Pedestrian Detection in Outdoor Images using Color and Gradients

Marcel Häselich Michael Klostermann Dietrich Paulus
Active Vision Group, University of Koblenz-Landau, 56070 Koblenz, Germany
{*mhaeselich, michaelk, paulus*}*@uni-koblenz.de*

***Abstract*— Pedestrian Detection in digital images is a task of huge importance for the development of autonomous systems and for the improvement of robots interacting with their environment. The challenges such a system has to overcome are the high inter-class variance of pedestrians and the demands of unstructured environments. Outdoor environments contain unknown regions, inhomogeneous illumination, and parts of the pedestrians can be occluded.**

In this work, a complete system for pedestrian detection is realized according to state-of-the-art techniques. As main features, we use the "Histograms of Oriented Gradients" in combination with the "Color Self-Similarity" feature as proposed by Walk et al. We describe and evaluate our complete detection approach and our new structure element is able to accelerate the Color Self-Similarity computations by a factor of four.

***Index Terms*— Human Detection, Computer Vision**

I. Introduction

The increasing amount of sensor information necessitates an automatic pedestrian detection to enhance or automate important tasks in everyday life. Autonomous robots need to perceive humans in their environment in order to avoid collisions or to interact with them. A key challenge is the high variability of pedestrians. Persons differ in their form, pose, clothing, and color and hence possess a high inter-class variance. Other difficulties arise from the outdoor environment. The area around the sensor is unknown and inhomogeneous and parts of the pedestrians can be occluded. Unknown ambient light conditions complicate the detection and may cause erroneous or falsified sensor data.

In our approach we use the "Histograms of Oriented Gradients" (HOG) [1] to encode the shape and appearance of a pedestrian. Walk et al. [2] present the "Color Self-Similarity" (CSS), a feature that describes color relations within a detection window and forms a well-suited addition to the gradient histograms. Our goal is to realize a fast, robust, and portable camera-based pedestrian detection for our autonomous outdoor robot. For this task we extend the approach of Walk et al. and integrate it into our robotic framework. We identify the computation of the color similarities as cost-intensive part and present a new structure element that accelerates the computations.

II. Related work

In the past years, a large number of approaches emerged and a significant progress could be observed. A recent publication of Dollar et al. [3] especially addresses this situation and presents an elaborate evaluation of sixteen state-of-the-art detectors. The evaluation yields that despite the steady progress over the last years, there is is still room for improvement in case of low resolution images or for partially occluded pedestrians.

Regarding the components of current detectors, the trainable system of Papageorgiou and Poggio [4] can be seen as pioneering work for many approaches. Their approach computes a feature vector from a detection window that is classified afterwards by a Support Vector Machine (SVM) [5] with polynomial kernel. The authors present a large number of features and select relevant ones manually to detect faces, persons and cars.

Viola et al. [6] present an approach that is based on their previous work on facial recognition [7]. Their approach uses AdaBoost [8] combined with decision trees for the learning. The classification is distributed on a cascade of true/false decisions and is further accelerated by computation principles for rectangular sums presented by Crow et al. [9].

Dalal and Triggs [1] present the HOG feature which is able to encode the shape of a pedestrian in a robust way. It is computed from normalized three-dimensional histograms quantized in position and gradient orientation. Dalal and Triggs use a linear SVM for classification. Prisacariu and Reid [10] demonstrate the real-time capability of the HOG feature without loosing quality using GPGPU techniques. In a subsequent approach, Dalal and Triggs [11] introduce the feature "Histograms of Oriented Flow" (HOF). This feature adopts the principle of HOG to the optical flow [12] to increase the classification quality.

The detection quality of HOG-based approaches can be increased through combination with complementary features and other learning approaches. Wojek et al. [13] focus on moving images and examine the quality of HOG, HOF and Haar-Wavelet features in combination with different learning algorithms. Besides the combination of linear SVM and HOG introduced by Dalal and Triggs, the histogram intersection kernel SVM presented by Maji et al. [14] is examined by Wojek et al. In addition, AdaBoost and the extension MLPBoost [15] are used. The comparison yields that movement information is helpful for detecting pedestrians moving sideways.

Walk et al. [2] present a pedestrian detection system that uses a histogram intersection kernel SVM in combination with HOG, HOF, and a new feature called CSS. The CSS feature encodes the color similarity of all cells within a detection window and is supposed to describe color relations from objects and background.

