Fast moving pedestrian detection based on motion segmentation and new motion features

Shanshan Zhang[1] · **Dominik A. Klein[2]** ·
Christian Bauckhage[3] · **Armin B. Cremers[1]**

Abstract The detection of moving pedestrians is of major importance for intelligent vehicles, since information about such persons and their tracks should be incorporated into reliable collision avoidance algorithms. In this paper, we propose a new approach to detect moving pedestrians aided by motion analysis. Our main contribution is to use motion information in two ways: on the one hand we localize blobs of moving objects for *regions of interest* (ROIs) selection by segmentation of an optical flow field in a pre-processing step, so as to significantly reduce the number of detection windows needed to be evaluated by a subsequent people classifier, resulting in a fast method suitable for real-time systems. On the other hand we designed a novel kind of features called *Motion Self Difference* (MSD) features as a complement to single image appearance features, e. g. *Histograms of Oriented Gradients* (HOG), to improve distinctness and thus classifier performance. Furthermore, we integrate our novel features in a two-layer classification scheme combining a HOG+*Support Vector Machines* (SVM) and a MSD+SVM detector. Experimental results on the *Daimler mono moving pedestrian detection* benchmark show that our approach obtains a log-average

✉ Shanshan Zhang
zhangs@iai.uni-bonn.de

Dominik A. Klein
kleind@iai.uni-bonn.de

Christian Bauckhage
christian.bauckhage@iais.fraunhofer.de

Armin B. Cremers
abc@iai.uni-bonn.de

[1] Institute of Computer Science III, University of Bonn, Römerstraße 164, 53117 Bonn, Germany

[2] Fraunhofer FKIE, Fraunhoferstraße 20, 53343 Wachtberg, Germany

[3] Fraunhofer IAIS, Schloss Birlinghoven, 53757 Sankt Augustin, Germany

 Springer

miss rate of 36 % in the FPPI range $[10^{-2}, 10^{0}]$, which is a clear improvement with respect to the naive HOG+SVM approach and better than several other state-of-the-art detectors. Moreover, our approach also reduces runtime per frame by an order of magnitude.

Keywords Pedestrian detection · Motion segmentation · Motion self difference features · Histograms of oriented gradients · Support vector machines

1 Introduction

Pedestrians are omnipresent but vulnerable participants in urban traffic environments. Therefore, it is of importance for vehicles to locate them early from on-board video data and to prevent collisions. From the perspective of active safety, moving pedestrians are more endangered than those who just stand along the street; in addition, walking pedestrians are more challenging to detect, since movements cause significant variations in appearance. An early attempt concerned with this topic was done by Curio et al. [7], which was restricted to detect pedestrians walking across the street. In this paper, we consider all moving pedestrians, including those who walk across and also along the street. General pedestrian detection techniques could be readily employed for moving pedestrian detection, but specialized solutions as presented here enhance effectiveness and efficiency at the same time.

The approach we demonstrate utilizes motion information from an optical flow estimation to improve results in two different ways: one is to select ROIs from a motion field segmentation to avoid exhaustive search over the whole image; the second are novel motion based features, motivated by the characteristic motion patterns exhibited by moving pedestrians against the background, as well as other moving objects, e. g. vehicles, bicycles and so on.

Our previous findings about moving pedestrian detection was published in [36], while in this article we provide several new enhancements and deeper insights into this challenge. Most notably are the following four contributions:

Moving object localization from graph-based motion field segmentation in video frames, motion is a very important cue to distinguish moving objects from background, as moving objects usually generate different flow fields than a static background and static objects, in terms of both magnitude and orientation. This assumption still holds in a moving camera situation as from a driving vehicle. We observed that the motion field responds more sensitively already to small scale ($\sim$ 50 pixels in image height) and low contrast pedestrians than single image cues (e. g. gradients, colors). Such demanding scenes which show only low response on pedestrians for still image appearance cues, nevertheless often exhibit distinguishable optical flow vectors. An example is given in Fig. 1, where we obtained a clearly separated blob from motion segmentation around the small scale and low contrast pedestrians. We implemented an efficient graph-based segmentation algorithm on motion fields as a pre-processing to localize moving objects, e. g. walking pedestrians and driving vehicles.

Height-prior detection window generation from blob hypothesis an obvious disadvantage for sliding window detectors is that a huge amount of detection windows have to be examined to detect people on various scales in a whole image. We successfully cope with this challenge by applying our efficient algorithm for detection window proposals. These
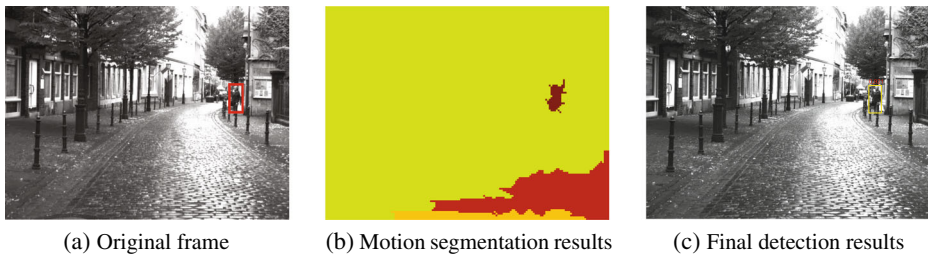
| (a) Original frame | (b) Motion segmentation results | (c) Final detection results |

Fig. 1 An example of using motion segmentation for ROIs selection when the pedestrian is of small scale and low contrast. (**a**) The *red bounding box* indicates the ground truth annotation. (**c**) The *yellow bounding box* denotes the final detection results, and the number above indicates the detection score output from our classification method

examined windows are placed along the width direction of each segmented blob considering the blob height and fixed aspect ratio as given knowledge to determine appropriate sizes. The motivation is that from our observation, a flow field of pedestrians always shows obvious boundaries along the upper and lower side of each person, while it sometimes presents weak or even no boundaries on the left and right side of human bodies. One explanation might be that pedestrians in images usually appear side-by-side instead of on-top of each other. See Fig. 4 for an example.

Motion self difference features we designed new motion based features for moving pedestrians. As exemplary shown in Fig. 5, one can notice that moving pedestrians exhibit rather characteristic motion patterns versus static backgrounds and non-articulated moving objects, e. g. driving cars, in terms of flow magnitudes. More specific, such objects exhibit a flow magnitude map reflecting their fixed outer shapes, while moving pedestrians show non-uniform structures due to inter-body motion. We make use of this property of walking pedestrians to design motion self difference features, which compute the difference between each neighboring pair of rectangular regions all over the human body w.r.t. optical flow vectors.

Moving pedestrian annotations we added "moving" annotations to the *Daimler mono pedestrian detection* benchmark. For each original pedestrian annotation, we differentiate between moving or static by examination of several video frames before and after the current frame. This was done manually by a single, expressly instructed person to avoid any bias and inhomogeneity. This was extensive, but valuable work for further research in this area. The ground truth data with moving/static annotations can be downloaded from http://www.iai.uni-bonn.de/~zhangs.

