Pedestrian detection and tracking in challenging surveillance videos

Kristof Van Beeck and Toon Goedemé

EAVISE - Campus De Nayer, KU Leuven, De Nayerlaan 5, 2860
Sint-Katelijne-Waver, Belgium
{kristof.vanbeeck,toon.goedeme}@kuleuven.be

Abstract. In this chapter we propose a novel approach for real-time robust pedestrian tracking in surveillance images. Typical surveillance images are challenging to analyse since the overall image quality is low (e.g. low resolution and high compression). Furthermore often birds-eye viewpoint wide-angle lenses are used to achieve maximum coverage with a minimal amount of cameras. These specific viewpoints make it unfeasible to directly apply existing pedestrian detection techniques. Moreover, real-time processing speeds are required. To overcome these problems we introduce a pedestrian detection and tracking framework which exploits and integrates these scene constraints to achieve high accuracy results. We performed extensive experiments on publically available challenging real-life video sequences concerning both speed and accuracy. Our approach achieves excellent accuracy results while still meeting the stringent real-time demands needed for these surveillance applications, using only a single-core CPU implementation.

Keywords: Pedestrian detection, Tracking, Surveillance, Computer vision, Real-time

1 INTRODUCTION

Reliable pedestrian detection and tracking in surveillance images opens up a wide variety of applications (e.g. abnormal behaviour detection, path prediction, intruder detection, people safety on e.g. movable bridges and crowd counting). In recent years, tremendous advances concerning pedestrian detection were published. Current state-of-the-art detectors achieve excellent accuracy results on publicly available datasets (see section 2). Unfortunately, directly applying these existing techniques on challenging surveillance images is not a trivial task. This is due to the inherent nature of these surveillance applications; often a large number of cameras are utilised since large areas need to be covered completely. Such scenarios impose severe constraints on the hardware: low-cost cameras are employed with wide-angle lenses, mounted high in a partly down-looking birds-eye view. Consequently image processing and analysis on these images is challenging.

Indeed, typical surveillance images are often captured at low-resolution and use high compression. Classic background subtraction based object detection

methods yield very noisy results at these high compression ratios. Moreover, these techniques do not differentiate between people and other objects. Due to their specific viewpoint (and wide-angle lens) standard pedestrian detectors - which are trained and evaluated on forward-looking images - are also unable to give accurate detection results on these images. Additionally, due to perspective effects some pedestrians to be detected appear very small in the image, which remains one of the most challenging tasks for current pedestrian detectors [15]. Furthermore, real-time processing speeds are required. In this chapter we pro-

Fig. 1. Example frame of one of the sequences of the CAVIAR dataset [8]

pose a flexible and fast pedestrian detection and tracking framework specifically addressing these challenging surveillance images. See figure 1 for a typical example frame of the publicly available surveillance dataset we used [8]. Our approach achieves excellent accuracy results at real-time processing speeds. We overcome the above mentioned challenges by the integration of three modalities: foreground segmentation approaches, the exploitation of scene constraints and an accurate pedestrian detector. This is done as follows. First, candidate regions in the image are generated. Using a calibrated scene distortion model, an early rejection of false patches is achieved. Next the candidate regions are warped to a standard viewing angle and used as input for a state-of-the-art pedestrian detector. As explained in section 3 our approach allows for the use of a highly accurate pedestrian detector which would otherwise be too computationally intensive for real-time applications. Finally, the detections are employed in a *tracking-by-detection* approach to further increase the accuracy. Note that, using our approach the actual scene calibration is trivial and easily performed. We demonstrate the effectiveness of our approach on challenging surveillance video sequences, and present extensive accuracy and speed results. Our approach is generalisable to other object classes. The remainder of this chapter is structured as follows. In section 2 we discuss related work on this topic, and distinguish our approach from existing work. Section 3 presents our framework in detail. Next we propose experimental results on challenging sequences in section 4. Finally, in section 5 we conclude our work and give final remarks on future work.

2 RELATED WORK

Pedestrian detection and tracking in general is a very active research topic. [10] initially proposed the use of *Histograms of Oriented Gradients* (HOG) for pedestrian detection. Their insights paved the way for numerous derived approaches; even today most state-of-the-art pedestrian detectors still rely on HOG features albeit in a more subtle manner (e.g. in combination with other features). A well-known example is the work of [17]. As opposed to the rigid model introduced by Dalal and Triggs they propose the inclusion of parts (representing e.g. the limbs or head of a pedestrian) to increase detection accuracy, coined the *Deformable Part Models* (DPM). In later work the authors tackled the inevitable increase in computational complexity by introducing a cascaded approach in which a fast rejection of negative detection windows is possible [16]. An extension was proposed using grammar models to cope with partial occlusion [19]. [21] published a new and fast training methodology for DPM models. As opposed to enriching the model with parts, [13] introduced the use of a rigid model with additional features, called *Channel Features* (ChnFtrs).

All previously mentioned approaches employ a sliding window paradigm: to cope with scale variations a scale-space feature pyramid is calculated and each layer is evaluated at each location. To speedup detection [12] proposed an approach which approximates features nearby to avoid a full feature pyramid calculation. Several other techniques have been proposed to speedup detection: using model scaling in stead of image scaling, GPU implementations [2] and search space minimisation techniques [1, 9, 27]. For several years, the DPM approaches remained among the top performing methods [14, 15]. However, the need of parts for pedestrian detection remains unclear [4]. Indeed, recent work on optimised rigid models - e.g. *Roerei* [3] and *ACF* [11] - in fact outperform the DPM detectors.

