Preceding Vehicle Detection Using Histograms of Oriented Gradients

MAO Ling[1], XIE Mei[1], HUANG Yi[2] and ZHANG Yuefei[1]
[1] School of Electronic Engineering/University of Electronic Science and Technology of China/ Chengdu,
Sichuan, China
[2] Department of Communication Engineering/Chongqing College of Electronic Engineering/ Shapingba,
Chongqing, China
maoling5@163.com

***Abstract*—This paper presents a monocular vision-based preceding vehicle detection system using Histogram of Oriented Gradient (HOG) based method and linear SVM classification. Our detection algorithm consists of three main components: HOG feature extraction, linear SVM classifier training and vehicles detection. Integral Image method is adopted to improve the HOG computational efficiency, and hard examples are generated to reduce false positives in the training phase. In detection step, the multiple overlapping detections due to multi-scale window searching are very well fused by non-maximum suppression based on mean-shift. The monocular system is tested under different traffic scenarios (e.g., simply structured highway, complex urban environments, local occlusion conditions), illustrating good performance.**

I. INTRODUCTION

THE researches on Advanced Driver Assistance System (ADAS) are developed quickly in the recent years [1]. As an important component of Intelligent Transportation System (ITS), ADAS is aimed to help drivers see the circumstances on road and driving situations and reduce traffic accidents [2]. In the last decades, monocular vision-based driving assistance systems have attracted more interests of the researchers for their low cost and for the high-fidelity information they give about the driver environment [3-4]. Vehicle accident statistics disclose that the main threats a driver is facing are from other vehicles. Consequently, robust and effective vehicle detection, especially preceding vehicle detection, is a primary step in most of these systems [3].

In monocular vision-based driving assistance systems, the camera mounted on the vehicle captures the preceding vehicles. The captured vehicles may have variances on local shape, color, view and are affected by local occlusion and illumination conditions etc, which challenges most of the existing object detection methods. It is needed to investigate more powerful features and detection methods to complete the preceding vehicle detection.

A majority of methods reported in the literature follow two basic steps: (1) Hypothesis Generation (HG) where the locations of possible vehicles in an image are hypothesized and (2) Hypothesis Verification (HV) where tests are performed to verify the presence of vehicles in an image [3].

Various monocular HG approaches have been suggested in literature and can be divided in two categories: (1) knowledge-based, and (2) motion-based. Knowledge-based methods employ information about vehicle shape and color as well as general information about streets, roads, and freeways. A good synthesis of these different clues such as shadow, edges, entropy and symmetry is given in [5]. However, the parameters (e.g., segmentation threshold) used in the approach above need usually be adjusted to situations, because shadow segmentation and edges extraction are sensitive to illumination changes and the condition of road surface. In other words, the knowledge-based approach is not so robust. Motion-based methods detect vehicles and obstacles using optical flow [6]. Generating a displacement vector for each pixel, however, is time consuming and thus impractical for a real-time system. Moreover it works well only under important relative motion situations such as for passing vehicles.

HV approaches can be classified mainly into two categories: (1) template based, and (2) appearance-based. Template-based methods usually use “loosely” predefined patterns of the vehicle class and perform a correlation between an input image and the template [7-8]. These templates could be very fast, however, their simplicity introduces some uncertainties and very accurate results couldn’t be expected.

Appearance-based methods acquire the characteristics of the vehicle class from a set of training images which capture the variability in vehicle appearance. Usually, the variability of the non-vehicle class is also modeled to improve performance. First, each training image is represented by a set of local or global features. Then, the decision boundary between the vehicle and non-vehicle classes are learned either by training a classifier or by modeling the probability distribution of the features in each class. PCA [9], local orientation coding (LOC) [10], Haar wavelet [11], Gabor filters [12-13] have been used for vehicle representation. However, we can imagine that how to organize the filter responses into effective feature set is not an easy task and computing the feature set is time consuming. Neural network

The work of this paper is supported by Guangdong Natural Science Fund (GNSF). The Project Number is: 8152840301000009.

978-1-4244-8223-8/10/$26.00 ©2010 IEEE

(NN) [9-11] and support vector machines (SVM) [12-13] are usually used as the classifiers.

