Vision-based Bicyclist Detection and Tracking for Intelligent Vehicles

Hyunggi Cho Paul E. Rybski Wende Zhang

CMU-RI-TR-10-11

January 2010

Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

Abstract

This paper presents a vision-based framework for intelligent vehicles to detect and track people riding bicycles in urban traffic environments. To deal with dramatic appearance changes of a bicycle according to different viewpoints as well as nonrigid nature of human appearance, a method is proposed which employs complementary detection and tracking algorithms. In the detection phase, we use multiple view-based detectors: frontal, rear, and right/left side view. For each view detector, a linear Support Vector Machine (SVM) is used for object classification in combination with Histograms of Oriented Gradients (HOG) which is one of the most discriminative features. Furthermore, a real-time enhancement for the detection process is implemented using the Integral Histogram method and a coarse-to-fine cascade approach. Tracking phase is performed by a multiple patch-based Lucas-Kanade tracker. We first run the Harris corner detector over the bounding box which is the result of our detector. Each of the corner points can be a good feature to track and, in consequence, becomes a template of each instance of multiple Lucas-Kanade trackers. To manage the set of patches efficiently, a novel method based on spectral clustering algorithm is proposed. Quantitative experiments have been conducted to show the effectiveness of each component of the proposed framework.

Contents

1 INTRODUCTION

Research on safety design of vehicles has focused on protecting drivers and passengers from accidents. Many concepts and devices have been developed, from new types of safety airbag and electronic equipment to intelligent driving assistance systems [12]. In the last few years, however, the trend of research has been extended to protecting vulnerable road users (VRUs) such as pedestrians, bicyclists, two wheelers, and other small vehicles [8]. This can be regarded as a natural trend to enrich total driving safety. Among these VRU's, as shown in Figure 1, pedestrians and bicyclists are the weakest traffic participants because there is no special protection device or mechanism against the consequences of accidents (save for helmets worn by bicyclists). For this reason, accurate and real-time pedestrian and bicyclist detection techniques have emerged as a hot research topic in the field of computer vision and Intelligent Transportation Systems (ITSs), and a great variety of approaches have been proposed in the research community (see Section 2 for more detail). To this end, different approaches use different sensors such as an ultrasonic sensor, thermopile sensor, laser scanner, microwave radar and cameras, and sometimes their fusion is exploited to result in more robust detection [8]. Since every sensor technology has its own advantages and limitations, sensor fusion in certain levels has recently been considered as a promising and desirable approach [13].

Figure 1: Examples of Vulnerable Road Users. Among these VRU's, pedestrians and bicyclists are the weakest traffic participants because there is no special protection device or mechanism against the consequences of accidents.

For our work, we are considering the use of video cameras as the primary sensor system for detecting and tracking bicyclists. Cameras are attractive in that they not only capture high-resolution views of scenes that include both color and texture information, but also in general are inexpensive as compared to other sensor technologies such as LIDAR or RADAR. However, despite their attractive aspects, vision-based bi-

cyclist and pedestrian detection is still a challenging problem due to the fact that people can appear quite different from each other due to differences in clothing/hairstyle, body pose, as well as motion. Real-world outdoor environments are complex and fluid and include cluttered backgrounds, changing illumination, and variable weather conditions which can further complicate the detection and tracking problem. Furthermore, because this application is for supporting autonomous vehicles, the sensors are mounted on moving platforms which once again increases the complexity. To tackle these difficulties, many interesting and promising vision techniques have been proposed from the computer vision and ITS communities. Some of this work is used already in practical real-time pedestrian detection systems [10], [1]. However, these systems mainly focus on pedestrians, not bicyclists; indeed there is a comparative lack of research about bicyclist detection and tracking. While these two problems share many common features, the bicyclist detection problem shows more challenging aspects, including dramatic appearance changes of a bicycle according to its viewpoints, and fast motion compared to a pedestrian. Therefore, our proposed bicyclist detection method is based on a robust shape feature extraction algorithm and is coupled to a computationally efficient tracking algorithm. The contributions of this work are summarized into the following two aspects:

Fast bicyclist detector: to deal with dramatic appearance changes of a bicycle according to viewpoints and, at the same time satisfy real-time constraints our application domain needs, we use four view-based detectors: frontal, rear and right/left side view. For each detector, we implement a Histogram Oriented Gradients (HOG) based detector [4] and apply it as a building block to our bicyclist detection/tracking framework. One problem of the HOG based detector is its slow performance, which exists for two reasons. First, the HOG descriptor basically uses a dense encoding scheme of the image region (or template). Second, it has to search for interesting objects in multi-level scale images. To solve this problem, we follow the approach proposed by *Zhu* and *Avidan* [25], applying two methods to speed up the HOG based detector. The first one is to use the concept of "Integral Histogram" [17] to speed up the feature extraction process. The other method is to use a boosting algorithm [7] to speed up the classification process. We use AdaBoost to select the best features and construct a cascade of classifiers.