In [18] the authors present the use of *convolutional neural networks* (R-CNN) for object detection, achieving unprecedented state-of-the-art accuracy results. This methodology existed for a long time, but its applicability to image classification tasks was highlighted by the work of [23]. Interestingly, their method steps away from the traditional sliding window approach, and utilises region proposals as input for the deep learning classifiers. Although currently not real-time, their framework is able to classify a large variety of classes simultaneously, making it ideal for large image database retrieval applications such as ImageNetimage [30]. Recently [20] presented a hybrid approach combining DPMs with CNNs, called *DeepPyramid DPM*.

Several pedestrian tracking algorithms exist. Due to recent advances in object detection techniques, *tracking-by-detection* has become increasingly popular. There, an object detector is combined with a reliable tracking algorithm (e.g. particle filtering); see for example [7]. Concerning existing work on pedestrian tracking in surveillance images many either operate on standard viewpoint and/or high-resolution images [6, 31], or employ thermal cameras to facilitate segmentation to reduce the search area [24].

In previous work we presented a real-time pedestrian detection framework for similar viewpoint images which are captured with a blind-spot camera mounted on a real truck [32]. These images are - apart from the viewpoint - challenging since the camera is moving. However, in this work we can fully exploit and integrate foreground segmentation methods to increase both accuracy and speed. Furthermore, we work with images captured from genuine surveillance cameras. These images are of low-resolution, low-quality and, due to the use of wide-angle lenses show large amounts of distortion and contain non-trivial viewpoints.

Existing work on the same dataset either employs clustering algorithms with GPU optimisation [25], or focusses on motion analysis by matching trained silhouette models [28]. We differ significantly from these previous works: we developed an accurate tracking framework in which we can employ a highly accurate pedestrian detector on these challenging images, and thus perform much better than existing methods. We achieve real-time processing speeds on a single-core CPU implementation. Our approach easily lends itself for multi-threaded implementation if higher computational speeds are needed.

3 ALGORITHM OVERVIEW

Running standard pedestrian detectors such as the *Deformable Part Models* on surveillance images as shown in figure 1 is unfeasible. Current pedestrian detectors are only trained on upright pedestrians at a fixed height. Scale invariance is achieved using a scale-space pyramid. Thus in order to achieve decent detections on these surveillance images the detectors ought to run on multiple rotations and scales of the same surveillance image, using both dense rotation and scale steps. Evaluating the total 4D rotation-scale search space in real-time evidently is impossible. Nonetheless, the use of a pedestrian detector could significantly increase the accuracy, as opposed to standard techniques which only rely on e.g. background subtraction with blob analysis due to time constraints. Therefore, to overcome these challenges we propose the integration of a foreground segmentation approach with a scene model and a highly accurate pedestrian detector. Our approach allows for the detection of pedestrians in challenging viewpoints (e.g. rotated) under large lens distortion at low computational complexity, with very high accuracy. To retrieve the scene model, a simple one-time calibration procedure is performed, no explicit lens or camera calibration is needed. Our algorithm briefly works as follows. As seen in figure 1, pedestrians appear rotated and scaled based on their position in the image. We exploit this scene knowledge throughout our detection and tracking pipeline. For each input image, after a preliminary segmentation, we generate region proposals which potentially contain pedestrians. The scene model is used to reduce the number of region proposals. Next, based on the position in the image we warp each valid potential region to an upright and fixed-height image patch. These patches are given as input to a state-of-the-art pedestrian detector, which evaluates a pedestrian model on a single scale only. This is the key advantage of our work: since only one scale and position needs to be evaluated we can use a highly accurate pedes-

trian detector which would otherwise be too time-consuming. Furthermore this approach allows for the detection of extremely small pedestrians, if the detection model is powerful enough. The detections are retransformed to the original input image, and employed in a tracking-by-detection framework to associate pedestrian tracks and handle missing detections. Since each region can be evaluated independently, a fast multi-threaded implementation of this approach is trivial. Figure 2 shows an overview of our approach. In the next subsections we describe further details of each step in our pipeline, and motivate important design choices.

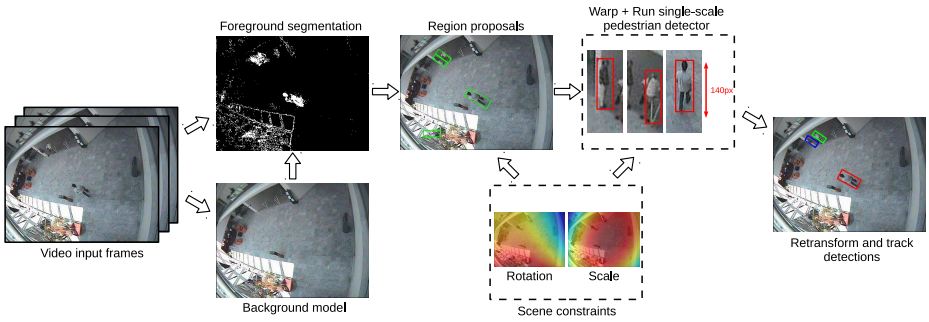

Fig. 2. Overview of our detection pipeline. After a first foreground segmentation step we extract region proposals which potentially contain pedestrians. Each region is warped to an upright fixed-height patch. Next, a highly accurate pedestrian detector is evaluated at a single scale. Finally, the detections are retransformed and tracked.

