2008 IEEE Intelligent Vehicles Symposium
Eindhoven University of Technology
Eindhoven, The Netherlands, June 4-6, 2008

A Two-staged Approach to Vision-based Pedestrian Recognition Using Haar and HOG Features

Philip Geismann
Department of Embedded Systems and Robotics
Technical University Munich
geismann@in.tum.de

Georg Schneider
TRW Automotive, Driver Assistance Systems
Carl-Spaeter-Strasse 8, 56070 Koblenz
georg.schneider@trw.com

Abstract— **This article presents a two-staged approach to recognize pedestrians in video sequences on board of a moving vehicle. The system combines the advantages of two feature families by splitting the recognition process into two stages: In the first stage, a fast search mechanism based on simple features is applied to detect interesting regions. The second stage uses a computationally more expensive, but also more accurate set of features on these regions to classify them into pedestrian and non-pedestrian. We compared various feature extraction configurations of different complexities regarding classification performance and speed. The complete system was evaluated on a number of labeled test videos taken from real-world drives and also compared against a publicly available pedestrian detector. This first system version analyzes only single image frames without using any temporal information like tracking. Still, it achieves good recognition performance at reasonable run time.**

I. Introduction

The quality of future driver assistance systems will depend on the ability to attentively perceive their environment. Therefore, such systems will have to detect and classify surrounding objects in a reliable manner. To avoid dangerous traffic situations, it is especially important to recognize pedestrians, who are the most vulnerable road participants.

There are various approaches that try to solve the pedestrian recognition task. Earlier solutions relied on a single kind of sensor, for example a multilayer laser scanner [1] or a video camera [5], [20]. By now, many authors present systems that merge information coming from different sensor types, a common combination consisting of distance measuring sensors like LIDAR and vision-based sensors [2], [3]. In a similar way, some authors use depth information of stereo-based vision to detect pedestrian candidates as input for following recognition processing. [16], [17], [18].

Such multi sensor systems, while showing good detection results due to the high amount of information, also lead to higher material costs and maintenance efforts. Our work is focused on a single camera sensor, which provides detailed information content at considerable low material cost. Additionally, there is a growing trend to install camera sensors for other automotive applications like a lane detection system anyway, so presumably they will be highly available. These properties make cameras a promising and realistic sensor for the problem of pedestrian recognition in a vehicle.

There are several ways to detect people in video sequences. Some methods exploit characteristic motion cues of the pedestrian (i.e. detect walking pedestrians [4]), others use the motion of the vehicle itself to segment the given scene (structure from motion [13]). Further on, a variety of motion-independent methods exist. These approaches mostly analyze the shape of human bodies in images, thus also being able to detect stationary pedestrians. Viola and Jones proposed a framework for rapid object recognition based on Haar-like wavelet features and an Adaboost classification cascade [6]. This framework was modified and used later in [8] for pedestrian detection, but it turned out to achieve a relatively high number of false positives, due to the high variety of pedestrian appearances. Better recognition results on a more challenging test set were presented in [9]. The authors used a dense grid of overlapping blocks to extract histograms of oriented gradients (HOG) in a search window. A linear SVM was trained to classify the given samples. The combination of HOG and SVM proved to be very capable, but time-consuming. Following works put the HOG/SVM framework into a cascaded classifier structure and significantly accelerated the detection process [12], [11]. Geronimo et al. [10] combined Haar features and a feature set similar to HOGs (called edge orientation histograms EOH) to train a single Adaboost classifier. This Adaboost method selects only discriminative Haar/HOG features, thus yielding good recognition results.

Since the HOG and Haar feature families proved their efficiency for the task of pedestrian recognition, we chose to use a combination of both. However, in contrast to [10], in our work they are employed in different steps of the recognition process. We divide the recognition process into two stages called *detection* of person-like regions and *classification* into person and non-person sets. [1] This way, recognition can be sped up by evaluating simple and fast Haar features in the detection stage and later perform classification by the powerful HOG/SVM combination only on meaningful regions.

