Fast Cyclist Detection by Cascaded Detector and Geometric Constraint

Wei Tian,[1] Martin Lauer[1]

[1]Institute of Measurement and Control Systems,
Karlsruhe Institute of Technology, 76131 Karlsruhe, Germany
Email: {wei.tian, martin.lauer}@kit.edu

In this paper we present a vision-based detection system for cyclists. We build cascaded detectors with different classifiers and shared features to detect cyclists from multiple viewpoints. To improve the performance, we reveal the dependence between the size and the position of an object in the image by a regression method. We also explore the applications of this geometric constraint with different camera setups. Based on experiments we demonstrate that our detector is suitable for real time applications.

1 INTRODUCTION

In recent years the protection of vulnerable road users [GT07], mainly pedestrians and cyclists, has become a focus of automobile industries. However, the technical levels to protect both groups are unbalanced. For pedestrian protection, vision-based detection systems have broadly been applied in driver assistance systems with good performance [DWSP09]. In the meanwhile, the available systems for cyclist detection are mostly radar, infrared and acoustics based [DNL02], which are sensitive to all objects in the surroundings, reducing the usability and robustness. Some researchers also implant inductive loop sensors under the lane surface to detect bicycles [KB08]. However, this approach takes high maintenance cost and the detection is only available at certain locations. In comparison, vision-based systems can be installed on vehicles as well as infrastructure elements and have demonstrated high detection precision, e.g. for pedestrians [DWSP12]. Therefore, it is promising to develop a vision-based detection system specially tailored to cyclists.

The first step to detect cyclists in the image is to solve the multi-view problem, because the appearance of a cyclist in the image varies obviously according to the viewpoint. This problem is rarely taken into consideration in pedestrian detection. Secondly, the detector should provide

real time performance, so that the activation of protection measurements is not delayed, once a cyclist is detected.

In this work we introduce a vision-based detection system for cyclists. To solve the multi-view problem, we build viewpoint-based detectors, each has a cascade structure for computational efficiency. Based on the geometric constraint, obtained by a regression method, a small ROI (region of interest), where objects can appear, is extracted. Since only this small ROI is detection relevant, the detection speed and precision can be further improved. When combining these two methods, we show that our detector is suitable for real time applications.

2 RELATED WORK AND CONTRIBUTION

Vision-based object detection, especially for the pedestrians, has become a hot topic among most of the researchers in the last ten years. Powerful features, such as Haar [VJ04], HOG [DT05] and its variant DPM [FMR08], are proposed to improve the detection precision significantly. In the mean while, efficient detector structures like cascade [VJ04], fast pyramid [DT05] and scaled models [BMTVG12] are introduced to achieve a real time performance. The orientation estimation is also attempted in some works, e.g. Enzweiler et al. in [EG10] utilized sample-dependent cluster priors and discriminative experts.

To detect cyclists, Rogers et al. used a model with two circles to represent bicycles [RP00]. Qiu et al. in [QYZ+03] utilized edge and motion information to detect both cyclists and pedestrians. A combination of DPMs and an extended Kalman filter was applied by Cho et al. in [CRZ10] to detect and track bicycles. This work is extended in [CRZ11] by him with the help of an interacting multiple model filter for 3-D tracking. In spite of a high detection rate, this approach suffers the fact that it can not distinguish between parked bicycles (without riders) and cyclists. And DPMs are known as time consuming without additional hardware acceleration [YLWL14]. Instead, HOG-LP (light and pyramid sampling) is used by Li et al. to detect crossing cyclists [LCX10]. Takahashi et al. in [TKM10] also added additional pedaling movement to improve detection performance. But the usabilities of both methods are limited, because cyclists from other directions are not well considered.

In this paper we show that cascaded detectors in combination with different classifiers and shared features are good choices to detect cyclists from multiple viewpoints. Another contribution is that we reveal the dependence between the size and the position of an object in the image (i.e. the geometric constraint) by a regression method. Moreover, we explore the application of the geometric constraint with different camera setups, e.g. with rotations, making it the third bright spot of this paper.

3 APPROACHES

This section consists of three parts. The first part introduces our detection framework, including the detector structure and its training algorithm. The second part focuses on the geometry based

Figure 1: Positive samples from KITTI dataset are divided into eight equidistant orientation bins I to VIII.

ROI extraction method. And the last part describes the combination of ROI extraction and the detection procedure in dealing with image scaling and camera rotations.

