Volume xx (200y), Number z, pp. 1–6

A robust, real-time pedestrian detector for video surveillance

Domonkos Varga[1,2], Tamás Szirányi[1,3]

[1] Computer and Automation Research Institute of the Hungarian Academy of Sciences
[2] Department of Networked Systems and Services, Budapest University of Technology
[3] Department of Material Handling and Logistics Systems, Budapest University of Technology

Abstract

Detecting different categories of objects in an image and video content is one of the fundamental tasks in computer vision research. Pedestrian detection is a hot research topic, with several applications including robotics, surveillance, and automotive safety. It is a challenging problem due to the variance of illumination, color, scale, pose, and so forth. Increasing interest in robust pedestrian detection algorithms is also coming from the visual surveillance community. The goal of this paper is to present our novel pedestrian detector in surveillance videos. A robust, novel feature extraction method is defined based on Local Binary Patterns and gradients. Occlusion handling is one of the most important problem in pedestrian detection. We propose an effective occlusion handling process, which consists of extensive part detectors. Our experiments also demonstrate that the pedestrian detector can provide robust input for a video surveillance system and it is able to work on different modalities.

1. Introduction

Pedestrian detection has been one of the most extensively studied problems in computer vision. One reason is that pedestrian detection is the first step for a number of applications such as smart video surveillance, people-finding for military applications, human-robot interaction, intelligent digital management, and driving assistance system. Pedestrian detection is a rapidly evolving area, as it provides the fundamental information for semantic understanding of the video footages. Because of the various style of clothing in appearance, different possible body articulations, different illumination conditions, the presence of occluding accessories, frequent occlusion between pedestrians, etc., the pedestrian detection is still a challenging problem in computer vision.

The aim of this paper is to present our novel pedestrian detector in surveillance videos. In video surveillance, the cameras are static and usually look down to the ground.

The rest of this paper is organized as follows. In Section 2, the related and previous works are reviewed. We describe the proposed pedestrian detector in Section 3. Section 4 shows experimental results and analysis. We draw the conclusions in Section 5.

2. Related Work

There is extensive literature on pedestrian detection algorithms. An extensive review on these algorithms is beyond the scope of this paper. We refer readers to comprehensive surveys[1,2] for more details about existing detectors. In this section, we review only the works related to our method.

Broadly speaking there are three major types of approaches for visual pedestrian detection: model-based, part-based, and feature-classifier-based.

In model-based pedestrian detection, an exact pedestrian model is defined. Then we search the image for matched positions with the pre-defined model to detect pedestrians. Model-based pedestrian detection corresponds to the generative models in pattern recognition. Most of the matching process is under the framework of the Bayesian theory to estimate the maximum posterior probability of the object class.

In consideration of the distinct pedestrian contours, the shape models are the most commonly used in pedestrian detection. The shape models can be discrete or continuous. Discrete shape models mean a set of contour examplars which are usually used for edge image matching. Gavrila[3] presented a probabilistic approach to hierarchical, exemplar-based shape matching. A template tree was constructed in order to represent and match the variety of shape exemplars.

(3/2016).

This tree was generated offline by a bottom-up clustering approach using stochastic optimization. Applying coarse-to-fine probabilistic matching strategy, Chamfer distance was used as the similarity measurement between two contours.

In feature-classifier-based pedestrian detection the detection windows are extracted (usually sliding-windows search) from video frames first. Next features are extracted from the detection window. A classifier is trained based on a large number of training samples. The classifier classifies the feature vectors as pedestrian class or non-pedestrian class. The feature-classifier-based algorithms differ from each other in two ways. They use different features or different classification algorithms.

