Vehicle and Pedestrian Detection Using Support Vector Machine and Histogram of Oriented Gradients Features

Zhiqian Chen [1], Kai Chen [2], James Chen [3]
[1] Department of Software Engineering, Peking University, Beijing, China
[2] School of Automation Engineering, Northeast Dianli University, Jilin, China
[3] Department of Computer Science, University of California, Los Angeles, USA
imczq@pku.edu.cn, zjcxck2jly@sina.com, James.chen@gmail.com

Abstract—**Vehicle and Pedestrian Detection is a key problem in computer vision, with several applications including robotics, surveillance and automotive safety. Much of the progress of the past few years has been driven by the availability of challenging public datasets. In this paper, we build up a vehicle and pedestrian detection system by combing Histogram of Oriented Gradients (HoG) feature and support vector machine (SVM). HoG feature provides a reasonable and feature invariant object representation, while SVM framework gives us a robust classifier that can control both the training set error and the classifier's complexity. A detailed system architecture design is presented and the testing experiments show that high performance in both accuracy and speed can be achieved by the developed system.**

Keywords-vehicle detection; pedestrian detection; support vector machine; histogram of oriented gradient; computer vision

I. Introduction

Detecting people in images is a problem with a long history [15, 16, 17, 18, 19, 20]; In the past two years there has been a surge of interest in vehicle and pedestrian detection [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]. Accurate vehicle and pedestrian detection would have immediate and far reaching impact to applications such as surveillance, robotics. Because of the rise in the popularity of automobiles over the last century, road accidents have become an important cause of fatalities. Therefore, the automotive applications [11, 12] embedded with vehicle and pedestrian detection are particularly compelling as they have the potential to save numerous lives.

However, due to the wide variety of appearances of objects, such as view point, occlusions, lighting and backgrounds, vehicle and pedestrian detection is a very difficult and challenging task. Generally the challenges are summarized by the following points:

(1) The appearance of pedestrians exhibits very high variability since they can change pose, wear different clothes, carry different objects, and have a considerable range of sizes. Similarly the occurrence of vehicle can be under different view point. This problem will be solved by learning mixture of model in our system to capture multi-view object.

(1) Pedestrians and vehicle must be identified in outdoor urban scenarios, that is, they must be detected in the context of a cluttered background (urban areas are more complex than highways) under a wide range of illumination and weather conditions that vary the quality of the sensed information (e.g., shadows and poor contrast in the visible spectrum). Therefore, an invariant feature that can well represent object is very important.

(3) The required performance is quite demanding in terms of system reaction time and robustness (i.e., false alarms versus misdetections). Yet, machine learning technique have seen great improvement, which means that some important classifiers learning methods, such as support vector machine (SVM) [14], boosting technique, and deep learning, have obtained state-of-art performance in object classification.

(4) The detection speed is also crucial in terms of system reaction time. Computational complexity is often a big issue in the field of image processing, machine learning, and pattern recognition. However, processing speed in object detection has recently seen great progress, enabling real-time operation without sacrificing quality. GPU device and parallel computing technique are very helpful to improve the ability of image processing in detection process.

In this paper, we will build up a vehicle and pedestrian detection system. In terms of representation, we use Histogram of Oriented Gradients (HOG) features [13], which are feature descriptors used in computer vision and image processing for the purpose of object detection. The technique counts occurrences of gradient orientation in localized portions of an image. This feature is similar to that of edge orientation histograms, scale-invariant feature transform descriptors, and shape contexts, but differs in that it is computed on a dense grid of uniformly spaced cells and uses overlapping local contrast normalization for improved accuracy. [13] HOG feature allows the system to identify the important characteristics of the people class while ignoring some variance in location and in the pixel-level representations. Additionally, using a large set of positive and negative examples, we train a support vector machine (SVM) [14] classifier to differentiate between target and non-target. Here we will model three targets for the system: people, front view cars and side-view cars. Because the difference between front-view and side-view cars is obvious, it is reasonable necessary to train two separate classifiers for the car class.

978-0-7695-5125-8/13 $26.00 © 2013 IEEE
DOI 10.1109/CSA.2013.92

To detect people or car in images, our core static detection system implements a brute force search to match each possible position to find the occurrence of the targets. It does not rely on motion or tracking and makes no assumptions about the scene structure, number of people in the scene, or camera movement. It directly applies this static approach or our trained model to video sequences.