Multiple patch-based Lucas-Kanade tracker: in order to efficiently deal with articulate bicyclist shape and obtain the trajectory of bicycles, we apply a multiple patch-based Lucas-Kanade tracker to our framework. First, good features to track are detected by running the Harris corner detector over the bounding box returned by our detector. Each of the corner-like points becomes a template of each instance of multiple Lucas-Kanade trackers. Since we cannot guarantee that all the multiple patches are necessarily found on the bicyclist, we need a high level patch management scheme not only to find outliers (patches from the background) but also to evolve the topology between multiple patches in an on-line fashion. Here, we propose a new method based on spectral clustering algorithm to control geometric constraints of multiple patches.

The structure of this paper is as follows. We begin with Section 2 by reviewing the previous work on the pedestrian detection and tracking problem. This is due to the fact that there is a lack of research on bicycle detection and tracking and we believe our problem is most related to the problem of pedestrians. In Section 3 we formulate

the detection problem in a static image and give the details of our view-based detectors. We then discuss the tracking problem in Section 4. The experimental results and comparisons are presented in Section 5. We conclude in Section 6.

2 RELATED WORK

Many interesting vision-based approaches for pedestrian detection and tracking have been proposed. Here, we only focus on research using a monocular camera in the visible spectrum. Thus, we omit work related to the use of infrared cameras and stereo vision. For the earlier work on this topic, refer to the surveys of *Gavrila* [9] and *Li* et al. [12]. More comprehensive surveys, including the most recent research efforts in the field, can be found in [8], [6], [5].

For the detection of pedestrians, various combinations of features and classifiers can be applied to recognize a pedestrian. Selecting the correct feature is important because overall performance of the system depends on the discriminative power of features used in detection algorithm. Recent research shows three main features: appearance, shape, and motion. Some of the features used for appearance-based detection are Haar wavelets [15], and Gabor filter outputs [2]. In [22], texture information is extracted using simple masks (called Haar-like features), and classification is performed based on integrating the weak classifiers obtained from these masks. As for the shape-based features, symmetry, edge template [10], histogram of oriented gradients (HOG) [4], [25], edgelet [24], and shapelet [18] have been exploited. Motion is also an important cue in detecting pedestrians. However, in the case of cameras installed on a moving vehicle, it is not easy to find independent moving objects. Thus, a more complicated method is required to compensate for ego-motion of the vehicle. Compared to motion cue, the beauty of shape-based approach is that it can recognize both moving and stationary pedestrians. In addition, the discriminative power of shape-based features is usually stronger than that of appearance-based features. For the tracking of pedestrian, a number of mathematical frameworks have been proposed. Kalman filter or particle filter-based methods, mean-shift algorithm, and optical flow-based methods are the most frequently used frameworks for such tasks. In [10], *Gavrila* and *Giebel* used an $\alpha - \beta$ tracker to overcome gaps in detection. Indeed, the tracker is a simplified Kalman filter with a constant velocity model and predetermined steady-state gains. Particle filters have shown robust performance in handling non-Gaussianity and non-linearity. *Smith* et al. [20] used a particle filter successfully to track a variable number of interacting people using a fixed camera. In [3], *Comaniciu* and *Meer* used a color histogram computed from a circular region as a representation of an object. Instead of performing an intensive search for locating the object, they use the mean-shift procedure. The mean-shift tracker maximizes the appearance similarity iteratively by comparing the histograms of the object and the candidate regions in the next image. Another approach to track a region is to compute its translation by using an optical flow-based method. Optical flow methods are used for generating dense flow fields by computing the flow vector of each pixel under the brightness constancy constraint. This computation is always carried out in the neighborhood of the pixel either algebraically [14] or geometrically. In [19], *Shi* and *Tomasi* proposed the KLT tracker

which iteratively computes the translation (du, dv) of a region (e.g., 25×25 patch) centered on an interest point. Once the new location of the interest point is obtained, the KLT tracker evaluates the quality of the tracked patch by computing the affine transformation between the corresponding patches in consecutive frames. If the sum of square difference between the current patch and the projected patch is small, they continue tracking the feature, otherwise the feature is eliminated.

3 FAST BICYCLIST DETECTION

The goal of the detection process is to recognize bicyclists and find their exact location from a static image. One of the most common solution is to define object detection as a binary classification problem using a traditional sliding window approach. The sliding window method evaluates a sub-image at multiple different scales and locations over the images. At each location, features are extracted from the sub-image and the classifier is run on these features to check whether the region contains interest objects. Naturally, next fundamental questions are "What features are most discriminative ?", "How can we speed up the whole detection process ?",and "Can the features cover all variability in appearance ?" We will discuss each of these questions and give our best answers in the following subsections.

3.1 Integral HOG Features

For the answer to the first question, a number of features have been explored. Some important features are discussed in Section 2. According to the recent comprehensive evaluation studies [5], [6], it is shown that histograms of oriented gradients (HOG) still shows best performance as a single feature relative to other existing feature sets.