In section 2 we present a detailed description of the detection and classification procedures. Section 3 quantitatively evaluates and discusses the experimental results achieved on a number of real world test drives. Section 4 concludes the paper and gives perspectives on future work.

[1]To be exact, the detection stage also must include a (simpler) classification step to make its decision about pedestrian candidates.

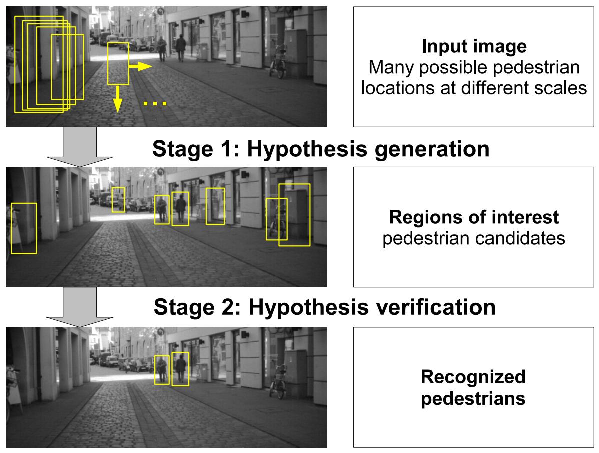
Input image
Many possible pedestrian locations at different scales
Stage 1: Hypothesis generation
Regions of interest
pedestrian candidates
Stage 2: Hypothesis verification
Recognized pedestrians

Fig. 1. Two-staged recognition process of our system.

II. Pedestrian recognition

The general architecture of the recognition system is depicted in figure 1. Following the object recognition paradigm, the recognition process has been separated into two steps: Hypothesis generation and hypothesis verification. The work flow is as follows: The first stage, the detection stage, generates hypotheses, i.e. rectangular regions which may contain pedestrians. These regions of interest (ROI) are then passed on to the second level, the classification stage, which analyzes them in a more thorough way. It confirms the presence of a pedestrian or discards the ROI if the classification is negative. The system output consists of bounding boxes describing size and position of the pedestrians in each frame.

A. *Detection stage*

In the hypothesis generation stage, we use a Haar detector similar to the one proposed in [6]. The Haar detector has very high detection speed, due to the two following facts (for details on feature extraction and classification see [6]):

- Simple feature set: Haar features provide a very basic set of features that can be calculated efficiently by using integral images.
- Cascaded classifier: Combining Adaboost classifiers in a cascade structure reduces classification time for negative input samples.

The OpenCV library[2] provides an efficient implementation of the object detection framework based on the work of Viola and Jones. It also contains a trained classifier for pedestrian detection which was used as reference detector.

1) Training process: In order to develop a hypothesis generation stage that detects as many pedestrians as possible, the detection rate should be kept high, even if many detections are false positives. For that reason, we carried out a series of training processes, using training samples from different sources that were all resized to 14×28 pixels. This also determines the size of the smallest pedestrian that can be detected in the image. The positive samples show a wide range of different poses and appearances. 500 positive samples were taken from our own database, 250 from the MIT[3] dataset and 250 from the INRIA[4] dataset. By using a mixture of training sets, we reduced sensor influence and improved results regarding generalization. The negative sample set was created by sub sampling 6000 street scene images that did not contain any pedestrians.

The training process is influenced by the training data itself, the minimal detection rate and the maximal false positive rate per cascade level. These parameters implicitly determine the number of evaluated Haar features in each cascade level and the total number of levels, which in turn are responsible for the overall detection speed of the classifier. Experiments showed that using a large number of training samples and moderate rates for detections (99.5%) and false positives (0.5) per cascade level yields best results. It leads to a Haar classifier cascade with only 11 levels. Even if an input sample has to pass through all stages, its detection speed is acceptable.