Zhu et al. [16] integrateAdaBoost with HOG features to

realize a human detection system. The authors use the work of Porikli [17], who showed that the fast computation of rectangular sums on integral images is portable to histograms. An integral image is created for each histogram bin and the resulting integral histogram can be used to compute arbitrary rectangular histograms. The resulting detector of Zhu et al. achieves an acceleration of factor 70, but is unable to uphold the quality of the original HOG variant.

Dollar et al. [18] generalize this idea and present the "Integral Channel Features". In their approach, the authors investigate various channels and use the boost approach to determine relevant features. For the detection of pedestrians, 8 channels consisting of gradient magnitude, gray-value image and gradient histogram (6 channels) are used.

Based on the Integral Channel Features Dollar et al. [19] publish a speed-optimized extension. Since the gradients and gradient histograms are not scale-invariant, the authors approximate the necessary pyramid of features from different scales by using neighborhood information. Although detection quality is affected by this approximation, pedestrians can be detected in multiple images per second.

Felzenszwalb et al. [20] propose a part-based approach for pedestrian detection. Their approach uses the HOG feature as well and defines a two-dimensional star pattern to model the part-whole relation of the pedestrians. Felzenszwalb et al. choose a latent SVM which is able to encode the relative position of the parts as hidden variables.

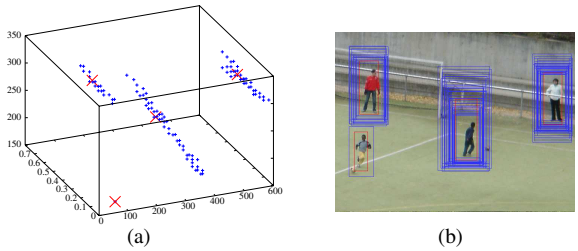
III. Approach

Despite the steady progress over the past years there is still room for improvements. In particular, detection is disappointing at low resolutions and for partially occluded pedestrians. Further, the performance in terms of run-time an memory consumption is not yet optimal. A detector, a classifier and the HOG and CSS features form the main components of our approach. Detections on an input image are represented by enclosing rectangles. Our trained classifier decides for each rectangle if a pedestrians is contained or not and how confident he is with his decision. Classification is not directly performed on the images but on their features. The following section describes our approach and is subdivided into parts for the detector, the classifier, and the features.

A. Detector

Our search for pedestrians in an input image uses a sliding window framework with a fixed window size. Each window needs a binary classification if a pedestrian is sufficiently contained. A fixed window size alone results in a pedestrian detection that is limited to pedestrians of a fix size and is not scale-invariant. Hence, the input image is transformed into multiple scales and the fixed-size window operates on each scale with the given offset. The input image is initially scaled by a factor $s_{\text{start}} \geq 1$ and subsequently shrunk by a factor $0 < s_{\text{shrink}} < 1$. The parameter s_{shrink} controls the resolution of the scale space and the number of scales is computed as

$$k_{\text{scales}} = \left\lfloor \frac{\log(\min(\frac{n_{\text{img}}}{n_{\text{win}}}, \frac{m_{\text{img}}}{m_{\text{win}}})) + \log(s_{\text{start}}))}{-\log(s_{\text{shrink}})} + 1 \right\rfloor \quad (1)$$



(a) (b)

Fig. 1. Non-maximum suppression using the mean shift algorithm. In the upper image detections are marked as blue plus-signs and estimated means are red crosses, corresponding to the bounding boxes in the lower image.

where $n_{\text{img}} \times m_{\text{img}}$ and $n_{\text{win}} \times m_{\text{win}}$ are the image and window size, respectively.

The classification of all windows on all scales yields a number of detections that accumulate around the pedestrians (cf. blue rectangles in Fig. 1 (b)). Those multiple detections for a single target need to be aligned. There exist different approaches to realize such a thinning or non-maximum suppression. Dollar et al. [18] for example perform a pairwise comparison of overlapping windows and discards the ones with the lowest decision value. For our approach we follow the method of Dalal et al. [11] and use the mean shift algorithm [21]with uncertainty-vector $\boldsymbol{\rho}$ given as

$$\boldsymbol{\rho} = (\rho_x, \rho_y, \rho_z) = (4, 8, \log(1.6))^{\mathrm{T}} \quad . \quad (2)$$

The uncertainty σ_i for the i-th detection are computed w.r.t. to the scaling factor s_i as

$$\sigma_i = \frac{1}{s_i}(\rho_x, \rho_y, \rho_z)^{\mathrm{T}} \quad (3)$$

Fig. 1 (a) shows an example in which detections are depicted as blue plus-signs and estimated means as red crosses. The corresponding bounding boxes are visualized in Fig. 1 (b) where the red boxes are the ones selected by the non-maximum suppression.

