Fast Human Detection Using a Cascade of Histograms of Oriented Gradients

Qiang Zhu[1], Shai Avidan[2], Mei-Chen Yeh[1], and Kwang-Ting Cheng[1]

[1]Electrical & Computer Engineering Department
University of California at Santa Barbara, CA, 93106, USA
[2]Mitsubishi Electric Research Laboratories
201 Broadway, Cambridge, MA, 02139, USA

{qzhu,myeh,timcheng}@ece.ucsb.edu, avidan@merl.com

Abstract

We integrate the cascade-of-rejectors approach with the Histograms of Oriented Gradients (HoG) features to achieve a fast and accurate human detection system. The features used in our system are HoGs of variable-size blocks that capture salient features of humans automatically. Using AdaBoost for feature selection, we identify the appropriate set of blocks, from a large set of possible blocks. In our system, we use the integral image representation and a rejection cascade which significantly speed up the computation. For a 320 $\times$ 280 image, the system can process 5 to 30 frames per second depending on the density in which we scan the image, while maintaining an accuracy level similar to existing methods.

1 Introduction

Human detection is a logical next step after the development of successful face detection algorithms. However, humans have been proven to be a much more difficult object to detect because of the wide variability in appearance due to clothing, articulation and illumination conditions that are common in outdoor scenes.

Recently, Dalal & Triggs [1] presented a human detection algorithm with excellent detection results. Their method uses a dense grid of Histograms of Oriented Gradients (HoG), computed over blocks of size 16×16 pixels to represent a detection window. This representation is proved to be powerful enough to classify humans using a linear SVM. Unfortunately, their method can only process 320×240 images at 1 FPS using a very sparse scanning methodology that evaluates roughly 800 detection windows per image.

We speed up their method, while increasing the number of detection windows for evaluation from 800 to $12,800$. The improvement is achieved by combining the cascade of rejector approach that is extensively used for face detection [13, 11] with the HoG features. However, we discovered that the use of fixed-size blocks, advocated by Dalal & Triggs is not informative enough to allow fast rejection in the early stages of the cascade. We therefore design a much larger set of blocks that vary in sizes, locations and aspect ratios. We then use AdaBoost to select the best blocks suited for detection and construct the rejector-based cascade. This results in a near real-time human detection system that matches existing methods in terms of accuracy and significantly outperforms them in terms of speed.

In the rest of the paper, we first give some background on human detection. We then describe the method proposed in [1], followed by the details of our framework and experimental results.

2 Background

There appear to be two leading approaches to the problem of human detection. Please refer to [3] for a detailed survey. One approach uses a single detection window analysis whereas the other approach uses a parts-based approach. Within each method, different authors offer different features and different classifiers to tackle the problem.

Under the single-detection-window approach, the work of Papgeorgiou and Poggio [8] uses Haar-based representation, combined with a polynomial SVM. The work of Gavrila and Philomin [4] compare edge images to an exemplar dataset using the chamfer distance. Viola *et al.* [14] extended their Haar-like wavelets to handle space-time information for moving-human detection.

Others have taken a parts-based approach that aims at

0-7695-2646-2/06 $20.00 (c) 2006 IEEE

dealing with the great variability in appearance due to body articulation. In this approach, each part is detected separately and a human is detected if some or all of its parts are presented in a geometrically plausible configuration. Felzenswalb & Huttenlocher [2] use pictorial structure approach where an object is described by its parts, connected with springs, and represent each part with Gaussian derivative filters of different scale and orientation. Ioffe & Forsyth [5] represent parts as projections of straight cylinders and propose efficient ways to incrementally assemble these segments into a full body assembly. Mikolajczyk *et al.* [7] represent parts as co-occurrences of local orientation features. The system proceeds by detecting features, then parts and eventually humans are detected based on assemblies of parts.

Dalal and Triggs [1] used the single window approach with a dense HoG representation that was successfully used for object representation [6, 10, 7].

3 The Dalal-Triggs Algorithm

We start with a short description of the Dalal & Triggs algorithm. 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. Each cell consists of a 9-bin Histogram of Oriented Gradients (HoG) and each block contains a concatenated vector of all its cells. Each block is thus represented by a $36 - D$ feature vector that is normalized to an L2 unit length. Each 64×128 detection window is represented by 7×15 blocks, giving a total of 3780 features per detection window. These features are then used to train a linear SVM classifier.