The rest of this article is organized as follows. In Section 2, we will present the system architecture of our system, which includes the introduction of HoG feature, support vector machine, and sliding widow detection technique. In Section 3, the implementation and experiment are presented. Finally, section 4 presents a summary of the paper.

II. System architecture

A. Histograms of Oriented Gradients

Histogram of Oriented Gradients (HOG) [13] features are a trending topic in object detection literature. HOG features are a robust way of describing local object appearances and shapes by their distribution of intensity gradients or edge directions, and have been used successfully as a low level feature in many object recognition tasks.

The HOG feature is related to the SIFT feature descriptor. SIFT is computed in sparse set of interest points, while HOG is intended to be run over a dense grid. HOG feature is implemented by the following way: First, the 2D gradient of the image is computed using a vertical and horizontal filter. Secondly, the image is divided into M cells of $N \times N$ pixels. A histogram with H bins is computed and normalized given the weighted gradient at each pixel, for each of the cells. Lastly, the concatenation of the histograms from each cell gives us a $N \times M$ length feature vector that represents the image. The figure 1 shows the architecture of computing HOG feature. The figure 2 shows us a test image and the visualization of its HOG feature.

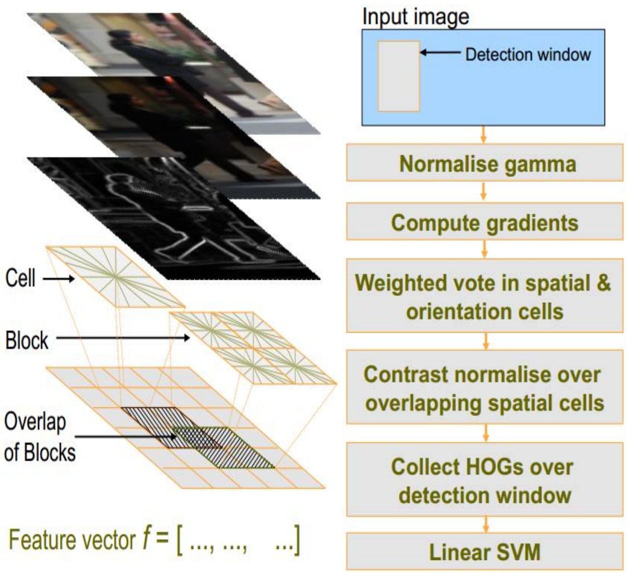
Cell
Block
Overlap of Blocks
Feature vector f = [..., ..., ...]
Input image
Detection window
Normalise gamma
Compute gradients
Weighted vote in spatial & orientation cells
Contrast normalise over overlapping spatial cells
Collect HOGs over detection window
Linear SVM

Figure 1. HOG feature

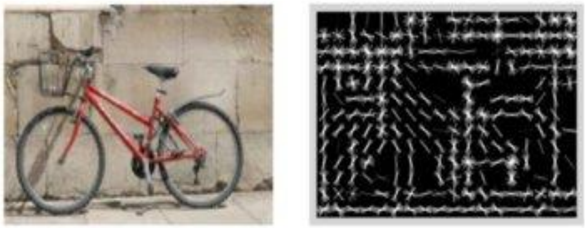

Figure 2. Testing image and its HOG feature

The HOG descriptor has a few important advantages over other descriptor methods. Because the HOG feature operates on localized cells, the method upholds invariance to geometric and photometric transformations. Also Dalal and Triggs [13] discovered, coarse spatial sampling, fine orientation sampling, and strong local photometric normalization permits the individual body movement of pedestrians to be ignored so long as they maintain a roughly upright position. Thus, the HOG feature is particularly suited for human detection. Actually, HOG feature perform very well in the object detection and classification, which has been justified by a lot of works in the past few years.

B. Support vector machine classification

Support vector machine (SVM) is a principled technique to train classifiers that are well-founded in statistical learning theory. [13] Unlike traditional training algorithms like back propagation which only minimizes training set error, one of the main attractions of using SVMs is that they minimize a bound on the empirical error and the complexity of the classifier, at the same time. Therefore, they are capable of learning in sparse, high dimensional spaces with relatively few training examples.

