PRECEDING VEHICLE DETECTION USING HISTOGRAMS OF ORIENTED GRADIENTS

Project Report

1411535: Computer Vision and Image Processing
Ammara Razzaq
U00045665

Preceding Vehicle Detection Using Histogram of Oriented Gradients

Project Report

1- Introduction

In driving assistance systems (ADAS), the camera mounted on vehicles capture preceding vehicles. The captured vehicle may have variance on local shape, color, view and are affected by local occlusion and illumination conditions etc, which challenges most of the existing object detection methods to complete the preceding vehicle detection.

In this project vehicle detection based on HOG and SVM classifier is implemented based on research paper “Preceding Vehicle Detection using Histogram of Oriented Gradients”.

2- Vehicle Detection System

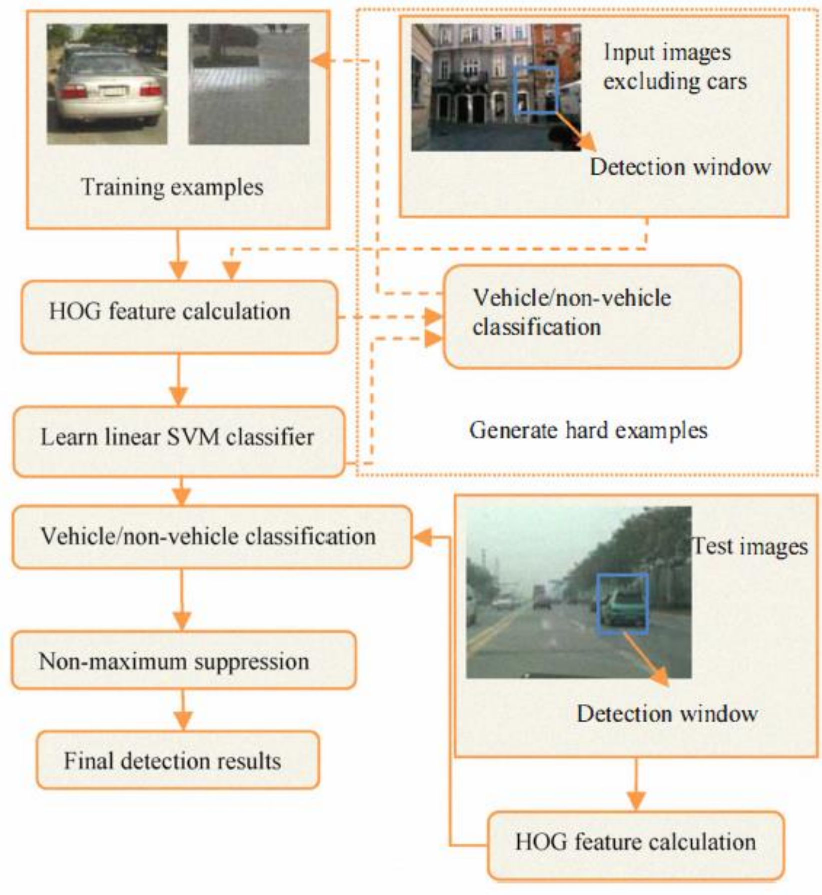


Fig. 1. The framework of vehicle detection algorithm based on HOG

The frame work of vehicle detection algorithm is based on following steps.

Step-1: HOG Feature Extraction

Step-2: Train Linear SVM Classifier

Step-4: Apply Hard-Negative Mining

Step-5: Vehicle Detection

Step-6: Non-Maximum Suppression

3- Implementation

Step-1: HOG Feature Extraction

HOG features of images are extracted to characterize the local object appearance and shape.

HOG features are extracted from training data consisting of positive and negative samples. Number of negative samples is much greater than number of positive samples. Each detection window is divided into cells of size 8x8 pixels and each group of 2x2 cells is integrated into a block in a sliding fashion, so blocks overlap with each other. The gradient angle of each histogram is divided into 9 bins.

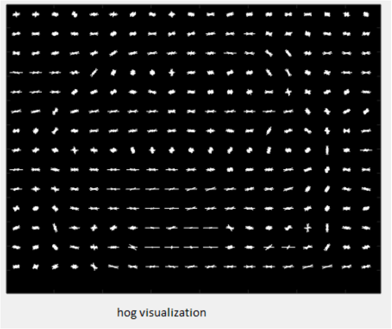
Each 128x128 detection window is represented by 15x15 blocks giving a total of 8100 features per detection window. This feature extraction is a dense representation that map local image regions to high dimension feature spaces. These features are then used to train a linear svm classifier.