4 A Fast Human Detection Framework

The Dalal & Triggs algorithm makes use of three key components: (1) the use of HoG as a basic building block, (2) the use of a dense grid of HoGs across the entire detection window to provide a good description of the detection window, and (3) a normalization step within each block that emphasizes relative behavior, with respect to the neighboring cells, as opposed to the absolute values.

An important factor, that is missing in their approach, is the use of blocks in multiple scales. They use fairly small block size (typically, 16×16 pixels) which might miss the "big picture", or global features of the entire detection window. Indeed, they report that adding blocks/cells of different scales would somewhat improve the results but would also significantly increase the computation cost. The capturing of the "big picture" is therefore relied on the dense set of small-scale blocks across the entire detection window.

To accelerate the detection process we use a cascade of rejectors and use AdaBoost to choose which features to evaluate in each stage, where each feature corresponds to one block. However, the small size of the blocks is proved to be a major obstacle. We found that none of these small size blocks was informative enough to reject enough patterns so as to accelerate the detection process. Therefore, we increase our feature space to include blocks of different sizes, locations and aspect ratios. As a result, we have $5,031$ blocks to choose from, compared to the 105 blocks used in the Dalal-Triggs algorithm. Moreover, we found that the first few stages of the cascade, that rejects the majority of detection windows, actually use large blocks and the small blocks are used much later in the cascade.

To support the fast evaluation of specific blocks, as are chosen by our AdaBoost-based feature selection algorithm, we use the integral image representation to efficiently compute the HoG of each block.

It is worth mentioning that Viola *et al.* [14] use a similar framework for detecting humans in a surveillance environment, where people to be detected are very small and usually have a clear background (road, wall, etc.). But, as claimed in their paper, the detection performance greatly relies on the available motion information. However, for the Dalal-Triggs's INRIA database which contains extremely complicated backgrounds and dramatic illumination changes, the Harr-wavelet feature achieves a much lower detection accuracy than that of the HoG feature. We will demonstrate this point in the Experiments section.

4.1 Integral Histograms of Orientated Gradients

The "Integral Image" [13] allows very fast evaluation of Harr-wavelet type features, known as rectangular filters. This led to a real-time face detection system that was later extended to a human detection system [14], using rectangular filters both in space and time. Recently, Porikli [9] suggested the "Integral Histogram" to efficiently compute histograms over arbitrary rectangular image regions.

Inspired by their work, we exploit a fast way of calculating the HoG feature. First, we discretize each pixel's orientation (including its magnitude) into 9 histogram bins. We compute and store an integral image for each bin of the HoG (resulting in 9 images in our case) and use them to compute efficiently the HoG for any rectangular image region. This requires 4×9 image access operations.

This approach, while fast to compute, differs from the method suggested by Dalal T̃riggs and in fact is inferior to it because of the following two reasons. First, Dalal & Triggs use a Gaussian mask and tri-linear interpolation in constructing the HoG for each block. We cannot use these steps because they don't fit well into our integral image ap-

proach. Second, Dalal & Triggs use an L2 normalization step for each block. Again, we replace the L2 normalization with L1 normalization which is faster to compute using the integral image. A more detailed analysis regarding the impacts of these variations is given in the Experiments section.

4.2 Variable-size Blocks

The use of HoG features in the Dalal & Triggs approach was restricted to a single scale (105 blocks of size 16×16 pixels). Moreover, they report that using blocks and cells at multiple scales improves results somewhat while the computational cost greatly increases.

We circumvent this problem by using feature selection. Specifically, for a 64×128 detection window we consider all blocks whose size ranges from 12×12 to 64×128. The ratio between block width and block height can be any of the following ratios $(1:1)$, $(1:2)$ and $(2:1)$. Moreover, we choose a small step-size, which can be any of $\{4, 6, 8\}$ pixels depending on the block size, to obtain a dense grid of overlapping blocks. In total, 5031 blocks are defined in a 64×128 detection window, each of which contains a 36-D histogram vector of concatenating the 9 orientation bins in 2×2 sub-regions. The advantages of using a set of variable-size blocks are twofold. First, towards a specific object category, the useful patterns tend to spread over different scales. The original 105 fixed-size blocks only encode very limited information. Second, some of the blocks in this large set of 5031 blocks might correspond to a semantic part in people, say human leg. A small number of fixed-size blocks is less likely to establish such mappings.