In most recently, Histograms of Oriented Gradient (HOG) descriptors proposed in [14] significantly outperforms the state-of-the-art feature sets for human detection, since it's robust even in cluttered backgrounds under difficult illumination.

The reviews above inspire us to propose a new algorithm to detect preceding vehicles, which is based on HOG [14]. This algorithm searches all possible areas and directly decides whether vehicles are included in the areas, which avoids much trouble encountered in the HG step. Similar to HV step, HOG descriptor is used to obtain the representation of vehicles. Some calculation skills could be easily adopted in this procedure, for example, Integral Image [15-16], and therefore this appearance-based method is in some sense more efficient than others, especially Gabor filters.

The framework of the detection algorithm is shown in Fig.1. Positive and negative samples for training are represented by HOG feature, which are then used to train a preliminary linear SVM classifier. To increase the detection accuracy, the preliminary linear SVM classifier is re-trained using the additional negative examples generated in the step "generate hard examples". Then the refined detector exhaustively scans the test images and finds the possible areas including cars. Finally we use the non-maximum suppression method based on mean shift [17] to fuse multiple overlapping detections to yield the final vehicle detections.

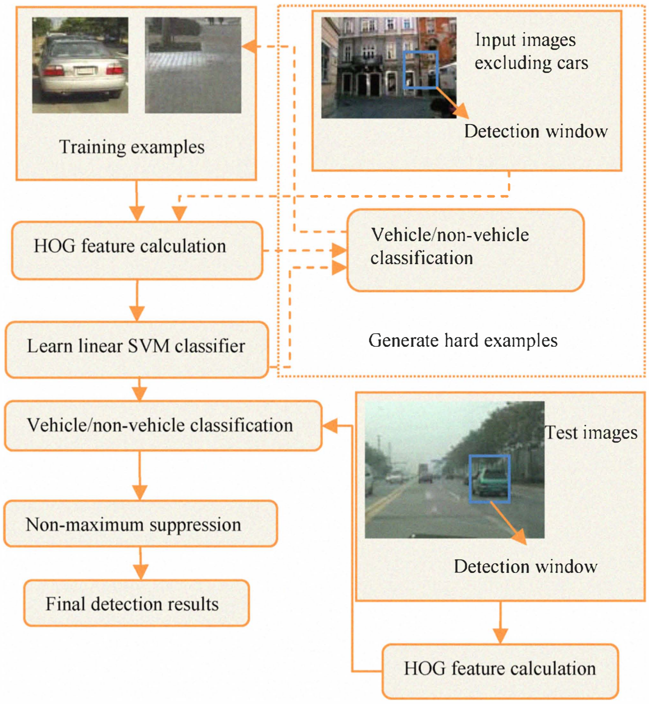


Fig. 1. The framework of vehicle detection algorithm based on HOG

The rest of the paper is organized as follows: In section II, our vehicle detection algorithm is described in detail. In section III, the performance of our algorithm is presented, and in section IV, the conclusions are described.

II. Vehicles Detection System

As is shown in fig.1, the system consists of three main components: HOG feature extraction, linear SVM classifier training and vehicles detection. The input image is firstly represented as HOG feature which is later used as the input of the training step or linear SVM classifier. The learning phase creates a binary classifier that provides cars/non-cars decisions for fixed sized image regions (windows); while the detection phase uses the classifier to perform a dense multi-scale scan reporting preliminary object decisions at each location of the test image. These preliminary decisions are then fused to obtain the final object detections. We will detail the three components in the following sections.

A. HOG Feature Extraction

Histogram of Oriented Gradient (HOG) descriptors presented in [14] provide excellent performance relative to other existing feature sets including wavelets. The basic hypothesis is that local object appearance and shape can often be characterized rather well by the distribution of local intensity gradient or edge directions, even without precise knowledge of the corresponding gradient or edge positions.

