Real-time Pedestrian Detection in a Truck's Blind Spot Camera

Kristof Van Beeck and Toon Goedemé
EAVISE, Campus De Nayer - KU Leuven, J. De Nayerlaan 5, 2860 Sint-Katelijne-Waver, Belgium
ESAT-PSI, KU Leuven, Kasteel Arenbergpark 10, 3100 Heverlee, Belgium

Keywords: Pedestrian Detection, Tracking, Real-time, Computer Vision, Active Safety Systems.

Abstract: In this paper we present a multi-pedestrian detection and tracking framework targeting a specific application: detecting vulnerable road users in a truck's blind spot zone. Research indicates that existing non-vision based safety solutions are not able to handle this problem completely. Therefore we aim to develop an active safety system which warns the truck driver if pedestrians are present in the truck's blind spot zone. Our system solely uses the vision input from the truck's blind spot camera to detect pedestrians. This is not a trivial task, since the application inherently requires real-time operation while at the same time attaining very high accuracy. Furthermore we need to cope with the large lens distortion and the extreme viewpoints introduced by the blind spot camera. To achieve this, we propose a fast and efficient pedestrian detection and tracking framework based on our novel *perspective warping window* approach. To evaluate our algorithm we recorded several realistically simulated blind spot scenarios with a genuine blind spot camera mounted on a real truck. We show that our algorithm achieves excellent accuracy results at real-time performance, using a single core CPU implementation only.

1 INTRODUCTION

Fast and meanwhile accurate pedestrian detection is necessary for many applications. Unfortunately these two demands are contradictory, and thus very difficult to unite. Even with today's cheaply available computational power it remains very challenging to achieve both goals. Indeed, recent state-of-the-art pedestrian detectors achieving real-time performance heavily rely on the use of parallel computing devices (e.g. multicore CPUs or GPUs) to perform this task. This often makes it unfeasible to use these algorithms in real-life applications, especially if these applications rely on embedded systems to perform their tasks.

In this paper we propose an efficient multi-pedestrian detection and tracking framework for a specific application: detection of pedestrians in a truck's blind spot zone. Statistics indicate that in the European Union alone, these blindspot accidents cause each year an estimated 1300 casualties (EU, 2006). Several commercial systems were developed to cope with this problem, both *active* and *passive* systems. Active safety systems automatically generate an alarm if pedestrians enter dangerous zones around the truck (e.g. ultrasonic distance sensors), whereas passive safety systems still rely on the focus of the truck driver (e.g. blind spot mirrors). However, none of these systems seem to adequately cope with this problem since each of these systems have their specific disadvantages. Active safety systems are unable to interpret the scene and are thus not able to distinguish static objects from actual pedestrians. Therefore they tend to generate many false alarms (e.g. with traffic signs). In practice the truck driver will find this annoying and often disables these type of systems. Existing passive safety systems are far from the perfect solution either. In fact, although blind spot mirrors are obliged by law in the European Union since 2003, the number of casualties did not decrease (Martensen, 2009). This is mainly due to the fact that these mirrors are not adjusted correctly; research indicates that truck drivers often use these mirrors to facilitate maneuvering. A passive blind-spot camera system with a monitor in the truck's cabin is always adjusted correctly, however it still relies on the attentiveness of the driver.

To overcome these problems we aim to develop an active safety system based on the truck's blind spot camera. Our final goal is to automatically detect vulnerable road users in the blind spot camera images, and warn the truck driver about their presence. Such an active safety system has multiple advantages

Van Beeck K. and Goedemé T..
Real-time Pedestrian Detection in a Truck's Blind Spot Camera.
DOI: 10.5220/0004821304120420
In *Proceedings of the 3rd International Conference on Pattern Recognition Applications and Methods* (ICPRAM-2014), pages 412-420
ISBN: 978-989-758-018-5

over existing systems: it is independent of the truck driver, it is always adjusted correctly and it is easily implemented in existing passive blind spot camera systems. Due to the specific nature of this problem, this is a challenging task. Vulnerable road users are a very diverse class: besides pedestrians also bicyclists, mopeds, children and wheelchair users are included. Furthermore the specific position and type of the blind spot camera induces several constraints on the captured images. These wide-angle blind spot cameras introduce severe distortion while the sideway-looking view implies a highly dynamical background. See figure 1 for an example frame from our blind spot dataset.

However, the most challenging part is undoubtly the hard real-time constraint, combined with the need for high accuracy. In this paper we present part of such a total safety solution: we propose an efficient multi-pedestrian tracking- and detection framework based on blind spot camera images. Our algorithm achieves both high accuracy and high detection speeds. Using a single-core CPU implementation we reach an average of 13 FPS on our datasets.