3.1 Detection Framework

To handle the multi-view problem of cyclists, we follow the idea of Ohn-Bar in [OBT14] by dividing the cyclists into subcategories according to the viewpoints. For training the detector we also use the KITTI dataset [GLU12], which consists of rich images captured in traffic scenes, including pedestrians, cyclists, cars, etc. According to the ground truth of the 3-D orientations, positive samples here are divided into eight equidistant orientation bins, as shown in Fig. 1, each with a range of $45°$. This Approach is also consistent with our previous work in [TL15]. As sliding window detection is applied in this work and samples even from the same bin present minor differences in the size, we scale the samples of each orientation bin to their average aspect ratios. We choose 0.5 for bin II and VI, 1.0 for bin IV and VIII and the aspect ratio for other bins is equal to 0.75. The samples have a minimal height of 80 pixels.

For each orientation bin we build a cascaded detector. As shown in Fig. 2, the cascade consists of $n + 1$ stages. Each of the first n stages is a DF (decision forest), composed of multiple boosted decision trees with a depth 2, which is the same as [OBT14]. But the last stage is replaced with one SVM (support vector machine). Such a structure has several advantages. At first, the powerful SVM allows to make our cascade very short, only consisting of 3 to 4 stages. In comparison, a cascade only made of DFs should have about 10 stages to achieve the same precision. With the help of such a simple structure, a fast detection can also be achieved. Although SVMs are computationally more demanding than DFs, the influence on speed is mere. Because the SVM is located at the end stage, only a few classifications are carried out. In the meanwhile, the whole training process is also short, because the time to train one SVM,

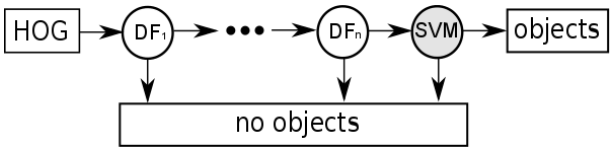


Figure 2: The cascade consists of n decision forests DF_1 to DF_n and one stage of support vector machine SVM. While the DFs make hypotheses by comparing some specific spatial and orientation bin values of an HOG vector with predefined thresholds, the classification of SVM is done with the whole HOG vector of an image patch, so that both a high precision and a fast detection can be achieved.

according to our experiments, is about 2 to 3 minutes. In contrast, a cascade only made of about 10 DFs can result in many hours of training process.

In this paper we use the same HOG features as [FMR08] both for the DFs and the SVM. Each DF only selects specific spatial and orientation bin values from a HOG vector of an image patch as inputs. Hypotheses are made by comparing those bin values with predefined thresholds. In contrast, the classification by SVM is done with the whole HOG vector. Since the features are shared by both kinds of classifiers, for computational efficiency, they are only calculated once at the beginning and saved for following detections.

To train the detectors, we use half of the training dataset from KITTI [GLU12], i.e. the dataset D, to generate positive and negative samples and the other half as a dataset T for validation. The training procedure of a detector from one orientation bin, shown in Algorithm 1, can be divided into two parts, (a) training a cascade only with DFs and (b) training the SVM at the last stage. Part (a) is done by the AdaBoost algorithm. The details about AdaBoost and training a single decision tree are described in [VJ04] [Dol]. In part (b), false positives from the trained cascade are considered as negative samples while the positive samples remain unchanged. These samples are used to train one SVM, which is added at the end of the cascade. The new detector is tested on the training dataset to collect false positives, which are added to the dataset of negative samples to retrain the SVM. This procedure is repeated until either the predefined precision p_0 or the number of iterations l is reached. The current precision p of the detector is achieved by running it on the validation dataset T.

To choose the best number n of DFs, detectors with different lengths are tested on the dataset T. The average false positives per image (FPPI) of each cascade length with respect to a recall of 90% are listed in Table 1. As can be seen, the minimum of the detection error is located at the value of $n = 2$. Besides that, the errors can neither by an increased nor by a decreased cascade be reduced. This fact implies that the classification power of the cascaded detector is influenced both by the DFs and the SVM. On one hand, a small number of DF stages increases the classification burden on the SVM while, on the other hand, a large number of DF stages reduces the number of negative samples, which are applied to train the SVM, so that the SVM is more likely to overfit the examples. Therefore, the number n should be chosen from a trade-off between these two arguments. According to the experiments, a value of 2 or 3 is a

Table 1: Average false positives per image (FPPI) of each cascade length with respect to a recall of 90%. n represents the number of DFs in the detector.

n	0	1	2	3	4	5	6	7
FPPI	0.22	0.16	0.11	0.12	0.14	0.16	0.18	0.19

reasonable choice.