In [4], *Dalal* and *Triggs* proposed a dense encoding scheme of local histograms of oriented gradients (HOG). The aim of this method is to describe an image by a set of local histograms. These histograms count occurrences of gradient orientation in a local part of the image. More specifically, feature extraction is implemented by dividing the image into small spatial regions (or "cells"). For each cell a local 1-D histogram of gradient directions is accumulated over the pixels found in that cell. To make the method invariant to illumination and shadowing, the authors also normalize the local responses. The HOG feature descriptor as an object representation has been used successfully to classify objects in combination with a linear SVM. However, dense HOG representation is unfortunately computationally too intensive for a real-time application. To solve this problem, *Zhu* and *Avidan* [25] proposed a novel method by exploiting the concept of Integral HOG. The Integral HOG is an extension of original HOG features for a fast evaluation. It is inspired by "integral image" [21] which allows very fast extraction of Haar-like features and the "Integral Histogram" [17] which allows efficient histogram computation over arbitrary rectangular image regions.

Following their work, we exploit a fast method of calculating the HOG features. The first difference with original HOG representation comes from how the result of gradients of the image is saved. For each bin of the HOG, an integral image is computed and is saved separately. Since we use nine orientation bins, nine integral images are

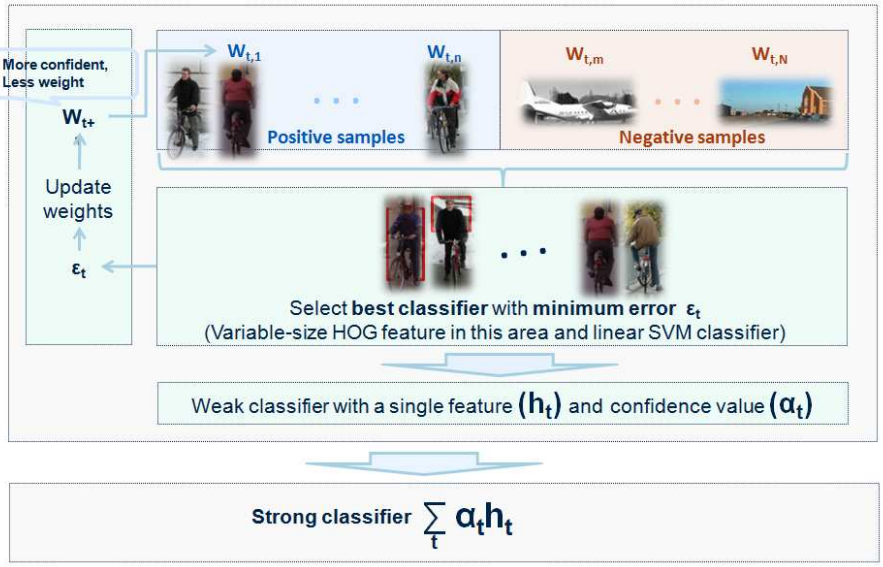


Figure 2: Feature selection process using AdaBoost

constructed. These integral images are used later to compute efficiently the HOG for any rectangular image region only with 4×9 image access operations.

3.2 AdaBoost Classifier

Both fast feature extraction and fast classification are crucial factors for the second question. We already have the half of the solution with the use of Integral HOG features. The second half of the solution comes from the idea proposed by *Viola* and *Jones* [21]. They used a variant of AdaBoost learning technique to find the best set of Haar-like features and to construct a cascade of classifiers. Although this rejector-based cascade is still run in the sliding window manner, it dramatically speeds up the detection process by focusing attention on more promising regions of the image. In other words, the goal of constructing a cascade is to match the complexity of a classifier that operates over a small number of features with the performance of a classifier that operates over a very large number of features. Similarly, *Zhu* and *Avidan* [25] followed the same approach with the Integral HOG features by varying the size of blocks, which is another key difference compared to the original HOG representation and a linear SVM as a weak classifier. While fixed-size blocks (typically, 16×16 pixels) are used in the original HOG, in the Integral HOG case, variable-size blocks are used instead. Combined with constructing a rejector-based cascade, weak classifiers with fixed-sized blocks are not informative enough to allow fast rejection in the early stages of the cascade. Thus, they use a much larger set of blocks that vary in size, location and aspect ratio and then use AdaBoost to select the best feature to be evaluated in each

stage, where each feature corresponds to one block. Then, they construct a cascade of classifiers using weak classifiers associated with these features.

In our system, there are over 2,909 variable-size blocks associated with frontal view detection window because we use a 64×128 detection window and consider variable block size ranges from 12×12 to 64×128 and width/height ratios $(1:1)$, $(1:2)$, and $(2:1)$. Even though the number of all possible variable-size blocks is very large, the primary assumption of AdaBoost, which has been proven empirically, is that a very small number of these features can be combined to form an effective classifier [21]. A graphical summary of the boosting process is shown in Figure 2. In the AdaBoost algorithm, each round of boosting selects one feature from the 2,909 potential features. It means that we can select one best classifier that minimizes the overall error. Afterwards, we re-weight all the data to focus on the mistakes. In our next iteration, we can find the next best classifier based on the weighted data. Finally, we construct a cascade of classifiers by combining all the classifiers at the end according to their confidence.