Using the SVM formulation, the classification step for a pattern using a polynomial of degree two is as follows:

$$f(\mathbf{x}) = \theta \left(\sum_{i=1}^{N_s} \alpha_i y_i (\mathbf{x} \cdot \mathbf{x}_i + 1)^2 + b \right) \qquad (1)$$

where N_s is the number of support vectors (i.e. training data points that define the decision boundary), and α_i are Lagrange parameters. The details of SVM can be found in paper [13]. Although the SVM is based on a linear discriminator, it is not restricted to making linear hypotheses. It is possible to make non-linear decisions by a non-linear mapping of the data to a higher dimensional space. The phenomenon is analogous to folding a flat sheet of paper into any three dimensional shape and then cutting it into two halves, the resultant non-linear boundary in the two dimensional space is revealed by unfolding the pieces. The SVM is non-parametric mathematical formulation allows these transformations to be applied efficiently and implicitly: the SVM's objective is a function of the dot product between pairs of vectors; the substitution of the original dot products with those computed in another space eliminates the need to transform the original data points explicitly to the higher space. The computation of dot products between vectors without explicitly mapping to another space is performed by a kernel function. There are several common kernel functions that are used such as the linear, polynomial kernel, Gaussian kernel and the sigmoidal kernel.[13]

C. Detecting vehicle and pedestrian with sliding window

The sliding window model for detection purpose is conceptually simple and natural, which independently classifies all image patches within the testing image as being object or non-object. Sliding window classification is the dominant paradigm in object detection and it is one of the most noticeable successes of computer vision in detecting face. For example, modern cameras and photo organization tools have face detection function. In our vehicle and pedestrian detection, we also use this sliding window model due to its simplicity and power.

To detect pedestrians or vehicle in a new image, we shift the detection window over all locations in the image. This will only detect target at a single scale, however, to achieve multi-scale detection, we incrementally resize the testing image and run the detection window over each of these resized images, which is equivalent to using resized detecting window to do detection within a fixed size of image in order to locate a potential target with different scales. It is important to reiterate that no motion or tracking is used. This brute force search over the image is quite time consuming. Using GPU and parallel computing technique can reduce the computation expense.

The ability to compute HOG in real time is directly related to being able to decompose the image and work on individual cells simultaneously. The main reason for the slowdown comes from sliding window approach which re-computes cells overlapped from previous windows. Caching these results would significantly improve our run time.

III. Implementation and Experiments

We learn three SVM classifiers: one for pedestrian, and two for cars at two viewing angles (45 degree and side views) respectively. Some training positive examples are shown in Figure 3 (45 degree view cars), Figure 4 (side view cars), and Figure 5 (pedestrians) respectively. Detailedly, we used the HOG (Histogram of Oriented Gradients) feature to represent each image patch, and then train the SVM for classification. Given a set of labeled training images (pedestrians, 45 degree cars and side view cars), a window around the object to be detected can be extracted, and HOG computed over it. These HOG feature vectors are then used to train a SVM classifier. We used the LIBSVM package [21] for SVM training, and implemented our own SVM classifier as a kernel on the GPU. We collect our own training date set of cars and pedestrians. This gave us a training set of more than 500 positive and 1000 negative examples. To perform detection, we took a sliding window approach, computing the HOG feature for each window and passing it as input to our SVM classifiers. Our implementation of HOG runs in real time: 0.06 seconds to compute a HOG feature with 16 by 16 cells and 8 bins per histogram for a 640 by 480 frame.