Since samples are divided into 8 orientation bins, we also train 8 detectors, which run on images one after another during detection. The final detection score is estimated by choosing the highest score of all the detectors with the help of the non-maximum-suppression algorithm. Besides the score, the ID of the corresponding viewpoint is also assigned to each detection, so that cyclists from different viewpoints in the image can be identified.

3.2 ROI Extraction by Geometric Constraint

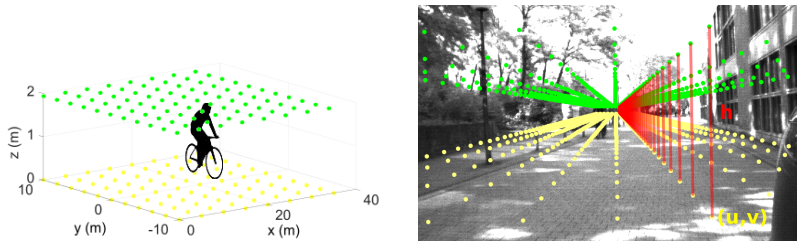
Considering that all of the 8 detectors are applied on each image, the detection can be time consuming and the errors from each detector are accumulated. For performance improvement, we integrate an ROI extraction method into our framework. By forward applying the ROI restriction, the detection only focuses on small image regions, so that the time cost for feature calculation and sliding window search can be reduced. As false positives outside the ROI can also be excluded, the precision increases as well (Fig. 4 (b)). In this paper, the ROI extraction is based on the fact that all cyclists are on the ground and the installation pose and position of the camera are known. Thus, the task is to find image regions, which represent objects standing on the ground. As solutions, three generally used methods are listed as follows.

- **Distance based method** It relies on distance measuring sensors. Benenson et al. used stereo cameras in [BMTVG12] to estimate the size and the position of pedestrians in the image. Since additional sensors are required, this method is inappropriate for ROI extraction only from monoscopic images.

- **Heuristic method** It is usually done manually to determine the image regions where objects are most likely to appear. E.g. Beck et al. in [BS14] simply used the lower image part to extract lanes. Since the ROI is selected manually, this method can only provide coarse locations of interesting objects.

- **Geometric method** It is based on the principles of a pinhole camera model. Sudowe et al. in [SL11] introduced the mathematical derivation between the size of an object in the real world and its position in the image, so that a significant improvement both on detection precision and speed is achieved.

In this paper we prefer the geometric method, since only a monocular camera is available and precise locating of objects is required. Unlike the approach in [SL11] we reveal the rela-



(a) (b)

Figure 3: The ground plane is presented in yellow with a height of $z_w = 0$ and the green one is the upper plane with a height of $z_w = S_{obj} = 2\mathrm{m}$. Both planes consist of grid points, which are displayed both in the real world coordinates (a) and in the image coordinates (b). Line in red represents the distance h between each point pair and (u, v) are image coordinates of the corresponding ground point.

tionship between the size and the position of an object in the image by a regression method, which can deal with image scaling and camera rotation efficiently (see Section 3.3).

Here we use the same camera model to introduce our method. Assuming that the camera is calibrated, a point from the real world with coordinates $\mathbf{X} = [x_w, y_w, z_w, 1]^T$ can be projected to a point in the image by

$$z \cdot \mathbf{U} = \mathbf{P} \cdot \mathbf{X}, \tag{1}$$

where $\mathbf{U} = [u, v, 1]^T$ contains the corresponding image coordinates, z is the depth of the object and $\mathbf{P}$ is the projection matrix.

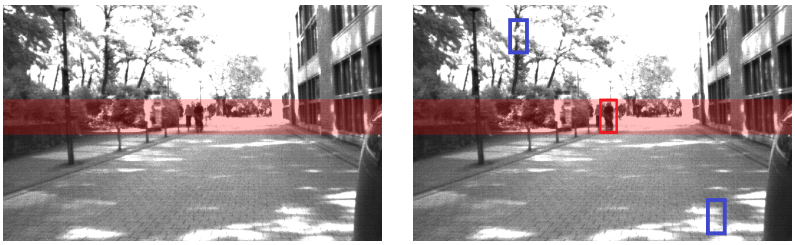
At first we build two horizontal planes in the real world. The first one with a height $z_w = 0$ corresponds to the ground plane and the second one has a height of $z_w = S_{obj}$, just above the head of the object. In the next step, we create grid points in both planes, as shown in Fig. 3 (a). For each point in the ground plane we associate it with the point with the same horizontal coordinates in the upper plane to create one pair. So the distance between each point pair is constant and corresponds to object's height S_{obj}, which here equals $2\mathrm{m}$.