3.3 View-based Detector

Our last question is that how we can deal with dramatic appearance changes of a bicyclist according to its viewpoints without violating the real-time constraints. HOG representation of some viewpoints of a bicyclist is visualized in Figure 3. In this paper, we propose to use four view-based detectors: frontal, rear and right/left side view. Of course, these four view Integral HOG representation of a bicycle cannot cover all variability in appearance, but we believe that this is a reasonable approach in that four views can capture pretty much of its characteristics and adding another view detector (i.e. 45^o or 135^o) does not improve detection accuracy enough to compensate additional computation. To support this argument, a comparison experiment between six view detector (including 45^o and 135^o views) and our four view detector is conducted and discussed in Section 5. For each view detector, as discussed in the previous subsection, the main concern is to find a set of variable-size blocks which maximize overall classification accuracy.

4 FAST BICYCLIST TRACKING

Once the bicyclist has been detected in the image, the next step is to track his/her position from frame to frame. Because of the relatively high cost of the detector, we are interested in finding an algorithm with a lower complexity in order to do tracking. Tracking exploits motion-related temporal constraints to find the correspondence of moving objects in the image sequence. To this end, several mathematical frameworks have been proposed (this is roughly discussed in Section 2). After performing a comparative investigation of these existing tracking techniques, we chose to apply a traditional Lucas-Kanade tracker [14] to our framework. The reason for this decision is that it can be integrated with our high-level patch management scheme to shows promising performance in general settings and various efficient extensions of the algorithm have been proposed to allow its real-time implementation.

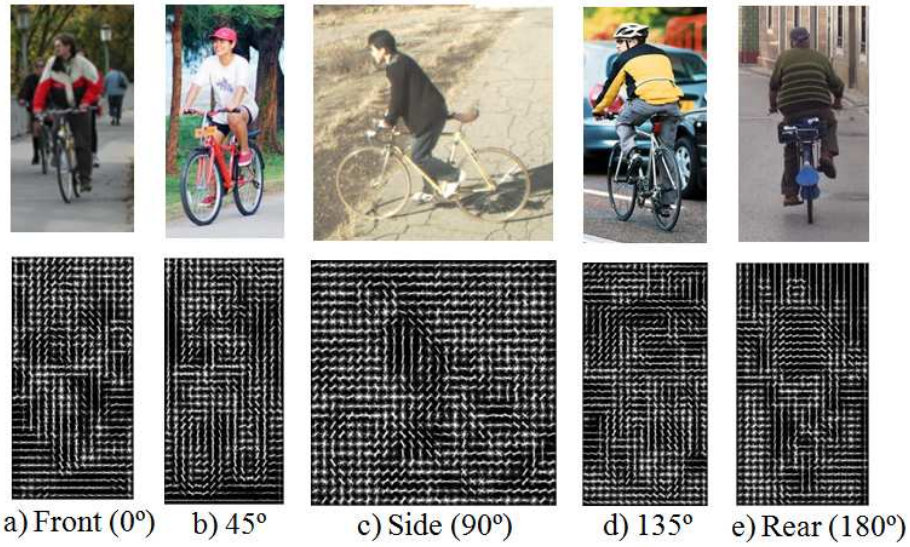


Figure 3: HOG representation of each viewpoint of a bicyclist

4.1 Multiple Patch-based Lucas-Kanade Tracker

The Lucas-Kanade tracker is one of the most popular versions of two-frame differential methods for motion estimation. The goal of the Lucas-Kanade algorithm is to compute optical flow by minimizing the sum of squared error between two subsequent images in the video sequence: the template T and the image I warped back onto the coordinate frame of the template. In the case of bicyclist tracking, the template is a region containing a bicyclist which is generated by the detection process. To deal with the fact that bicyclists are non-rigid (the person's legs are typically in constant motion, and the appearance of the bicycle changes drastically between frontal and side views), we propose a multiple patch-based approach of the Lucas-Kanade algorithm. Rather than using one big template, we find a set of good features using a Harris corner detector [11] and then try to track each of these multiple small patches independently using the Lucas-Kanade algorithm. However, as illustrated in Figure 4(a), we cannot guarantee that all the multiple patches are necessarily found on the bicyclist. While most of them are on the bicyclist, showing similar optical flow vectors, some of them are on a background, showing quite different motion vectors. Thus, an additional step is required to filter out these unnecessary patches. We propose a novel mechanism that we will describe below.

4.2 Control Scheme of Multiple Patches

We propose a new method to control geometric constraints of multiple patches based on spectral clustering algorithm [23]. Spectral clustering is a popular graph based

modern clustering algorithm. It is not only simple to implement but also can be solved efficiently by standard linear algebra methods. We found that a spectral clustering algorithm gives a formal mathematical tool to tackle our problem in a consistent way. In our problem context, spectral clustering is based on random walks on a similarity graph constructed by the multiple patches. Then, spectral clustering can be interpreted as trying to find a partition of the graph such that the random walk stays long within the same cluster and seldom jumps between clusters. The random walk can be formulated via following three steps:

- **Step 1:** Construct a similarity graph.
- **Step 2:** Assign weights to the edges in the graph.
- **Step 3:** Define a transition probability matrix.

