UDACITY

PROJECT SPECIFICATION

Vehicle Detection and Tracking

Writeup / README

CRITERIA	MEETS SPECIFICATIONS	REVIEWER TIPS
Provide a Writeup / README that includes all the rubric points and how you addressed each one. You can submit your writeup as markdown or pdf. Here is a template writeup for this project you can use as a guide and a starting point.	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.	Please don't fail the student on this rubric item just because something is missing from the writeup/README. If something is missing from the writeup but you can see it was executed correctly in the project code, please don't fail the project solely because the explanation wasn't there. If the writeup is missing entirely please select "Can't Review".

Histogram of Oriented Gradients (HOG)

CRITERIA	MEETS SPECIFICATIONS	REVIEWER TIPS
Explain how (and identify where in your code) you extracted HOG features from the training images. Explain how you settled on your final choice of HOG parameters.	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.	Students should provide some discussion of how they went about extracting HOG features, how they settled on their choices of parameter settings.
Describe how (and identify where in your code) you trained a classifier using your selected HOG features (and color features if you used them).	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.	Students are free to use HOG features alone, or include color features (spatially binned or histograms) in their feature vector here.

Sliding Window Search

CRITERIA	MEETS SPECIFICATIONS	REVIEWER TIPS

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?	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.	In this case, there are various approaches that may be taken with regard to which region of the image to search, how large to make the tiles and with how much overlap. There are not particular restrictions, but some justification should be provided for the choices made.
Show some examples of test images to demonstrate how your pipeline is working. What did you do to try to minimize false positives and reliably detect cars?	Some discussion is given around using methods like thresholding the decision function, hard negative mining, or feature vector tuning to improve the reliability of the classifier (less false positives and more reliable car detections)	Lots of flexibility here. The basic idea is that students try to optimize the performance of their classifier in some way before moving on to the video processing stage.

Video Implementation

CRITERIA	MEETS SPECIFICATIONS	REVIEWER TIPS
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.)	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.	Here, we're not looking for any specific accuracy level (minimum criterion would be at least more true positives than false positives) but the result should demonstrate that the classifier is doing a reasonable job of identifying cars in video frames without too many false positives
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.	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.	The most likely way students will implement this step is to require that a positive detection be made at or near the same position in several subsequent frames of video. However, they are free to try other parameters / methods for false positive rejection, as long as they can show it improves their detection accuracy.

Discussion

CRITERIA	MEETS SPECIFICATIONS	REVIEWER TIPS
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?	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.	This point is quite flexible. As long as a student has taken some time to reflect on the project and included some thoughts this point should pass.

Suggestions to Make Your Project Stand Out!

A stand out submission for this project will be a pipeline that runs in near real time (at least several frames per second on a good laptop) and does a great job of identifying and tracking vehicles in the frame with a minimum of false positives. As an optional challenge, combine this vehicle detection pipeline with the lane finding implementation from the last project! As an additional optional challenge, record your own video and run your pipeline on it to detect vehicles under different conditions.

Reviewer FAQ Student FAQ