UDACITY

PROJECT

Vehicle Detection and Tracking

A part of the Self Driving Car Engineer Nanodegree Program

PROJECT REVIEW

CODE REVIEW

NOTES

SHARE YOUR ACCOMPLISHMENT!

Meets Specifications

Writeup / README

The writeup / README should include a statement and supporting figures / images that explain how each rubric item was addressed, and specifically where in the code each step was handled.

Good job addressing the rubric items in the 'momr_writeup.md' file with references to supporting images.

Histogram of Oriented Gradients (HOG)

Explanation given for methods used to extract HOG features, including which color space was chosen, which HOG parameters (orientations, pixels_per_cell, cells_per_block), and why.

Good job choosing the 'YUV' color space. Good work computing the HOG values for all three color channels. Well done using a binned histogram of the three color channels and a downsampled (32x32) pixel image as the input features for the classifier.

Well done mentioning in the 'momr_writeup.md' file, "I tried various combinations of parameters and used them to train a simple classifer (a quick one to train) to get a feeling about how changing parameters change classifer performance. Then picked the combination with best performance and started to focus more on building a more robust classifier as described in the next section."

The HOG features extracted from the training data have been used to train a classifier, could be SVM, Decision Tree or other. Features should be scaled to zero mean and unit variance before training the classifier.

Good job using StandardScaler() to normalize the dataset by features. Nice job splitting the dataset into a training and holdout testing set with train_test_split. Good work using a support vector machine classifier to train the data and test the accuracy of the model on the holdout test data.

Sliding Window Search

A sliding window approach has been implemented, where overlapping tiles in each test image are classified as vehicle or non-vehicle. Some justification has been given for the particular implementation chosen.

Well done computing the HOG values for the entire image and using the sub-sampled values for each window. Good job limiting the sliding windows to only the area of the image where cars are most likely to occur for the project video.

Good job using multiple scales and a cells_per_step of 2 for the sliding window.

Some discussion is given around how you improved the reliability of the classifier i.e., fewer false positives and more reliable car detections (this could be things like choice of feature vector, thresholding the decision function, hard negative mining etc.)

Nice job mentioning in the 'momr_writeup.md' file, "The code for building my final classifer can be found under the "2.3 Combining Features" section of my notebook. I trained different classifers using the parameters I picked in the previous section. After normalizing my features and spliting up data into randomized training and test sets. I tested Linear SVM, Random Forest, Ada Boost, and a VotingClassifier consist of all the previously mentioned classifers. And found that Linear SVM is the best interms of Accuracy (99.2%) to Seconds to predict (4 ms) ratio, and that is why I choosed it to be used later on in my video pipeline code."

Two other methods, among many, that could be used to try to decrease the false positives are thresholding the decision function and hard negative mining.