2) Detection process: To further increase the processing speed, the 750×400 input image is downsampled to $1/4$ of its original size. We use a common nearest-neighbor resizing method without any blurring or additional image manipulation. The Haar detector analyzes the low resolution image by sliding a search window of a certain scale across possible image positions. Scales range from 14×28 to 70×140 pixels, feasible positions are defined by a scale-dependent step size. The scaling is done by resizing the features of the Haar detector itself, since they can be calculated at any scale with the same computational cost when using an integral image. Since the training samples are quite small, the AdaBoost learning algorithm is limited in capturing minor details. Thus, we found that downsampling the image has little effect on the detection performance if we focus on processing speed and disregard the number of false positives.

B. *Classification stage*

In the second stage, the hypotheses are verified by a computationally more expensive mechanism. It uses a different set of features as well as a different classifier than the previous stage. While the feature extraction is slower, the classification performance in total is more accurate, meaning that the number of false positives is low. The features are extracted over a search window with the size of 64×128 pixels which allows to capture a reasonable amount of local information.

1) Feature extraction: The extracted features are based on histograms of local gradients (HOG) that are computed over a grid of overlapping rectangular blocks in the search window. For each block, its histogram describes the frequency of the occurring gradient directions inside that block. As stated in [9], histograms with nine direction bins (ranging from $0°$ to $180°$ with a step size of $20°$) seem to be suitable for pedestrian classification.

[2]www.opencv.org

[3]http://cbcl.mit.edu/software-datasets/PedestrianData.html

[4]http://pascal.inrialpes.fr/data/human/

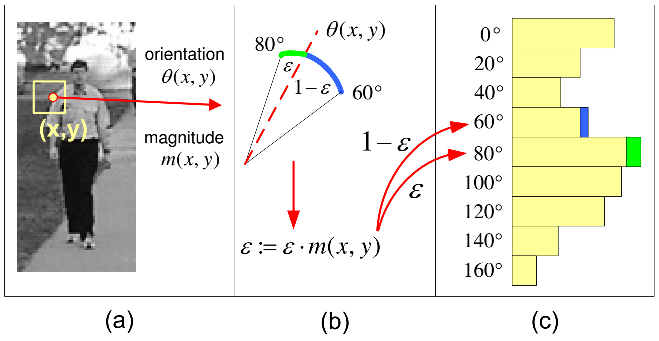

Fig. 2. Feature extraction as it is done for each pixel (x,y) inside a cell/block: (a) computation of gradient orientation and magnitude (b) orientation interpolation and weighting (c) vote binning for the used directions.

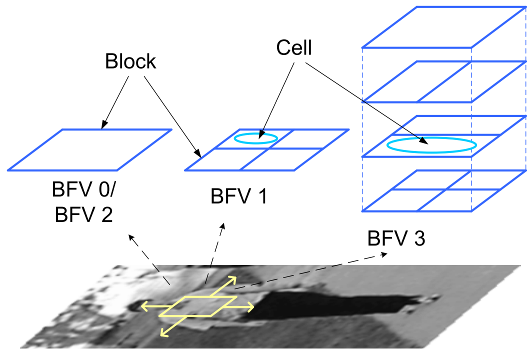

Fig. 3. Four different block feature variants.

Figure 2 illustrates the feature extraction process for one block feature. We compute the gradients by applying Sobel filters in both x- and y-direction. For each pixel covered by a block we compute its vote, a value representing the gradient magnitude. The vote is then distributed proportionally among the two neighboring bins of the pixel's gradient direction. Since the blocks are rectangular and the histogram only uses sums of direction occurrences, we employ integral images (see [14]) to efficiently build block histograms: Computing one integral image for each gradient direction allows us to save time in case of overlapping blocks and search windows.

In order to determine the most promising way to extract feature values of a block histogram, four different block feature variants (BFV) have been compared against each other. Figure 3 shows how each variant is defined by dividing the block into sub cells. $BFV0$ and $BFV2$ use the whole histogram of the block itself, while $BFV1$ divides the block into four sub cells, $BFV3$ into nine sub cells. The block feature variants also differ in the extraction of the actual feature values. A feature vector $\vec{b_i}$ of a single block i is computed as follows ($h_{d[,s]}$ denotes the histogram entry of block i for direction d in the optional sub cell s):