2 Related work

In recent years, the topic of pedestrian detection has attracted intensive attention in the computer vision community [11, 13], but the currently best performing approaches are still far from being reliable and real-time capable.

The arguably most prominent pedestrian detector is the HOG+SVM approach proposed by Dalal et al. [8]. From then on, HOG was considered to be an efficient feature for people detection, as it accurately describes the human-specific, steady head-shoulder and mutable lower body appearance in a rich gradient set. Most later on developed detectors employ it

at least as one basic feature, sometimes complemented with others [10, 31, 32, 35]. Apart from that, shape features are also frequently considered a useful cue: Gavrila et al. [18] introduced the *Hausdorff distance transform* and a template hierarchy to match image edges to a set of shape templates in a fast way.

Compared to appearance based features, motion features for dynamic scenes have not yet been intensively investigated in state-of-the-art detectors, due to some irregular influence from camera motion, which is difficult to remove. The *Histogram of Optical Flow* (HOF) features [9] were proposed as HOG-like motion features, which were successful on a per-window basis, but achieved only minimal benefit for full image detection in a recent evaluation [11]. Afterwards, Walk et al. [31] proposed a number of modifications to HOF features, which improve the performance modestly. More recently, Park et al. [28] computed temporal differences as features, following a weak stabilization based on coarse optical flow estimations through multiple frames. Besides, Bouchrika et al. [4] also tried to analyze person walking patterns for improving the detection accuracy.

In terms of classifiers, SVM has been very frequently used in different approaches for people detection. A linear kernel SVM [8] is computationally much faster than a latent SVM [16], while the latter can satisfy requirements for part-based models. In order to improve efficiency, Walk et al. [31] developed MPL-Boost, which is adaptive to learn multiple pose classifiers in parallel.

Techniques for ROIs extraction are of remarkable importance, although they are not widely discussed in previous work [19]. After determination of ROIs, unlikely regions such as the uniform ground plane and sky [24] can be avoided to be scanned for people, resulting in a significant reduction of the number of candidate detection windows for a more costly classifier to examine. Gualdi et al. [20] proposed a statistical based ROIs search approach which used Monte Carlo sampling for estimating the likelihood density function with Gaussian kernels; Kamijo et al. [22] applied a spatio-temporal Markov Random Field (MRF) model [23] for foreground objects extraction from background scenes; Some researchers used symmetry and pedestrian size constraints to select ROIs [2, 3, 5]; Itti et al. [21] employed a biologically inspired attentional algorithm that selects candidate regions according to a saliency map built on color, intensity, and gradient orientation of pixels; Elzein et al. [12] and Lim et al. [25] selected moving object regions based on temporal differences computed between successive frames; Enzweiler et al. [14] used Bayes' rule to estimate the posterior for the presence of a pedestrian in a certain image region based on motion parallax features. In some cases, when stereo images are provided, more information can be exploited. For example, Franke et al. [17] proposed to merge stereo processing and motion analysis to detect moving objects; Benenson et al. [1] estimated the stixel world to restrict search regions.

3 Overview on our approach

A flow chart of our approach is shown in Fig. 2. The whole procedure can be divided into four main parts:

Motion segmentation Graph-based motion segmentation is applied on the optical flow field, which is computed from two down-sampled, consecutive input frames.

ROIs selection and detection window generation Interesting segments are considered blobs of pedestrians. They are selected according to permissible sizes and positions from
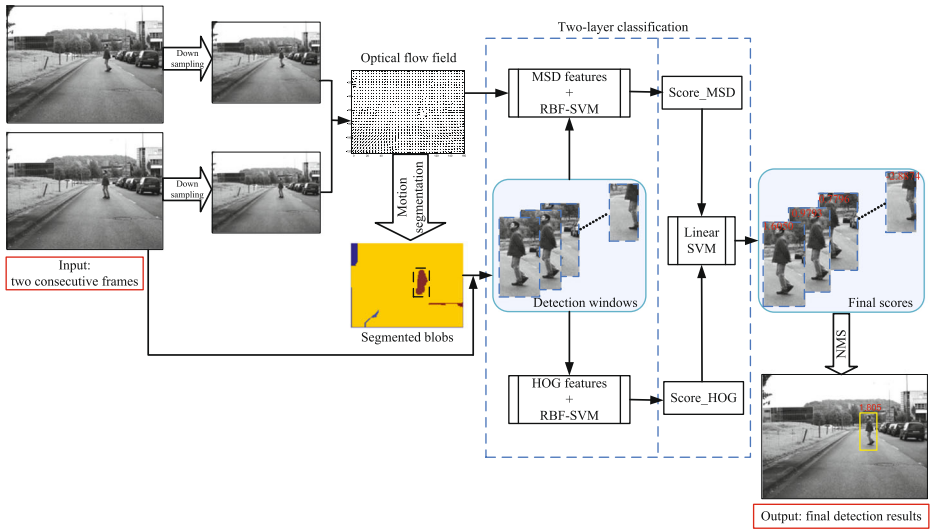
Fig. 2 The flow chart of our approach. See more details in Section 3

prior knowledge. For each one, a line of detection windows is defined using a height-prior principle based on the blob's height progression.

Feature computation HOG features and MSD features are extracted from each detection window.

Classification We propose a two-layer classification scheme, adaptive to two different kinds of features. First, we train a SVM with Radial Basis Function kernel (RBF-SVM) for each kind of feature separately. Therefore, we get two primary scores for every detection window from two individually trained classifiers. Next, a linear SVM trained on those two primary scores decides about the merged confidence for each detection window, and the final detection results are obtained by a *non-maximum suppression* (NMS) approach [16] to suppress less confident windows nearby.

4 Graph-based motion segmentation for detection window generation

We apply motion segmentation as a pre-processing step to select ROIs, where moving objects may appear. Since the magnitudes of optical flow vectors are related to the distance between camera and subjects, also static persons are distinguishable from the background when the camera itself moves, especially if the following distance ratio R_z is high:

$$R_z = \frac{\mathrm{distance}(bg, cam)}{\mathrm{distance}(person, cam)}. \tag{1}$$

Therefore, although we focus on moving pedestrian detection in this paper, we state that our approach for ROIs selection can be extended for static pedestrian detection given an accurate optical flow estimation. The more accurate optical flow is, the weaker such constraint on R_z.

4.1 Graph-based motion segmentation

We use a robust optical flow estimation algorithm introduced by Liu [26], which produces a layered flow field from two consecutive frames I_t and I_{t+1}. Because we consider optical flow computation to be too time consuming for the full resolution frame, we resize the input frames to 1/16 at first, preserving the original aspect ratio. Fortunately, we find the image down-sampling causes little performance loss, while the overall consumed time is significantly reduced.