Another way to view our approach is as an implicit way of doing parts-based detection using a single window approach. The most informative parts, i.e. the blocks, are automatically selected using the AdaBoost algorithm. The HoG features we use are robust to small and local changes, while the variable size blocks can capture the "global picture".

4.3 Training the Cascade

We construct rejection cascade similar to the one proposed in [13], with the following modifications. Each feature in our scheme corresponds to the $36D$ vector used to describe a block. The weak classifiers we use are the separating hyperplane computed using a linear SVM. Finally, because evaluating each of the $5,301$ possible blocks in each stage is very time consuming, we adopt a sampling method suggested by Scholkopf & Smola [12](pp. 180). They show that one can find, with a high probability, the maximum of m random variables, in a small number of trials. More specifically, in order to obtain an estimate that is with probability 0.95 among the best 0.05 of all estimates, a random sub-sample of size $\frac{log0.05}{log0.95} \approx 59$ will guarantee nearly as good performance as if we considered all the random variables. In practice we sample 250 blocks, at random, in each round.

For each level of the cascade we construct a strong classifier consisting of several weak classifiers (linear SVMs in our case). In each level of the cascade we keep adding weak classifiers until the predefined quality requirements are met. In our case we require the minimum detection rate to be 0.9975 and the maximum false positive to be 0.7 in each stage. The training process took a few days using a PC with 1.8GHz CPU and 2GB memory.

Algorithm 1 Training the cascade

Input: F_{target}: target overall false positive rate
f_{max}: maximum acceptable false positive rate per cascade level
d_{min}: minimum acceptable detection per cascade level
Pos: set of positive samples
Neg: set of negative samples

initialize: i = 0, D_i = 1.0, F_i = 1.0
loop $F_i > F_{target}$
$i = i + 1$
$f_i = 1.0$
loop $f_i > f_{max}$
1) train 250 (%5 at random) linear SVMs using Pos and Neg samples
2) add the best SVM into the strong classifier, update the weight in AdaBoost manner
3) evaluate Pos and Neg by current strong classifier
4) decrease threshold until d_{min} holds
5) compute f_i under this threshold
loop end
$F_{i+1} = F_i \times f_i$
$D_{i+1} = D_i \times d_{min}$
Empty set Neg
if $F_i > F_{target}$ then evaluate the current cascaded detector on the negative,i.e. non-human,images and add misclassified samples into set Neg.
loop end

Output: A i-levels cascade
each level has a boosted classifier of SVMs
Final training accuracy: F_i and D_i

(a) (b)

Figure 1. HoG vs. Rectangular filters. (a) The best HoG filter. (b) The best rectangular filter.

5 Experiments

We implemented the Dalal & Triggs algorithm using the same training/testing databases they provided. We observed that our implementation resulted in perfectly consistent results with those reported in their paper. The values of all parameters suggested in their paper result in the best accuracy and our miss-rate/FPPW (false positives per window) curve is very similar to the one they reported. At 10^{-4} FPPW, the detection rate is about 88%. Note that, in their experiments, the best two curves, "Ker. R-HOG" and "Lin. R2-HOG", have a significant higher computational cost. Therefore, we only consider "Lin. R-HOG" proposed in their experiments, which achieves a similar accuracy to the more expensive "Ker. R-HOG" and "Lin. R2-HOG", as the FPPW goes down.