3.1 Foreground segmentation

First we perform a foreground segmentation step to identify moving regions in the static camera images. Several segmentation approaches are applicable ranging from basic background subtraction methods to more advanced motion estimation methods. Since we employ scene constraints further on to reduce the number of region proposals, our approach allows for the use of a coarse segmentation. For this step we thus prefer low computational complexity over high accuracy, excluding time-consuming techniques (e.g. optical flow). Hence, we rely on background estimation techniques, which generate a statistical model of the scene. Several popular methods exist. Since a comprehensive comparison of these techniques is out of the scope of this work, we refer to [5] and [26] for a detailed overview. Concerning background subtraction, the main challenges in typical surveillance images arise from changing lightning conditions and camera shake. Based on these comparative works we opted for the method of [33], which employs *Gaussian Mixture Models* (GMM). These methods haven proven to cope well with (limited) background motion. Their proposed method is an

extension of the original GMM where the number of Gaussian components per pixel is automatically selected. This effectively reduces memory requirements and increases the computation speed, making it ideal for this application. A qualitative segmentation output example is shown in the overview figure (fig. 2).

3.2 Modelling scene constraints

As previously mentioned, the pedestrians in the surveillance images appear rotated and scaled. Since the position of the surveillance camera is fixed with respect to the ground plane both parameters only depend on the position in the image. If we know the rotation and average pedestrian height for each pixel position $\mathbf{x} = [x, y]$ we can exploit this scene knowledge to achieve fast and accurate pedestrian detection, similar to [32]. During the generation of the region proposals this information can be used to reject regions which diverge too much from the expected region properties, thus limiting the search regions. For each valid proposed region, we use the transformation parameters to warp each patch to an upright, fixed-scale image patch, allowing the use of an accurate pedestrian detector whilst being real-time. To retrieve these transformation parameters a

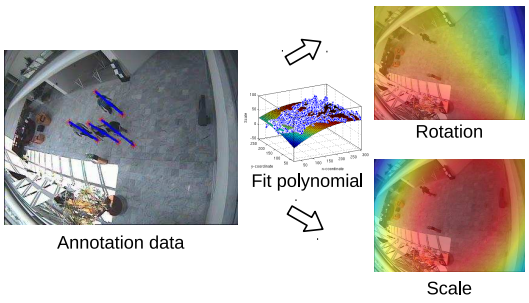

Fig. 3. A one-time calibration step is needed. The transformation parameters are extracted from the annotations.

one-time offline calibration needs to be performed (see figure 3). However, the scene calibration as proposed here is easy to perform and trivial. For this, we extracted the rotation and height of each annotated pedestrian from the dataset, giving the scale and rotation for that specific point. Next we interpolated the datapoints using a second order 2D polynomial function $f_i(\mathbf{x})$ for both parameters:

$$f_i(\mathbf{x}) = p_0 + p_1 x + p_2 y + p_3 x^2 + p_4 xy + p_5 y^2 \tag{1}$$

Both $f_{scale}(\mathbf{x})$ and $f_{rotation}(\mathbf{x})$ are used as *Lookup functions* (LUFs): at each position in the image they define the expected region properties and transformation parameters.

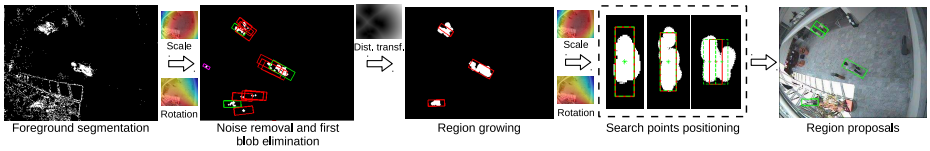

Fig. 4. Our region proposals pipeline. After foreground segmentation and noise removal a first blob elimination is performed. Next we perform region growing using a distance transform. Finally, we determine the optimal search points.

3.3 Generation of region proposals

In a next step we refine the segmentation and generate region proposals which need to be warped and evaluated using our single-scale pedestrian detector. Since we employ a pedestrian detector in the next stage to validate each region we are allowed to propose more regions than needed, i.e. regions without pedestrians. An accurate detector should indeed negatively classify such patches. However, it is important to early reject false patches, since they lead to useless computations and lower processing speeds. This stage thus tries to balance between optimal accuracy and speed, generating an optimal amount of search locations. Figure 4 gives an overview of our region proposal calculation. Let us now discuss each consecutive step in this pipeline.

First elimination. As a preprocessing step, we first eliminate noise in the segmentation which remained after the background subtraction step (due to e.g. changing lightning conditions). This is simply done using *morphological opening*. Next, we perform a connected component analysis (using 8-connectivity), and test the local scene model for each blob. That is, we construct a bounding box of the expected scale and rotation around the centroid of each blob. We reject two types of regions: extremely small ones (25 pixels or less) due to the high SNR there (drawn in magenta in the second step of figure 4), and those that diverge from an area constraint (drawn in red). For this constraint, we require that the area of the connected component should be larger than a minimal percentage of the expected area (15%). This step eliminates most invalid regions.

Region growing. In the case of insufficient contrast, the foreground segmentation performs suboptimal (i.e. tends to split a valid pedestrian in multiple blobs, as seen for the largest pedestrian in figure 4). For each remaining valid region we therefore perform region growing based on the Euclidean distance transform, joining regions nearby. This has a second advantage: multiple pedestrians which are nearby are joined into a single detection region, even if one of them was removed after the first elimination. This is also illustrated in figure 4: after the first elimination only one of both small pedestrians is maintained. However, after region growing both are connected.