In previous work (Van Beeck et al., 2011; Van Beeck et al., 2012) we proposed our initial *warping window approach*. However, this initial approach was based solely on a naive similarity warp, running up against its limit (e.g. w.r.t. accuracy for our application). In this paper we propose our *perspective warping window approach*: we extensively redesigned and improved our previous work making it more elegant and accurate, without significantly increasing the algorithmic complexity. Moreover, we even obtain higher computation speeds. Figure 2 concisely compares our previous and our improved novel approach presented here.

Our proposed algorithm briefly works as follows. Traditional state-of-the-art pedestrian detectors use a *sliding window* paradigm: each possible position and scale in the image is evaluated. This however is unfeasible in real-time applications. Instead, we proposed our *warping window* approach: we eliminate the need to perform a full scale-space search using the exploitation of scene constraints. That is, at each position in the input image we locally model the transformation induced by the distortion. During detection, we can then warp the regions of interest (ROIs) in the image and use a standard pedestrian detector at a single scale on each ROI.

Figure 1: Example frame from our blind spot camera.

Figure 2: Similarity vs perspective transformation model.

This approach is integrated in a tracking-by-detection framework and combined with temporal information, making it more robust while reducing the detection time. We performed extensive experiments to evaluate our algorithm concerning both speed and accuracy. For this we recorded several realistically simulated dangerous blind spot scenarios.

The remainder of this paper is organised as follows. In the next section we describe related work concerning this topic. Section 3 describes our algorithm in more detail, while in section 4 we propose our experiments and evaluation results. We then conclude our work in section 5.

2 RELATED WORK

In the past few years the accuracy of pedestrian detectors has been significantly improved. Currently, even on challenging datasets excellent accuracy results are presented (Dollár et al., 2012).

Initially, Dalal and Triggs proposed a pedestrian detection framework based on the Histograms of Oriented Gradients (HOG) combined with an SVM (Support Vector Machine) for classification (Dalal and Triggs, 2005). This idea was further refined in Felzenszwalb et al. (2008) where the authors extended the concept with a part-based HOG model rather than a single rigid template. Evidently, this increases calculation time. To partially cope with this problem they proposed a more efficient cascaded framework (Felzenszwalb et al., 2010). Apart from increasing the model complexity, one can opt to increase the number of features to improve detection accuracy. Indeed, such a detector is presented in (Dollár et al., 2009a), called *Integral Channel Features*. However, each of these detectors still uses a sliding window ap-

proach. Across the entire image the features are calculated at all scales. To avoid such an exhaustive full scale-space search several optimisation techniques were proposed; e.g. Lampert et al. (2009) proposed an efficient subwindow search. Dollár et al. (2010) introduced the *Fastest Pedestrian Detector in the West (FPDW)* approach, in which they approximate feature responses from scales nearby thus eliminating the need to fully construct the scale-space pyramid. Extensive comparative works have been published (Enzweiler and Gavrila, 2009; Dollár et al., 2009b) to determine the most accurate approach. Both conclude that the HOG-based approach outperforms existing methods.

More recently, a benchmark between sixteen state-of-the-art pedestrian detectors was presented (Dollár et al., 2012). The authors conclude that part-based HOG detectors still achieve the highest accuracy, while the FPDW is one order of magnitude faster with only small loss in accuracy. Based on these conclusions we chose the part-based HOG model as the base detector in our framework.

Concerning speed, several GPU optimisations were proposed. Prisacariu and Reid (2009) proposed a fast GPU implementation of the standard HOG model. Pedersoli et al. (2013) presented a pedestrian detection system using a GPU implementation of the part-based HOG model. Benenson et al. (2012a) proposed work in which they perform model rescaling instead of image rescaling, and combined with their stixel world approximation they achieve fast pedestrian detection (Benenson et al., 2012b). Recently the authors proposed their *Roerei* detector (Benenson et al., 2013). Based on a single rigid model they achieve excellent accuracy results. However, in real-life applications using embedded systems such high-end GPU computing devices are often not available. Therefore our algorithm focuses on real-time performance, while maintaining high accuracy, on standard hardware.

Speed optimisation is also achieved using pedestrian tracking algorithms, of which several are proposed in the literature. They often rely on a fixed camera, and use a form of background modelling to achieve tracking (Viola et al., 2005; Seitner and Hanbury, 2006). Since in our application we have to work with moving camera images, this cannot be used. Pedestrian tracking algorithms based on moving cameras mostly use a forward-looking view (Ess et al., 2008) or employ disparity information (Gavrila and Munder, 2007). Cho et al. (2012) proposed a pedestrian tracking framework related to our work, exploiting scene constraints to achieve real-time detection. However, they use a basic ground-plane assumption whereas our approach is much more flexible and generic. Moreover, our specific datasets are much more challenging due to the severe distortion.