B. Classifier

The task of the classifier is to perform a binary classification between pedestrian and non-pedestrian as precise as possible. Therefore, we train an SVM for the detection windows. We use a linear Soft-Margin SVM [22], [23] with a feature vector that will be described in Sec. III-D. Given the two classes ω_1 and ω_2 with corresponding feature vectors $\boldsymbol{x}_i$ with $i = 1, ..., l$, their affiliation is encoded by y_i defined as

$$y_i = \begin{cases} -1, & \text{if } \boldsymbol{x}_i \text{ belongs to } \omega_1 \\ +1, & \text{if } \boldsymbol{x}_i \text{ belongs to } \omega_2 \end{cases} \quad . \quad (4)$$

The linear Soft-Margin SVM approach computes a separating hyperplane that discriminates between the positive and negative feature vectors. The cost-function that needs to be minimized is given as

$$\min_{\boldsymbol{w}, b, \boldsymbol{\xi}} \quad \frac{1}{2}\boldsymbol{w}^{\mathrm{T}}\boldsymbol{w} + C\sum_{i=0}^{l}\xi_i \quad (5)$$

$$\text{subject to} \quad y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0 \quad (6)$$

where $\boldsymbol{w}$ is the normal of the hyperplane, b encodes the distance of the hyperplane to the origin of the coordinate system, and C is a weighting factor. The vector $\boldsymbol{\xi} = (\xi_0, ..., \xi_i, ..., \xi_l)$ describes the influence of correct and false classifications on the hyperplane. The solution yields the hyperplane defined by the normal vector $\boldsymbol{w}$ and the variable b. A feature vector $\boldsymbol{x}$ can now be classified according to

$$\text{sign}\left(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b\right) \quad . \tag{7}$$

For this work, we used LIBLINEAR from Fan et al. [24] which is optimized for linear SVMs with a plenty of data.

C. Training

For the training of the SVM we worked with 615 positive and 1218 negative samples. Positive samples are extracted from annotations and negative samples are generated at random at all scales from images without pedestrians. In order to reduce the number of false positives, we perform an extended bootstrapping in which we retrain our SVM with difficult samples. The model created in the first iteration is therefore applied on images without pedestrians on the whole scale space. Since all detections are false positives, we add them as such to our model. This step is repeatable and lowers the number of false negatives, but also influences the number of true positives. True positives that were close to the hyperplane before an iteration might not be detected again by the classifier afterwards. We observed that two bootstrapping phases adequately reduce the false negatives while preserving the number of true positives.

D. Features

Features encode occurring patterns as multi-dimensional feature vectors. For the previously described detection windows, we compute the HOG and CSS features as follows.

Histograms of Oriented Gradients

The HOG feature introduced by Dalal and Triggs [1] is able to bundle the gradient information and still allows variations in the pose. In the first step, the gradient magnitude and orientation are calculated. Gradients are calculated from a convolution of the image with the filter cores $[-1\ 0\ 1]$ in x- and $[-1\ 0\ 1]^{\mathrm{T}}$ in y-direction. The gradient magnitude is computed as

$$g_{\text{mag}}(x,y) = \sqrt{\left(\frac{\mathrm{d}}{\mathrm{dx}} f(x,y)\right)^2 + \left(\frac{\mathrm{d}}{\mathrm{dy}} f(x,y)\right)^2} \tag{8}$$

and the gradient orientation is defined as

$$g_{\text{ori}}(x,y) = \tan^{-1}\left(\frac{\frac{\mathrm{d}}{\mathrm{dy}} f(x,y)}{\frac{\mathrm{d}}{\mathrm{dx}} f(x,y)}\right) \quad . \tag{9}$$

In the next step the image is segmented into equidistant HOG-blocks. For each HOG-block, the gradient magnitudes are weighted with a two-dimensional Gaussian f_g

$$\mathrm{f}_g(x,y) = \exp\left(-\left(\frac{(x-x_0)^2}{2\sigma^2} + \frac{(y-y_0)^2}{2\sigma^2}\right)\right) \tag{10}$$

with (x_0, y_0) as block center and σ as half the block width.

Afterwards, a three-dimensional histogram is computed for each HOG-block with a resolution of $\xi{\times}\xi{\times}\beta$, where β denotes the resolution of the gradient orientation. Each pixel of each block is registered in the histogram according to its relative position in the block and its gradient orientation. Entries in the histogram are made using trilinear interpolation. The resulting vector $\boldsymbol{h}$ is normalized using the $\mathcal{L}_2-\text{Hys}$ normalization. Our first feature vector is formed by concatenation of all normalized histograms of all HOG-blocks which are contained in the detection window.