We treat the optical flow field as a two-channel image, which is called flow image $I_f(t)$ in the following procedure, and its two channels f_x and f_y denote flow vector elements in direction of the corresponding axis. Before segmentation, we first employ gentle Gaussian smoothing ($\sigma = 0.8$) on the flow image in order to reduce noise while only slightly influencing the region edges. After that, segmentation is processed following an efficient graph based algorithm proposed by Felzenszwalb [15], which we ported to work on flow images.

First, we build a 4D feature space by mapping each pixel p_i to a vector

$$\vec{v}(i) = (x(i), y(i), f_x(i), f_y(i))^{\mathsf{T}}, \tag{2}$$

where $x(i)$ and $y(i)$ are the coordinates of p_i in $I_f(t)$, and $f_x(i)$ respectively $f_y(i)$ are its flow elements.

Second, we construct the graph $G = (V, E)$ by connecting each vertex (feature vector) to its 8 nearest neighbors within this 4D-space. The weight $\mathrm{w}(v_i, v_j)$ of an edge, formed by vertices v_i and v_j, is the L_2 (Euclidean) distance between the two corresponding points in feature space.

Finally, we iterate on the set of edges to combine components according to weight $\mathrm{w}(v_i, v_j)$ until a segmentation S which satisfies

$$\mathrm{w}(v_i, v_j) > \mathrm{MInt}(C_i, C_j), \quad v_i, v_j \in V, \quad i \neq j \tag{3}$$

has been found. Here, C_i and C_j represent the components which v_i and v_j belong to, and MInt is the minimum internal difference, defined as:

$$\mathrm{MInt}(C_i, C_j) = \min(\max_{p,q \in C_i} \mathrm{w}(p, q) + \tau(C_i),$$
$$\max_{p,q \in C_j} \mathrm{w}(p, q) + \tau(C_j)). \tag{4}$$

The threshold function τ is used to control the degree of segmentation. The boundary between two components is evidenced when the difference between them is greater than their internal difference. τ controls the gap between intra- and inter-component differences. In our method, we define the function τ as follows:

$$\tau(C) = \frac{k}{|C|}, \tag{5}$$

where, k is a constant number, and $|C|$ is the number of pixels inside component C. This means, for small components we require stronger evidence for boundary. Intuitively, a bigger number of k produces larger segmented components.

4.2 Analysis of interesting blobs

Assuming we obtain n segments from the previous step, only a subset is selected as ROIs. Interesting blobs are those probably containing moving pedestrians, chosen according to the following weak constraints of the blobs' sizes and positions: from our observation, those blobs belonging to the background usually have a large extent and huge span from left to

right, so one can reject them easily; those blobs smaller than the lower limit of pedestrian size we want to detect(cf. Section 7.1) are considered as noises and should be discarded; assuming pedestrians are standing on the ground, the lower boundary of each interesting blob can not go above the vanishing point.

The i-th segment comprises a set of $s(i)$ unique pixels

$$S_i = \{(x_1, y_1), (x_2, y_2), ..., (x_{s(i)}, y_{s(i)})\}_i, \tag{6}$$

that is inspected to fulfill the following conditions:

$$w_{\min} \leq \max_{a,b \in [1,s(i)]} \{|x_a - x_b|\}_i \leq w_{\max}, \tag{7}$$

$$h_{\min} \leq \max_{a,b \in [1,s(i)]} \{|y_a - y_b|\}_i \leq h_{\max}, \tag{8}$$

$$\max_{k \in [1,s(i)]} \{y_k\}_i \geq \xi y_v, \tag{9}$$

where y_v indicates the vertical coordinate of the vanishing point, which can be computed given camera parameters; $\xi(\xi < 1)$ is the tolerance parameter.

Some number m of interesting blobs remain which satisfy the above requirements and undergo further examinations for inside pedestrian detection.

4.3 Detection window generation

In this subsection, we describe how detection windows are created from interesting blobs as the input of our classifier. We generate detection window coordinates from blobs in $I_f(t)$, but crop them from the original frame I_t, thus a coordinate transformation is necessary to reverse down-sampling (cf. Section 4.1). In order to crop reasonable detection windows, a novel height-prior algorithm as illustrated in Algorithm 1 is proposed to obtain window coordinates from segmented blobs in motion frames. For each blob, we employ a sliding window strategy to generate detection windows from left to right. At each x-coordinate, the height of each detection window is adjusted to the upper and lower boundary of the blob at current coordinate. In order to include some context information for feature computation, we use a ratio of m_b to grow the height value. After fixing the height of the detection window, the width is calculated using a constant aspect ratio r_{wh} to this height. This aspect ratio is a fixed parameter predefined by the training examples when learning the classification model. Figure 3 shows two examples of detection windows at different positions.

Algorithm 1 Generating detection windows from selected interesting blobs

1: initialize the list of detection windows: $T_{\text{wnd}} \leftarrow \emptyset$;
2: **for** each interesting blob $i \in [1, m]$ **do**
3:     set $x_{\min} \leftarrow \min_{k \in [1,s(i)]} \{x_k\}$ and $x_{\max} \leftarrow \max_{k \in [1,s(i)]} \{x_k\}$;
4:     **for** $u = x_{\min}$ to $x_{\max}$ **do**
5:         $j \leftarrow u - x_{\min} + 1$;
6:         upper boundary: $y_t \leftarrow \min_{x_k = u, k \in [1,s(i)]} \{y_k\}$;
7:         lower boundary: $y_b \leftarrow \max_{x_k = u, k \in [1,s(i)]} \{y_k\}$;
8:         height: $h \leftarrow |y_b - y_t|(1 + 2m_b)$;
9:         width: $w \leftarrow h \cdot r_{\text{wh}}$;
10:        center point: $(x_c, y_c) \leftarrow (u, \frac{y_t + y_b}{2})$;
11:        append (h, w, x_c, y_c) to T_{wnd};
12:    **end for**
13: **end for**
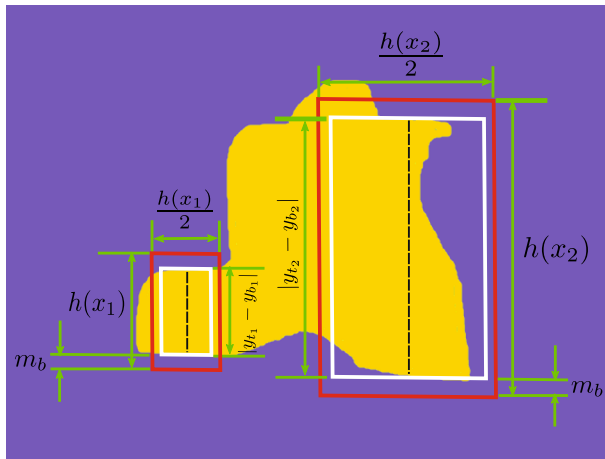14: **return** T_{wnd}

Fig. 3 Generation of detection windows from a blob: two examples are drawn at different horizontal positions

The reason for this height-prior rule is that from our observation, the optical flow around human bodies usually exhibits more apparent edges in the head-to-sky and feet-to-ground areas; in contrast, the flow field can easily form a horizontally connected region when a group of people walking shoulder by shoulder as illustrated in Fig. 4.