We significantly differ from all of the previously mentioned approaches. We aim to develop a monocular multi-pedestrian tracking framework with a challenging backwards/sideways looking view, targeting high accuracy at real-time performance. Furthermore, most of these classic sliding window approaches assume only object scale variation. Other geometrical variations (e.g. rotation (Huang et al., 2005) and aspect ratio (Mathias et al., 2013)) are usually covered by an exhaustive search approach. Our proposed warping approach offers a solution that can even cope with perspective distortion. In fact, without our warping window paradigm it would be unfeasible in practice to perform such an exhaustive search in a perspective distortion space.

3 ALGORITHM OVERVIEW

As mentioned above, existing pedestrian detectors employ a sliding window approach. Across all positions and scales in the image the features are calculated and evaluated, making it almost impossible to meet the stringent real-time demands needed in most safety applications. To achieve real-time detection speeds with high accuracy we propose our novel *perspective warping window* approach.

Our idea is mainly based on the following observation. Looking at an example frame from our dataset (see figure 1) one clearly notices that the pedestrians appear rotated, scaled and perspectively transformed. This is due to the specific position and the wide-angle lens of our blind spot camera. The crux of the matter is that this transformation only depends on the position in the image. Thus each pixel coordinate $\mathbf{x} = [x,y]$ uniquely defines the transformation at that specific position. If at each pixel position this transformation is known, we can dramatically speedup pedestrian detection. Based on this transformation we can locally warp each region of interest to upright pedestrians at a fixed height, and run a single-scale pedestrian detector on each warped ROI image patch. This approach effectively eliminates the need to construct a scale-rotation-transformation-space pyramid, and thus is very fast. Moreover, this approach is easily generalisable to other applications where such distortion occurs due to non-standard camera viewpoints and/or wide-angle lens distortions (e.g. surveillance cameras). To determine this transformation at each pixel coordinate a one-time calibration step is needed. To further increase both accuracy and speed, we inte-

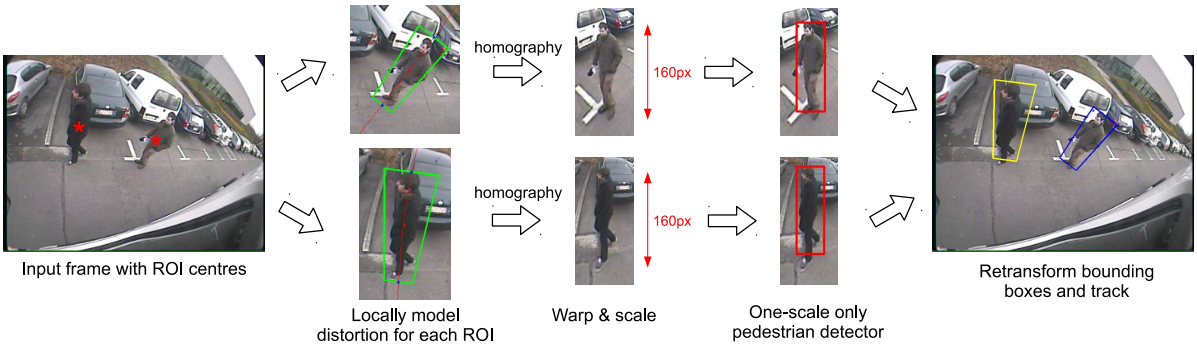

Figure 3: Illustration of our novel perspective warping window approach. At each position in the image we locally model the distortion, warp the ROIs to a standard scale and use a one-scale only pedestrian detector.

grate this warping window approach into an efficient tracking-by-detection framework. We use temporal information to predict future positions of pedestrians, thus further reducing the search space. Below we describe these steps in more detail. In subsection 3.1 we describe how our new perspective warping approach models the transformation, and motivate important algorithmic design choices such as the pedestrian detector, and the optimal scale parameter. In subsection 3.2 we then show how we integrate each of these steps into our total framework, thus describing our complete algorithm.

3.1 Warp Approach

Figure 3 illustrates our perspective warping window approach. Starting from input images as given in figure 1, pedestrians appear rotated, scaled and perspectively distorted. If we assume a flat ground-plane, these transformation parameters only depend on the specific position in the image. If we know the transformation we can model the perspective distortion for that ROI, extract and warp the ROI image patch to a fixed-scale (160 pixels - motivated further in this work) and perform pedestrian detection on a single scale only. We thus eliminate the need to construct a scale-space pyramid. Note that, although we perform detection on a single scale only, the pedestrian model still provides some invariance with respect to the pedestrian height. However, if large deviations from the standard height (e.g. children) need to be detected, an extra scale needs to be evaluated. The coordinates of the detected bounding boxes are then retransformed and fed into our tracking framework. Next we describe further details of our algorithm: how this position-specific transformation is mathematically modeled and how the calibration is performed. We further motivate the choice of our baseline pedestrian detector and determine the optimal fixed-height parameter.

