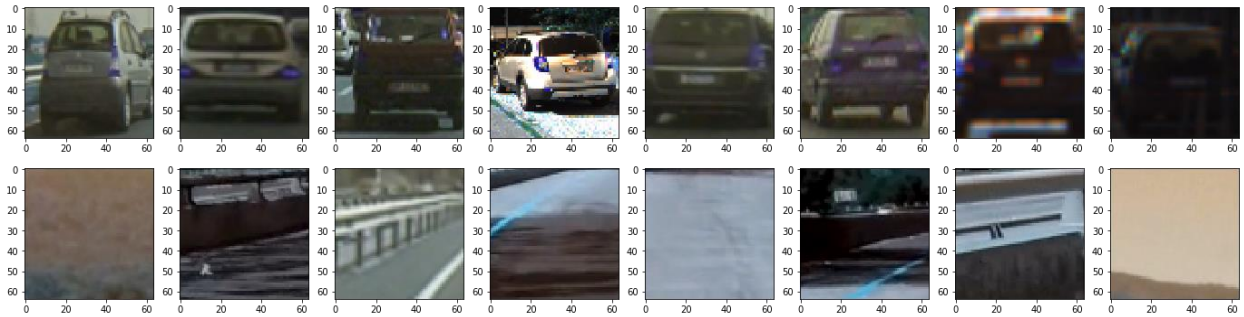
Vehicle Detection and Tracking

1: Data summary and visualization

As always, the first step is the data exploration to understand what dataset we are using. For this project data was provided by Udacity. It included 8,792 images of cars and 8.968 images of non-cars. Image shape was 64x64 RGB image. Figure 1 shows a few of the car and not cars images. Car images have picture of cars from the back or the side from various angles. The lighting seems to differ a lot, with some images look to be with a lot of sun lights while others are images from the darkness.

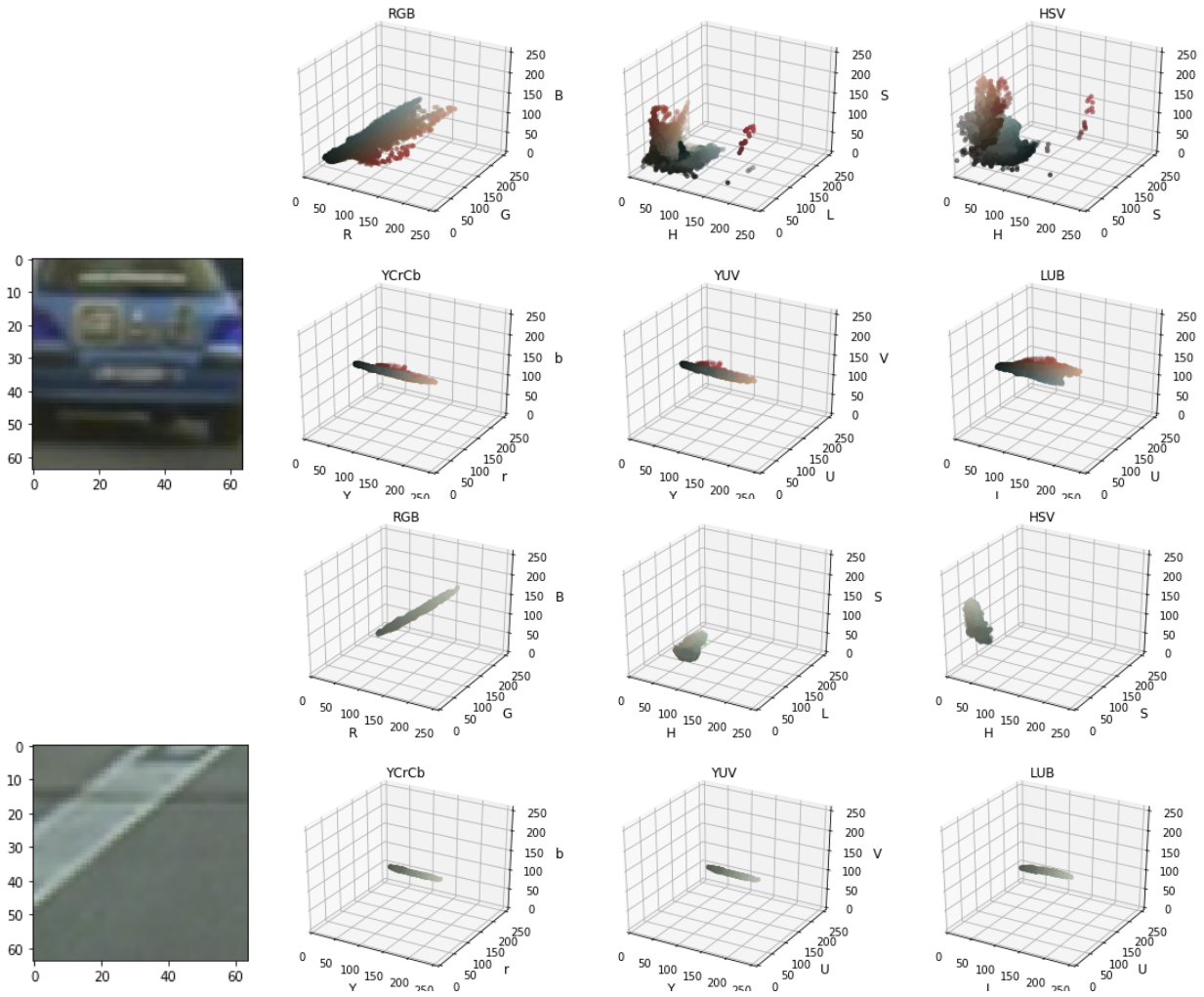
Figure 1: sample images



2: Visualize color spaces

I then visualized cars and non-cars images across different color spaces, including RGB, HLS, HSV, YCrCb, YUV, and LUB. Figure 2 has a car and non-car images along with their corresponding color spaces. One thing I noticed was that YCrCb and YUV were nearly identical in most images. Through iterations, they seem to produce most distinguishiable features. At the end, YCrCb was used to separate car images from non-car images.