We emphasize two details turned out to be important to obtain good performance in this procedure: one is to keep the aspect ratio r_{wh} of each detection window fixed to the same as the classification model (64/128); the other is to add surrounding context with a fixed proportion of m_b applied to height of the detection window to include gradient information between a person and background during training and classification. In our approach, we choose m_b to be 0.2.

(a) Multiple pedestrians (b) Segmented blobs

Fig. 4 An example of blobs each containing multiple people walking side by side. The *green bounding boxes* in (**a**) indicate ground truth annotations

5 Motion self difference features

Typical urban traffic environments consist of various objects, including buildings, vehicles, pedestrians and so on. In this paper, we categorize all the above objects into: static backgrounds, including all the static objects, e. g. buildings and static pedestrians etc.; and moving objects, e. g. walking pedestrians, driving cars, etc.. After the motion segmentation procedure, blobs probably containing moving objects are selected, but inevitably some static scene parts may also be included due to high R_z (cf. (1)), errors from the optical flow estimation, or segmentation algorithm. Our task in the classification procedure is to distinguish moving pedestrians from all other objects, thus we observe the optical flow magnitude maps of various objects to seek representative features for moving pedestrians.

We observed that moving pedestrians exhibit rather different flow patterns from static backgrounds including static pedestrians and other moving objects, e. g. moving cars. The major reason is that moving pedestrians generate non-consistent motions from different body parts. See Fig. 5 for examples. More specific, from the flow magnitude maps of static backgrounds and rigid moving objects, one can clearly see the constant silhouette of the objects; in contrast, from the flow magnitude maps of moving pedestrians, only the silhouette of the upper bodies stays constant, but there is a significant difference between two legs, since only one leg at a time moves. We call this inter-body motion, caused by the non-rigidity of human body. Based on the above observation, we propose MSD features to describe the inter-body *relative* motion for moving pedestrians. The feature extraction procedure is listed in Algorithm 2.
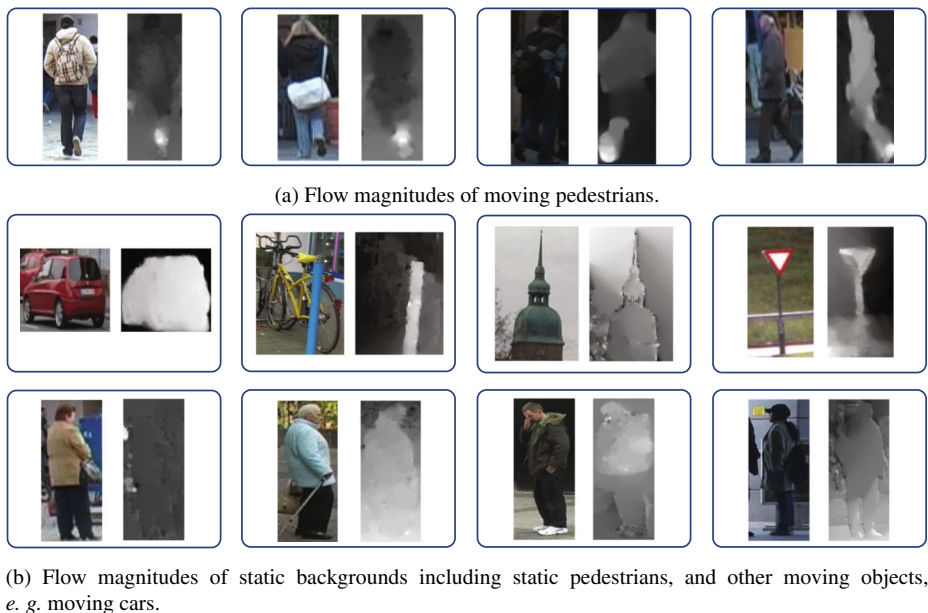

(a) Flow magnitudes of moving pedestrians.

(b) Flow magnitudes of static backgrounds including static pedestrians, and other moving objects, *e. g.* moving cars.

Fig. 5 Several examples of flow magnitudes for various objects in a typical urban traffic environment. Each *blue bounding box* contains a pair of the original image and its corresponding flow magnitude map

First, we resize each detection window to the size of our pedestrian model: 64×128 pixels, and divide the resized window into $s \times s$ pixel square-sized regions, which we call *cells* in the following. We denote the number of cells along the horizontal and vertical directions as nc_h and nc_v, respectively.

Second, we compute two histograms for each cell using trilinear interpolation [8] in terms of f_x and f_y respectively. For the whole detection window, this yields two histogram sets, denoted

$$H_x = \{h_x(i, j) | 1 \le i \le nc_h, 1 \le j \le nc_v\}, \tag{10}$$

and

$$H_y = \{h_y(i, j) | 1 \le i \le nc_h, 1 \le j \le nc_v\}. \tag{11}$$

Third, we compute the difference between histograms in H_x and H_y respectively. For each cell, we consider all of its eight nearest neighboring cells and compute the difference between the centering cell and each neighboring cell. The difference measure we utilize is histogram intersection (HI). Given two histograms h_p and h_q each containing n bins, histogram intersection is defined as

$$\mathrm{HI}(h_p, h_q) = \frac{\sum_{k=1}^{n} \min(h_p(k), h_q(k))}{\sum_{k=1}^{n} h_p(k)}. \tag{12}$$

However, if we iterate over all cells, redundancy will emerge since some cell pairs are considered for twice. Therefore, we employ a tabu list to solve this problem. Those cell pairs which have been considered once will be appended to the tabu list and a second visit is thus prohibited.

Choosing the cell size of s to be 8 pixels for a 64×128 pixels detection window, we obtain a 1072 dimensional feature vector. Notably, those cells along detection window borders do not have eight neighboring cells.

Algorithm 2 Extracting motion self difference features

1: compute histogram sets H_x and H_y for the detection window;
2: initialize feature set: $F_{\mathrm{MSD}} \leftarrow \emptyset$;
3: initialize cell pair tabu set: $P \leftarrow \emptyset$;
4: initialize cell pair index set used for each cell:
   $dx = \{0, 1, 1, 1, 0, -1, -1, -1\}, dy = \{-1, -1, 0, 1, 1, 1, 0, -1\}$;
5: **for** $x_c = 1$ to nc_h **do**
6:     **for** $y_c = 1$ to nc_v **do**
7:         **for** $i = 1$ to 8 **do**
8:             $x_n = x_c + dx[i], y_n = y_c + dy[i]$;
9:             **if** $x_n \notin [1, nc_h] \ || \ y_n \notin [1, nc_v]$ **then**
10:                break;
11:             **else if** $[(x_c, y_c), (x_n, y_n)] \notin P$ AND $[(x_n, y_n), (x_c, y_c)] \notin P$ **then**
12:                compute $\mathrm{HI}(h_x(x_c, y_c), h_x(x_n, y_n))$ using Eq. 12 and append it to F_{MSD};
13:                compute $\mathrm{HI}(h_y(x_c, y_c), h_y(x_n, y_n))$ using Eq. 12 and append it to F_{MSD};
14:                append $[(x_c, y_c), (x_n, y_n)]$ to the tabu set P;
15:             **end if**
16:         **end for**
17:     **end for**
18: **end for**
19: **return** F_{MSD}

---

6 Two-layer classification

In this paper, we use two different kinds of features: HOG, which is appearance based; MSD, which is motion based. Generally, there are two classification strategies to cope with multiple features. One is to combine features, i. e. to concatenate all the features into a long feature vector and train one classifier as a whole; another is to combine classifiers, i. e. to train one classifier for each set of features individually, and then combine the outputs of these classifiers to make a joint decision.