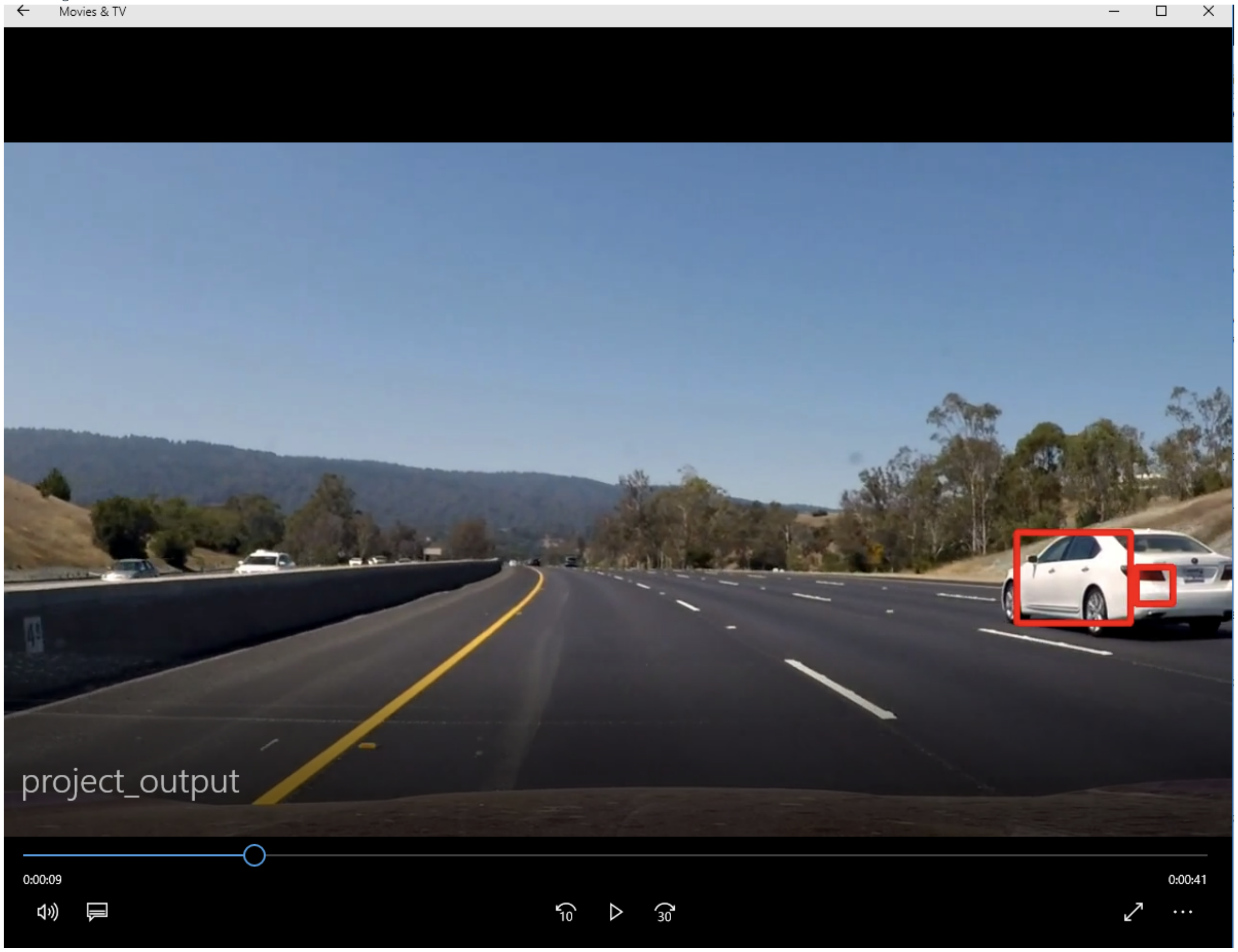
Quoted from the discussion link below, for svm decision function, "The desion function tells us on which side of the hyperplane generated by the classifier we are (and how far we are away from it)." We could try using a decision function threshold to increase the "confidence" level needed to classify a vehicle. This could help decrease the number of false positive. This is a link to a stackoverflow discussion on svm decision function.

Hard negative mining is when we take the false positives from the output and insert it back into the training data and retrain the model. One way to do this is to limit the window search to the area(s) of the image where no expected vehicles are to occur. Then all of the classified vehicles would be false positives, and all of the detected sub images could be outputted and inserted into the training set to have the model retrained.

Video Implementation

The sliding-window search plus classifier has been used to search for and identify vehicles in the videos provided. Video output has been generated with detected vehicle positions drawn (bounding boxes, circles, cubes, etc.) on each frame of video.

Good job outputting the annotated project video. It looks good. There are still some frames where multiple bounding boxes are labeled for a single vehicle. Trying to use an additional scale, a larger scale, could help draw larger bounding boxes that could help the bounding boxes overlap one another to try to remove this effect of having multiple bounding boxes for one vehicle.



A method, such as requiring that a detection be found at or near the same position in several subsequent frames, (could be a heat map showing the location of repeat detections) is implemented as a means of rejecting false positives, and this demonstrably reduces the number of false positives. Same or similar method used to draw bounding boxes (or circles, cubes, etc.) around high-confidence detections where multiple overlapping detections occur.

Well done implementing a thresholded heatmap and incorporating bounding boxes across multiple consecutive frames to try to reduce the jittering of the bounding boxes.

Discussion

Discussion includes some consideration of problems/issues faced, what could be improved about their algorithm/pipeline, and what hypothetical cases would cause their pipeline to fail.

Nice job on the discussion. Good job mentioning the heat map that was implemented. Well done mentioning, "The pipeline will likely fail for detecting Trucks or motorbikes, if not because of the classifer, then this will be because of the window search algorithm assumes implecitly certain size of vehicles to search for.

Two things that I wanted to try but I hadn't enough time are: using a DNN instead of my current classifier. Second is using YOLO :)"

DOWNLOAD PROJECT

RETURN TO PATH

Rate this review

Student FAQ