Defining search points. Finally, we define exact search locations where the pedestrian detector will be applied. This is done as follows. Each remaining region is again verified against the scene constraints since, due to the previous step, these regions could have grown significantly. This is the case when multiple (possibly previously invalid) regions are joined. Note that we do not reject regions at this stage. We locally evaluate each region and use the expected height and rotation to estimate the number of possible pedestrians. Based on the size of the region we first evaluate if multiple search points are necessary for this region. If so, we define a linear grid over the entire region of which the step size depends on the ratio of the expected and actual region parameters, and eliminate grid points which are located outside the segmented region.

The final region proposals are visualised as the green rectangles shown in the rightmost image in figure 4. As seen, our regions accurately predict possible pedestrians in the image. This is the power of this approach: by combining foreground segmentation and scene model constraints the search space for the computationally expensive pedestrian detector can be enormously restricted. Slight deviations from the exact pedestrian position are allowed since we employ a sliding-window approach in the final warped patch.

3.4 Warping patches

Our scene model has another advantage: for each image location we know how a pedestrian is locally distorted. Each region proposal is warped to an upright pedestrian at a fixed-scale. Using this approach we are able to accurately detect even rotated and extremely small pedestrians, using a single-scale pedestrian detector only. The region proposals I are warped such that $I_{warp} = TI$ where transformation matrix T simply consists of a Euclidean transformation of which the parameters are extracted from the LUFs:

$$T = \begin{bmatrix} s\cos\theta & -s\sin\theta & t_x \\ s\sin\theta & s\cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

Note that the optimal scale to which the patches are warped highly depends on which pedestrian detector is used. This is dicussed in the next section, where we motivate the choice of pedestrian detector and determine the optimal scale.

3.5 Pedestrian detector

The warped image patches are now classified by a pedestrian detector. In fact, the method described in the previous sections is generic and can be combined with each existing pedestrian detection algorithm. As discussed in section 2, recent R-CNN based detection methods currently achieve top accuracy results concerning object detection in general. However, their performance is far from real-time, and they are more suited for multi-class large database retrieval tasks. Rigid pedestrian detectors (such as ChnFtrs) currently offer the best trade-off between speed

and accuracy when a full-scale space pyramid needs to be constructed. However, since we need to evaluate a single scale only, no scale-space pyramid needs to be constructed. Therefore we are able to use an accurate pedestrian detector which would otherwise be too time-consuming, such as the *Deformable Part Models*. Moreover, since a rigid model does not allow for any deformation, using it in our single-scale approach is even unfeasible in a direct manner. Since natural slight height variations exist between pedestrians (and due to small calibration errors), the detection accuracy significantly drops when using these models on a single-scale. Given this information, we opted to use the cascaded DPM model [16]. When used out-of-the-box this detector works as follows. First a scale-space pyramid is constructed in which for each layer HOG features are calculated resulting in a feature pyramid. Next, this pyramid is evaluated using a sliding window with the pedestrian model shown in figure 5. A pedestrian is represented as a

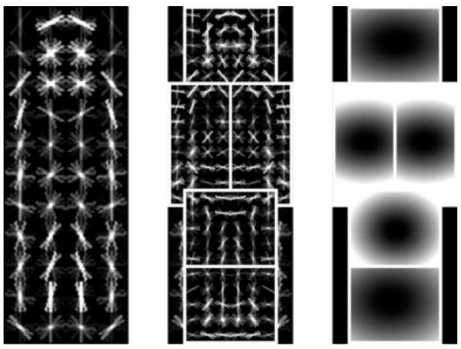

Fig. 5. Pedestrian model used in our implementation. (L) Root model. (M) Different parts. (R) Deformation costs.

root model (left), several parts representing e.g. the limbs and head which are calculated at twice the resolution of the root model (middle), and a deformation cost which penalises large deviations from the expected part locations (right). The responses of both root filter and part filters are summed to give a final detection score. We altered this detector into a single-scale only implementation, and performed experiments to simultaneously determine the optimal scale factor to which the region proposals need to be warped, and the optimal detection threshold. This is done as follows. We extracted about 6000 annotated pedestrians from the CAVIAR dataset and warped them to different scales (heights). Combined with 6000 negative patches we calculated the accuracy in function of the height and detection score threshold. The results are shown in figure 6.

As can be seen, at low resolutions the accuracy drops significantly, since only very limited spatial information is available. At high resolution similar behaviour is seen, since the pedestrians mismatch the detection model. Concerning the detection threshold, the detection accuracy is low at both high values (high *false*

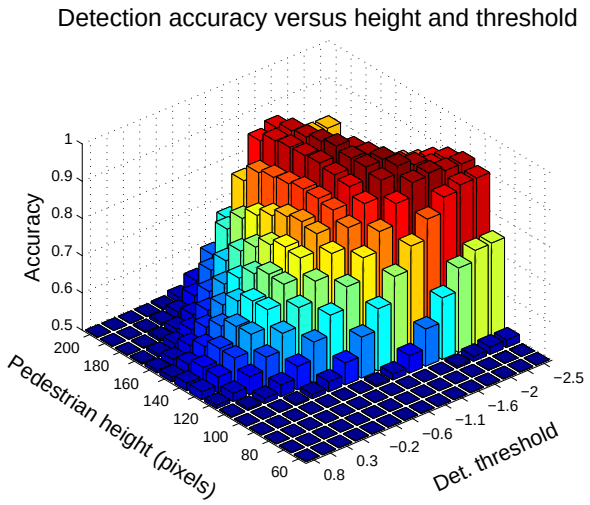

Fig. 6. The accuracy versus the pedestrian height and detection threshold for the single-scale cascaded DPM detector.

negative rate) and low values (high *false positive rate*). Figure 7 displays the optimal threshold slice extracted from figure 6. The accuracy is almost constant between 130-170 pixels. However, at larger pedestrian heights the detection time significantly increases. We therefore used 140 pixels as our optimal rescale height to which the region proposals will be warped such that a one-scale pedestrian model can be directly applied.