The first method is easy to implement, but it might suffer from the curse of dimensionality; the second method consists of a two-layer classification procedure, more adaptable to high variation among different kinds of features, which may generate quite different boundaries during training. Therefore, we apply the two-layer classification scheme in this article (cf. the illustration in Fig. 2).

In the first layer, same as in [8], we choose support vector machines as classifiers for both HOG and MSD features but in two separate spaces. We use a RBF kernel instead of a linear kernel, which has been widely used in similar applications. Apparently, a linear kernel is most often chosen due to its efficiency but not quality of results. Since in our approach we examine much less detection windows than previous methods did, the time consumed by a SVM based on RBF kernel is still tolerable.

In the second classification layer, there are multiple possible ways to combine the decision scores from the first layer [6]. In this case, a linear kernel SVM suffices, since there is an only two dimensional feature space consisting of the two decision scores from the first layer classifiers. The trained linear kernel SVM outputs the final decision score for the whole classification procedure.

Similar to [8], we also use the INRIA person dataset as training dataset for HOG features, while we use the TUD-Brussel dataset [34] for MSD features since this dataset provides motion pair annotations. A multi-round training strategy is used for both features, i. e. the final model is produced by retraining using an augmented set (initial training data + hard negative samples).

After receiving a final score from our classification procedure for each detection window, the final detections in the current frame are determined by a pairwise non-max suppression method [16], which discards the lesser-confident of every pair of detection windows that overlap sufficiently according to (13).

7 Experiments

To our knowledge, none of the current public pedestrian datasets provide "moving" annotations for pedestrians. Therefore, we have to choose one as our basis dataset and then manually annotate moving pedestrians. In order to choose an appropriate dataset for experiments, we have the following criteria: first, it should contain consecutive video frames for optical flow computation; second, the size of the dataset should be big, with a large number of annotated pedestrians; third, pedestrians of bigger size are preferable, because it makes adding moving annotations easier.

Based on the above criteria, we took the *Daimler mono pedestrian detection* benchmark [13], which was captured by a monochrome camera installed on a vehicle driving through urban environment, as our basis dataset. This dataset consists of 21,790 consecutive frames (640×480), as well as 56,492 pedestrian annotations. The on-board camera setting and large amount of pedestrian annotations meet our purpose of an extensive test

for intelligent vehicle applications. Next, we manually determined each pedestrian annotation to be moving or static, through observing multiple consecutive video frames before and after the current time point. Then we added an additional label of "moving" to the ground truth data, so as to record the movement status for each pedestrian annotation.

In this section, we explain our evaluation protocol, examine performance of our detector under different configurations, and evaluate the performance of our approach in comparison to several other methods using the same protocol. In addition, we discuss the runtimes of all the detectors considered in this article in Section 7.4.

7.1 Evaluation protocol

In the following, we explain details of our evaluation protocol in four aspects, which are consistent to the conventions in this field.

Ground truth regulation [1] (1) *Ignored* data selection: In our experiments, three categories of pedestrians are not considered: first, static pedestrians; second, pedestrians of less than 50 pixels in image height, which corresponds to a real-world height of 1 meter at a distance of 25 meters in this dataset; third, pedestrians that are heavily occluded (36-100 % occlusion). All the above data is marked with an *ignore* label, which means they need not be matched, however, matches are not considered as mistakes either. (2) Aspect ratio standardization: because most of the detectors use windows with aspect ratio of 0.5, it is important that the ground truth is annotated in the same way to obtain meaningful performances [11]. We standardize all ground truth bounding boxes by keeping the original height and center while adjusting the width. From our observation, performances of different detectors stay stable for various choices of r_{wh}.

Detection results filtering Detection results are filtered out using an expanded filtering method [11], so that detection results far outside the evaluation scale range should not be considered. When evaluating a scale range of $[S_1, S_2]$, only detections in $[S_1/\xi, S_2\xi]$ are considered for evaluation. In our evaluation, we set $\xi = 1.25$.

Matching rules Filtered ground truth bounding boxes and detection results bounding boxes are annotated by B_{gt}, and B_{dt} respectively. A detected bounding box and a ground truth bounding box match if and only if the ratio of overlap to the union of their areas exceeds a given threshold [13]:

$$\mathrm{match}(B_{\mathrm{dt}}, B_{\mathrm{gt}}) = \frac{\mathrm{area}(B_{\mathrm{dt}}) \cap \mathrm{area}(B_{\mathrm{gt}})}{\mathrm{area}(B_{\mathrm{dt}}) \cup \mathrm{area}(B_{\mathrm{gt}})} \overset{!}{>} 0.25 . \tag{13}$$

Performance measurements We perform full image evaluation instead of per-window evaluation as the former one provides a natural measure of error of an overall detection system. In order to compare different detectors, we plot miss rate against false positives per image (FPPI) curves in logarithmic scales by varying the threshold on the detection confidence of the classifiers. We only plot the curves in FPPI range between $(-\infty, 10^0]$ as more than 10^0 FPPI is anyway unacceptable for intelligent vehicle applications. In addition to this miss rate vs. FPPI curves, we calculate a single, numerical measurement to summarize

---

[1]Regulated ground truth annotations can be downloaded at http://www.iai.uni-bonn.de/~zhangs

detector performance. We use the *log-average miss rate* [11], which is computed by averaging the miss rates at nine FPPI rates evenly sampled in log-space in the range of $[10^{-2}, 10^0]$. This *log-average miss rate* generally gives a more stable and informative assessment of the overall performance for different detectors than the single miss rate at 10^{-1} FPPI, according to [11].

7.2 Comparisons of our detectors under different configurations

To examine the improvements brought by motion segmentation and motion self difference features, we define two detectors with different configurations: MoSeg+HOG, using motion segmentation for ROIs selection and HOG features for classification; MoSeg+HOG+MSD, using motion segmentation for ROIs selection and, both HOG and MSD features for classification. We compare the above two detectors with a solely HOG feature based detector to observe the performance gains from both modifications.