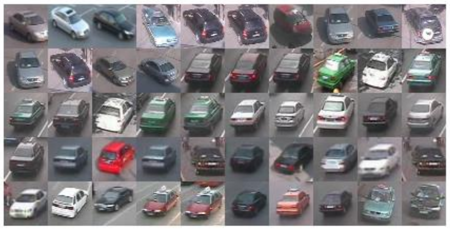

Figure 3. Training positives for 45 degree view cars

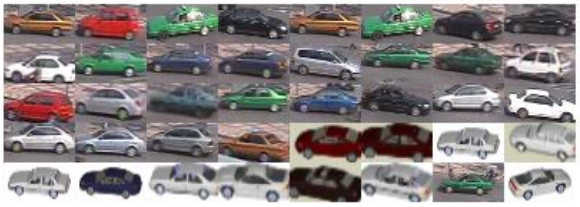

Figure 4. Training positives for side view cars

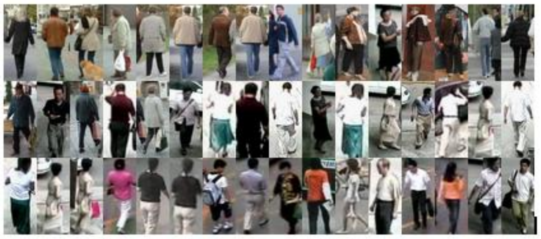

Figure 5. Training positives for pedestrians

One obvious attraction of SVM framework for our detection system is that it controls both the training error and the complexity of the decision classifier at the same time. This can be contrasted with other training techniques like back propagation, which only minimize training error. Because there is no controlling of the classifier complexity in this type of system, such as back propagation, it will tend to overfit the data and provide poor generalization.

We made significant effort in collecting a large number of positive examples for pedestrians and two types of vehicles. What if we only had a small number of training positive for our detection problem? Figure 6 quantifies the performance of our detection system in 1200 testing pedestrian examples, when trained on 1, 10, and the full set of 1300 people, each using the same set of 1000 negative training points. The figure reports the average performance in 15 runs. We can find that even with a single example of our positive class, the SVM finds a decision surface that performs quite well. With 10 positive examples, the performance almost approaches that of the system trained with the full data set of 1300 positive examples. The same conclusion and experimental results happened in the other two vehicle detection.

In order to test our system, we use some image frame from video surveillance. The detection results of 8 testing image are shown in Figure 7. From the result we may find that the performance of detecting pedestrians and cars are pretty good and works stable. Even though the environment surrounding the target object may easily influence the detection, (e.g. the colors of some cars are similar to their background which may easily result in false negative), our models can still give us a satisfied results and performance.