3.6 Tracking

The resulting detections are then retransformed to the input image coordinates. Next a *non-maxima suppression* step is performed, in which overlapping detections are filtered; only the highest scoring detection is kept. To link detections over multiple frames and to cope with occasional missing detections we integrate our approach in a *tracking-by-detection* framework. For this we employ the well-know Kalman filter [22]. For each new detection, a Kalman filter is initialised. We employ a constant velocity motion model. The state vector x_k thus consists of the centre of mass of each detection, the velocity and the scale: $x_k = \begin{bmatrix} x\ y\ v_x\ v_y \end{bmatrix}^T$. Our process matrix A thus equals:

$$A = \begin{bmatrix} 1\,0\,1\,0 \\ 0\,1\,0\,1 \\ 0\,0\,1\,0 \\ 0\,0\,0\,1 \end{bmatrix} \tag{3}$$

Using this motion model we predict the position of the pedestrians in the next frame. When a new frame is processed, we try to match each running tracker with

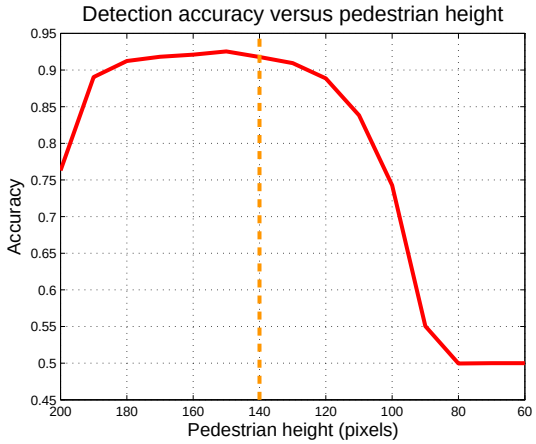

Fig. 7. The optimal threshold slice displaying the accuracy versus the pedestrian height.

a new detection as follows. We construct a circular region - based on the scale of that tracked detection - around the estimated new centroid. If the centroid of a new detection is found within that region, the detection is associated with this track, and the Kalman filter is updated. If multiple detections are found, we take the closest one based on the Euclidean distance. If no detection can be associated with a running track, we update the Kalman filter with its estimated position. If this occurs for multiple frames in a row, the track is discarded. For detections without an associated track, evidently a new Kalman filter is instantiated. Furthermore the exact size of the bounding boxes are averaged over multiple frames. See figure 8 for two qualitative tracking sequences of our proposed algorithm.

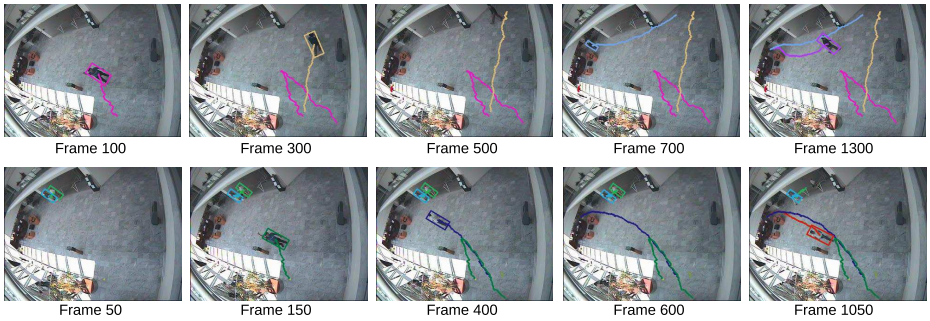

Fig. 8. Qualitative tracking example on two of the evaluation sequences (top and bottom row). See *http://youtu.be/kWoKBPQoeQI* for a video.

4 EXPERIMENTS AND RESULTS

We performed extensive experiments concerning both speed and accuracy on the publicly available CAVIAR dataset [8]. This dataset was recorded at the entrance lobby of the INRIA labs with a wide-angle camera lens. The images are taken with a resolution of 384×288 at 25 frames per second, and are compressed using MPEG2. See figure 1 for an example frame. The dataset is divided into six different scenarios: *walk*, *browse*, *meet*, *leave bags*, *rest* and *fight*. Each scenario is again subdivided into multiple sequences, making a total of 28 sequences. We used all sets for testing. Note that some sequences contain pedestrians which are inherently undetectable with our proposed framework. For example, the *fight* sequences include scenarios with people in specific fighting poses, and the *rest* sequences contain scenarios where people fall on the floor or rest in e.g. chairs thus violating our scene constraints. Table 1 gives a textual overview of each scenario. For each scenario we give a *difficulty* measure, i.e. an indication of the complexity of the sequences of each scenario. Easy scenarios are composed of simple sequences with only few people and low interaction whereas difficult scenarios contain many occlusions and challenging poses. In total, our evaluation set consists of about 26400 frames, containing about 36200 annotations. Our algorithm is implemented in Matlab, with time-consuming parts (e.g. the detection and transformation) in C and OpenCV (using *mexopencv* as interface). Our test hardware consists of an Intel Xeon E5 CPU which runs at 3.1 GHz. All speed test are performed on a single CPU core. However, a multi-threaded CPU implementation to further increase the processing speed is trivial.