Papageorgiou and Poggio[4] introduced a dense overcomplete representation using Haar wavelets. The images were mapped from the space of pixels to an overcomplete dictionary of Haar wavelet features. They used three different types of 2-dimensional non-standard Haar wavelets with an overlap of 75%: vertical, horizontal, and diagonal. Histograms of oriented gradients (HOG) have been proposed by Dalal and Triggs[5]. First, each detection window is decomposed into cells of size 8×8 pixels and each group of 2×2 cells are integrated into a block with an overlap of 50%. A 9-bin histogram of oriented gradients is computed for each cell. Each block is represented by the concatenated histograms of all its cells. This concatenated histogram is normalized to an L2 unit length. Each 128×64 detection window is represented by 15×7 blocks, giving a 3780 dimensional feature vector per detection window. These feature vectors are then used to train a linear SVM classifier. Local Binary Pattern (LBP) is a simple, but very efficient texture operator which labels the pixels of an image by thresholding the neighborhood of each pixel and considers the result as a binary number[6]. Later extensions of LBP operator use neighborhoods of different sizes. The notation (P,R) is used for the neighborhood description, where P is the number of sampling points on a circle of radius R. Formally, we can write:

$$LBP_{P,R}(x,y) = \sum_{i=0}^{P-1} s(u_i - u_c) \cdot 2^i, s(x) = \begin{cases} 1 & x \geq 0 \\ 0 & \text{otherwise,} \end{cases} \quad (1)$$

where u_c corresponds to the graylevel of the center pixel and u_i to the graylevels of P equally spaced pixels on a circle of radius R. A histogram of the labelled image $f_l(x,y)$ can be computed and it can serve as an input for different machine learning algorithms[7].

Fusion of features can improve the detection performance, but it is not wise to combine every feature blindly. The most popular approach for improving detection quality is to combine the features computed over the input image. Wang et al. combined HOG and LBP[8]. They used two kinds of detectors, i.e., global detector for whole scanning windows and part detectors for local regions, are learned from training data using linear SVM. For each ambiguous scanning window, an occlusion likelihood map was constructed by using the response of each block of the HOG feature to the global detector. The occlusion likelihood map is then segmented by Meanshift. The segmented portion of the window with a majority of negative response is inferred as an occlusion region. Dollár et al. presented Integral Channel Features (ICF) for pedestrian detection task[9]. The general idea behind ICF is that multiple registered image channels are computed using linear and non-linear transformations of the input image, and then features such as local sums, histograms, and Haar features and their various generalizations are efficiently computed using integral images.

3. Our system architecture

Figure 1 presents our system overview, the input video frames are segmented in order to determine the foreground. Using the result of the foreground segmentation, we rapidly filter out negative regions, while keeping the positive regions. The detection system scans the image all relevant positions and scales to detect a pedestrian. The so-called feature pyramid is derived from the standard image pyramid in order to accelerate the feature extraction. The detection window scans the feature pyramid and extracts the feature vector with the help of it. The feature component encodes the visual appearance of the pedestrian, while the classifier component determines for each sliding-window independently whether it contains a pedestrian or not.

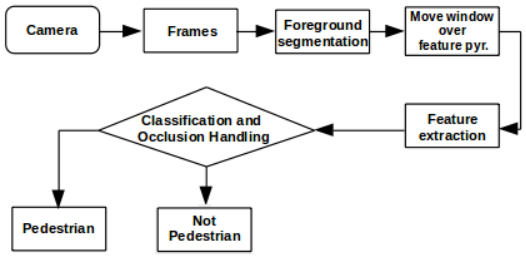

Figure 1: *Architecture of our pedestrian detection system.*

To train our system, we gathered a set of 13,500 greyscale sample images of pedestrians as positive training examples, together with their left-right reflections. The positive examples have been aligned and scaled to the dimensions 128×64. The images of the pedestrians were taken from public pedestrian datasets[10] and from our surveillance and traffic videos. We made a database of negative samples too, which consists of 16,000 non-pedestrian images. In order to improve the performance we put 7,000 vertical structures like poles, trees or street signs to the negative samples. The vertical structures are common false positive detections in pedestrian detection.

3.1. Foreground segmentation

In order to reduce detection time and eliminate false detections from background, multi-scale wavelet transformation (WT) using frame difference was applied to segment foreground. A signal is broken into similar (low-pass) and discontinuous (high-pass) subsignals by the low-pass and high-pass filters of WT[11].