Color Self-Similarity

Our second feature is the CSS-feature introduced by Walk et al. [2]. Complementary to the shape description of the HOG-feature, CSS encodes the color information. Dollar et al. [18] have shown that integral channel features are able to record a peak in the Hue channel in the area of the head, but not for the rest of the body. CSS circumvents this limitation by considering color similarities instead of accessing the color values directly.

For the computation of the CSS-feature, the image is initially converted to the HSV space. The detection window is subdivided into quadratic cells of size $\zeta \times \zeta$ and for each cell a three-dimensional color histogram of $3 \times 3 \times 3$ is constructed. Similar to the HOG-feature computation, the histogram is filled with the pixels contained in the cell and a trilinear interpolation is performed. Afterwards, for each cell the color similarity to any other cell of the window is computed. The measure that is used to calculate the similarity is the histogram intersection d_{HI}, which is calculated for two histograms H and V as

$$d_{\text{HI}}(H,V) = \sum_k \min(H(k), V(k)) \ , \tag{11}$$

where k iterates over all classes of the histogram. This similarity can be normalized by the value $d_{max} = \zeta^2$, which is given by the number of pixels contained in one cell, yielding the similarity measure

$$d_{\text{CSS}}(H,V) = \frac{d_{HI}(H,V)}{\zeta^2} \tag{12}$$

with the range of values $[0, 1]$. In order to map all similarities from each cell to all other cells, $n(n-1)/2$ values are required, where n is the number of cells of a detection window. Those $n(n-1)/2$ form the feature vector of the CSS-feature. The feature is able to describe the similarity of body parts and the similarity of background regions, which are interrupted by the appearance of pedestrians.

For an example of 128 cells, $\zeta = 8$, and a detection window of size 64×128, there are 8128 similarities computations required. The computation of the similarities for one window is illustrated in Fig. 2 (a). In the image, a circle represents the computation of a similarity with the cell marked with a cross. The complexity for each detection window w.r.t. the number of cells n is given by $\mathrm{O}\left(n(n-1)/2\right)$. Upon initial inspection, the similarity calculations can only be reduced if ζ is increased in order to decrease n, which could result in a quality loss. The inspection of the sliding window reveals