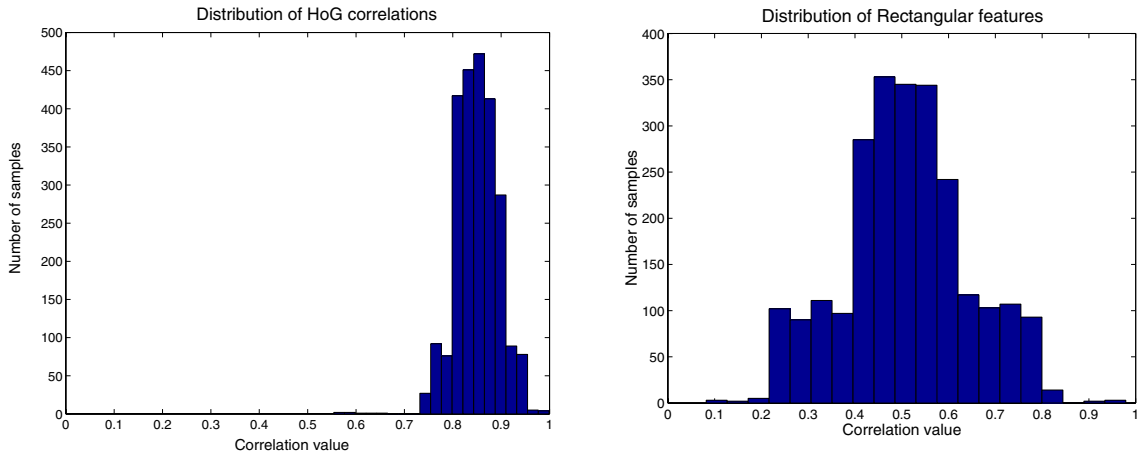
As our main contribution is integrating the HoG feature into a cascade framework, a straightforward comparison is applying the original Viola-Jones Detector to human detection task. The Intel OpenCV library provides an efficient implementation of the Viola-Jones face detector. After slight modifications, we adapt it to human detection (note that we do not use any motion information, as the input does not consist of pairs of images). The resulting cascade has 18 stages and about 800 weak classifiers. The training process took eight days using a PC with 1.4GHz CUP and 512MB memory. This detector resulted in a very low hit rate (about 50%) under the same testing dataset. A more detailed comparison of this rectangular filter to the HoG filter will be presented later.

To understand this large discrepancy between the two types of filters we designed an experiment that compares the stability of HoG features to that of the rectangular features. To do this we picked the best HoG feature and the best rectangular feature (shown in figure 1) and analyzed their stability as follows: we computed the HoG feature on each of the 2,418 training images and took its mean. We then measured the correlation between the HoG feature, as computed for each image, and the mean HoG feature. We repeated the experiment with the best rectangular feature. The results are shown in figure 2. As can be seen, the HoG feature is much more stable (the peak of the distribution, that measures the correlation, is around 0.85, and the variance is about 0.1) than the rectangular feature (the peak is around 0.5 and the variance is about 0.3).

In the next experiment we evaluated the importance of using blocks of different sizes. Figure 3 shows three histograms. Each histogram shows the distribution of the classification accuracy (measured by the same error rate) of blocks using a linear SVM. The first histogram is of all blocks of size 16×16 pixels, the second histogram is of blocks of size 40×80 pixels and the third histogram is of all $5,031$ blocks. The figure clearly shows that the 16×16 blocks are the least informative and that increasing the block population does contribute significantly to the performance of our system.

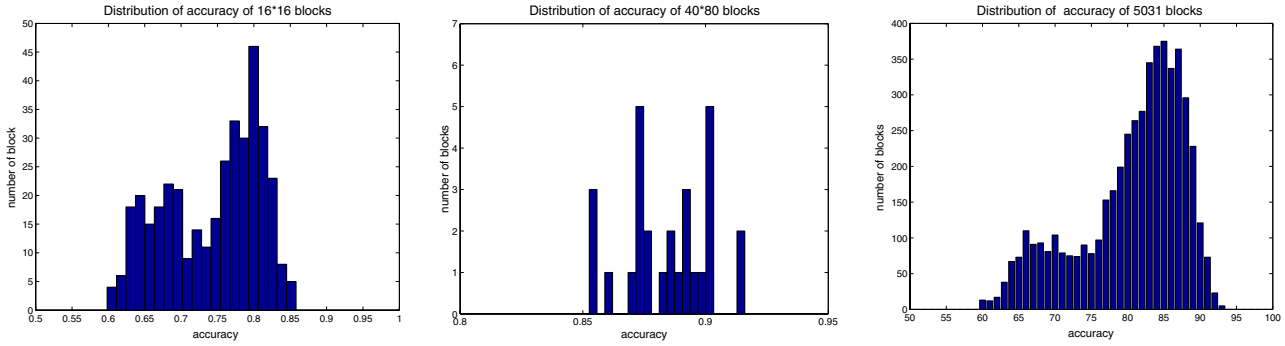
Figure 4 gives some details about our cascade. The cascade consists of 30 levels and we found that, on average, 4.6 block evaluations are needed to classify a detection window. This is a speedup more than 20X, compared to the 105 blocks that must be evaluated for every detection window in the Dalal & Triggs algorithm. It is worth mentioning that in our rectangular-based cascade, we found that, on average, we need to evaluate 19.4 blocks to classify a detection window. Note that a direct comparison of the computational costs between the two types of cascades (HoG-based and rectangular-based) is difficult because the HoG-based method requires the creation of 9 integral images, a normalization step for each block and different weak classifiers (linear SVMs for the HoG Vs. a threshold function for the rectangular case).