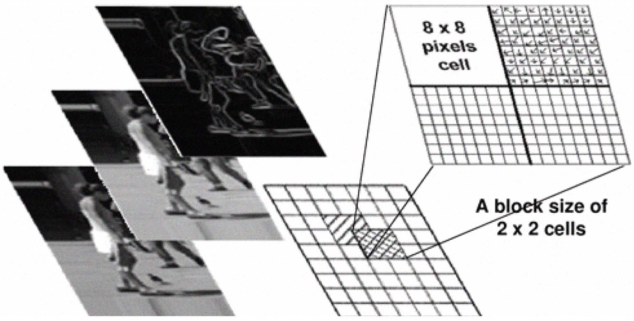


Fig. 2. HOG features. Each block consists of a grid of spatial cells. For each cell, the weighted vote of image gradients in orientation histograms is computed

In [14], each detection window is divided into cells of size 8×8 pixels and each group of 2×2 cells is integrated into a block in a sliding fashion, so blocks overlap with each other. For each pixel $I(x,y)$, the gradient magnitude $m(x,y)$ (3) and orientation $\theta(x,y)$ (4) is computed in these cells. Then a local one-dimensional orientation histogram of gradients is formed from the gradient orientations of sample points within a cell. Each histogram divides the gradient angle range into a predefined number of bins (e.g. 9 bins). The gradient magnitudes vote into the orientation histogram (see Fig.2). Each block contains a concatenated vector of all its cells. In other words, each block is represented by a 36-D feature vector that is normalized to an L2 unit length (5). Each 64×128 detection window is represented by 7×15 blocks, giving a total of 3780 features per detection window. Apparently 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.

$$dx = I(x+1,y) - I(x-1,y) \quad (1)$$

$$dy = I(x,y+1) - I(x,y-1) \quad (2)$$

$$m(x,y) = \sqrt{dx^2 + dy^2} \quad (3)$$

$$\theta(x,y) = \tan^{-1}\left(\frac{dy}{dx}\right) \quad (4)$$

$$\mathbf{v} \leftarrow \mathbf{v} / \sqrt{\|\mathbf{v}\|_2^2 + \varepsilon^2} \quad (5)$$

The original method of computing histograms is not efficient. In this paper we adopt Integral Image [15] to efficiently compute histograms over cells.

The integral image at location x, y contains the sum of the pixels above and to the left of x, y, inclusive

$$ii(x,y) = \sum_{x' \le x, y' \le y} i(x', y') \quad (6)$$

where $ii(x,y)$ is the integral image and $i(x,y)$ is the original image. Using the following pair of recurrences

$$s(x,y) = s(x,y-1) + i(x,y) \quad (7)$$

$$ii(x,y) = ii(x-1,y) + s(x,y) \quad (8)$$

where $s(x,y)$ is the cumulative row sum, $s(x,-1)=0$, and $ii(-1,y)=0$, the integral image can be computed in one pass over the original image.

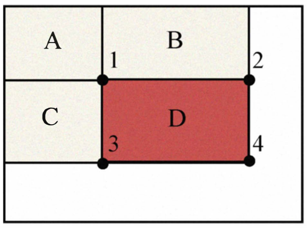


Fig. 3. The sum of the pixels within rectangle D can be computed with four array references. The value of the integral image at location 1 is the sum of the pixels in rectangle. The value at location 2 is A+B, at location 3 is A+C, and at location 4 is A+B+C+D. The sum within D can be computed as 4+1-(2+3)

Using the integral image any rectangular sum can be computed in four array references (see Fig. 3). Clearly the difference between two rectangular sums can be computed in eight references. Since the two-rectangle features defined above involve adjacent rectangular sums they can be computed in six array references, eight in the case of the three-rectangle features, and nine for four-rectangle features.

B. Train Linear SVM Classifier

We use linear SVM other than RBF kernel SVM as our binary classifier because the number of HOG features is large, one may not need to map data to a higher dimensional space and linear SVM is usually computationally faster.

The training data include positive and negative examples which are fixed resolution image windows. Each positive window usually contains only one centered instance of the object, and negative windows are usually randomly sub-sampled and cropped from set of images not containing any instances of the object. Extract HOG feature from all the training data, obtain high-dimension feature vectors and then train the linear SVM classifier using these vectors, see Fig. 1.