- **Block feature variant 0 (BFV 0)**
  Direct evaluation of the nine different oriented gradient histogram entries:

$$\vec{b_i} = (h_1, ..., h_9)^T, \ \vec{b_i} \in \mathbb{R}^9 \tag{1}$$

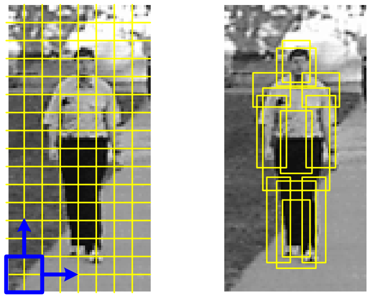

Fig. 4. *(a) Dense descriptor: Grid of 105 overlapping equal-sized blocks (b) Custom descriptor: 12 most discriminative variable-sized blocks.*

- **Block feature variant 1 (BFV 1)**
  Concatenation ("∘") of the histogram entries in each of the four cells:

$$\vec{b_i} = ((h_{1,1}, ..., h_{9,1})^T \circ \ldots \circ (h_{1,4}, ..., h_{9,4})^T), \ \vec{b_i} \in \mathbb{R}^{36} \tag{2}$$

- **Block feature variant 2 (BFV 2)**
  Evaluation of all quotients of the histogram entries:

$$\vec{b_i} = \left(\frac{h_1}{h_2}, \frac{h_1}{h_3}, \ldots, \frac{h_8}{h_9} \right), \ \vec{b_i} \in \mathbb{R}^{36} \tag{3}$$

  This block feature captures strong directions that stand out from other directions.
- **Block feature variant 3 (BFV 3)**
  Concatenation of the histogram entries in each of the nine cells:

$$\vec{b_i} = ((h_{1,1}, ..., h_{9,1})^T \circ \ldots \circ (h_{1,9}, ..., h_{9,9})^T), \ \vec{b_i} \in \mathbb{R}^{81} \tag{4}$$

Extraction of the block features also includes block-wise normalization, i.e. every $\vec{b_i}$ is normalized using the L2 norm (to length 1). The single block vectors are then concatenated into one feature vector $\vec{x}$ that describes the search window instance. The arrangement of the block features inside the search window is called *descriptor*. It defines how an input sample is finally transformed into feature space and thus has a strong impact on classification performance. We evaluated two descriptor configurations with different numbers of blocks and block arrangements: First, a dense descriptor (inspired by the original Dalal/Triggs descriptor presented in [9]) with $7 \times 15 = 105$ overlapping blocks was used (figure 4 (a)). In order to analyze if such a complex descriptor is necessary, we developed a leaner descriptor with custom sized blocks and positions. To find out the most informative block positions, we employed a feature selection method based on AdaBoost that searched the whole space of different block sizes and locations inside the search window. The blocks lying in the middle of the search window (approximately covering the human torso), in the head/shoulder regions and in the leg area turned out to be highly discriminative.

Utilizing this observation, we manually created a second descriptor (figure 4 (b)) that combined the insights of the applied AdaBoost learning technique as well as *a priori* assumptions about characteristics of the human silhouette. Note that this descriptor only has 12 blocks in total, making it computationally much simpler than the dense descriptor.

2) Training process: We use a linear support vector machine (SVM) as classifier [15]. SVMs demonstrated good proficiency in the field of pedestrian recognition, with decent classification speed in the linear variant. We trained one SVM classifier for each descriptor and BFV combination, obtaining 8 alternatives in total. The linear SVM was adjusted to a soft-margin, thus allowing relatively far training outliers. Training input consisted of 1454 positive and 5000 negative samples of 64×128 pixel size, all originating from both our own database and the INRIA dataset. We obtained the database by selecting pedestrians from real world images of the original resolution. To reduce the influence of rescaling effects on the training data, we chose training samples that already covered roughly the size of 64×128 pixels.

3) Classification process: The classification stage receives regions of interest as input, i.e. sub images that contain hypotheses generated by the first-stage-detector. Each input sample is scaled to the classifier standard size of 64×128 pixels using the nearest-neighbor method. Depending on the chosen descriptor/BFV combination, the features are extracted, transformed into a feature vector and sent to the SVM classifier. The SVM outputs a score that describes the classification confidence of the given input sample.