For computing histograms efficiently some image preprocessing is performed. In the research paper each image is converted to an integral image as image pre-processing step. For this project I have used another technique which is more efficient image representation. In this technique a global threshold for image is computed that is used to convert the intensity image to a binary image. To compute the global threshold Otsu’s method is used which chooses the threshold to minimize the intra-class variance of the black and white pixels. Graythresh() and im2bw() Matlab functions are used for this purpose.

ExtractHOGFeatures() function of Matlab is used to extract HOG features. cellSize and blockSize Values are default values of matlab function i.e, cellSize = [8 8] and blockSize = [2 2].

trainingFeatures and trainingLabels are calculated for training SVM Classifier. Training labels are ‘0’ for negative training data and ‘1’ for positive training data.

Below figure shows HOG features of a test image.



hog visualization

Step-2: Train Linear SVM Classifier

A linear SVM classifier is trained on positive and negative training data which are fixed resolution image windows.

Linear SVM classifier is used as binary classifier because the number of HOG features is large, and linear SVM is usually computationally faster. Each positive instance usually contains only one centred instance of the object, and negative windows are usually randomly sub-sampled and cropped from set of images not containing any instance of the object.

A linear SVM Classifier is trained on training data using trainingFeatures and trainingLabels. Svmtrain() function is used for this purpose. Now, the initial classifier is ready, but in the negative training set used in training classifier, running preliminary classifier will generate many false positives.

Step-3: Apply Hard Negative Mining

To reduce false positives and make full use of the training images, the preliminary detector exhaustively scans the negative training images for hard examples (false positives), and then the classifier is retrained using this augmented training set (original positives and negatives and hard examples) to produce the final detector.

For the purpose of hard negative mining sliding window technique is used to exhaustively scan the images. Sliding window is a fixed size window that scans the whole image using a fixed step size. If the step size is larger, sliding window will be faster, but it can miss some detections too. So an optimal step size should be used. Window size will be same as the training image sizes.

At each window HOG features are computed and classifier is applied. If classifier (incorrectly) classifies a given window as an object, feature vectors are recorded associated with the false positive patch. Classifier is retrained using original positive and negative training set and additional hard examples to produce final detector.

After retraining, final detector is now ready to detect the vehicles on test data.

Step-4: Vehicle Detection

Classifier is now trained and can be applied on test dataset. For each image in the test data set and for each scale of image, sliding window technique is applied. At each window HOG descriptors are extracted and classifier is applied. If the classifier detects an object, bounding box of window is recorded. Vehicle detector is run on test data and it gives multiple detections around each vehicle. These detections need to be fused together.

To get only one bounding box around each vehicle non-maximum suppression is applied.

Step-5: Non-Maximum Suppression

After scanning of image is finished, non-maximum suppression is applied to remove redundant and overlapping bounding boxes.

Non-maximum suppression method developed by Tomasz is used. This method is one of the fastest method so far. Using this method, one bounding box among many around each vehicle is selected according to SVM score of each bounding box.

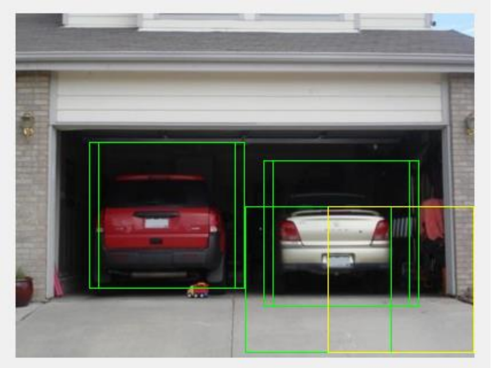
In the research paper, mean shift technique is used for non-maximum suppression. More robust algorithm for non-maximum suppression can be applied. There is a more accurate and fast algorithm developed by Dr. Tomasz Malisiewicz which he used for his dissertation and his ICCV 2011 paper, *Ensemble of Exemplar-SVMs for Object Detection and Beyond*. I have used Dr. Tomasz algorithm for non-maximum suppression which gives more accurate bounding boxes.

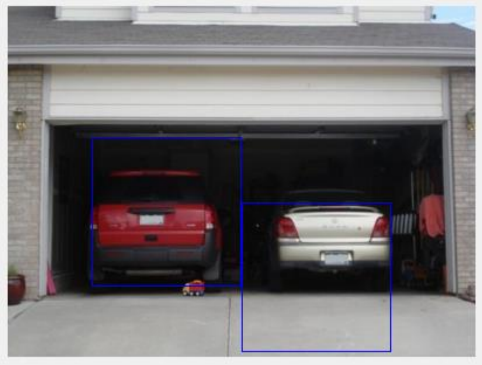
The input to this algorithm is set of overlapped bounding boxes and their scores. Score of each bounding box is computed using svmdecision() function of Matlab.