Actually in the set of images not containing any instances of the object used in the training, running the preliminarily trained classifier would generate many false positives. To reduce false positives and make full use of the training images, we use the preliminary detector exhaustively scan the negative training images for hard examples (false positives), and then the classifier is re-trained using this augmented training set (original positives and negatives, and hard examples) to produce the final detector.

C. Vehicles Detection

During the detection phase, the binary window classifier is scanned across the test image at multiple scales. This typically produces multiple overlapping detections for each object instance. These detections need to be fused together. [17] proposes a solution based on representing detections in a position scale pyramid. Each detection provides a weighted point in this 3-D space and the weights are the detection's confidence score. A non parametric density estimator is run to estimate the corresponding density function and the resulting modes (peaks) of the density function constitute the final detections, with positions, scales and detection scores given by value of the peaks. In this sense we call this process as non-maximum suppression. In practice we use Gaussian kernel mean shift for the mode estimation.

Let $[x_i, y_i]$ and s_i denote the detection position and scale, respectively, for the i - th detection. The detection confidence score is denoted by $t(w_i)$. Let $y_i, i = 1 \ldots n$ be the set of detections ($y = [x, y, s]$ in 3-D position and scale space) generated by the detector. Assume that each point also has an associated symmetric positive definite 3×3 bandwidth or covariance matrix $\mathbf{H}_i$, defining the smoothing width for the detected position and scale estimate. Overlapping detections are fused by representing the n points as a kernel density estimate and searching for local modes. If the smoothing kernel is a Gaussian, the weighted kernel density estimate at a point y is given by (see [18-19])

$$\hat{f}(y) = \frac{1}{n(2\pi)^{3/2}} \sum_{i=1}^{n} |\mathbf{H}_i|^{-1/2} t(w_i) \exp\left(-\frac{\mathrm{D}^2[y, y_i, \mathbf{H}_i]}{2}\right) \quad (9)$$

Where

$$\mathrm{D}^2[y, y_i, \mathbf{H}_i] \equiv (y - y_i)^{\mathrm{T}} \mathbf{H}_i^{-1} (y - y_i) \quad (10)$$

is the Mahalanabois distance between y and y_i. In this paper uncertainty $\mathbf{H}_i$ is a diagonal matrix as follow

$$\mathrm{diag}[\mathbf{H}_i] = [(s_i\sigma_x)^2, (s_i\sigma_y)^2, (\sigma_s)^2] \quad (11)$$

where σ_x, σ_y and σ_s are user supplied smoothing values, and s_i is the i - th scale.

Let ϖ_i be the weights defined as

$$\varpi_i(y) = \frac{|\mathbf{H}_i|^{-1/2}\, t(w_i)\exp\left(-\mathrm{D}^2[y, y_i, \mathbf{H}_i]/2\right)}{\sum_{i=1}^{n} |\mathbf{H}_i|^{-1/2}\, t(w_i)\exp\left(-\mathrm{D}^2[y, y_i, \mathbf{H}_i]/2\right)} \qquad (12)$$

Let

$$\mathbf{H}_h^{-1}(y) = \sum_{i=1}^{n} \varpi_i(y)\mathbf{H}_i^{-1} \qquad (13)$$

Then the mode can be iteratively estimated by computing

$$y_m = \mathbf{H}_h(y_m)[\sum_{i=1}^{n} \varpi_i(y_m)\mathbf{H}_i^{-1} y_i] \qquad (14)$$

starting from some y_i until y_m does not change anymore.

III. Experiments and Results

The proposed algorithm is evaluated on a group of test images captured from several video stream shot downtown. The camera is Sony XR520, mounted near the rear-view mirror.

We use the MIT car data set as the positives [20] and selected negative images from INRIA data set as negatives [21]. The MIT car data set includes 516 128×128 images containing front and rear view cars. In experiments, we reflect these images and get 1032 images. 1804 128×128 windows randomly sub-sampled and cropped from the negative images not containing any instances of the cars used as negative examples in the preliminarily training procedure (see fig 4). The initial data sets are all transformed to high-dimension feature vectors and train the preliminary linear SVM classifier. Then re-train classifier to get the final vehicle detector after getting the hard examples.