Transformation Modelling. Figure 4 illustrates how the transformation is locally modeled. We use a perspective distortion model in the lens-distortion-corrected image. At each position, the height and width (at the ground) are known after a one-time calibration step (see further). These are visualised as two heat maps (the so-called look-up-functions or LUFs) in figure 4. The transformation coordinates are determined as follows. Each ROI centre coordinate (indicated with the red asterisk in the leftmost image) is first transformed into the undistorted image. This lens undistortion is simply based on the traditionally used radial lens distortion model:

$$\mathbf{x}' = \mathbf{x}(1 + k_1 r^2 + k_2 r^4) \quad (1)$$

$$r^2 = x^2 + y^2 \quad (2)$$

Here, $\mathbf{x}'$ denotes the corrected pixel coordinate, $\mathbf{x}$ the input coordinate and k_1 and k_2 indicate the radial distortion coefficients.

Next we calculate the vantage line through this ROI centre in the undistorted image, and determine the height and width (at the bottom) from the two LUFs. Based on these data we construct the perspective model in the undistorted image. The rotation of the image patch is determined from the angle of the vantage line, and the length ratio between the top and bottom is calculated based on the distance to the vantage point (visualised in the middle of figure 4). We thus locally model the pedestrians as if they are planar objects standing upright, faced towards the camera (that is, perpendicular to the optical axis of our blind spot camera). Our experiments show that this is a valid approximation for pedestrians. These coordinates are then retransformed to the distorted input image. Note that evidently only the coordinates are transformed, the middle image displayed here is only used for visualisation purposes. Based on the coordinates in the distorted image, and the known calibration data we apply a homography on the ROI image patch, thereby effectively undoing the local perspective distortion (visualised in fig. 3).

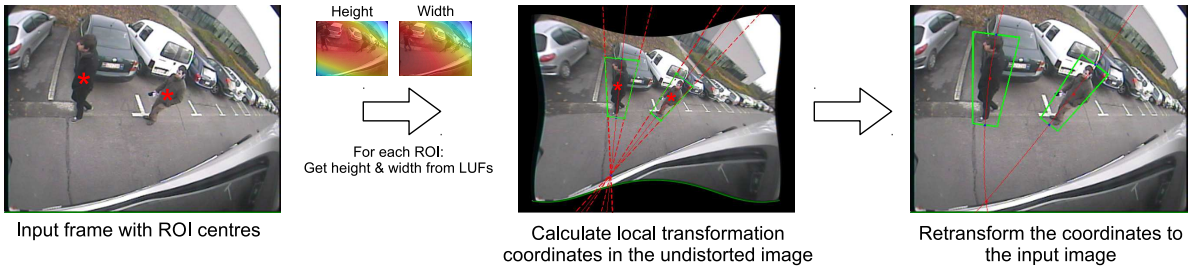

Figure 4: The transformation is modeled as a perspective transformation, calculated in the undistorted image.

Calibration. To obtain these two LUFs, a one-time calibration step is needed. To achieve this, we manually annotated about 200 calibration images. We utilised a planar calibration board of 0.5×1.80m, and captured calibration positions homogeneously spread over the entire image (figure 5). The labeling was performed in the undistorted image. These images yield the vantage point, and the height and width of a pedestrian (at the ground) at each position for that image. Next we interpolated these datapoints using two-dimensional second order polynomial functions for both the height and the width: $f_h(x,y)$ and $f_w(x,y)$ with:

$$f_i(x,y) = p_0 + p_1x + p_2y + p_3x^2 + p_4xy + p_5y^2 \quad (3)$$

Both functions are displayed as heat maps in figure 5: for each pixel coordinate they effectively give the height and width of the calibration pattern at that location. If for some reason the position of the camera w.r.t. the ground place changes, a recalibration needs to be performed. This is highly unlikely though, due to the robust camera mounting on the truck. Thus to summarise, detection is composed of four different steps: calculate the local perspective distortion model at each ROI centre, perform a homography and transform the pedestrians to an undistorted, upright position at a fixed height of 160 pixels, run a pedestrian detector at one scale, and finally retransform the coordinates of the detected bounding boxes to the original input image.