Table 1. Overview of the difference sequences of the CAVIAR dataset.

Scenario	**# frames**	**difficulty**	**comments**
Walk	3045	easy	Few people, low interaction.
Browse	6654	medium	People browse at e.g. reception desk.
Leave bags	5839	medium	Leaving objects behind.
Rest	4220	medium	Resting on floor and in chairs.
Fight	2492	difficult	People fighting. Difficult poses.
Meet	4123	difficult	Group meetings, multiple occlusions.

4.1 Accuracy

Figure 9 displays the accuracy results of our algorithm, using *precision-recall* curves. We give results for all scenarios mentioned above, ranging from easy (e.g. *walk* - limited number of persons) through difficult (e.g. *meet* - multiple persons with occlusions). For the sake of clarity we spread the accuracy results of the six sequences over two separate plots, based on their difficult. The left accuracy plot groups the easy and medium scenarios, the right plot gives the accuracy results for the more difficult scenarios. We exclude small pedestrians

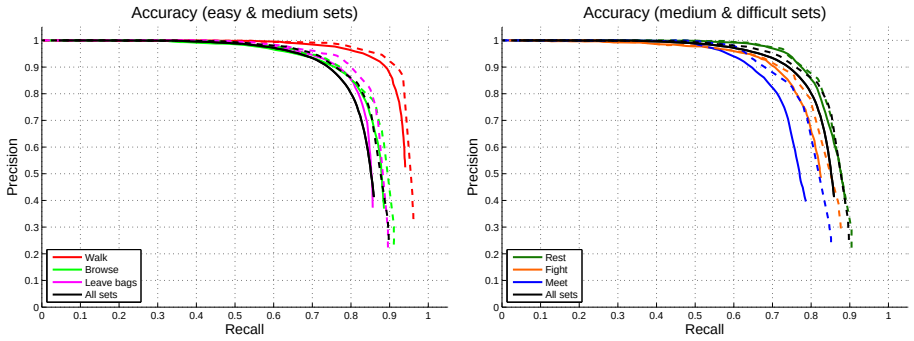

Fig. 9. The accuracy of our algorithm over the CAVIAR dataset. Solid lines indicate the results without tracking, dotted lines include tracking. The accuracy results for the six scenarios are divided over two graphs based on their difficulty for the sake of clarity. The black curve (*All sets*) indicates the average accuracy over all six scenarios.

from the annotations (smaller than 20 pixels), and remove annotations in the top left corner of the image (on the balcony) and the bottom left corner of the image (people behind the covered reception desk). Furthermore we discard annotations close to the image border, since the pedestrians are not completely visible there (the annotation is strict and already starts when part of a pedestrian enters the frame). The solid lines in figure 9 indicate the accuracy without tracking, whereas the dotted lines show the accuracy with tracking. The black curves on both figures indicate the total average accuracy over the entire evaluation set (all six scenarios). To indicate the difficulty, in figure 10 we display some extracted annotations which are warped to an upright position. As can be seen, these low-resolution output images contain severe compression artifacts. Even for humans they are sometimes difficult to recognize as a pedestrian. However, we achieve excellent accuracy results given these strict dataset annotations and challenging nature of these images. As observed, on some difficult scenarios (e.g. *Meet* and *Rest*) a lower accuracy is obtained. This is mainly due to two reasons: these

Fig. 10. Example of warped annotations. Low-resolution and high-compression artifacts are noticeable.

sets contain many long-term occlusions and poses which a standard pedestrian detector is unable to detect (e.g. sitting in a chair, lying on the floor). Since our tracker handles missing detections, the accuracy significantly improves.

4.2 Speed

The exact calculation time depends on the number of region proposals per image. Figure 11 therefore displays the speed of our algorithm (in frames per second), versus the number of region proposals. Evidently, the processing speed decreases

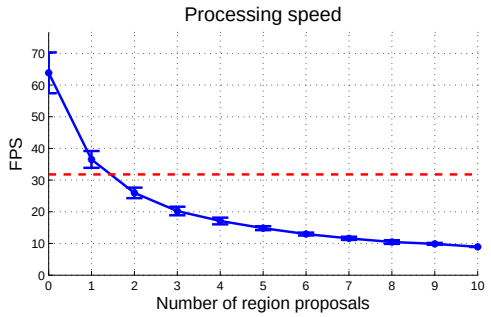

Fig. 11. The processing speed of our algorithm versus the number of region proposals.

when multiple region proposals need to be evaluated. However, even at e.g. four region proposals we still achieve 17 fps. Over the entire evaluation set we achieve an average of 32 frames per second, indicated with the dotted red line. Note that all experimental results are performed on a single CPU core. In fact, each region proposal can be evaluated independently, thus allowing for an easy multi-threaded implementation. Figure 12 visualises the individual calculation times for each important step in the entire algorithm pipeline, for a varying number of region proposals. As visualised, generation of the region proposals takes about 15-20ms. The warping operation is very fast: on average 1ms per region proposal is needed. Concerning the pedestrian evaluation step, the average feature calculation time per region is about 3ms whereas the model evaluation takes 4ms. The time needed to retransform the coordinates is negligible.