In first step, we can connect each patch to its k nearest neighbors, or connect each patch to all neighbors within ε. A similarity function, which defines the edge weights in the second step, plays a pivotal role in getting good clustering performance. Here, we use a magnitude difference of optical flow vectors as well as distance among multiple patches. Note that the direction of optical flow vectors cannot be a good measure since it shows convergent or divergent pattern of flow when a bicycle shows longitudinal motion. Our similarity function is thus defined by:

$$s(x_i, x_j) = e^{-(\alpha\|x_i - x_j\| + \beta\|m_i - m_j\|)} \tag{1}$$

where α and β are constants. The closer the patches and the smaller a magnitude difference of the patches, the higher the weight. Finally, we define a Markov random walk over the similarity graph by constructing a transition probability matrix from the edge weights. Formally, the transition probability of jumping in one step from patch i to patch j is proportional to the edge weight w_{ij} and is given by $p_{ij} = w_{ij}/d_i$ where $d_i = \sum_j w_{ij}$. Then, the transition probability matrix of the random walk is defined by:

$$P = D^{-1}W. \tag{2}$$

Spectral clustering, which can be viewed as a outliers detection process in this case, is performed by normalized spectral clustering algorithm proposed by Ng, Jordan, and Weiss [23]. Algorithm 1 describes the whole process of our patch management scheme. We run this algorithm at every frame and we filter out bad patches by averaging its clustering results.

5 EXPERIMENTAL RESULTS

We evaluated our detection and tracking framework using various real world datasets. We first conducted frontal and side view bicyclist detection experiments using a new bicyclist dataset which we collected from the Internet. The original HOG based detector was tested first and then real-time enhancement using the Integral HOG and

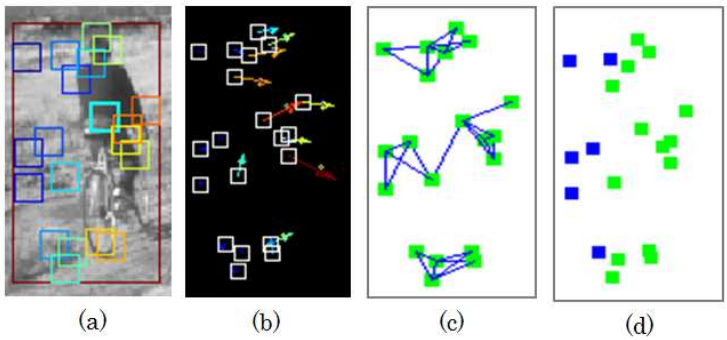


Figure 4: Example of patch management scheme. (a) Selected patches by Harris corner detector (b) Optical flow of each patch (c) 3-nn similarity graph (d) Clustering result

Algorithm 1 Patch management scheme based on spectral clustering algorithm

Require: Similarity matrix $S \in \mathbb{R}$, number of clusters k

1: Construct a similarity graph using K-nn.
2: Make its weighted adjacency matrix W using (1).
3: Compute the normalized Laplacian: $L_{sym} = D^{-1/2}LD^{-1/2}$.
4: Compute the first k eigenvectors $v_1, ..., v_k$ of L_{sym}.
5: Let $V \in \mathbb{R}^{n \times k}$ be the matrix containing the vectors $v_1, ..., v_k$ as columns.
6: Form the matrix $U \in \mathbb{R}^{n \times k}$ form V by normalizing the row sums to have norm 1, that is $u_{ij} = v_{ij}/(\sum_k v_{ik}^2)^{1/2}$.
7: **for** $i = 1, ..., n$ **do**
8: let $y_i \in \mathbb{R}^k$ be the vector corresponding to the i-th row of U.
9: **end for**
10: Cluster the patches$(y_i)_{i=1,...,n}$ with k-means algorithm into clusters $C_1, ..., C_k$.

Ensure: Clusters $A_1, ..., A_k$ with $A_i = \{j | y_j \in C_i\}$

AdaBoost was compared with the first method. With regard to bicycle tracking, we collected video data from suburban environments using our autonomous vehicle "Boss", Carnegie Mellon University's first place winning vehicle from the 2007 DARPA Urban Challenge. Six video sequences were recorded. Three categories of the sequences are from stationary Boss and the other three sets are from moving Boss. Tracking experiments for each case using the multiple patch-based Lucas-Kanade algorithm are also conducted.

5.1 Probabilistic Analysis of Detector

From a practical perspective, it is very important to construct a classifier producing a posterior probability. The probabilistic output of a classifier can help in post-processing such as when combining more classifiers together or when generating a Precision-

Recall (PR) curve to analyze a classifier's performance. However, the output of Support Vector Machines (SVMs) and AdaBoost that we use in this work is an uncalibrated value, not a probability. To solve this problem, we implement Platt's method [16] which convert a classifier output to a calibrated posterior probability for probabilistic analysis. According to [16], the motivation for this method is using a parametric model to fit the posterior $P(y = 1|f)$ directly instead of estimating the class-conditional densities $p(f|y)$. The author trains an SVM first and then trains the parameters of an additional sigmoid function to map the SVM outputs into probabilities. As for the parametric model, the author suggests using a form of sigmoid, which is expressed by:

$$P(y = 1|f) = \frac{1}{1 + \exp(Af + B)} \tag{3}$$

The parameters of A and B are fit using maximum likelihood estimation from a training set. We implement their method to convert the outputs of SVMs/Adaboost to posterior probabilities successfully. Furthermore, we use their probability outputs to generate an PR curve for a comparison of the performance of original HOG and Integral HOG based detectors.