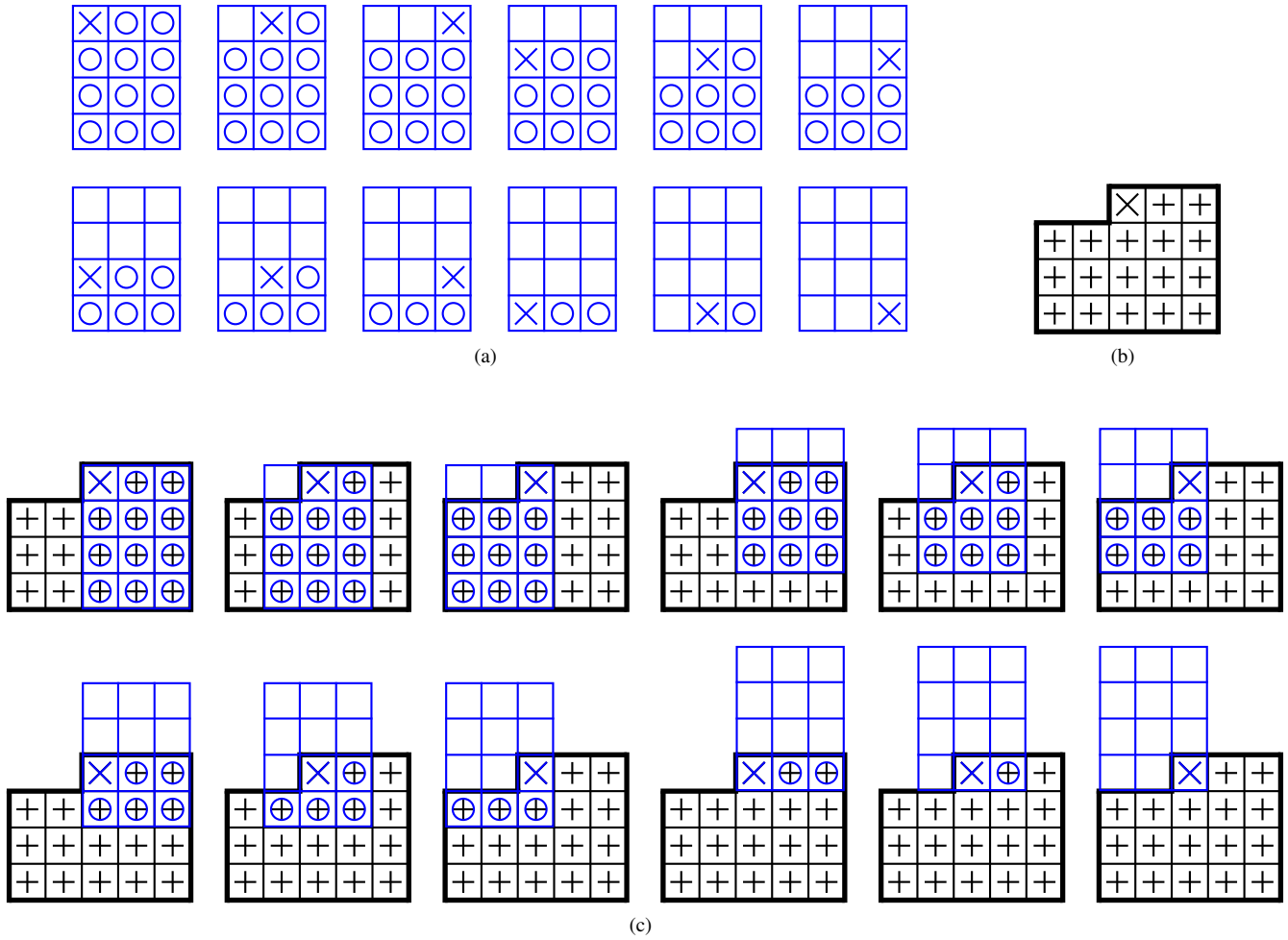


Fig. 2. Application and benefit of the new structure element. Similarity computations for the CSS are shown in (a) and the structure element that is used to accelerate the computations is visualized in (b). The current cell is marked with an cross, elements contained in the structure element are depicted as plus-signs, and cells that are used for computation are marked with a circle. Extraction of the similarity computations from the new structure element is illustrated in in (c).

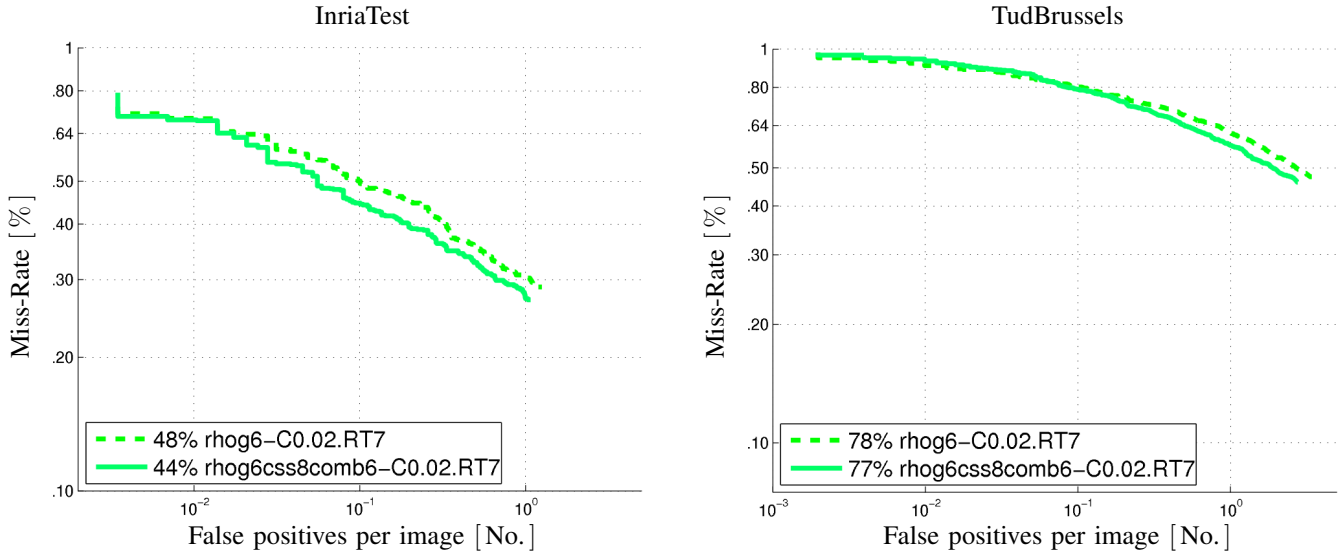


Fig. 3. Comparison of the HOG-based classification versus the HOG + CSS-based variant. The evaluation was performed on the INRIA (*InriaTest*) [1] and the TudBrussels (*TudBrussels*) [13] datasets.