Typically, we yield several positive classification bounding boxes for one pedestrian. These boxes vary in size and position and surround the correct pedestrian location. This effect can result from overlapping hypotheses, discrete search window step size and the fact that the descriptors allow some local fuzziness. Therefore, in a last processing step, clusters of multi-scale bounding boxes are aggregated into one single average bounding box. Since the system has no information about the expected number of pedestrians in one frame, we employ a free clustering technique without a priori knowledge about cluster numbers. The procedure combines multiple nearby bounding boxes (”clusters”) into one average bounding box. To determine size and position of the resulting box, it computes the average size and position of the cluster, preferring bounding boxes that received a high classification score. We decided to perform clustering after the classification stage rather than after the hypothesis generation, because in that case some detections could be lost due to imprecise hypothesis aggregation. Furthermore, the HOG feature extraction employs integral images, which allow the fast histogram calculation in overlapping areas. Therefore, the additional computational effort is small.

III. Results and discussion

To assess the performance of the recognition system, we carried out separate tests for the hypothesis generation stage, the classification stage and the combined recognition system.

A. Hypothesis generation test

We evaluated five test videos that were captured during test drives in real-world scenes. The videos vary in complexity, meaning that some videos contain more distracting structures and human-like shapes than others. In each video frame, every pedestrian occurrence was manually labeled using a label tool. We used the following scheme for pedestrian labeling: The bounding box is drawn in such a way that it encloses the pedestrian from head to feet and the torso tightly, thereby possibly neglecting outstretched arms. To determine the complete bounding box dimension for later evaluation, a virtual margin is generated around the drawn box. After adding a height-dependent margin above and below the pedestrian, the box width can be calculated such that the width/height ratio becomes 1:2. This way, the labeled data always fits the training data format, which also has a margin around the pedestrian (see figure 4) and a 1:2-ratio. The length of the test videos ranges from 75 to 670 frames with 70 to 500 pedestrian occurrences.

TABLE I

Average detection rates and false positives per frame (achieved by single frame processing).

rate	method	1	2	3	4	5
detection rate	HypoGen	71%	94%	100%	81%	97%
detection rate	OpenCV	59%	16%	70%	74%	18%
false positive rate	HypoGen	13.3	9.2	10	4.3	16.2
false positive rate	OpenCV	0.55	0.3	0.3	0.7	0.9

Our hypothesis generation system outputs a surrounding box for each detected pedestrian candidate. These boxes are compared to the labeled ground truth data using the relation of the PASCAL object challenge:[5] A predicted pedestrian counts as correctly detected if the condition $|A \cap B| \, / \, |A \cup B| > 0.5$ is fulfilled, where A denotes the predicted algorithm output box, B the labeled ground truth box.

The two upper rows in table I show the average detection rates of the 11-staged AdaBoost Haar detector (”HypoGen”) compared to the reference OpenCV cascade for pedestrian detection. The detection rate points out the fraction of all pedestrian occurrences that could be detected by the algorithm. The rates illustrate the diversity of the videos, ranging from average to good detection rates. Our Haar cascade detection rates are clearly higher in all videos. Admittedly, the OpenCV detector is meant to be a full recognition system: As the two lower rows in table I show, its false positive rate per frame is very low, whereas our cascade has average 4 to 17 mis-detections in each frame. Again, we use this detector only for the generation of hypotheses, so the amount of false positives just increases the processing time of the full recognition system. Note that high detection rates are especially important here since they present an upper bound on the combined recognition system.