The HSV color space is related to human color perception and it separates chromaticity and luminosity. That is why it is selected to be used here. We define a foreground mask in the following way[12]:

$$P_f = \begin{cases} 1, & E_{\Delta V} \geq T_{\Delta V} \wedge E_{\Delta S} \geq T_{\Delta S} \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

where ΔV and ΔS are the difference between the two successive frames of the value and the saturation component, respectively; $E_{\Delta V}$, $E_{\Delta S}$ stand for multi-scale WT across ΔV, and ΔS, respectively; $T_{\Delta V}$, $T_{\Delta S}$ represent a threshold value of ΔV, and ΔS, respectively.

In order to remove ghost effects the WT-based edge detection is used to extract edges of current frame,

$$P_e = \begin{cases} 1 & E_V \geq T_V \\ 0 & \text{otherwise,} \end{cases} \quad (3)$$

where V is the value component of current frame, T_V stands for a threshold value for E_V.

A bitwise AND operation is applied on P_f and P_e to extract the whole foreground region mask:

$$P = P_f \bullet P_e. \quad (4)$$

3.2. Multi-scale Center-symmetric Local Binary Pattern Operator

The original LBP operator labels the pixel of an image by thresholding the 3-by-3 neighborhood of each pixel with central pixel value and the result is taken as a binary number. Later extensions of the LBP operator use neighborhoods of different sizes. The notation (P,R) is used for the neighborhood description, where P is the number of sampling points on a circle of radius R.

The Center-symmetric Local Binary Pattern (CS-LBP) was introduced by Heikkilä et al[13]. In CS-LBP, pixel values are not compared to the center pixel but to the opposing pixel symmetrically with respect to the center pixel. We can see that for 8 neighbors, original LBP produces $2^8 = 256$ different binary patterns, whereas for CS-LBP this number is only $2^4 = 16$. The idea of Multi-scale Center-symmetric Local Binary Pattern is based on the simple principle of varying the radius R of the CS-LBP operator and combining the resulting histograms[14]. The neighborhood is described with two parameters $P,R = \{R_1, R_2, ..., R_{n_R}\}$, where n_R is the number of radii utilized in the process of computation. Each pixel in Multi-scale CS-LBP image is described with n_R values. The Multi-scale CS-LBP histogram for different values of $R = \{R_1, R_2, ..., R_{n_R}\}$ can be determined by summing $\mathbf{h}^{(1)}, \mathbf{h}^{(2)}, ..., \mathbf{h}^{(n_R)}$ vectors:

$$\mathbf{h} = \sum_{i=1}^{n_R} \mathbf{h}^{(i)}. \quad (5)$$

In our experiments, we used the following parameters: $P = 8$, $R_1 = 1$, $R_2 = 2$, $R_3 = 3$, and $n_R = 3$.

3.3. Feature extraction

In this subsection, we introduce the implementation details of our feature extraction method. The key steps are as follows.

1. We normalize the gray-level of the input image to reduce the illumination variance in different images. After the gray-level normalization, all input images have gray-level ranging from 0 to 1.
2. We obtain 11 layers of the input image in the following way: first, we compute the gradient magnitude of each pixel of the input gray-scale image (detection window), then we repeat this computation ten times on the previous derivative image. Considering the speed of the computation, we compute an approximation of the gradient using Sobel operator.
3. The detection window and each of the 11 layers of the detection window are split into equally sized overlapping blocks. The rate of overlapping is 50%. In our case, the size of the detection window is 128×64 and the size of the blocks is 16×16.
4. We take the detection window and the multi-scale CS-LBP histograms ($P = 8$, $R_1 = 1$, $R_2 = 2$, $R_3 = 3$, $n_R = 3$) are extracted from each block independently. Let $\mathbf{v}_i$ be the unnormalized descriptor of the ith block, $\mathbf{f}$ be the descriptor of the detection window. We obtain $\mathbf{f}$ in the following way:
   - $\mathbf{f} = [\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_N]$;
   - $\mathbf{f} \leftarrow \mathbf{f} / \sqrt{|| \mathbf{f} ||_1 + \varepsilon}$.
5. We take each layers one after the other and the multi-scale CS-LBP histograms are extracted from each block independently. Let $\mathbf{v}_{i,j}$ be the unnormalized descriptor of the ith block in the jth layer, $\mathbf{g}_j$ be the descriptor of the jth layer. We obtain $\mathbf{g}_j$ in the following way:
   - $\mathbf{g}_j = [\mathbf{v}_{1,j}, \mathbf{v}_{2,j}, ..., \mathbf{v}_{N,j}]$;
   - $\mathbf{g}_j \leftarrow \mathbf{g}_j / \sqrt{|| \mathbf{g}_j ||_1 + \varepsilon}$.
6. We obtain the feature vector of the detection window in the following way:

$$\mathbf{F} = \mathbf{f} + \sum_{j=1}^{11} \frac{1}{j+1} \mathbf{g}_j. \quad (6)$$

