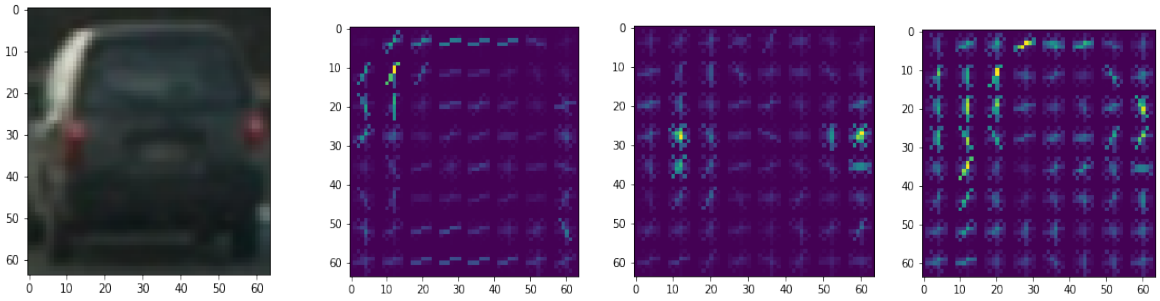


Non Vehicle

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()'.

Here is an example using the `YCrCb` color space and HOG parameters of `orientations=8`, `pixels_per_cell=(8, 8)` and `cells_per_block=(2, 2)`:

Original Image	Red Channel HOG	Blue Channel HOG	Green Channel HOG



2. Explain how you settled on your final choice of HOG parameters.

I tried various combinations of parameters and finally found that YUV and YcrCb give better accuracy . Also using ALL channels gives better accuracy when classifying .

color_space = 'YCrCb' pix_per_cell = 8
orient = 9 cell_per_block = 2
hog_channel = "ALL"

3. Describe how (and identify where in your code) you trained a classifier using your selected HOG features (and color features if you used them).

I trained using hog features as well as spatial and color features .Feature extraction is defined in extract_features() methd present in utils.py .
I trained a linear SVM using these parameters.

color_space = 'YCrCb' # Can be RGB, HSV, LUV, HLS, YUV, YCrCb
orient = 9 # HOG orientations
pix_per_cell = 8 # HOG pixels per cell
cell_per_block = 2 # HOG cells per block
hog_channel = "ALL" # Can be 0, 1, 2, or "ALL"
spatial_size = (32, 32) # Spatial binning dimensions
hist_bins = 64   # Number of histogram bins
spatial_feat = True # Spatial features on or off
hist_feat = True # Histogram features on or off
hog_feat = True

Sliding Window Search

1. Describe how (and identify where in your code) you implemented a sliding window search. How did you decide what scales to search and how much to overlap windows?

I used technique defined in lecture .I used multiple window sizes ranging from 200 to 720 incrementing by 16 ie. 200, 216 , 232, 248 and so on .

To avoid false positive I clipped the area from 400 to 600 in y direction and 600 to 1210 in x direction and also applied heat map with threshold of 2 .

I implemented this in predict_draw method in the ipynb file .

2.Show some examples of test images to demonstrate how your pipeline is working. What did you do to optimize the performance of your classifier?

Ultimately I searched using YCrCb 3-channel HOG features plus spatially binned color and histograms of color in the feature vector, which provided a nice result.

3**. Provide a link to your final video output. Your pipeline should perform reasonably well on the entire project video (somewhat wobbly or unstable bounding boxes are ok**

as long as you are identifying the vehicles most of the time with minimal false positives.)

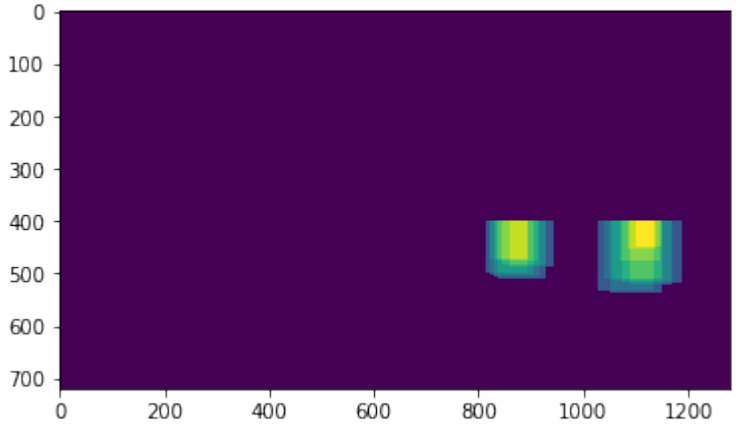
Here's a link to my video
https://youtu.be/PTVVfX1DkWA

Describe how (and identify where in your code) you implemented some kind of filter for false positives and some method for combining overlapping bounding boxes.

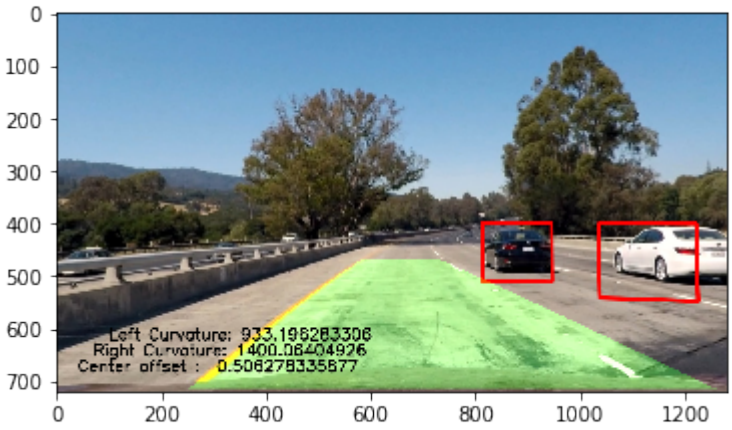
I recorded the positions of positive detections in each frame of the video. From the positive detections I created a heatmap and then thresholded that map to identify vehicle positions. I then used `scipy.ndimage.measurements.label()` to identify individual blobs in the heatmap. I then assumed each blob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

Here's an example result showing the heatmap from a series of frames of video, the result of `scipy.ndimage.measurements.label()` and the bounding boxes then overlaid on the last frame of video:

HeatMap Image



Output Image



Discussion

1. Briefly discuss any problems / issues you faced in your implementation of this project. Where will your pipeline likely fail? What could you do to make it more robust?

Initially in many frames there was no car being detected . At times immediately after detection in one frame the next frame could not detect a car .

As a result the output is not so smooth .This can be seen here https://www.youtube.com/watch?v=vkgoLKgwc_A .

To fix this I combined boxes detected in last 10 frames and used those to create heatmap and then thresholded the map by 2 .

The problem with this technique is that in case of false positives wrong detections remain at least for 10 frames . But disappear after that .If we improve accuracy of classifier then we can improve this technique.