1- Bounding boxes are sorted according to their svm scores.
2- Instead of using an inner for loop to loop over each of individual boxes, the code is vectorized using min and max function, this allows to find maximum and minimum values across the axis rather than just individual scalars.
3- Width and height of each rectangle is computed.
4- Using values in step-3 overlap ratio is computed.
5- Entries that are greater than the supplied threshold are deleted.(value of supplied threshold normally falls in the range of 0.3 – 0.5)

Using vectorized code instead of for loops improves speed up to 100 times.

Below figures show test images before and after applying non-maximum suppression.





4- Experiments and Results

The proposed system is evaluated on a group of test images gathered from internet.

For training purposes MIT car dataset is used as positive samples and selected negative images from INRIA dataset as negative samples. MIT car dataset includes 516 128x128 images containing front and rear view of cars. 1025 [128 128] negative samples are used not containing any instances of vehicles.

Below figure has some positive and negative samples from training data set.

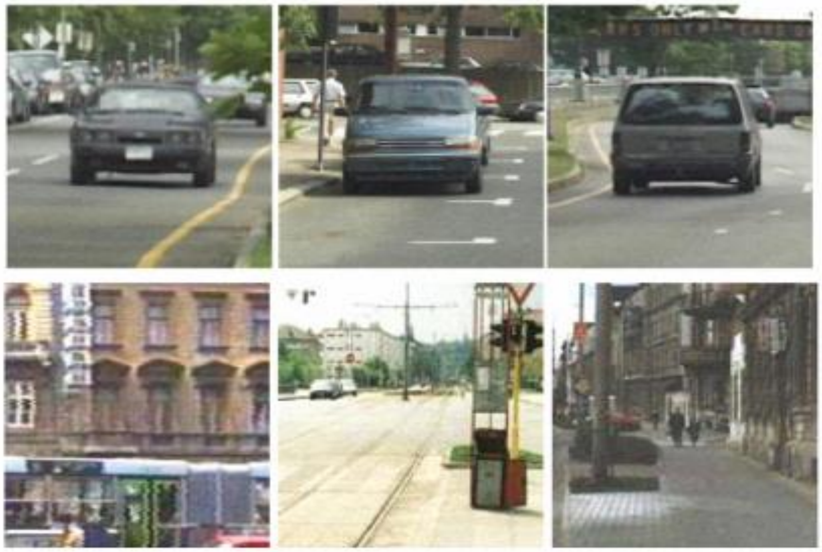
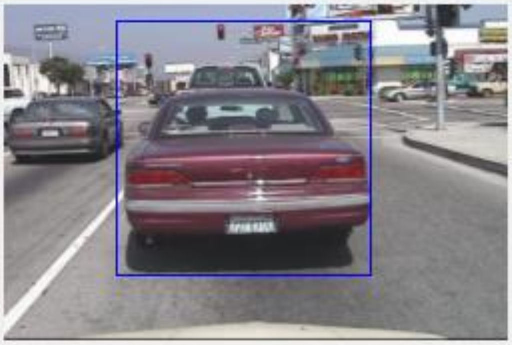


Fig. 4. Training examples. Positives(top line) and negatives(bottom line)

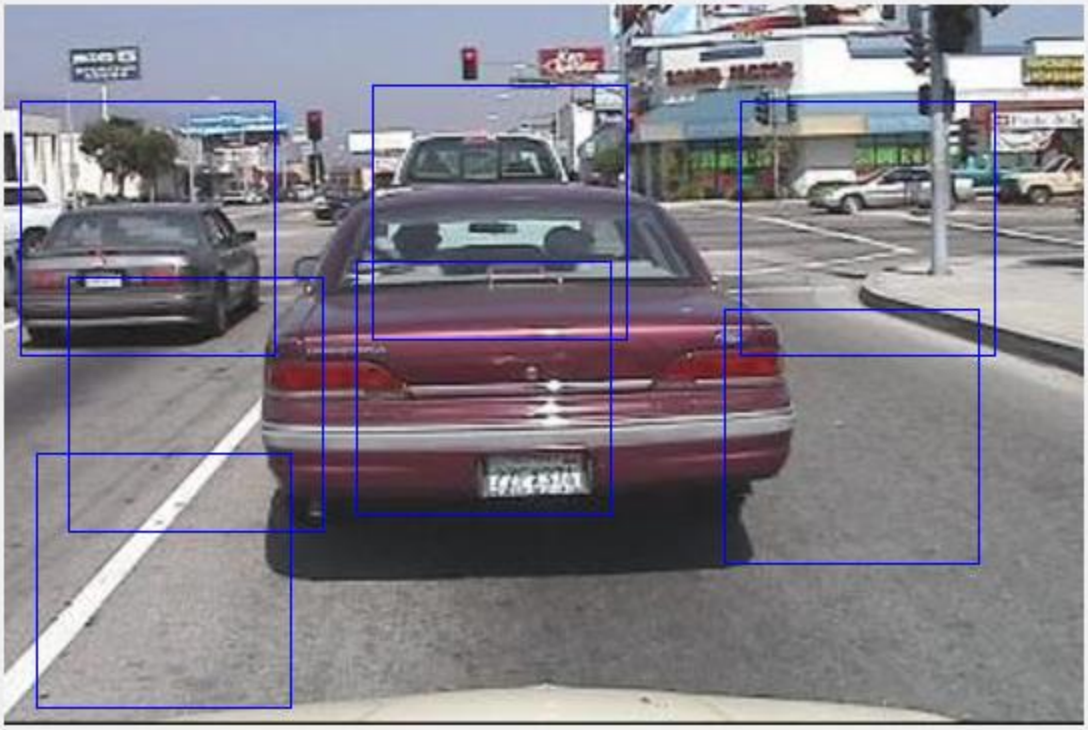
The initial datasets are all transformed to high-dimension feature vectors and train the preliminary linear SVM classifier. Then retrain to get the final detector after getting the hard examples.