B. Hypothesis verification test

To find out which combination of descriptor and block feature variant performs best, we carried out a series of tests on an image test set to measure the feature extraction/classifier

[5]www.pascal-network.org

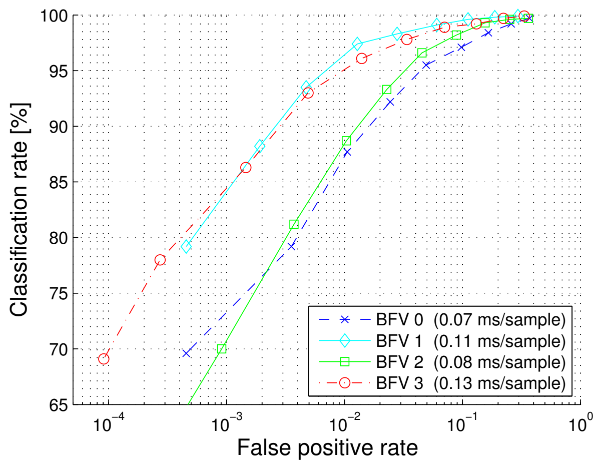

Fig. 5. Classification performance and speed of custom descriptor.

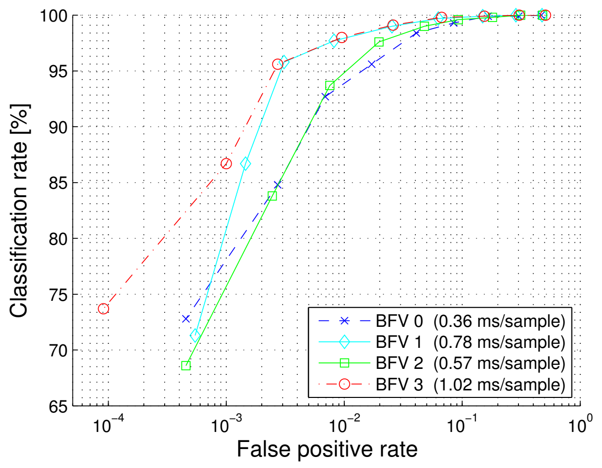

Fig. 6. Classification performance and speed of dense descriptor.

quality. The test set consists of 1000 positive and 11000 negative samples from our videos as well as the MIT and INRIA test sets.

The resulting ROC curves of the four block feature variants are depicted in figure 5 for the customized descriptor, figure 6 for the dense descriptor, respectively. They were created by shifting the SVM decision hyperplane between the two classes using a classification threshold.

In both descriptor configurations, the more complex methods $BFV1$ and $BFV3$ outperform the simpler ones. Furthermore the calculation of gradient quotients to detect main directions ($BFV2$) yields lower results than $BFV1$ which has the same dimensionality but uses the unaltered values. In general, the ROC curves of the customized descriptor and the dense descriptor present similar characteristics. Interestingly, the custom descriptor variant of $BFV1$ has lower false positive rates at comparable detection rates than the combination of $BFV1$ and dense descriptor, which represents the original Dalal/Triggs method. However, the other dense descriptor's curves are steeper and reach higher detection rates at lower false positive rates.

The overall best performance (regarding low false positive rates) is achieved by the dense descriptor with the most complex BFV variant $BFV3$ that evaluates nine sub cells. Since it has the largest area-under-ROC (a quality measure for classifiers) of all evaluated combinations, it can be assumed to have the lowest effective false positive rate. This potential comes at a high cost: Figures 5 and 5 also show the classification times of all block feature variants achieved on a 3.2 GHz Pentium 4 with 3GB RAM. The classification time includes the duration of both feature extraction and classification, so it mainly depends on the dimensionality of the chosen combination. The most complex combination (possessing the lowest false positive rate) also has the highest classification time (more than $1ms$). On the other hand, by accepting a slightly worse false positive rate, the classification time can be dramatically reduced if a customized descriptor is used.

TABLE II
RECOGNITION RATES OF DENSE AND CUSTOM DESCRIPTOR WITH BFV 3 AT A FALSE POSITIVE RATE OF 0.2 PER FRAME

Video	1	2	3	4	5
Dense descriptor	52%	70%	91%	61%	55%
Custom descriptor	45%	53%	85%	69%	58%

C. Combined system test

For the concluding test of the whole recognition system, the five test videos were evaluated in the same way as it was done for the hypothesis generation stage, only this time the ROIs were sent to the SVM classifier. Table II outlines exemplary recognition rates for custom and dense descriptor with $BFV3$ at a false positive rate of 0.2 per frame.