The overall length of the feature vector for a 128×64 detection window is $15 \times 7 \times 16 = 1680$ because each window is represented by 15×7 blocks. Experiments on different pedestrian datasets show that the proposed feature with linear Support Vector Machine (SVM) performs well. It can be seen that $\mathbf{f}$ mainly captures the contours with some scale information, while $\mathbf{g}_{11}$ captures the detailed texture, the rest $\mathbf{g}_i$-s capture special edges or textures. That is why the weights of the layers in Eq. 6 have descending coefficients. We will report about the effect of the number of the layers in Section 4.

3.4. Feature representation

In many applications such as video surveillance, detection speed is as important as accuracy. A standard pipeline for performing multi-scale detection is to create a densely sampled image pyramid then the detection system scans all images of the pyramid to detect a pedestrian. In order to accelerate the scanning and feature extraction process, we define a feature pyramid using a standard image pyramid.

We obtain the eleven layers of an image of the standard pyramid as described in the previous subsection. The multi-scale CS-LBP operator ($P = 8$, $R_1 = 1$, $R_2 = 2$, $R_3 = 3$, $n_R = 3$) is applied to the image and its eleven layers. In this way, we correspond 12 values to each pixel of the image. An image of the standard pyramid can be substituted by an $(W - 2 \cdot R_3) \times (H - 2 \cdot R_3) \times 5$ array where W stands for the width of the image and H is the height of the image. Using the feature pyramid derived from a standard image pyramid, the time of the feature extraction and thereby the scanning process can be reduced.

3.5. Occlusion handling

The linear SVM finds the optimal hyperplane that divides the space between positive and negative samples. Let be $\mathbf{x} \in \mathbb{R}^n$ a new input then the decision function of the holistic classifier can be defined as:

$$H(\mathbf{x}) = \beta + \mathbf{w}^T \mathbf{x}, \tag{7}$$

where $\mathbf{w}$ stands for the weighting vector, and β represents the constant bias of the learned hyperplane.

In our occlusion handling method, we determine first whether the score of the holistic classifier is ambiguous. The response of a linear SVM classifier is ambiguous if it is close to 0. When the output is ambiguous, an occlusion inference process is applied (Fig. 2).

We consider pedestrian as a rigid object and define a human body grid of $2m \times m$, where $2m$ and m indicate the numbers of cells in horizontal and vertical direction, respectively. Each cell is a square and has equal size. We ensure each part to be a rectangle. The possible sizes of the parts can be defined as

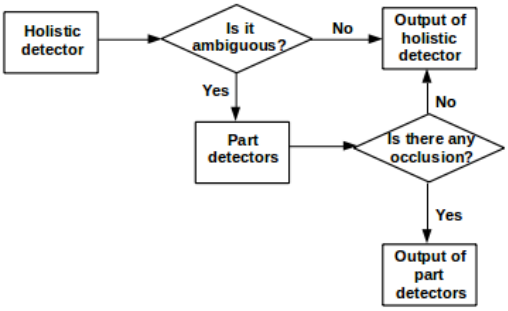

Figure 2: *Occlusion handling scheme.*

$$S = \{(w,h) \mid W_{min} \leq w \leq m, H_{min} \leq h \leq 2m, w,h \in \mathbb{N}^+\}, \tag{8}$$

where w and h stand for the width and height of a part in terms of the number of cells they contain. W_{min} and H_{min} are used to avoid subtle parts. Then, for each $(w,h) \in S$, we slide a $h \times w$ window over the human body grid to generate parts at different positions. The entire part pool can be defined as follows

$$P = \{(x,y,w,h,i) \mid x,y \in \mathbb{N}^+, (w,h) \in S, i \in I\}, \tag{9}$$

where x and y stand for the coordinates of the top-left cell in the part and i is a unique id. For instance, the part representing the full body is defined as $(1,1,m,2m,I_1)$.