5.2 Performance Analysis of Detector

In our bicycle detection experiments, we used 130 of the normalized images (64×128 for front view and 128×128 for side view) along with their left-right reflections as positive training samples. For the negative samples, we used 1218 images from the INRIA Person dataset[1] (ten times the positive examples) that included backgrounds that do not contain either pedestrians or bicycles. In the first experiment, we used 95 images with 124 labeled bicycles (front view:42, 45^o or 135^o view:42, side view:40) as a test set. We trained a linear SVM using the training set and fitted a sigmoid function to the classifier output. The coefficients A and B for the sigmoid function (Equation 3) were found to be -4.8260 and 0.5641, respectively. Second, we ran the original HOG detector on the test images. Based on the statistics of our test set and detection results, we computed basic metrics and generated a PR curve (Figure 5) for better analysis. The Hit Rate (HR) and the number of False Positives (FP) of the classifier are shown in Table 1. As discussed in Section 3 C, we feel the four view Integral HOG detector is better in terms of its accuracy/efficiency.

Table 1: Classification rates for bicyclists using HOGs

Detectors	**Hit Rate (%)**	**False Positives (#)**
Original HOG (four view)	65.12	175
Integral HOG (four view)	65.12	226
Integral HOG (six view)	65.12	248

[1] http://pascal.inrialpes.fr/data/human/, accessed on May 5 2009

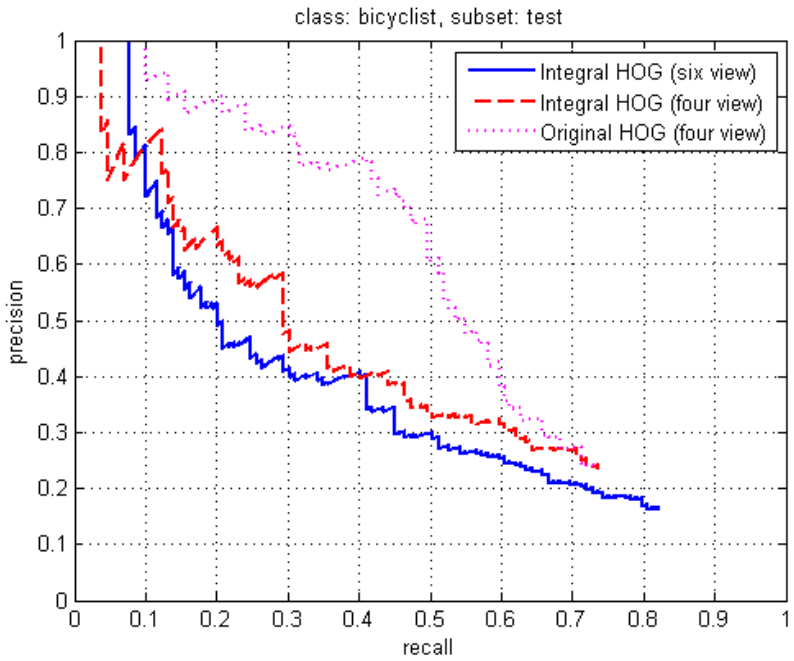


Figure 5: PR curve for three detectors. The magenta plot shows the response for the brute-force HOG implementation and the blue and red plot show the responses of the Integral HOG implementation for six view and four view, respectively .

For the real-time enhancement method, we found out the most informative blocks from which Integral HOG features can be extracted. As an example of bicyclist detection, the first feature selected by the AdaBoost algorithm seemed to be the overall shape of a bicyclist. Secondary and tertiary features included the person's head, their torso, and the wheel of a bicycle. Several of these selected regions and their confidence values (alpha) as generated by the AdaBoost algorithm are shown in Figure 6. Following the same approach which *Zhu* et al. proposed in [25], the next step is to construct a cascade of weak classifiers. The cascade consists of 5 levels where the weak classifiers are linear SVMs using a 36-D feature of each block. Our overall feature set consist of 2,909 blocks of different sizes, locations and aspect ratios. The first two levels in our cascade only contain four linear SVM classifiers each, and reject 70% of the detection windows. Thus, the average number of blocks to be evaluated for each detection window is as low as 6.4. The Hit Rate (HR) and the number of False Positive (FP) of this cascaded classifier are compared between six view and four view-based detector. As shown in Table 1, while six view-based detector increases hit rate slightly, it also entails more false positives. In addition, an ROC curve was compared with previous original HOG case in Figure 5. While this approach shows comparable results with the

Figure 6: Several of the selected regions and confidence values (alpha) as identified by the AdaBoost algorithm.