In the detection phase, each test image is detected at different scales to detect vehicles of all sizes. The scale range used in tests is from 0.5 to 1.5 and the scale step is 0.1. Scale decides the size of the vehicle detected.

Figure shows the detection results for scale = 1.5 and scale =0.7. Small vehicles could not be detected at smaller scale but can be detected at larger scale. With Scale = 1.5 all vehicles are detected with some false positives as well.



Scale = 0.7



scale = 1.5

Performance Parameters

The performance parameters for test data set are as follows:

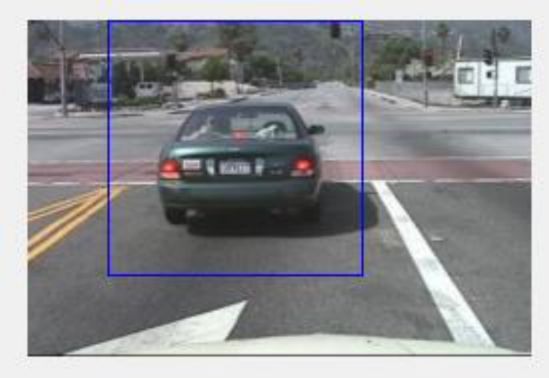
(1) TPR = Number of Detected Vehicles /Total Number of Vehicles

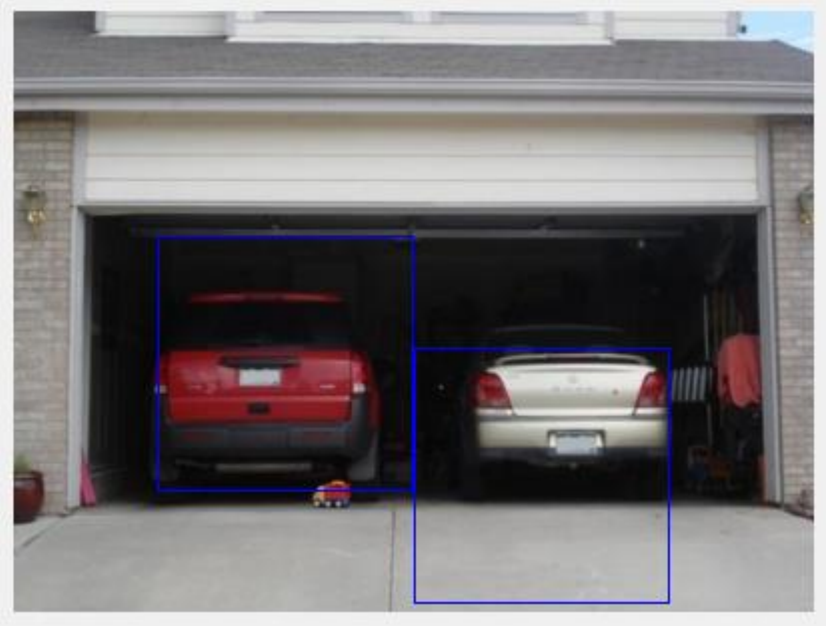
(2) FPR = Number of false positive/Total number of frames

Test dataset includes around hundred images on which tests have been performed. After running the detector on test dataset TPR is 95.83% and FPR is 0.3%.

Total number of vehicles in test dataset were 120 and 115 were the total number of detected vehicles.

Below figures shows experiment results for single and multiple vehicles.





5- Improvements in the detection system

The improvements that are done in the project are already discussed above.

1. Image Pre-Processing

   Instead of using the integral image, binary image is used, using which histograms can be computed more efficiently.

2. Non-Maximum Suppression

   Instead of using the mean shift algorithm for non-maximum suppression presented in paper, a faster non-maximum suppression algorithm is applied which is much accurate and fast that mean shift algorithm.