Next, we inspected the most informative blocks that were chosen by our system. Figure 5(a) shows the best five blocks with minimum error rates. We observe that the best



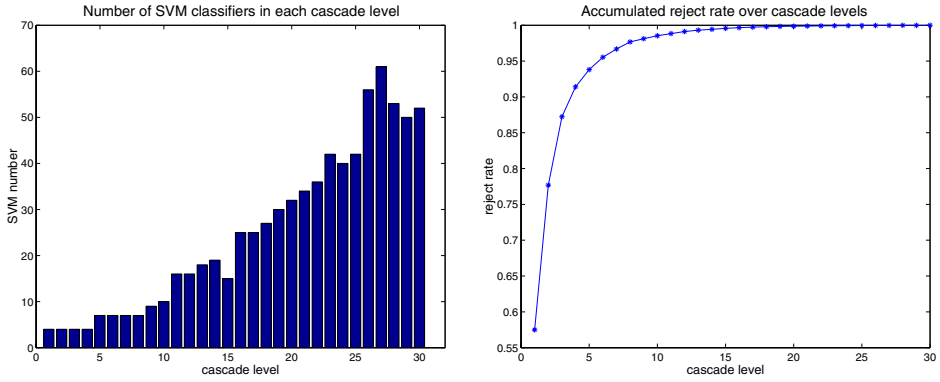
(a) (b)

Figure 2. Stability of HoG vs. Rectangular filters. These histograms measure the stability of these two kinds of features. (a) Stability of the HoG feature. (b) Stability of the rectangular feature. The horizontal axis denotes the correlation between a feature and the mean of features. The peak of the HoG histogram is closer to 1 and much more narrow.



(a) (b) (c)

Figure 3. Classification score of blocks of different sizes. This shows that (a) small blocks (16×16 pixel) are much less informative than (b) large blocks (40×80 pixels). Note that there are much fewer large blocks to choose from. (c) Histogram of all 5,031 blocks considered in the feature selection process.



(a) (b)

Figure 4. Cascade details (using HoGs of variable-size blocks). The cascade consists of 30 levels where the weak classifiers are linear SVM using a 36-D feature of each block, and the feature set consists of $5,031$ blocks of different sizes, locations and aspect ratios. (a) The number of SVM classifiers in each level. (b) The rejection rate as cumulative sum over cascade levels. The first four levels in our cascade only contain four SVM classifiers each, and reject about 90% of the detection windows. Therefore, the average number of blocks to be evaluated for each detection window is as low as 4.6.

block size is about 36×80 (which is roughly the human size presented in the training image). From (b) to (d), we visualize the blocks selected in our cascade of level-1,2,8 as well. Note that during the cascade training we randomly evaluate 5% of all possible blocks in each level. Therefore, the "best" blocks chosen in a cascade may not the best ones globally. However, we observe that the blocks located in some specific positions, say the body, the legs, the head, etc., achieve a much higher accuracy and thus are more likely to be selected. While Dalal & Triggs algorithm only uses a fixed-scale block of 16×16, which incurs a much higher error rate in our experiment, we found that 16×16 pixel blocks appear only after the 8-th stage of our cascade. The early stages of the cascade use much larger block sizes. Another interesting observation is that most selected blocks cluster in the central area, which demonstrates that AdaBoost is very efficient in selecting the most informative blocks as opposed to the blocks in background.

Finally, we compare four approaches: Dalal and Triggs, a cascade of rectangular features and our approach with L1-norm and with L2-norm. A comparison of miss-rate/FPPW curves is presented in figure 6. As for the cascade detector, we use different stages to obtain a pair of (miss rate, FPPW). For example, for a 30-stages cascade, we can get 30 pairs to plot a curve. We notice that both L1-Norm and L2-Norm work slightly better in low FPPW region than Dalal & Triggs algorithm. However, as the miss rate reduces, our approach incurs a higher FPPW. Note that the difference between the L1-Norm and L2-Norm results is small in our framework, contrary to the conclusion in Dalal and Triggs' experiments. Figure 7 shows some typical results of our algorithm.