4.3 Comparative evaluation

Figure 13 illustrates the accuracy improvement we achieved as compared to a basic background subtraction technique, i.e. interpreting the foreground blobs that are large enough as pedestrians. As seen, on these challenging images these naive methods yield poor results. The inclusion of our scene model and the application of a state-of-the-art pedestrian detector raises the accuracy enormously.

A quantitative comparison with other work using precision-recall curves on this dataset is difficult, since to the best of our knowledge no such accuracy

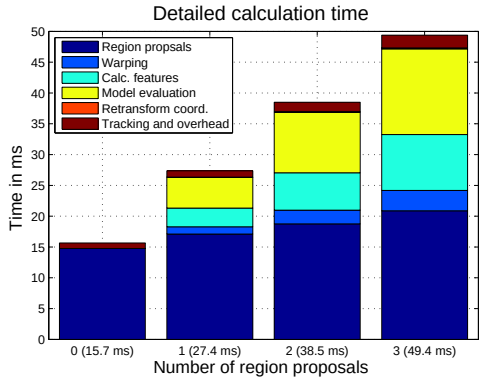

Fig. 12. An overview of the calculation time for each step in the algorithm versus the number of region proposals.

results similar to our work exist. Existing work on these specific sequences of the CAVIAR dataset often focusses on activity recognition (e.g. *fight*) and anomaly detection. However, [29] present accuracy experiments using *tracking failure* measurements on 11 tracks of the CAVIAR dataset. For this, the authors consider a track lost if the tracking failed for 20 frames or more. In their work a multi-hypothesis tracking approach (particle filter) is used. They achieve a tracking failure percentage of 33.64% with $N = 20$ particles and 16.82% when $N = 50$. Using our approach we achieve a tracking failure of 9.1% on the same sequences relying only on a single hypothesis tracker (Kalman filter). As a final

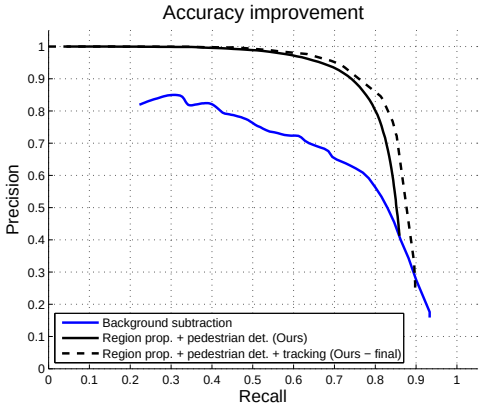

Fig. 13. The obtained accuracy improvement compared to a naive background subtraction approach.

qualitative analysis we compare our approach with a naive detection approach, that is running the standard deformable part model detector on all scales and all rotations. For this, we need to upscale the image five times (the smallest pedestrian to be detected is only 25 pixels high, and the height of the detection model equals 120 pixels), and use a rotation step size of 10 degrees. Using this approach, the calculation time for a single frame increases to about 13 minutes. Figure 14 displays the detections found using this naive approach (left), and the output of our algorithm (right). As seen, the naive approach yields several false positives and fails to detect all pedestrians. Our algorithm achieves excellent accuracy results with minimal computational cost (89ms for this frame).

Fig. 14. Qualitative comparison between running a detector on all scales and rotations (left) versus the output of our algorithm (right).

5 CONCLUSIONS AND FUTURE WORK

We presented a fast and accurate pedestrian detection and tracking framework targeting challenging surveillance videos. Our proposed algorithm integrates foreground segmentation methods with scene constraints to generate region proposals, which are then warped and evaluated by a single-scale pedestrian detector. Using this approach we can employ a highly accurate pedestrian detector for non-trivial camera-viewpoint images where existing pedestrian detectors fail, while still achieving real-time performance. We performed extensive evaluation experiments concerning both accuracy and speed on the publicly available CAVIAR dataset. This dataset consists of typical low-resolution high-compression surveillance images taken with a wide-angle lens from a challenging viewpoint. We show that our approach achieves both excellent accuracy and processing speeds using a single-core CPU implementation only. Furthermore, our proposed method easily lends itself for a multi-threaded implementation.

To improve the detection accuracy on very difficult scenarios (e.g. long-term occlusions, people in chairs or people lying on the floor) several further optimisations are possible. To cope with challenging poses an upperbody detector or an evaluation at multiple rotations could be employed. For this, the rotation

should be included in the tracker. Additional features (e.g. color information) could be used to enable person reidentification. Furthermore, the scene calibration currently is based on annotation data. In the future we plan to investigate if an automated calibration method can be implemented (using e.g. an offline exhaustive search over all scales and rotations).

ACKNOWLEDGEMENTS

The authors would like to acknowledge that the dataset used here is from the EC Funded CAVIAR project/IST 2001 37540 [8].