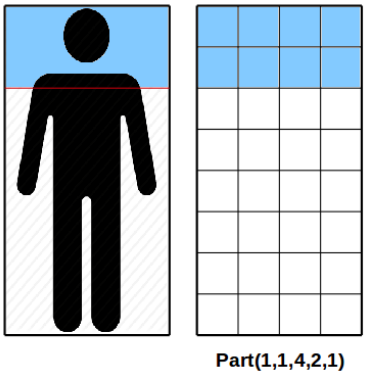

Figure 3: *Part prototype example, (x,y,w,h,i) is defined in Eq. 9. The head-shoulder part with 2 grids in height and 4 grids in width.*

In our implementation, we have used the following parameters $m = 4$, $W_{min} = 2$, $H_{min} = 2$, and the step size is one. For each part, a linear SVM was trained. If the output of the holistic detector is ambiguous, we run the part detectors. We take only into account the results of the part detectors with the five highest scores.

4. Experimental results

We perform the experiments on CAVIAR sequences, which is captured in a corridor with resolution 384×288 pixels. In this paper, we use per-image performance, plotting detection rate versus false positives per-image (FPPI). Figure 4 shows some sample detections on the CAVIAR sequences. Figure 5 shows the detection rate versus false positive per-image (FPPI) for the presented detector and six other systems. The six other systems we compare include Dalal and Triggs HOG+SVM system[5], Lie et al. HOG+AdaBoost system[15], Papageorgiou et al. Haar+SVM system[4], Monteiro et al. Haar+AdaBoost system[16], a PHOG+HIKSVM system[17], and a system based on Aggregated Channel Features[18]. Table 1 summarizes the speed comparison.

Figure 4: *Some detections on CAVIAR sequences.*

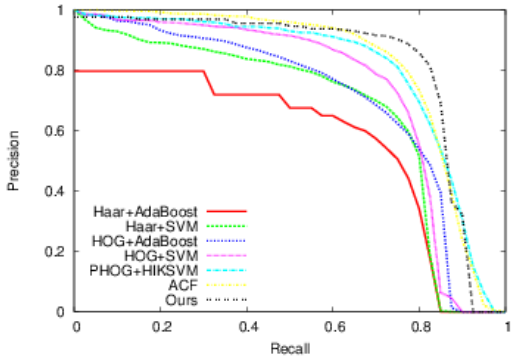

Figure 5: *Detection rate versus false positive per-image (FPPI) curves for pedestrian detectors.* 2×2 *is the step size and* 1.09 *is the scale factor of the sliding-window detection.*

Figure 6 demonstrates the effect of the number of layers. Over 11 layers we experience no significant performance improvement.

In order to prove the discriminative power of our feature, we applied our algorithm to the video frames of an infrared surveillance camera. The presented feature extraction method captures mainly gradient information and some

Table 1: *Speed comparison.*

Method	Speed
Haar+AdaBoost[16]	15.63 fps
Haar+SVM[4]	13.56 fps
HOG+AdaBoost[15]	9.48 fps
HOG+SVM[5]	4.27 fps
PHOG+HIKSVM[17]	6.19 fps
ACF[18]	14.03 fps
Ours	9.68 fps

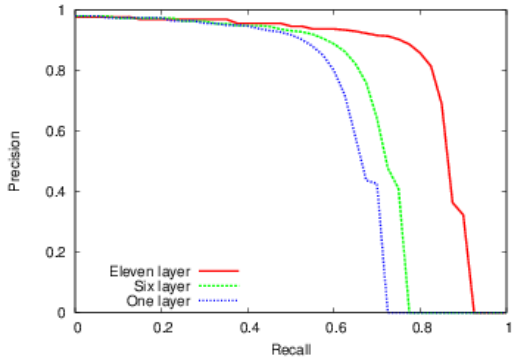

Figure 6: *Detection rate versus false positive per-image (FPPI) curves with respect to the number of the layers in the proposed detector.* 2×2 *is the step size and* 1.09 *is the scale factor of the sliding-window detection.*

texture and scale information. That is why we could build a detector that shows high invariance to illumination and clothing, and performs well in infrared images too. Figure 7 shows some sample detections in infrared images.