Pedestrian Detector. Based on the comparative works given in section 2 we conclude that, since we aim for high accuracy, HOG models are most suited. The FPDW has only slightly lower accuracy and is much faster. However, since we need to evaluate only one scale, no feature pyramid is constructed, thus this speed advantage is here not relevant. Since we know the scale at each position, this allows us to use a pedestrian detector with very high accuracy, which would otherwise be too computationally expensive for real-time operation. Thus, the choice for our baseline pedestrian detector goes to the top-accuracy state-of-the-art HOG based detector: the cascaded part-based HOG detector from (Felzenszwalb et al., 2010). Let us briefly discuss how this pedestrian detector works. The detector uses a pretrained model, consisting of HOG features (see fig. 6). It consists of a root filter and a number of part filters representing the head and limbs of the pedestrian. To use this model, first a scale-space pyramid is constructed, using repeated smoothing and subsampling. For each pyramid layer the HOG features are computed. Then, for a specific scale the response of the root filter and the feature map is combined with the response of the part filters to calculate a final detection score. On our 640×480 resolution images this detector off-the-shelf needs an average of 2.3s per frame, while their cascaded version (which we use in our framework) needs on average 0.67s per frame. We altered this detector into a single-scale detector and when used in our framework we achieve real-time performance (see section 4).

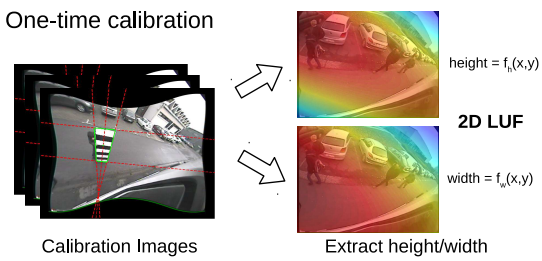

Figure 5: A one-time calibration is needed to determine the local perspective distortion.

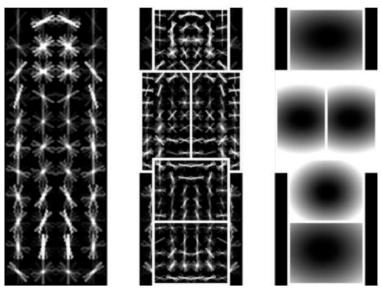

Figure 6: The pedestrian model of our detector. (L) Root filter (M) Part filters (R) Score distribution over parts.

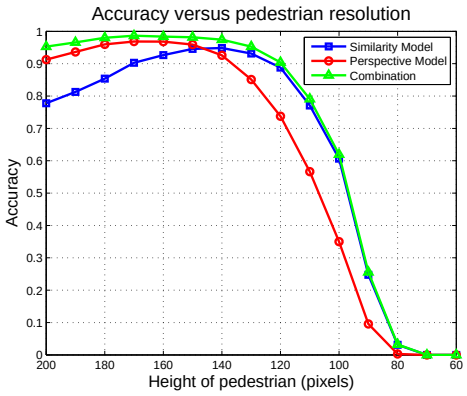

Figure 7: Determining the optimal scale parameter.

Determining the Optimal Scale Factor. As already mentioned, we rescale the pedestrians to a fixed height. For this, an optimal value needs to be determined. To achieve this, we extracted 6000 pedestrian images from our dataset, and performed the warp operation as given above. These pedestrians were warped to fixed heights, and we then performed accuracy measurements to determine the optimal height. Besides our novel *perspective transformation model* presented in this paper, we also warped the pedestrians using the *similarity transformation model* as explained in (Van Beeck et al., 2012), simply consisting of a rotation and scaling operation (see fig. 2 for a qualitative comparison). This was done to analyse the benefit of our more complex perspective model. Figure 7 displays our results. Besides the individual transformations, we also give the combined accuracy. Note that the optimal resolution of the perspective and similarity transformation model differs. For the first the optimal height lies at 160 pixels, whereas the latter reaches its optimum at 140 pixels. As can be seen,

Figure 8: Performance of the two transformation models in function of the position. Colored dots indicate which model performed the detection. Red: perspective model. Blue: similarity model. Green: both models. Yellow indicates missed detections.

Figure 9: Example of five initialisation coordinates together with their corresponding transformation ROIs.

the perspective model has a clear accuracy advantage over the similarity model. If both models were combined, an even higher accuracy is achieved. This, however, would double the calculation time. Figure 8 shows where each model performs best in function of the position in the image. Red dots indicate where the perspective model worked, blue where the similarity model worked and green were both models found the detection. Yellow indicates a missed detection. The perspective model obviously performs much better than the similarity model. The similarity model performs slightly better only at the image border, due to the small calibration error there. The perspective model performs better close to the truck because of the large amount of viewpoint distortion there. Note that if we analyse positions where both models found the detection, the perspective model achieves the best detection score in 69% of these cases, further indicating its clear advantage over the similarity transformation model.