From Fig. 6, we can see that supplementing motion segmentation for ROIs selection to original HOG detector results in a significant decrease of miss rate by 12 % on average; moreover, the integration of our novel, motion based MSD features yields an additional performance improvement of 7 % w.r.t. the average miss rate, compared to the above detector MoSeg+HOG. Therefore, our optimal detector is MoSeg+HOG+MSD, which results in an overall improvement of 19 % w.r.t. the average miss rate of the original HOG detector.

7.3 Comparisons against state-of-the-art detectors

We also compare our proposed MoSeg+HOG+MSD detector against several state-of-the-art detectors from a recent survey [11] in this field: original HOG [8] as the most similar approach; VJ [30], Shapelet [29] and MultiFtr [33] as comparison for different features;
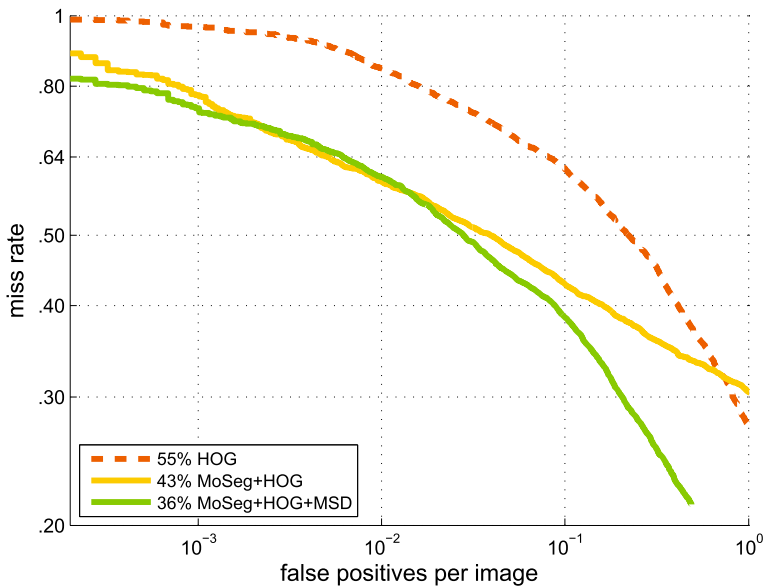

Fig. 6 Performance comparison of our detectors under different configurations on Daimler Monocular Pedestrian Dataset
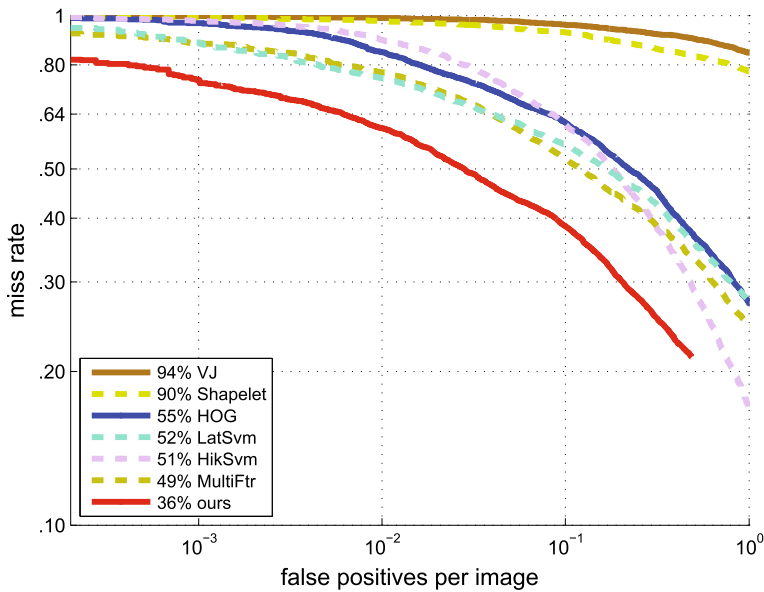
Fig. 7 Performance comparison of state-of-the-art detectors on Daimler Monocular Pedestrian Dataset

HikSvm [27] and LatSvm [16] as comparison for different SVM kernels. Here, we do not consider those detectors who use color information, as the dataset only contains gray scale images. Moreover, we do not discuss those detectors whose runtimes are on minute-level per frame (640×480 pixels) reported by [11], as they are far from real-time requirements for intelligent vehicle applications. All the original detection results[2] of the 6 selected detectors plus our own detection results are evaluated using the same evaluation protocol explained earlier in Section 7.1.

The miss rate vs. FPPI curves of above detectors are shown in Fig. 7, where the *log-average miss rate* values are indicated in the left bottom rectangle within this figure. Generally, our approach obtains a *log-average miss rate* of 36 %, lower than other state-of-the-art detectors, and our miss rate is consistently lower in the FPPI range of $[10^{-3}, 10^{-1}]$, which is an useful range for intelligent vehicle applications (Fig. 8).

7.4 Runtime analysis

We perform our experiments on a modern computer with Intel Core-i5 CPU (2.4GHz), and our source code is written in Matlab. Our runtime for one single frame is 1.93 seconds on average obtained by evaluating the whole dataset of 21,790 frames.

As it would be infeasible work to re-implement every other detector and rerun them on our computer, we instead normalize all the runtimes using the data in [11] to the rate of a single machine. Here, we explain the normalization procedure in more detail. First, we found there is one detector's code publicly available, among the 15 detectors selected in [11]. We downloaded it and ran it on our computer. After obtaining its runtime on our computer, we compare it with its runtime reported by [11], thus we get a relative speed ratio