Figure 7: *Some detections on infrared images.*

5. Conclusions

In this paper, we proposed a novel pedestrian detection system and reported on experimental results. We have presented our novel feature extraction method based on multi-scale CS-LBP operator and gradients. We combined the pedestrian detection with foreground segmentation in order to filter out effectively the false detections. The performance of pedestrian detection was also improved by handling occlusion with an extensive part pool. Finally, the FPPI curves and sample detections was presented on CAVIAR sequences and on infrared images.

Acknowledgements

This work has been supported by the EU FP7 Programme (FP7-SEC-2011-1) No. 285320 (PROACTIVE project). The research was also partially supported by the Hungarian Scientific Research Fund (No. OTKA 106374).

References

1. R. Benenson, M. Omran, J. Hosang, and B. Schiele. Ten years of pedestrian detection, what have we learned?. *Computer Vision-ECCV 2014 Workshops*, 613–627, 2014.
2. P. Dollár, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: A benchmark. *IEEE Conference on Computer Vision and Pattern Recognition*, 304–311, 2009.
3. D.M. Gavrila. A bayesian, exemplar-based approach to hierarchical shape matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **29**(8):1408–1421, 2007.
4. C. Papageorgiou and T. Poggio. A trainable system for object detection. *International Journal of Computer Vision*, **38**(1):15–33, 2000.
5. N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *IEEE Conference on Computer Vision and Pattern Recognition*, 886–893, 2005.
6. M. Pietikäinen. Image analysis with local binary patterns. *Image Analysis*, 115–118, 2005.
7. X. Zhao, Z. He, S. Zhang, and D. Liang. Robust pedestrian detection in thermal infrared imagery using a shape distribution histogram feature and modified sparse representation classification. *Pattern Recognition*, **48**(6):1947–1960, 2015.
8. X. Wang, T. Han, S. Zhang, and S. Yan. An HOG-LBP human detector with partial occlusion handling. *IEEE International Conference on Computer Vision*, 32–39, 2009.
9. P. Dollár, Z. Tu, P. Perona, and S. Belongie. Integral Channel Features. *In proceedings of British Machine Vision Conference*, **2**(3):1–11, 2009.
10. C.G. Keller, M. Enzweiler, and D.M. Gavrila. A new benchmark for stereo-based pedestrian detection. *IEEE Intelligent Vehicles Symposium*, 691–696, 2011.
11. Y.-P. Guan. Spatio-temporal motion-based foreground segmentation and shadow suppression. *Computer Vision, IET*, **4**(1):50–60, 2010.
12. R. Xu, Y. Guan, and Y. Huang. Multiple human detection and tracking based on head detection for real-time video surveillance. *Multimedia Tools and Applications*, **74**(3):729–742, 2015.
13. M. Heikkilä, M. Pietikäinen, and C. Schmid. Description of interest regions with center-symmetric local binary patterns. *Computer Vision, Graphics and Image Processing*, 58–69, 2006.
14. D. Varga, T. Szirányi, A. Kiss, L. Spórás, and L. Havasi. A Multi-View Pedestrian Tracking Method in an Uncalibrated Camera Network. *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 37–44, 2015.
15. G. Lie, G. Ping-shu, Z. Yi-bing, Z. Ming-heng, and L. Lin-hui. Pedestrian Detection Based on HOG Features Optimized by Gentle AdaBoost in ROI. *Journal of Convergence Information Technology*, **8**(2):1–9, 2013.
16. G. Monteiro, P. Peixoto, and U. Nunes. Vision-based pedestrian detection using Haar-like features. *Robotica*, **24**:46–50, 2006.
17. S. Maji, A. Berg, and J. Malik. Classification using intersection kernel support vector machines is efficient. *IEEE Conference on Computer Vision and Pattern Recognition*, 1–8, 2008.
18. P. Dollár, R. Appel, S. Belongie, and P. Perona. Fast feature pyramids for object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **36**(8):1532–1545, 2014.