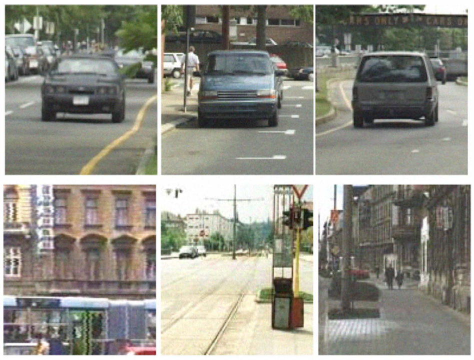
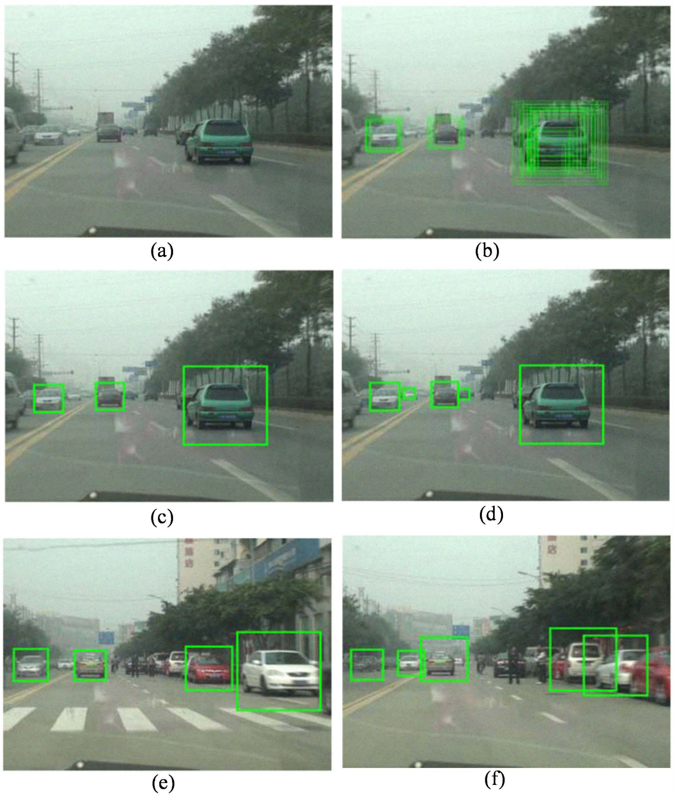


Fig. 4. Training examples. Positives(top line) and negatives(bottom line)

In the detection phase, the key point is deciding the scale range and scale step since the wider the scale range is and the smaller the scale step is, the better the detection result is but more time the computation costs. The scale range used in test is from 0.1 to 2.0, and the scale step is 0.1. It's notable that the scale somewhat decides the size of the vehicles detected, for example, the smaller the scale is, the bigger the size of the car detected in the test image is.

The other parameters are $\sigma_x = 32, \sigma_y$=32and$\sigma_s = 100$. Experimentally $\sigma_x$andσ_y affect the detection results remarkably as the change of their value, but σ_s is not so much.



(a)

(b)

(c)

(d)

(e)

(f)

Fig. 5. Vehicle detection examples in different backgrounds

Fig.5 shows some of the results of the detection examples using our algorithm. Fig.6 (a) is a test image, and (b) shows the original detection results which include many multiple overlapping detections. After applying the non-maximum suppression, the correct and accurate detections are obtained as shown in Fig.5 (c). Here if alter the maximum scale from 2.0 to 3.0, we get finer detections that the farther and smaller cars are detected in Fig.5 (d). It's notable that the right small car is not detected because its gray value is similar to the tree. Fig 5(e) and (f) show other test results, and we could see that the detector is somewhat robust to occlusion and different views.

IV. Conclusions

In this paper we have presented a system that uses a single frontal camera for vehicle detection based on HOG. Experimental results show that this system is effective and robust, can achieve a high reliability target detection with low false positive rate in demanding situations such as complex urban environments, local occlusion and little side view.