---

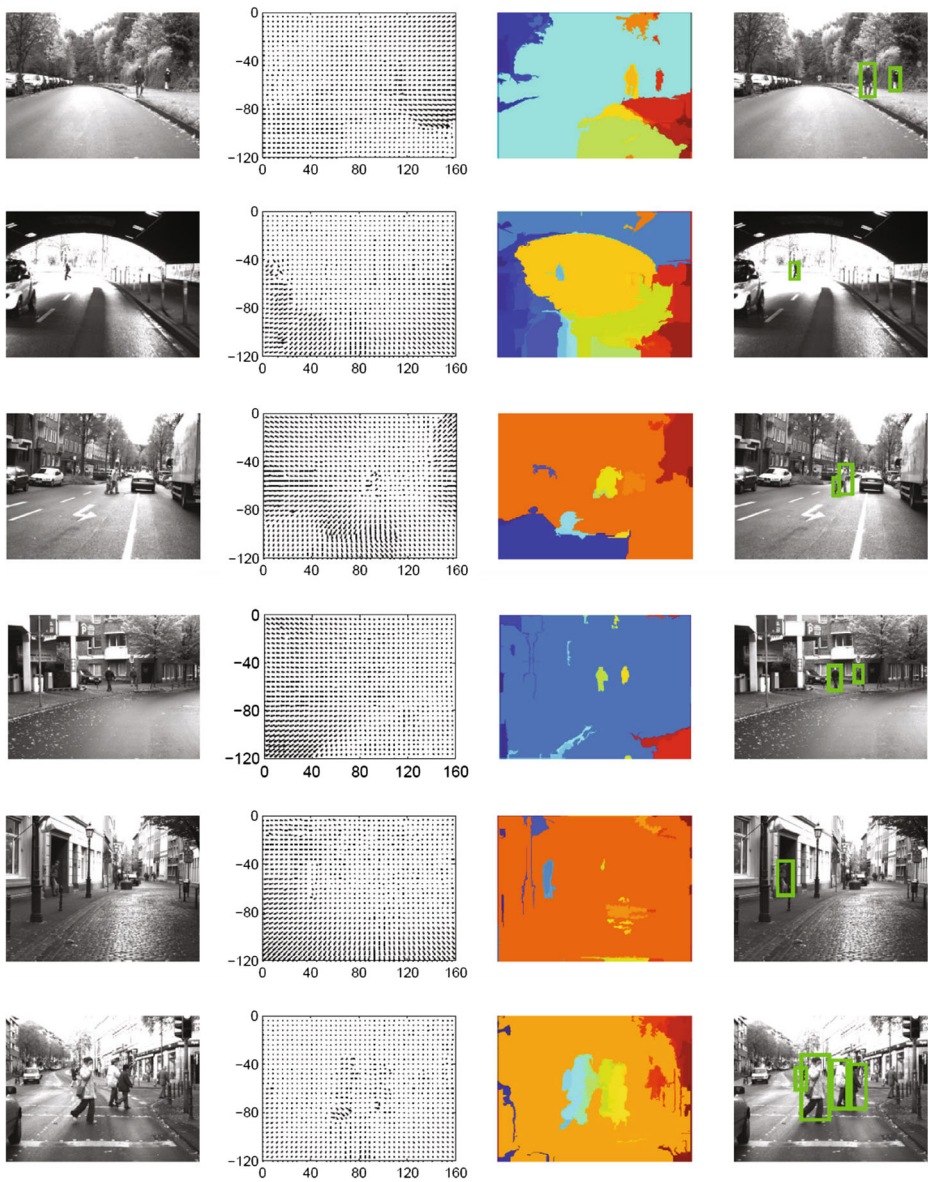[2]http://www.vision.caltech.edu/Image_Datasets/CaltechPedestrians

Fig. 8 Some exemplary results from our approach in different scenarios. In each row, the original frame, optical flow field, motion segmentation blobs and final detection results are shown from *left* to *right*

between our computer and the computer used in [11]. Then we use this ratio to normalize all the runtimes of detectors to the rate of our computer. A similar method was used in [11]. From Table 1, we can find, among the other six state-of-the-art detectors, LatSvm [16] is the fastest one with an average runtime of 10.86 seconds on a 640×480 frame, but our approach is around 10 times faster. The total runtime of our detector is 1.93 seconds, consisting of 0.63 seconds for ROIs selection and 1.60 seconds for classification. Among all the detectors

Table 1 Runtimes (in seconds) of different detectors by normalizing to the rate of a single machine

Detector	ROIs selection	Features	Classifier	Runtime
VJ [30]	×	Haar-like	AdaBoost	11.97
Shapelet [29]	×	Gradients	AdaBoost	106.43
HOG [8]	×	HOG	linSVM	19.72
LatSvm [16]	×	HOG	latent SVM	10.86
HikSvm [27]	×	HOG	HIK-SVM	29.58
MultiFtr [33]	×	HOG+Haar	AdaBoost	62.63
ours	√	HOG+MSD	RBF-SVM+linSVM	1.93

listed in Table 1, our detector is the fastest and the only one who applies ROIs selection, which demonstrates that ROIs selection is very important for enhancing efficiency.

7.5 Discussion

Based on the experimental results reported in Section 7.3, we want to highlight two aspects of our comparisons in more detail: first, our modifications to the HOG approach result in a significant decrease of miss rate by 19 % on average, which confirms our improvements are successful; moreover, we outperform MultiFtr, which combines various kinds of appearance features, such as Haar-like features, shapelets etc., thus an integration of temporal information and spatial information seems to be more informative than spatial information alone, and we managed to exploit it.

Another important aspect of evaluating a detector is the speed. The runtime for each detector reported in Section 7.4 are evaluated for one single frame. For our detector, the computational costs come from two parts: ROI selection and detection algorithm. For other detectors, the costs only come from detection algorithm. Therefore, there is a trade-off between additional ROI selection costs and reduced detection costs. From Table 1, we can see that our detector is significantly faster than other detectors considered for comparison. This improvement indicates that our ROI selection method is an effective way to reduce the runtime of the whole detector. More importantly, our detector ensures reasonable performance while reducing the runtime.

8 Conclusion and future work

In this paper, we proposed a new approach utilizing motion information for moving pedestrian detection. A main contribution is to utilize motion information from optical flow estimation for ROIs selection. This is accomplished by motion segmentation and blob selection. Furthermore, the motion field is used for novel motion features extraction through computing self differences between two neighboring regions' histograms of flow vectors. These novel features have been shown to produce distinctive patterns for moving pedestrians versus static backgrounds, static pedestrians, and other moving objects, thus increased classification performance.

Our approach significantly reduced the number of examined detection windows by explicit generation of suitable pedestrian hypotheses inside motion field blobs (ROIs),

resulting in less false positives as well as lower runtime. Furthermore, a two-layer classification scheme enhanced the detection confidence by combination of a HOG+SVM with a MSD+SVM classifier using a linear SVM.

Experimental results on *Daimler mono pedestrian detection* benchmark showed that our approach obtained a lower *log-average miss rate* than several state-of-the-art detectors. At the same time, we improved the efficiency by an order of magnitude.

In the future, we will focus on motion pattern analysis for other moving objects e. g. moving cars, so as to extend our approach to recognize various moving objects, enabling a better understanding of the urban traffic environment for collision avoidance and other tasks.