3.2 Tracking Framework

To further improve the accuracy and detection speed we integrated our warping window approach in a tracking-by-detection framework. This is implemented as follows. Instead of a full frame search, we use initialisation coordinates (which define transformation ROIs) at the border of the image, and initially only perform detection there. See figure 9 for an example. If a pedestrian is detected, a linear Kalman filter is instantiated for this detection. As a motion model we use a constant velocity assumption. Our experiments indicate that this assumption holds for our application. The state vector x_k consists of the centre of mass of each detection and the velocity: $x_k = \begin{bmatrix} x & y & v_x & v_y \end{bmatrix}^T$. Based on the update equation $\hat{x}_k^- = A\hat{x}_{k-1}$ we estimate the next position of the pedestrian. Here, $\hat{x}_k^-$ indicates the *a priori* state estimate and $\hat{x}_k$ indicates the *a posteriori* state estimate, at

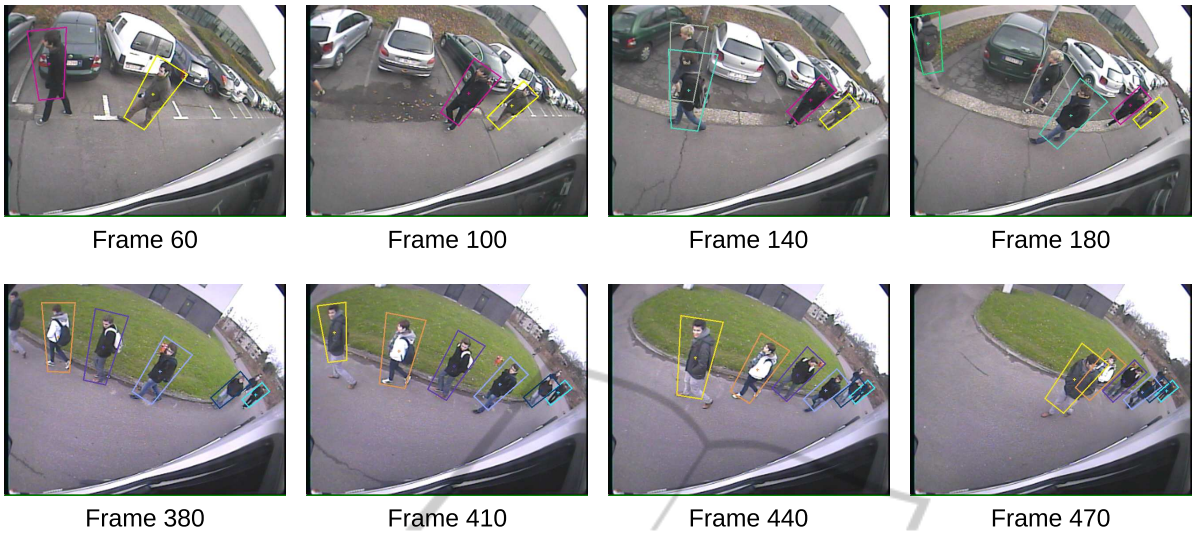

Figure 10: Qualitative tracking sequences over two of our datasets (top and bottom row) - see http://youtu.be/gbnysSoSR1Q for a video.

timestep k. The process matrix A thus equals:

$$A = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4)$$

Based on this motion model we predict the position (that is, the centre of mass) of the pedestrian in the next frame. Each estimated new pixel coordinate is then used as input for our warping window approach: we calculate the transformation model, warp the ROI and perform pedestrian detection on this ROI. For each pedestrian that is being tracked, our algorithm verifies if a new detection is found. This is evaluated by constructing a circular region around the estimated coordinate based on the scale of that tracked instance. If a new detection is found in this region, the Kalman filter is updated and the new position is predicted. If multiple detections are found, we associate the closest based on the Euclidean distance. The bounding box coordinates of tracked instances are averaged to assure smooth transitions between frames. If for tracked pedestrians no new detection is found, the Kalman filter is updated based on the estimated position. In this case we apply a dynamic score strategy, and lower the detection threshold for that instance (within certain boundaries). This ensures that pedestrians which are difficult to detect (e.g. partially occluded or a temporarily low HOG response) can still be tracked. If no detection is found for multiple frames in a row, the tracker is discarded. Evidently, if a new detection is found for which no previous tracker exists, tracking starts from there on. Figure 10 qualitatively illustrates tracking sequences on two of our datasets.