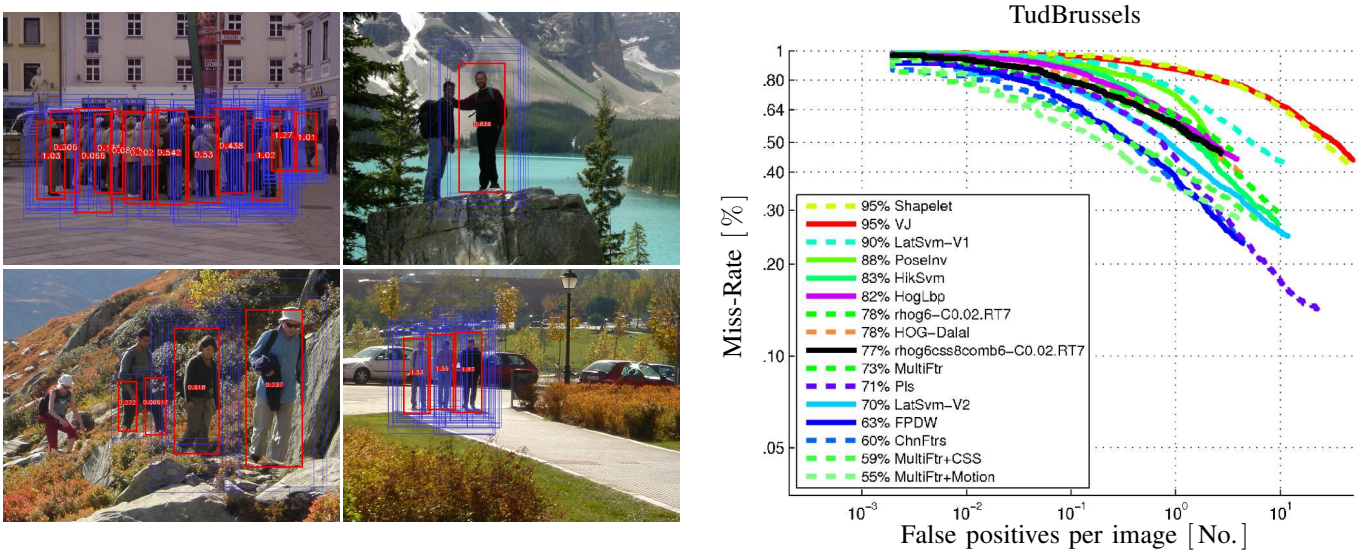


Fig. 4. A selection of some detection results and a comparison with other state-of-the-art approaches.

another possibility, which is to dissolve redundancies by sophisticated precomputations.

Our approach works as follows. In the first place, the whole image is divided into cells of size $\zeta \times \zeta$. Our goal is to compute all relevant similarities for all cells in advance. The window size determines the region called *support* that influences the feature of a single cell. For the example from Fig. 2 (a) with a detection window of 3×4 cells, the support is constructed as the structure element shown in Fig. 2 (b). The support $\mathrm{sp}_x \times \mathrm{sp}_y$ w.r.t. the number of cells of the detection window in x- and y-direction is defined as $\mathrm{sp}_x = 2c_x - 1$ and $sp_y = c_y$.

It is now possible to create a buffer for each cell of the image that stores the support information (cf. the example in Fig. 2 (b)). Fig. 2 (c) visualizes how all the similarities displayed in Fig. 2 (a) can be extracted from this structure element. The application of the structure element is possible for all cell windows w.r.t. boundary treatment. Thus, the computation of the histogram intersections no longer depends on the number of windows w_{img} but on the number of cells c_{img}: $\mathrm{O}\left(\left(\mathrm{sp}_x \mathrm{sp}_y - c_x\right) c_{\mathrm{img}}\right) = \mathrm{O}\left(2\, n\, c_{\mathrm{img}}\right)$ It needs to be taken into account that not for all the windows all histogram intersection are calculated, but the buffer is read-out each time, which in turn is square in n. The evaluation described in the next chapter will reveal an acceleration of factor four without any loss in quality. The memory complexity is given as $\mathrm{O}\left(\left(sp_x sp_y - c_x\right) c_{img}\right)$, since for each cell, all similarities to all other cells of the structure element are stored.