The classification results mirror the test video complexity differences that the hypothesis generation test already implied. However, two aspects of the combined system can be observed: At this false positive rate, not every pedestrian detected in the first stage is being classified correctly by the second stage, as the recognition rates never quite reach the maximal detection rates of table I. This may be caused by hypothesis bounding boxes that do not exactly cover the pedestrian area and thus are inaccurate or ”bad-fitted”, as the authors of [21] call the effect. Also, the classifiers cannot maintain the good false positive rate of the single classification test. This follows from the fact that the classifiers receive regions that already are hard examples and may contain possibly human-shaped objects or structures. Still, the second stage is able to identify and discard most of the negative inputs. In general, the dense descriptor shows better recognition rates, but also has a classification time that is 8 times higher than that of the custom descriptor.

To present a more detailed behavior of the recognition system, we plotted ROC curves of dense and custom descriptor on two exemplary test videos (fig. 7), where both descriptors use $BFV3$. First, note that with an increasing number of false positives per frame, the ROC curves of the descriptor variants converge as the recognition rates

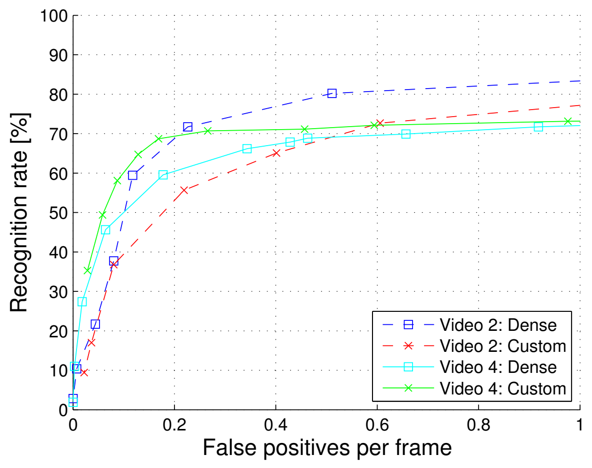

Fig. 7. Recognition rates for video 2 and 4 of dense and custom descriptor.

are bounded only by the detection rate of the hypothesis generation stage. Therefore, the interesting range to identify descriptor differences is at low false positive rates. In video 2, the dense descriptor's whole ROC curve lies above the custom descriptor's curve, thus outperforming the latter. However, in video 4 the custom descriptor clearly shows better results than the dense descriptor. The videos 2 and 4 have been chosen to illustrate that the dense descriptor, while generally being the "better" variant in terms of low false positive rates, does not necessarily achieve the best results under all circumstances. Although we tried to gather different real world scenes in our videos, naturally they can only represent a small fraction of all possible driving situations. Again, note that the rates of table II and fig. 7 originate from a single frame processing system and are likely to improve when integrating information of previous frames by tracking.

IV. Conclusion and Future Work

In this paper a two-staged approach to vision-based pedestrian detection has been presented. The first stage employs a detection mechanism based on Haar features. The second stage classifies the incoming samples using HOG-based features and a linear SVM. Different arrangements and block feature variants have been evaluated to identify the most effective descriptor configuration. The combined system exploits the fact that Haar features can be calculated rapidly and do not need high-detailed input for reasonable detection performance. Thus, the HOG features for subsequent classification only have to be evaluated at meaningful positions, making the system faster than comparable detection systems based on HOG features.

Since we tested the recognition system on our own real world data, detection and recognition rates are hard to compare to other complete recognition systems. However, since using the OpenCV pedestrian detector as a reference implementation on our videos resulted in significantly lower detection rates, we believe the presented system marks an improvement compared to systems solely based on one feature set.

Future work especially aims at the improvement of the hypothesis generation stage: If possible, the detection rate should be close to 100 % without considering false positives. To find out the most suitable parameters in detection and classification stage, we plan to apply evolutionary optimization techniques to the problem, as it was successfully done in [7] to find the best symmetry operator for vehicle detection. Further research also will focus on the implementation of a tracking mechanism to improve robustness, e.g. to increase detection rates and to reduce false positives per frame.