4 EXPERIMENTS & RESULTS

We performed extensive experiments concerning both speed and accuracy. Our datasets consists of simulated dangerous blind spot scenarios, recorded with a real truck. We used a commercial blind spot camera (Orlaco CCC115°) with a resolution of 640×480 at 15 frames per second. This camera has a 115 degree wide-angle lens. See figure 11 for the exact position of the camera. Five different scenarios were recorded, each in which the truck driver makes a right turn and the pedestrians react differently (e.g. the truck driver lets the pedestrians pass, or the truck driver keeps on driving, simulating a near-accident). This resulted in a total of about 11000 frames. For our accuracy and speed experiments we labelled around 3200 pedestrians. Our implementation is CPU-based only, and the hardware consists of an Intel Xeon E5 CPU which runs at 3.1 GHz. Note that all speed experiments are performed on a single core. The algorithm is mainly implemented in Matlab, with time-consuming parts

Figure 11: Our test truck with the mounted commercial blind spot camera (circled in red).

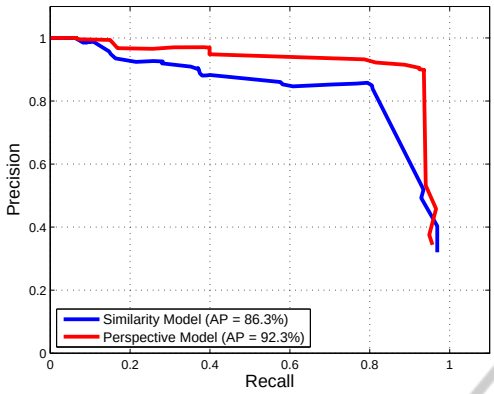

Figure 12: Precision-recall graph over our dataset.

(such as the homography) in OpenCV, using *mexopencv*.

Accuracy Results. Figure 12 displays the precision-recall graph of our algorithm as calculated over our datasets. The red PR curve indicates our novel perspective transformation approach, while the blue PR curve represents our previous similarity transformation approach. They are calculated as follows. For each detected pedestrian in our algorithm, we look for a labeled instance in a circular region (based on the scale) around the centre of our detection. If such an instance is found, this is counted as being a *true positive*. If this is not the case, this detection is counted as being a *false positive*. Each labeled pedestrian which is not detected accounts for a *false negative*. The PR-graph is then determined as: $precision = \frac{TP}{TP+FP}$ and $recall = \frac{TP}{TP+FN}$. We notice that, although both achieve very good accuracy results, our novel perspective warping window approach has a clear accuracy advantage over our similarity warping window approach. Indeed, the average precision (AP) for the similarity model equals 86.3%, whereas for the perspective model $AP = 92.3\%$. With the perspective model, at a recall rate of 94%, we still achieve a precision of 90%. Such high accuracy results are due to our warping window approach. Since we know the scale at each position, the number of false positives is minimized. Furthermore this allows us to use a sensitive pedestrian detection threshold.

Speed Results. As mentioned in section 3.1, if used out-of-the-box the baseline pedestrian detector takes 670ms (i.e. 1.5 fps). Since in our framework we only need to perform detection at a single scale and ROI, the calculation time drastically decreases. For each default search region and tracked pedestrian in the image we need to perform a warp operation and detection. Thus, the total calculation time evidently

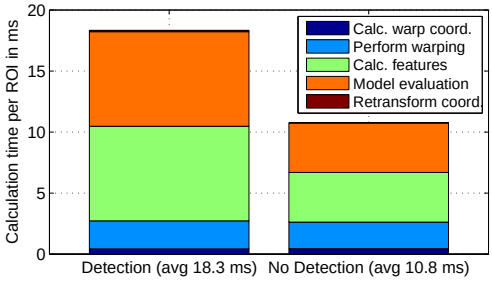

Figure 13: Calculation time per ROI.

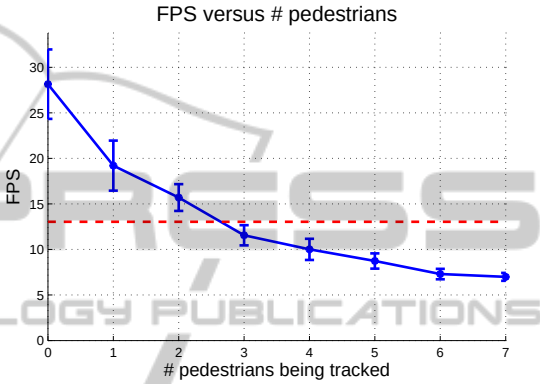

Figure 14: Speed performance versus the number of tracked pedestrians (dotted red line indicates the average fps).