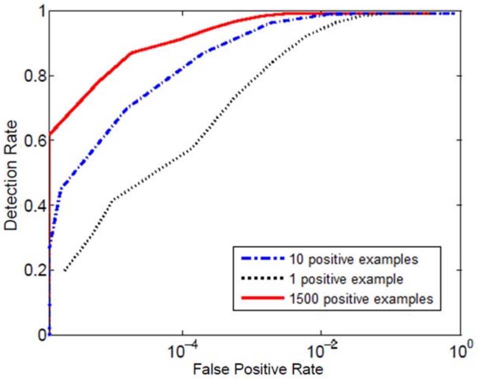

Figure 6. Training with few positive examples

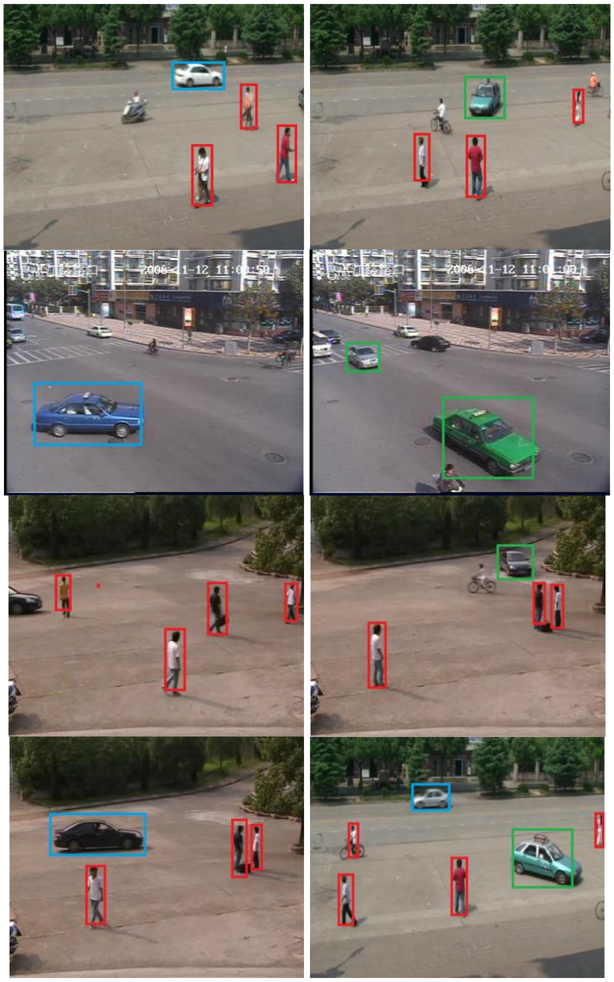

Figure 7. Detection results

IV. CONCLUSIONS

Vehicle and pedestrians detection is one of the main challenges of computer vision. Machine learning has been shown to be an effective technique for object detection. This paper builds a vehicle and pedestrian detection system by using SVM algorithm and HOG features. The experiments show high performance in both accuracy and speed of the developed system.

REFERENCES

[1] P. Dollar, B. Babenko, S. Belongie, P. Perona, and Z. Tu. “Multiple component learning for object detection”. In ECCV, 2008.

[2] A. Ess, B. Leibe, K. Schindler, and L. van Gool. “A mobile vision system for robust multi-person tracking”. In CVPR, 2008.

[3] P. Felzenszwalb, D. McAllester, and D. Ramanan. “A discriminatively trained, multiscale, deformable part model”. In CVPR, 2008.

[4] B. Leibe, N. Cornelis, K. Cornelis, and L. V. Gool. “Dynamic 3D scene analysis from a moving vehicle”. In CVPR, 2007.

[5] Z. Lin and L. S. Davis. “A pose-invariant descriptor for human detection and segmentation”. In ECCV, 2008.

[6] S. Maji, A. Berg, and J. Malik. “Classification using intersection kernel SVMs is efficient”. In CVPR, 2008

[7] S. Munder, C. Schňorr, and D. Gavrila. “Pedestrian detection and tracking using a mixture of view-based shape-texture models”. In IEEE Transactions on Intelligent Transportation Systems, 2008.

[8] P. Sabzmeydani and G. Mori. “Detecting pedestrians by learning shapelet features”. In CVPR, 2007.

[9] E. Seemann, M. Fritz, and B. Schiele. “Towards robust pedestrian detection in crowded image sequences”. In CVPR, 2007.

[10] V. Sharma and J. Davis. “Integrating appearance and motion cues for simultaneous detection and segmentation of ped”. In ICCV, 2007.

[11] D. M. Gavrila and S. Munder. “Multi-cue pedestrian detection and tracking from a moving vehicle”. IJCV, pages 41–59, 2007.

[12] D. Geronimo, A. Sappa, A. Lopez, and D. Ponsa. “Adaptive image sampling and windows classification for on-board pedestrian detection”. In Inter. Conf. on Computer Vision Systems, 2005.

[13] N. Dalal and B. Triggs. “Histograms of Oriented Gradients for Human Detection”. CVPR, 2005.

[14] Cortes, Corinna; and Vapnik, Vladimir N.; "Support-Vector Networks", Machine Learning, 20, 1995.

[15] T. Tsukiyama and Y. Shirai. “Detection of the movements of persons from a sparse sequence of tv images”. PR, 18(3-4):207–213, 1985.

D. M. Gavrila and V. Philomin. “Real-time object detection for “smart” vehicles”. In ICCV.1999.

[16] Y. Song, X. Feng, and P. Perona. “Towards detection of human motion”. In CVPR, 2000.

[17] C. Papageorgiou and T. Poggio. “A trainable system for object detection”.IJCV, 38(1):15–33, 2000.

[18] S. Ioffe and D. A. Forsyth. Probabilistic methods for finding people.IJCV, pages 45–68, 2001.

[19] P. Viola, M. Jones, and D. Snow. “Detecting pedestrians using patternsof motion and appearance.” In CVPR, 2003.

[20] K. Mikolajczyk, C. Schmid, and A. Zisserman. “Human detection based on a prob. assembly of robust part det”. In ECCV, 2004.

[21] Chih-Chung Chang, Chih-Jen Lin. “LIBSVM: A library for support vector machines”. ACM TIST 2(3): 27 (2011).