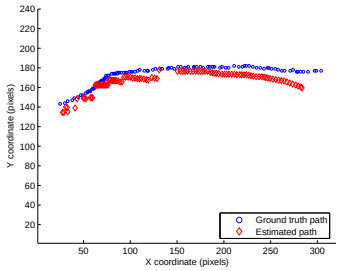
original HOG in terms of accuracy, in terms of speed, it shows a up to 30X speedup over the naive implementation using the sliding windows.

Table 2: Details of two image sequences used in the evaluation

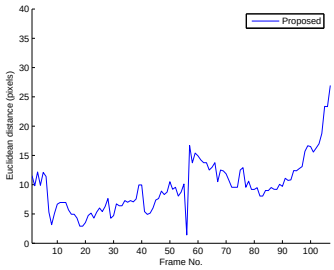
Sequences	Size	Frame-number	FPS	Bicyclist-number
'stationary'	320×240	107	15	1
'moving'	320×240	30	15	1

5.3 Performance Analysis of Tracker

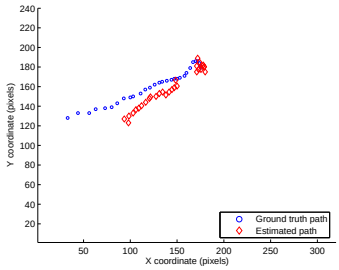
Tracking experiments were conducted on the six videos which we collected from Boss. Three videos of a person riding a bicycle were recorded from Boss's cameras while the vehicle was stationary and three more videos were obtained from Boss while it was in motion. Here, we evaluate the Lucas-Kanade tracking algorithm with a patch management scheme based on spectral clustering algorithm using two image sequences, each of which is one of the most challenging sequence from the two cases. In the stationary



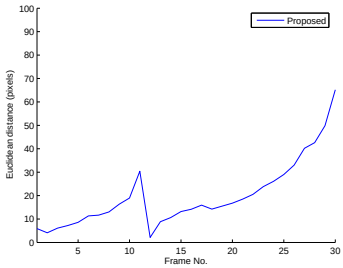
(a) Comparison between the estimated path and the ground truth trajectory



(b) Illustration of tracking accuracy in sequence "stationary case": the Euclidean distance between the estimated bicyclist position and the ground truth is plotted against frame numbers.



(c) Comparison between the estimated path and the ground truth trajectory



(d) Illustration of tracking accuracy in sequence "moving case": the Euclidean distance between the estimated bicyclist position and the ground truth is plotted against frame numbers.

Figure 7: Performance analysis of the tracker : stationary case on top, moving case on bottom.

case, a bicyclist rides along the road in front of Boss and makes a "u-turn" so that the left side, rear, and right-side of the bicycle are seen and must be tracked. In moving case, both the bicyclist and Boss are moving along the road in the same direction and Boss overtakes and passes the bicyclist. The moving imagery situation has a background which undergoes ego-motion that depends on the camera motion as well as the scene structure. Table 2 describes each image sequence.

For the performance of tracking, as partially shown in Figure 8, stationary cases show better tracking performance compared to moving cases. More detailed analysis for each case is investigated by computing tracking errors between the ground truth trajectory and the estimated path of a bicyclist. These errors refer to the Euclidean distance between the bicyclist detection (centers of bounding boxes are considered) and the ground truth created by a trained professional. Figure 7(b), 7(d) clearly illustrate the performance comparison of our approach in two different sequences (see Figures for the details). An abrupt change in Figure 7(b), 7(d) is due to the irregularity of video

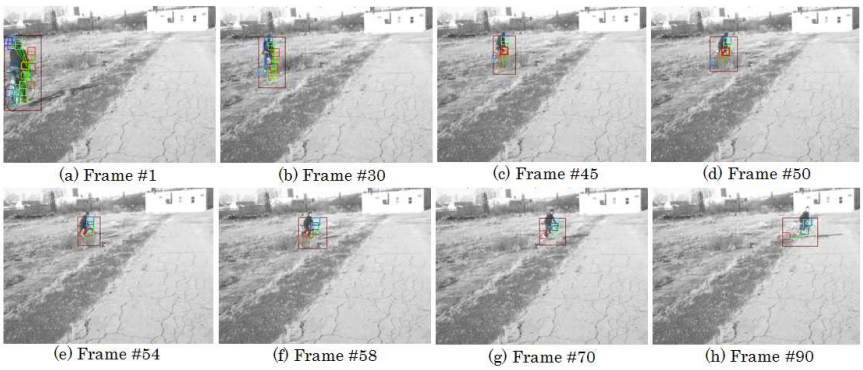


Figure 8: Tracking results for bicyclist: stationary case

logging process. In our six videos, the Lucas-Kanade tracker with a patch management scheme based on spectral clustering algorithm successfully tracks the bicyclist save for the case in which the bicyclist and Boss are both moving in the opposite direction and pass each other. In this case, the default set of parameters for the tracker could not account for the fast relative motion of the bicycle and additional tuning work is required to address this issue.