depends on the number of tracked pedestrians per image. Figure 13 displays the average calculation time per ROI. Note that if a detection is found, the average calculation time equals 18.3ms, while if no detection is found the average calculation time drops to 10.8ms. This calculation time per region is independent of the position in the image. The average detection time per ROI is subdivided into five steps: the calculation of the warp coordinates, the time needed to perform the warp operation, calculation of the HOG features, evaluation of the pedestrian model, and finally the retransformation of the detected coordinates to the input image. The total warp time (*calc. warp coord.* and *perform warping*) only equals about 3 ms. Most time is spent on the actual pedestrian detection. The time needed to perform the retransformation of the coordinates is negligible. Figure 14 displays the frames per second as a function of the number of tracked pedestrians we reached on our datasets. If no pedestrians are tracked we achieve 28.2 fps. On average we achieve 13.0 fps (with an average of 3.4 pedestrians), while our worst-case framerate equals 7.0 fps.

5 CONCLUSIONS & FUTURE WORK

In this work we proposed a multi-pedestrian tracking framework achieving excellent accuracy and speed results on a single-core CPU implementation. The algorithm is based on our novel perspective warping window approach. We proposed this approach to allow for efficient pedestrian detection on the challenging, highly distorted camera images from a blind-spot camera, with minimal CPU resources. However, this approach is easily generalisable to other applications with non-standard camera-viewpoints.

In the future we plan to further extend our framework to multi-class detection: we aim to develop a complete vulnerable road users detection system, starting with bicyclists. Furthermore we aim to investigate if the inclusion of other features (e.g. motion information) could further increase the robustness of our framework.

REFERENCES

Benenson, R., Markus, M., Tuytelaars, T., and Van Gool, L. (2013). Seeking the strongest rigid detector. In *Proceedings of CVPR*, pages 3666–3673, Portland, Oregon.

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.

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.

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.

Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *Proceedings of CVPR*, volume 2, pages 886–893.

Dollár, P., Belongie, S., and Perona, P. (2010). The fastest pedestrian detector in the west. In *Proceedings of BMVC*, pages 68.1–68.11.

Dollár, P., Tu, Z., Perona, P., and Belongie, S. (2009a). Integral channel features. In *Proceedings of BMVC*, pages 91.1–91.11.

Dollár, P., Wojek, C., Schiele, B., and Perona, P. (2009b). Pedestrian detection: A benchmark. In *Proceedings of CVPR*, pages 304–311.

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.

Enzweiler, M. and Gavrila, D. M. (2009). Monocular pedestrian detection: Survey and experiments. In *IEEE PAMI*, volume 31, pages 2179–2195.

Ess, A., Leibe, B., Schindler, K., and Van Gool, L. (2008). A mobile vision system for robust multi-person tracking. In *Proceedings of CVPR*, pages 1–8.

EU (22 february 2006). Commision of the european communities, european road safety action programme: mid-term review.

Felzenszwalb, P., Girschick, R., and McAllester, D. (2010). Cascade object detection with deformable part models. In *Proceedings of CVPR*, pages 2241–2248.

Felzenszwalb, P., McAllester, D., and Ramanan, D. (2008). A discriminatively trained, multiscale, deformable part model. In *Proceedings of CVPR*.

Gavrila, D. and Munder, S. (2007). Multi-cue pedestrian detection and tracking from a moving vehicle. In *IJCV*, volume 73, pages 41–59.

Huang, C., Ai, H., Li, Y., and Lao, S. (2005). Vector boosting for rotation invariant multi-view face detection. In *ICCV*, pages 446–453.

Lampert, C., Blaschko, M., and Hoffmann, T. (2009). Efficient subwindow search: A branch and bound framework for object localization. In *IEEE PAMI*, volume 31, pages 2129–2142.

Martensen, H. (2009). Themarapport vrachtwagenongevallen 2000 - 2007 (BIVV).

Mathias, M., Timofte, R., Benenson, R., and Van Gool, L. (2013). Traffic sign recognition - how far are we from the solution? In *ICJNN*.

Pedersoli, M., Gonzalez, J., Hu, X., and Roca, X. (2013). Toward real-time pedestrian detection based on a deformable template model. *In IEEE ITS*.

Prisacariu, V. and Reid, I. (2009). fastHOG - a real-time gpu implementation of HOG. Technical report, Department of Engineering Science, Oxford University.

Seitner, F. and Hanbury, A. (2006). Fast pedestrian tracking based on spatial features and colour. In *Proceedings of CVWW*, pages 105–110.

Van Beeck, K., Goedemé, T., and Tuytelaars, T. (2011). Towards an automatic blind spot camera: Robust real-time pedestrian tracking from a moving camera. In *Proceedings of MVA*, Nara, Japan.

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*.

Viola, P., Jones, M., and Snow, D. (2005). Detecting pedestrians using patterns of motion and appearance. In *IJCV*, volume 63, pages 153–161.