References

1. Benenson, R., Mathias, M., Timofte, R., and Van Gool, L. (2012a). Fast stixels computation for fast pedestrian detection. In *ECCV, CVVT workshop*, pages 11–20.
2. Benenson, R., Mathias, M., Timofte, R., and Van Gool, L. (2012b). Pedestrian detection at 100 frames per second. In *Proceedings of CVPR*, pages 2903–2910.
3. Benenson, R., Mathias, M., Tuytelaars, T., and Van Gool, L. (2013). Seeking the strongest rigid detector. In *Proc. of CVPR*, pages 3666–3673, Portland, Oregon.
4. Benenson, R., Omran, M., Hosang, J., and Schiele, B. (2014). Ten years of pedestrian detection, what have we learned? In *ECCV, CVRSUAD workshop*.
5. Benezeth, Y., Jodoin, P.-M., Emile, B., Laurent, H., and Rosenberger, C. (2008). Review and evaluation of commonly-implemented background subtraction algorithms. In *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*, pages 1–4. IEEE.
6. Benfold, B. and Reid, I. (2011). Stable multi-target tracking in real-time surveillance video. In *CVPR*, pages 3457–3464.
7. Breitenstein, M. D., Reichlin, F., Leibe, B., Koller-Meier, E., and Van Gool, L. (2011). Online multiperson tracking-by-detection from a single, uncalibrated camera. *IEEE PAMI*, 33(9):1820–1833.
8. CAVIAR project (2005). The CAVIAR project: Context aware vision using image-based active recognition. http://homepages.inf.ed.ac.uk/rbf/CAVIAR/.
9. Cho, H., Rybski, P., Bar-Hillel, A., and Zhang, W. (2012). Real-time pedestrian detection with deformable part models. In *IEEE Intelligent Vehicles Symposium*, pages 1035–1042.
10. Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *Proceedings of CVPR*, volume 2, pages 886–893.
11. Dollár, P., Appel, R., Belongie, S., and Perona, P. (2014). Fast feature pyramids for object detection.
12. Dollár, P., Belongie, S., and Perona, P. (2010). The fastest pedestrian detector in the west. In *Proceedings of BMVC*, pages 68.1–68.11.
13. Dollár, P., Tu, Z., Perona, P., and Belongie, S. (2009a). Integral channel features. In *Proc. of BMVC*, pages 91.1–91.11.
14. Dollár, P., Wojek, C., Schiele, B., and Perona, P. (2009b). Pedestrian detection: A benchmark. In *Proceedings of CVPR*, pages 304–311.
15. Dollár, P., Wojek, C., Schiele, B., and Perona, P. (2012). Pedestrian detection: An evaluation of the state of the art. *In IEEE PAMI*, 34:743–761.

16. Felzenszwalb, P., Girschick, R., and McAllester, D. (2010). Cascade object detection with deformable part models. In *Proceedings of CVPR*, pages 2241–2248.
17. Felzenszwalb, P., McAllester, D., and Ramanan, D. (2008). A discriminatively trained, multiscale, deformable part model. In *Proceedings of CVPR*.
18. Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2014a). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Computer Vision and Pattern Recognition*.
19. Girshick, R., Felzenszwalb, P., and McAllester, D. (2012). Discriminatively trained deformable part models, release 5. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
20. Girshick, R. B., Iandola, F. N., Darrell, T., and Malik, J. (2014b). Deformable part models are convolutional neural networks. *CoRR*, abs/1409.5403.
21. Girshick, R. B. and Malik, J. (2013). Training deformable part models with decorrelated features. In *IEEE International Conference on Computer Vision, ICCV 2013, Sydney, Australia, December 1-8, 2013*.
22. Kalman, R. (1960). A new approach to linear filtering and prediction problems. In *Transaction of the ASME Journal of Basic Engineering*, volume 82.
23. Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc.
24. Leykin, A. and Hammoud, R. (2010). Pedestrian tracking by fusion of thermal-visible surveillance videos. *Machine Vision and Applications*, 21(4):587–595.
25. Orts-Escolano, S., Garcia-Rodriguez, J., Morell, V., Cazorla, M., Azorin, J., and Garcia-Chamizo, J. M. (2014). Parallel computational intelligence-based multi-camera surveillance system. *Journal of Sensor and Actuator Networks*, 3(2):95–112.
26. Parks, D. H. and Fels, S. S. (2008). Evaluation of background subtraction algorithms with post-processing. In *Advanced Video and Signal Based Surveillance, 2008. AVSS'08. IEEE Fifth International Conference on*, pages 192–199. IEEE.
27. Pedersoli, M., Gonzalez, J., Hu, X., and Roca, X. (2013). Toward real-time pedestrian detection based on a deformable template model. *In IEEE ITS*.
28. Rogez, G., Orrite, C., Guerrero, J. J., and Torr, P. H. S. (2014a). Exploiting projective geometry for view-invariant monocular human motion analysis in man-made environments. *Computer Vision and Image Understanding*, 120:126–140.
29. Rogez, G., Rihan, J., Guerrero, J. J., and Orrite, C. (2014b). Monocular 3D gait tracking in surveillance scenes. *IEEE Transactions on Systems Man and Cybernetics Part B (Cybernetics)*, 44(6):894–909.
30. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. (2014). Imagenet large scale visual recognition challenge.
31. Singh, V. K., Wu, B., and Nevatia, R. (2008). Pedestrian tracking by associating tracklets using detection residuals. In *Motion and video Computing, 2008. WMVC 2008. IEEE Workshop on*, pages 1–8. IEEE.
32. Van Beeck, K., Tuytelaars, T., and Goedemé, T. (2012). A warping window approach to real-time vision-based pedestrian detection in a truck's blind spot zone. In *Proceedings of ICINCO*.
33. Zivkovic, Z. and van der Heijden, F. (2006). Efficient adaptive density estimation per image pixel for the task of background subtraction. *Pattern recognition letters*, 27(7):773–780.