References

1. Benenson R, Mathias M, Timofte R, Van Gool L (2012) Pedestrian detection at 100 frames per second. In: Proceedings of the IEEE conference on computer vision and pattern recognition
2. Bertozzi M, Broggi A, Chapuis R, Chausse F, Fascioli A, Tibaldi A (2003) Shape-based pedestrian detection and localization. In: Proceedings of the IEEE conference on intelligent transportation systems
3. Bertozzi M, Broggi A, Fascioli A, Tibaldi A, Chapuis R, Chausse F (2004) Pedestrian localization and tracking system with kalman filtering. In: Proceedings of the IEEE intelligent vehicles symposium
4. Bouchrika I, Carter JN, Nixon MS, Mörzinger R, Thallinger G (2010) Using gait features for improving walking people detection. In: Proceedings of the international conference on pattern recognition
5. Broggi A, Fascioli A, Fedriga I, Tibaldi A, Rose M (2003) Stereo-based preprocessing for human shape localization in unstructured environments. In: Proceedings of the IEEE intelligent vehicles symposium
6. Chen K, Wang L, Chi H (1997) Methods of combining multiple classifiers with different features and their applications to text-independent speaker identification. Int J Pattern Recog Artif Intell 11(3):417–445
7. Curio C, Edelbrunner J, Kalinke T, Tzomakas C, Von Seelen W (2000) Walking pedestrian recognition. IEEE Trans Intell Transport Syst 1(3):155–163
8. Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition
9. Dalal N, Triggs B, Schmid C (2006) Human detection using oriented histograms of flow and appearance. In: Proceedings of the European conference on computer vision
10. Ding J, Wang Y, Geng W (2013) An hog-ct human detector with histogram-based search. Multimed Tools Appl 63(3):791–807
11. Dollár P, Wojek C, Schiele B, Perona P (2011) Pedestrian detection: an evaluation of the state of the art. IEEE Trans Pattern Anal Mach Intell 34(4):743–761
12. Elzein H, Lakshmanan S, Watta P (2003) A motion and shape-based pedestrian detection algorithm. In: Proceedings of the IEEE intelligent vehicles symposium
13. Enzweiler M, Gavrila DM (2009) Monocular pedestrian detection: survey and experiments, vol 31
14. Enzweiler M, Kanter P, Gavrila DM (2008) Monocular pedestrian recognition using motion parallax. In: Proceedings of the IEEE intelligent vehicles symposium
15. Felzenszwalb PF, Huttenlocher DP (2004) Efficient graph-based image segmentation. Int J Comput Vis 59(2):167–181
16. Felzenszwalb PF, McAllester D, Ramanan D (2008) A discriminatively trained, multiscale, deformable part model. In: Proceedings of the IEEE conference on computer vision and pattern recognition
17. Franke U, Heinrich S (2002) Fast obstacle detection for urban traffic situations. IEEE Trans Intell Transp Syst 3(3):173–181
18. Gavrila DM, Munder S (2007) Multi-cue pedestrian detection and tracking from a moving vehicle. Int J Comput Vis 73(1):41–59
19. Gerónimo D, López AM, Sappa AD, Graf T (2010) Survey of pedestrian detection for advanced driver assistance systems. IEEE Trans Pattern Anal Mach Intell 32(7):1239–1258
20. Gualdi G, Prati A, Cucchiara R (2010) Multi-stage sampling with boosting cascades for pedestrian detection in images and videos. In: Proceedings of the European conference on computer vision
21. Itti L, Koch C, Niebur E (1998) A model of saliency-based visual attention for rapid scene analysis. IEEE Trans Pattern Anal Mach Intell 20(11):1254–1259

22. Kamijo S, Fujimura K, Shibayama Y (2010) Pedestrian detection algorithm for on-board cameras of multi view angles. In: Proceedings of the IEEE intelligent vehicles symposium
23. Kamijo S, Matsushita Y, Ikeuchi K, Sakauchi M (2000) Occlusion robust tracking utilizing spatio-temporal markov random field model. In: Proceedings of the international conference on pattern recognition
24. Leibe B, Cornelis N, Cornelis K, Van Gool L (2007) Dynamic 3d scene analysis from a moving vehicle. In: Proceedings of the IEEE conference on computer vision and pattern recognition
25. Lim J, Kim W (2013) Detecting and tracking of multiple pedestrians using motion, color information and the adaboost algorithm. Multimed Tools Appl 65(1):161–179
26. Liu C (2009) Beyond pixels: exploring new representations and applications for motion analysis. Ph.D. thesis, Massachusetts Institute of Technology
27. Maji S, Berg AC, Malik J (2008) Classification using intersection kernel support vector machines is efficient. In: Proceedings of the IEEE conference on computer vision and pattern recognition
28. Park D, Zitnick CL, Ramanan D, Dollár P (2013) Exploring weak stabilization for motion feature extraction. In: Proceedings of the IEEE conference on computer vision and pattern recognition
29. Sabzmeydani P, Mori G (2007) Detecting pedestrians by learning shapelet features. In: Proceedings of the IEEE conference on computer vision and pattern recognition
30. Viola P, Jones MJ (2004) Robust real-time face detection. Int J Comput Vis 57(2):137–154
31. Walk S, Majer N, Schindler K, Schiele B (2010) New features and insights for pedestrian detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition
32. Wang X, Han TX (2009) An HOG-LBP human detector with partial occlusion handling. In: Proceedings of the international conference on computer vision
33. Wojek C, Schiele B (2008) A performance evaluation of single and multi-feature people detection. In: Proceedings of the symposium of the german association for pattern recognition (DAGM)
34. Wojek C, Walk S, Schiele B (2009) Multi-cue onboard pedestrian detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition
35. Zhang S, Bauckhage C, Cremers A (2014) Informed haar-like features improve pedestrian detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition
36. Zhang S, Bauckhage C, Klein D, Cremers A (2013) Moving pedestrian detection based on motion segmentation. In: Proceedings of the IEEE workshop on robot vision (WoRV)

Shanshan Zhang received the M.Sc. degree in signal and information processing from Tongji University, Shanghai, China, in 2011. She is currently working towards the Ph.D. degree in the Intelligent Vision Systems group, University of Bonn. She was once a visiting researcher in Multimedia Information Research Division, National Institute of Informatics, Tokyo, Japan. Her main research interests include computer vision and pattern recognition and their applications in intelligent transportation systems.

Dominik A. Klein holds a masters degree in computer science (Dipl. Inform.) from the University of Bonn. There, since 2009 he is strenghtening the Intelligent Vision Systems group as a researcher and Ph.D. candidate. His main areas of interest include robotics, pattern recognition, object tracking, and biologically inspired vision systems.

Christian Bauckhage received the M.Sc. and Ph.D. degree in Computer Science from Bielefeld University, Germany, in 1998 and 2002, respectively. He is currently a professor of media informatics and pattern recognition at the University of Bonn and a lead scientist for media engineering at the Fraunhofer Institute for Intelligent Analysis and Information Systems (IAIS) in Bonn, Germany. Prior to his current position, he was a postdoctoral researcher at the Centre for Vision Research in Toronto, Canada, and worked as a senior research scientist at Deutsche Telekom Laboratories in Berlin, where he conducted and coordinated industrial ICT research. His research focuses on large scale, descriptive data mining and efficient statistical machine learning for multimedia applications. In these areas, he regularly publishes conference papers and journal articles and frequently serves on programm committees and editorial boards

Armin B. Cremers received his doctoral degree in Mathematics in 1972 from the University of Karlsruhe. He has served on the Computer Science Faculties of the University of Southern California, the University of Dortmund, and, since 1990, the University of Bonn where he is Head of the Artificial Intelligence / Intelligent Vision Systems Research Groups. In 2002 he became Founding Director of the Bonn-Aachen International Center for Information Technology (B-IT), jointly with RWTH Aachen University and Fraunhofer Society.