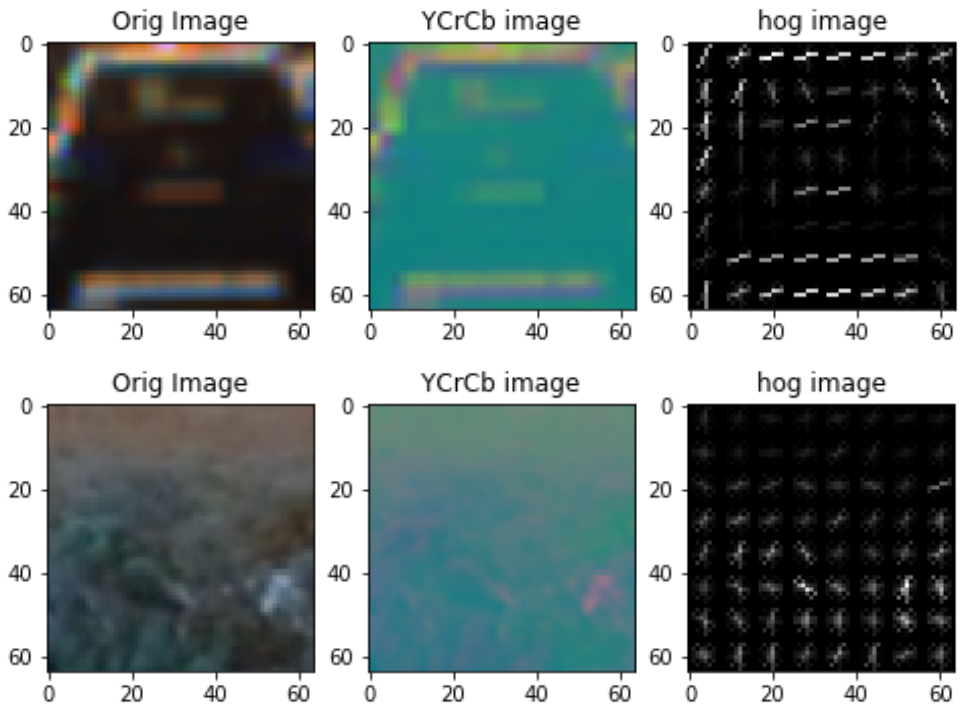
Figure 2: Colorspace for a car and non-car images



3: Visualize HOG features

I then compared the HOG features for car and non-car images as seen in figure 3.

Figure 3: HOG features for car and non-car images



4: Train the data in SVM after combining and normalizing features

Before training the data, I combined the color spatial, color histogram and hog features of YCrCb converted images. This is because the features may have values that vary widely in magnitude, so normalizing try prevent larger value to get a dominant effect.

For the machine learning classifier, I used the SVM. I tried both linearSVC and SVC, and also played around with parameters, including the penalty parameter C. At the end, I decided to use the default SVC model.

I used the spatial size of 16 and histogram bins of 32. For HOG, I used orientations of 8, pixels per cells of 8 and 2 cells per block. In addition, I split the training and test data into 80/20 split. This resulted in the accuracy of 99.3%.

5: Sliding Window

I then used the sliding windows in the test images and tried to identify if the car is in that window. The method did a fairly good job of finding cars as seen in figure 4.

Figure 4: Identifying cars in test images using sliding windows

5: Adding heatmap functionality

To remove false positives, I then addedd the heatmap functionality to only select boxes that meets the certain threshold. I tried various threshold values and decided on 5 at the end. As seen in Figure 5, it did remove the car box on the last test image.

Figure 5: Using heatmap to isolate car boxes

6: Use pipeline to run on a video

Finally, I used the same pipeline to create detect cars on the video. Given the different sizes of the cars, scales of 1.00, 1.25, 1.50, 1.75 and 2.00 were used. Since smaller cars only appear further away, scales of 1.00 and 1.25 were only used in the mid- to mid-bottom section of the road. In addition, I used 10 frames to accumulate the positive detection to create a heat map to offset potential false positives that may appear. The map was then thresholded to identify vehicle positions. This has resulted in smoother bounding boxes.

7: Challenges that I faced and improvements that can be made

- As always, data is a cruicial to any machine learning project. This particularly dataset lacked white cars so detecting a white car became a challenge. Getting more data is very valuable, but there are also other methods such as augmenting the existing data by flipping the image across the x-coordinate or changing the lightening.
- Additionally, reviewer has suggested that hard negative mining could improve the reliability of the classifier.
- While I did play with the C parameter on SVC, I probably could not quite the optimal value and ended up using the default value. I could have used GridSearch method to try out different C values.

- Another challenge that I ran into was trying to balance the runtime versus more detailed data extraction. Feature extraction took a lot of time to run, thus it became an art to make the features small as possible without losing too much information.
- To get fancier, I can apply the lane detection technique from previous lessons and try to estimate the distance to the bounding boxed car. Given this estimated distance and the size of the box, I can better estimate whether if this is a reasonable size for a car.