6 Conclusions

We demonstrate a near real-time human detection system that matches the detection performance of previous methods with a up to 70X speedup. This is achieved by integrating the cascade-of-rejectors concept with the Histogram of Oriented Gradients (HoG) features. Central to the success of our approach is that the features are chosen from a large set of blocks at multiple sizes, locations and aspect ratios.

References

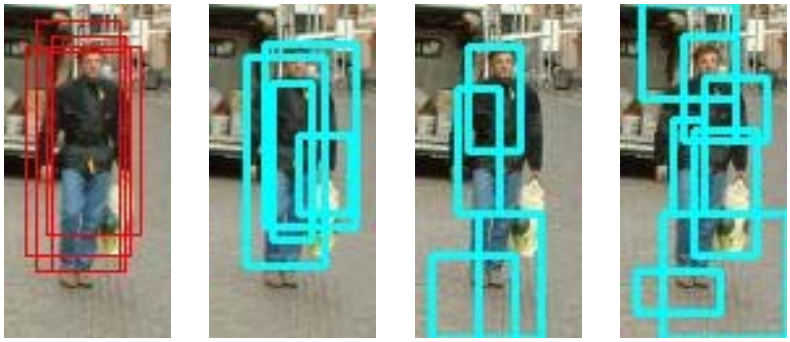
[1] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *Coference on Computer Vision and Pattern Recognition (CVPR)*, 2005.

[2] P. Felzenszwalb and D. Huttenlocher. Pictorial structures for object recognition. *Internation Journal of Computer Vision (IJCV)*, 61(1):55–79, 2005.

[3] D. M. Gavrila. The visual analysis of human movement: A survey. *Journal of Computer Vision and Image Understanding (CVIU)*, 73(1):82–98, 1999.

[4] D. M. Gavrila and V. Philomin. Real-time object detection for smart vehicles. *Coference on Computer Vision and Pattern Recognition (CVPR)*, 1999.

[5] S. Ioffe and D. Forsyth. Probabilistic methods for finding people. *International Journal of Computer Vision (IJCV)*, 43(1):45–68, 2001.



(a) (b) (c) (d)

Figure 5. Visualizing the selected blocks. (a) Top 5 blocks with minimum error rates (over all $5,031$ blocks). (b) Blocks in level-1 of the cascade. (c) Blocks in level-2. (d) Blocks in level-8. The blocks in figures (b)-(d) are the top blocks from a random sample of 250 blocks out of the total of $5,031$ blocks. Note that most blocks are much larger than the 16×16 pixel blocks used by Dalal & Triggs.