Detection runs at 1 sec/frame on a P-IV 2GHz computer with 2GB memory and tracking runs at 0.6 sec/frame. Tracking is much more robust than detection in that, the target is hardly lost; however, in the detection stage, a target may not be detected all the time. Thus, we interleave detection and tracking stages by applying detection every five frames or any time the tracking of a bicyclist is lost to find a balance between robust output and finding new bicyclists.

6 CONCLUSIONS AND FUTURE WORK

This paper presents a fast bicyclist detection and tracking framework. To robustly detect bicycles, we have implemented a system that uses Histograms of Oriented Gradients (HOG) descriptors to extract features from images and then employ a linear Support Vector Machine (SVM) to classify whether a given sub-image contains a bicycle. We have affected a dramatic speedup for the detection process by integrating a cascaded classifier concept in combination with HOG features of variable-size blocks. Once the bicycle has been detected in the image, the object is tracked in subsequent video frames with a robust and flexible implementation of the Lucas-Kanade tracking algorithm modified to operate over multiple small image patches. This multi-patch tracker allows our system to effectively track the object even when it changes orientations in the image. We have implemented a novel patch management scheme and integrated the method into our framework. Several experiments shows the effectiveness of each component of the proposed framework. As part of our future work, we will develop a tracking method which takes into account the bicycle motion kinematics.

7 ACKNOWLEDGMENTS

This project was funded by General Motors through the General Motors-Carnegie Mellon Autonomous Driving Collaborative Research Lab.

References

[1] M. Bertozzi, A. Broggi, M. Cellario, A. Fascioli, P. Lombardi, and M. Porta. Artificial vision in road vehicles. *Proceedings of the IEEE*, 90(7):1258–1271, 2002.

[2] H. Cheng, N. Zheng, and J. Qin. Pedestrian detection using sparse gabor filters and support vector machine. *IEEE Intelligent Vehicle Symposium*, pages 583–587, 2005.

[3] D. Comaniciu, V. Ramesh, and P. Meer. Kernel-based object tracking. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 25(5):564–577, 2003.

[4] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *Conference on Computer Vision and Pattern Recognition*, 2005.

[5] P. Dollar, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: A benchmark. *Conference on Computer Vision and Pattern Recognition*, 2009.

[6] M. Enzweiler and D. Gavrila. Monocular pedestrian detection: Survey and experiments. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 2008.

[7] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Computational Learning Theory: Eurocolt '95*, pages 23–37, 1995.

[8] T. Gandhi and M. M. Trivedi. Pedestrian protection systems: Issues, survey, and challenges. *IEEE Transaction on Intelligent Transportation System*, 8(3):413–430, 2007.

[9] D. M. Gavrila. Sensor-based pedestrian protection. *IEEE Intelligent System*, 16(6):77–81, 2001.

[10] D. M. Gavrila, J. Giebel, and S. Munder. Vision-based pedestrian detection: The protector system. *IEEE Intelligent Vehicle Symposium*, pages 13–18, 2004.

[11] C. Harris and M. Stephens. A combined corner and edge detector. *Alvey Vision Conference*, pages 147–152, 1988.

[12] Z. Li, L. L. K. Wang, and F. Wang. A review on vision-based pedestrian detection for intelligent vehicles. *Conference on Vehicular Electronics and Safety*, 2006.

[13] J. Llinas and D. Hall. An introduction to multi-sensor data fusion. *International Symposium on Circuits and Systems*, 1998.

[14] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. *International Joint Conference on Artificial Intelligence*, 1981.

[15] C. Papageorgiou and T. Poggio. A trainable system for object detection. *International Journal of Computer Vision*, 38(1):15–33, 2000.

[16] J. C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in Large Margin Classifiers*, 1999.

[17] F. Porikli. Integral histogram: A fast way to extract histograms in cartesian spaces. *Conference on Computer Vision and Pattern Recognition*, 2005.

[18] P. Sabzmeydani and G. Mori. Detecting pedestrians by learning shapelet features. *Conference on Computer Vision and Pattern Recognition*, 2007.

[19] J. Shi and C. Tomasi. Good features to track. *Conference on Computer Vision and Pattern Recognition*, pages 593–600, 1994.

[20] K. Smith, D. Gatica-Perez, and J. M. Odobez. Using particles to track varying numbers of interacting people. *Conference on Computer Vision and Pattern Recognition*, 2005.

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

[22] P. Viola, M. J. Jones, and D. Snow. Detecting pedestrians using patterns of motion and appearance. *International Journal of Computer Vision*, 63(2):153–161, 2005.

[23] U. von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007.

[24] B. Wu and R. Nevatia. Detection of multiple, partially occluded humans in a single image by bayesian combination of edgelet part detectors. *International Journal of Computer Vision*, 1:90–97, 2005.

[25] Q. Zhu, S. Avidan, M. Yeh, and K. Cheng. Fast human detection using a cascade of histograms of oriented gradients. *Conference on Computer Vision and Pattern Recognition*, 2006.