References

[1] J. C. McCall and M.M. Trivedi, “Video-Based Lane Estimation and Tracking for Driver Assistance: Survey, System, and Evaluation,” *IEEE Trans. Intelligent Transportation Systems*, vol. 7, no. 1, pp. 20-37, March 2006.

[2] J.-F. Liu, Y-F Su, M-K Ko and P-N Yu, “Development of a Vision-Based Driver Assistance System with Lane Departure Warning and Forward Collision Warning Functions,” in *Conf. Rec. 2008 IEEE Digital Image Computing: Techniques and Applications,* pp. 480–485.

[3] Z-H Sun, G Bebis and R Miller, “On-Road Vehicle Detection: A Review,” *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 28, no. 5, pp. 694-711, May 2006.

[4] M Bertozzi, A Broggi and A Fascioli, “Vision-Based Intelligent Vehicles: State of the Art and Perspectives,” *Robotics and Autonomous Systems,* vol. 32, pp. 1-16, 2000.

[5] M B van Leeuwen and F C.A. Groen, “Vehicle Detection with a Mobile Camera Spotting Midrange Distant and Passing Cars,” *IEEE J. Robotics and Automation Magazine*, pp. 37-43, March 2005.

[6] A. Giachetti, M. Campani, and V. Torre, “The Use of Optical Flow for Road Navigation,” *IEEE Trans. Robotics and Automation*, vol. 14, no. 1, pp. 34-48, 1998.

[7] U. Handmann, T. Kalinke, C. Tzomakas, M. Werner, and W. Seelen, “An Image Processing System for Driver Assistance,” *Image and Vision Computing*, vol. 18, no. 5, 2000.

[8] A. Bensrhair, M. Bertozzi, A. Broggi, P. Miche, S. Mousset, and G. Moulminet, “A Cooperative Approach to Vision-Based Vehicle Detection,” in *Conf. Rec. 2001* IEEE *Intelligent Transportation Systems*, pp. 209-214.

[9] N. Matthews, P. An, D. Charnley, and C. Harris, “Vehicle Detection and Recognition in Greyscale Iimagery,” *Control Eng.Pract*, vol. A, no. A, pp. a73-479, 1996.

[10] C. Georick, N. Detlev, and M. Werner, “Artificial Neural Networks in Real-Time Car Detection and Tracking Applications,” *Pattern Recognition Letters*, vo1. 17, no. A, pp. 335-343, 1996.

[11] C. Papageorgiou, and T. Poggio, “A Trainable System for Object Detection,” *International Journal of Computer Vision*, vol. 38, no.1, pp. 15-33, 2000.

[12] S. Thomas, W. Lior, B. Stanley, et al. “Robust Object Recognition with Cortex-Like Mechanisms,” *IEEE Tran. on Pattern Analysis and Machine Intelligence*, vol.29, no.3, March 2007.

[13] Z. Sun, B. George, and M. Ronald, “On-Road Vehicle Detection Using Evolutionary Gabor Filter Optimization,” *IEEE Tran. on Intelligent Transportation Systems*, vol.6, no.2, June 2005.

[14] D. Navneet, and T. Bill, “Histograms of Oriented Gradients for Human Detection,” *Computer Vision and Pattern Recognition*, vol.1, pp. 886-893, June 2005.

[15] P. Viola and M. Jones, “Rapid object detection using a boosted cascade of simple features,” *Computer Vision and Pattern Recognition*, 2001.

[16] F. Porikli, “Integral Histogram: A Fast Way to Extract Higtograms in Cartesian Spaces,” *Computer Vision and Pattern Recognition*, 2005.

[17] D. Navneet, “Finding People in Images and Videos,” Ph.D. dissertation, 2006.

[18] D. Comaniciu and P. Meer, “Mean Shift: A Robust Approach toward Feature Space Analysis,” *IEEE Tran. on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 603–619, 2002.

[19] D. Comaniciu, “An Algorithm for Data-Driven Bandwidth Selection,” *IEEE Tran. on Pattern Analysis and Machine Intelligence*, vol. 25, no. 2, pp. 281–288, 2003b.

[20] http://cbcl.mit.edu/software-datasets/CarData.html.

[21] http://lear.inrialpes.fr/data.