References

[1] K. Fuerstenberg and K. Ch and V. Willhoeft, "Pedestrian Recognition in Urban Traffic using a vehicle based Multilayer Laserscanner", *Proceedings of IV 2002*, 2002.

[2] B. Elias and P. Mahonen, "Pedestrian Recognition Based on 3D Image Data", *IEEE Symposium on Industrial Electronics*, 2007.

[3] M. Szarvas, U. Sakai and Jun Ogata, "Real-time Pedestrian Detection Using LIDAR and Convolutional Neural Networks", *Intelligent Vehicles Symposium*, 2006

[4] S. Bota and S. Nedesvchi, "Multi-Feature Walking Pedestrian Detection Using Dense Stereo and Motion", *WIT 2007*, 2007.

[5] D. Gavrila, "Pedestrian Detection from a Moving Vehicle", *Proceedings of the 6th European Conference on Computer Vision-Part 2*, 2000

[6] P. Viola and M. Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features", *IEEE Conference On Computer Vision And Pattern Recognition*, 2001.

[7] U. Kadow, G. Schneider and A. Vukotich, "Radar-Vision based Vehicle Recognition with Evolutionary Optimized and Boosted Features", *IEEE Intelligent Vehicles Symposium*, 2007

[8] P. Viola, M. Jones and D. Snow, "Detecting Pedestrians Using Patterns Of Motion And Appearance", *International Journal of Computer Vision*, 2005.

[9] N. Dalal and B. Triggs, "Histograms of Oriented Gradients for Human Detection", *IEEE Conference On Computer Vision And Pattern Recognition*, 2005.

[10] D. Geronimo, A. Lopez, D. Ponsa and A. D. Sappa, "Haar Wavelets and Edge Orientation Histograms for On-Board Pedestrian Detection", it Proceedings of the Iberian Conference on Pattern Recognition and Image Analysis, 2007.

[11] Q. Zhu, S. Avidan, M. Yeh and K. Cheng: "Fast Human Detection Using a Cascade of Histograms of Oriented Gradients", *IEEE Conference On Computer Vision And Pattern Recognition*, 2006.

[12] I. Laptev, "Improvements of Object Detection Using Boosted Histograms", *Proceedings of the British Machine Vision Conference*, 2006.

[13] H. Jin, P. Favaro and S. Soatto, "A Semi-direct Approach to Structure from Motion", *The Visual Computer, 192:1–18*, 2003.

[14] F. Porikli, "Integral histogram: a fast way to extract histograms in Cartesian spaces", *IEEE Computer Society Conference on On Computer Vision And Pattern Recognition*, 2005.

[15] C.J.C. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition", *Data Mining and Knowledge Discovery*, 1998.

[16] L. Zhao and C. E. Thorpe: "Stereo and Neural Network-Based Pedestrian Detection", *IEEE Transactions on Intelligent Transportation Systems*, 2000.

[17] G. Grubb, A. Zelinsky, L. Nilsson and M. Rilbe: "3D Vision sensing for improved pedestrian safety", *IEEE Intelligent Vehicles Symposium*, 2004.

[18] D. Gavrila and S. Munder: "Multi-Cue Pedestrian Detection and Tracking from a Moving Vehicle", *International Journal of Computer Vision*, 2007.

[19] A. Broggi, M. Bertozzi, A. Fascioli and M. Sechi: "Shape-based Pedestrian Detection", *IEEE Intelligent Vehicles Symposium*, 2000.

[20] A. Shashua, Y. Gdalyahu and G. Hayun: "Pedestrian detection for driving assistance systems: single-frame classification and system level performance", *IEEE Intelligent Vehicles Symposium*, 2004.

[21] D. Fernndez, I. Parra, M. A. Sotelo and P. Revenga: "Bounding Box Accuracy in Pedestrian Detection for Intelligent Transportation Systems", *IEEE Industrial Electronics, IECON*, 2006.