IV. Evaluation

The evaluation of the implemented pedestrian detection system follows the instructions of Dollar et al. [3] and uses the proposed scripts to achieve a precise comparison with existing detectors. Our ground truth consists of the images from the *InriaTest* [1] and the *TudBrussels* [13] datasets. InriaTest contains 288 images of varying size, illumination, and scenery and is a loose collection of holiday photographs. The 508 images of TudBrussels were gathered from a camera mounted on a car driving through the city of Brussels. Both image dataset are completely disjoint from the training data. In order to determine if the rectangle surrounding a detected pedestrian corresponds to the rectangle of a pedestrian annotated in the ground truth, the PASCAL-criteria [25] is used. The criteria states that two rectangles sufficiently overlap if they share at least 50 % of their area, resulting in a true positive (TP) matching between detection and ground truth. Bounding boxes of detected pedestrians that fail this criteria are classified as false positives (FP), vice versa as false negatives (FN) for bounding boxes of the ground truth.

In case of multiple detected bounding boxes corresponding to a single ground truth annotation, only the one with highest decision value will be used while the others are classified as FP. This in turn means that our approach has to conduct a very effective non-maximum suppression to reduce the impact of these false positives on the classification quality.

The quality of our detector is described by an evaluation curve whose linear x-axis describes the false positives per image and whose logarithmic y-axis encodes the miss-rate. False positives per image are computed as (FP) / (#images) and the miss-rate is defined as (FN) / (TP+FN). Fig. 3 shows the results for HOG versus HOG + CSS classifiers on both datasets. The key in the lower part shows the miss-rate at 10^{-1} false positives per image as reference point.

For our approach, a cell size of 6 of the HOG feature yielded the best results. Linear SVMs have a single parameter C which has a strong influence on the classification quality. In this work, a C-value of 0.02 yielded the best results. The cell size of the CSS feature is another parameter of our classifier. For the experiments we used 8×8 pixels per CSS cell. In this work, we observed that 2 retraining phases (RT) mainly lead to convergence for the HOG feature alone whereas the combination of features requires 7 retrainings.

TABLE I

COMPARISON OF THE DIFFERENT DETECTOR RUN-TIMES REFLECTING THE KEY FEATURE OF OUR APPROACH.

Detector	Resolution								
	320×240			640×480			1280×960		
	μ	σ	fps	μ	σ	fps	μ	σ	fps
HOG-Dalal	n. a.	n. a.	n. a.	4.1841s	n. a.	0.239	18.519s	n. a.	0.054
rhog8	0.176 s	0.007 s	5.691	0.939 s	0.023 s	1.065	4.166 s	0.094 s	0.240
rhog6	0.233 s	0.008 s	4.286	1.389 s	0.024 s	0.720	6.607 s	0.096 s	0.151
rhog6css8combo	1.500 s	0.008 s	0.667	13.886 s	0.027 s	0.072	78.986 s	0.106 s	0.013
rhog6css8-struct-combo	0.488 s	0.009 s	2.051	3.612 s	0.028 s	0.277	19.093 s	0.109 s	0.052

Each combination of parameters can be seen as a new detector which is why the name of our classifier is formed from these parameters as rhog6(HOG cell size)+css8(CSS cell size)-C0.02(SVM C-value)+RT7(# retrainings). Our classifier is compared to the state-of-the-art in Fig. 4.

The run-time of the system is evaluated one the TudBrussels dataset (508 images, resolution 640×480) and the result is shown in Table I. Since the run-time mainly depends on the size of the images, we used the interpolated sizes of 320×240 and 1280×960 as additional material. The run-times include loading the data, the detection itself, and the non-maximum suppression. A detailed description of HOG-Dollar is available in [3]. The 2 variants rhog6 and rhog8 represent the the feature with 6×6 and 8×8 pixels per cell. HOG features combined with the CSS features are represented by rhog6css8comb6 and rhog6css8-struct-combo, where the latter is our variant with the new structure element described in Sec. III-D. The values for HOG-Dollar are taken from the original paper and the system used for all other measurements is an Intel(R) Quad Core(TM) with 2.66 GHz and 8 GB RAM.

V. CONCLUSION

We presented our pedestrian detection approach for pedestrian detection in monocular images. In our work, we used HOG features in combination with CSS features. The combination of these features allows the integration of robust color information into the established Histograms of Oriented Gradients approach. Our evaluation reveals that the combination of the HOG and the CSS feature results in an improved detection quality. For the computation of the CSS feature, we presented a new structure element that is able to accelerate the computations by factor 4 without any influence on the feature quality.