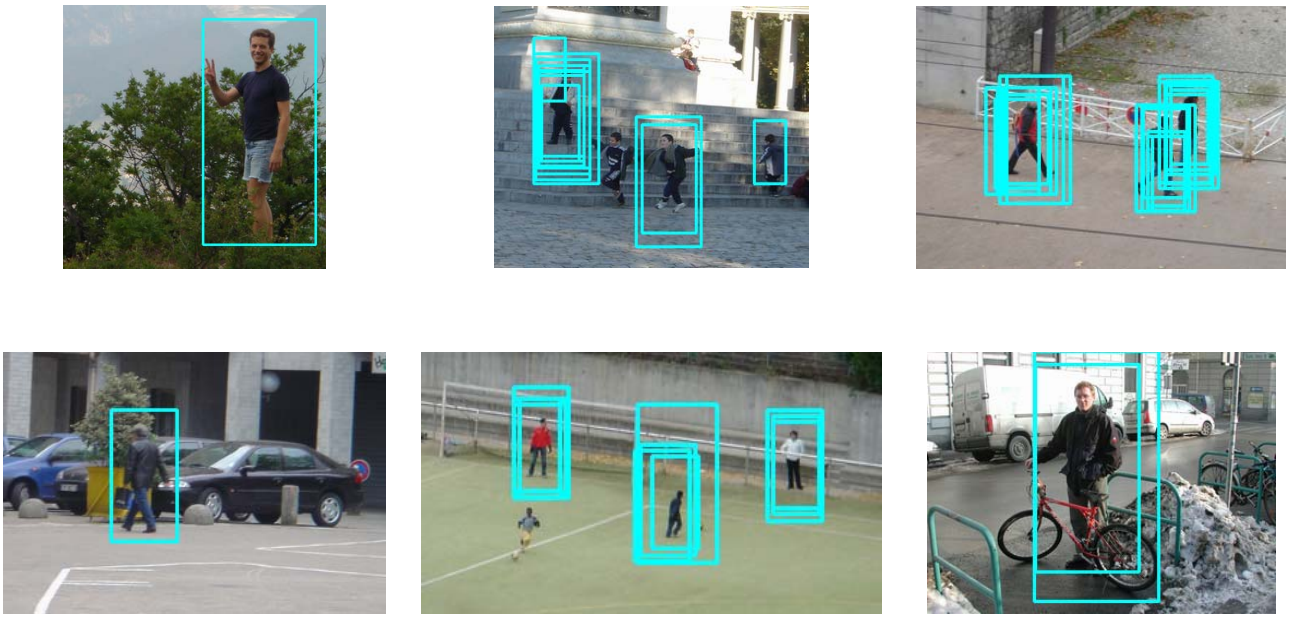


Figure 7. Results using a dense scan. No detection post-processing was applied to the images.

	Sparse scan (800 windows per image)	Dense scan ($12,800$ windows per image)
Dalal & Triggs	500ms	7sec
Cascade of Rect. features	11ms	55ms
Our approach (L1-norm)	26ms	106ms
Our approach (L2-norm)	30ms	250ms

Table 1. Time required to evaluate a 240×320 image. Sparse scan refers to the way used in Dalal & Triggs experiment. Dense scan is using a smaller step-size to produce more hypothesis in a testing image. Our method is much faster than the Dalal & Triggs algorithm, and the gap increases as the scan density increases. (For the Dalal & Triggs case, the reported results are based on our implementation of their algorithm). The Rectangular features are the fastest to compute but its accuracy is not as good. see figure 6 for accuracy comparison.

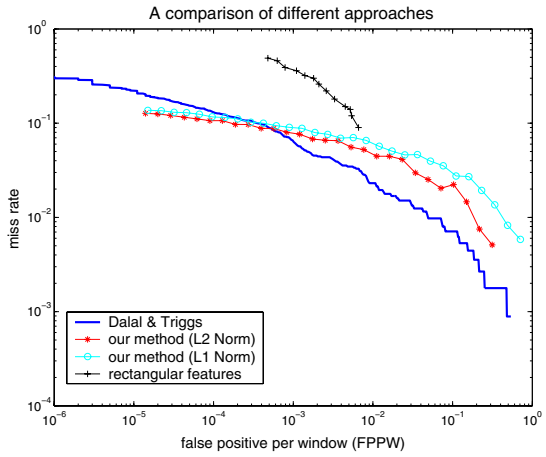


Figure 6. Comparing the Dalal & Triggs algorithm, a Rectangular filter detector and our cascade of the HoG method. Our method (using either L1 or L2 norms) is comparable to the Dalal & Triggs method, especially when the FPPW goes down.

[6] D. Lowe. Distinctive image features from scale-invariant keypoints. *Internation Journal of Computer Vision (IJCV)*, 60(2):91–110, 2004.

[7] K. Mikolajczyk, C. Schmid, and A. Zisserman. Human detection based on a probabilistic assembly of robust part detectors. *European Conference on Computer Vision (ECCV)*, 2004.

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

[9] F. Porikli. Integral histogram: A fast way to extract higtograms in cartesian spaces. *Coference on Computer Vision and Pattern Recognition (CVPR)*, 2005.

[10] J. M. S. Belongie and J. Puzicha. Shape matching object recongition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 24(24):509–522, 2002.

[11] H. Schneiderman. Feature-centric evaluation for efficient cascaded object detection. *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2004.

[12] B. Scholkopf and A. Smola. *Learning with Kernels Support Vector Machines, Regularization, Optimization and Beyond*. MIT Press, Cambridge, MA, 2002.

[13] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. *Coference on Computer Vision and Pattern Recognition (CVPR)*, 2001.

[14] P. Viola, M. Jones, and D. Snow. Detecting pedestrians using patterons of motion and appearance. *International Coference on Computer Vision (ICCV)*, 2003.