ACKNOWLEDGMENT

This work was partially funded under research contract PA 599/11-1 by the Deutsche Forschungsgemeinschaft (DFG).

REFERENCES

[1] N. Dalal and B. Triggs, "Histograms of Oriented Gradients for Human Detection," in *Proc. of the CVPR*, 2005, pp. 886–893.

[2] S. Walk, N. Majer, K. Schindler, and B. Schiele, "New Features and Insights for Pedestrian Detection," in *Proc. of the CVPR*, 2010, pp. 1030–1037.

[3] P. Dollar, C. Wojek, B. Schiele, and P. Perona, "Pedestrian Detection: An Evaluation of the State of the Art," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 34, no. 4, pp. 743–761, 2012.

[4] C. Papageorgiou and T. Poggio, "A Trainable System for Object Detection," *Int. Journal of Computer Vision*, vol. 38, no. 1, pp. 15–33, 2000.

[5] V. Vapnick and C. Cortes, "Support-Vector Networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[6] P. A. Viola, M. J. Jones, and D. Snow, "Detecting Pedestrians Using Patterns of Motion and Appearance," in *Proc. of the ICCV*, 2003, pp. 734–741.

[7] P. A. Viola and M. J. Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features," in *Proc. of the CVPR*, 2011, pp. 511–518.

[8] Y. Freund and R. E. Schapire, "A Decision-Theoretic Generalization of Online Learning and an Application to Boosting," in *Proc. of the COLT*, 1995, pp. 23–37.

[9] F. C. Crow, "Summed-Area Tables for Texture Mapping," in *Proc. of the SIGGRAPH*, 1984, pp. 207–212.

[10] V. Prisacariu and I. Reid, "fastHOG - a real-time GPU implementation of HOG," Department of Engineering Science, Oxford University, Oxford, UK, Tech. Rep., 2009.

[11] N. Dalal, B. Triggs, and C. Schmid, "Human Detection Using Oriented Histograms of Flow and Appearance," in *Proc. of the ECCV*, 2006, pp. 428–441.

[12] B. K. P. Horn and B. G. Schunck, "Determining Optical Flow," *Artificial Intelligence*, vol. 17, no. 1-2, pp. 185–203, 1981.

[13] C. Wojek, S. Walk, and B. Schiele, "Multi-Cue Onboard Pedestrian Detection," in *Proc. of the CVPR*, 2009, pp. 794–801.

[14] S. Maji, A. C. Berg, and J. Malik, "Classification Using Intersection Kernel Support Vector Machines is Efficient," in *Proc. of the CVPR*, 2008, pp. 2245–2260.

[15] G. D. C. Cavalcanti, J. P. Magalhaes, R. M. Barreto, and T. I. Ren, "MLPBoost: A Combined AdaBoost / Multi-Layer Perceptron Network Approach for Face Detection," in *Proc. of the SMC*, 2012, pp. 2350–2353.

[16] Q. Zhu, M.-C. Yeh, K.-T. Cheng, and S. Avidan, "Fast Human Detection Using a Cascade of Histograms of Oriented Gradients," in *Proc. of the CVPR*, 2006, pp. 1491–1498.

[17] F. Porikli, "Integral Histogram: A Fast Way To Extract Histograms in Cartesian Spaces," in *Proc. of the CVPR*, 2005, pp. 829–836.

[18] P. Dollar, Z. Tu, P. Perona, and S. Belongie, "Integral Channel Features," in *Proc. of the BMVC*, 2009, pp. 91.1–91.11.

[19] P. Dollar, S. Belongie, and P. Perona, "The Fastest Pedestrian Detector in the West," in *Proc. of the BMVC*, 2010, pp. 68.1–68.11.

[20] P. F. Felzenszwalb, R. B. Girshick, D. A. McAllester, and D. Ramanan, "Object Detection with Discriminatively Trained Part Based Models," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1627–1645, 2010.

[21] D. Comaniciu and P. Meer, "Mean Shift: A Robust Approach Toward Feature Space Analysis," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 603–619, 2002.

[22] C. Cortes and V. Vapnick, "Support-Vector Networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[23] S. Theodoridis and K. Koutroumbas, *Pattern Recognition*. Academic Press, 2009.

[24] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C. J. Lin, "LIBLINEAR: A Library for Large Linear Classification," *Journal of Machine Learning Research*, vol. 9, no. 1, pp. 1871–1874, 2008.

[25] M. Everingham, L. J. Van Gool, C. K. I. Williams, J. M. Winn, and A. Zisserman, "The Pascal Visual Object Classes (VOC) Challenge," *Int. Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, 2010.