With the help of (1) we project the grid points of both planes into the image. Then we calculate distances between the projected point pairs (Fig. 3 (b)) and store them in a vector $\mathbf{h}$. According to [SL11], the size and the position of an object in the image are related by a linear equation

$$\mathbf{h} = \mathbf{U_g} \cdot \mathbf{B}, \tag{2}$$

where

$$\mathbf{U_g} = \begin{bmatrix} u_1 & v_1 & 1 \\ u_2 & v_2 & 1 \\ \vdots & \vdots & \vdots \\ u_n & v_n & 1 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}.$$



(a) (b)

Figure 4: The red regions in (a) and (b) represent the ROI extracted by our geometric method. This image is captured on the KIT campus with a size of 1312×1082 pixels. The roll angle of the camera is zero and the detection window has a height of 80 pixels. The height of the object in the real world is assumed to be $2\mathrm{m}$. Red rectangle in (b) denotes the detected object in ROI and false positives outside ROI are marked in blue.

Matrix $\mathbf{U_g}$ is a coordinate matrix. Each row of it corresponds to the location of one ground point in the image. And $\mathbf{B}$ is a parameter matrix, which is easy to obtain by some regression method, e.g. the least square algorithm. For simplification, (2) can be rewritten as

$$h = \begin{bmatrix} u & v & 1 \end{bmatrix} \cdot \begin{bmatrix} a \\ b \\ c \end{bmatrix}, \tag{3}$$

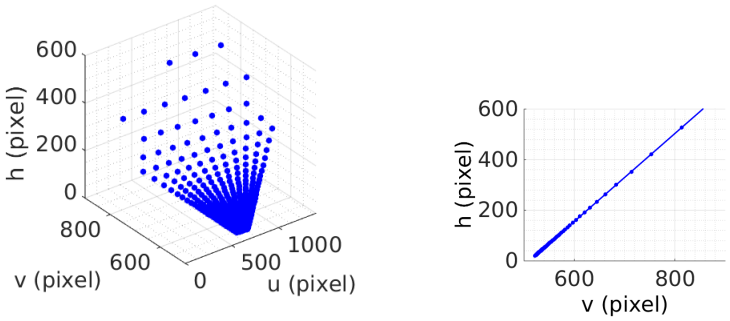
where h is exactly the object's height with respect to the ground point $[u, v, 1]^T$ in the image. If Euler angles are used to describe the camera model and the roll angle is zero, then the object's size is only dependent on its vertical coordinate v in the image. Hence, (3) can be further simplified as

$$h = \begin{bmatrix} v & 1 \end{bmatrix} \cdot \begin{bmatrix} b \\ c \end{bmatrix}. \tag{4}$$

Therefore, the ROI for valid objects can be restricted to a region, which is bounded by two horizontal lines (Fig. 4) and in accordance with the results in [SL11]. For validation of (3) and (4), we plot the measured heights of an object in dependence of the ground points in Fig. 5 (a). As can be seen, the plotted points are located in one plane, so the linear assumption has been proven. If we project this plane onto the axis of coordinate v (Fig. 5 (b)), then the points are only located in one line, which means the height of the object is only related to its vertical coordinate in the image and is consistent with (4).

3.3 ROI based Detection for Image Scaling and Camera Rotation

Instead of cutting off the ROIs as separate image patches, here we restrict the locations of the detector directly on the input image. Since sliding window detection is utilized in this work and



(a) (b)

Figure 5: Plot (a) shows the dependence between measured object's heights h and foot point coordinates (u, v) in the image. Because all the points are located in one plane, the linear relationship has been proven. Plot (b) shows the projection of the plane onto the axis of vertical coordinate v. As our camera has a zero roll angle, the points are located in one line, which means in this case that the height h of the object is only related to its vertical coordinate v.

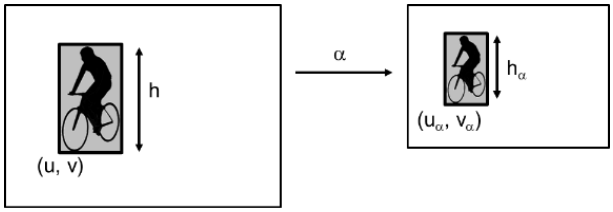


Figure 6: Scaling an image by factor $\alpha\,(0 < \alpha < 1)$. The height h and the bottom left corner (u, v) of the object are scaled to h_α and (u_α, v_α) respectively.

the window size is fixed, we can locate the valid position of an object in the image by

$$v_{obj} = \frac{h_{win} - c}{b} \tag{5}$$

with the window height h_{win} and the vertical image coordinate v_{obj} of the foot point of the cyclist. Hence, the searching area of the detector is limited.

However, this procedure is only appropriate to detect objects, which are as big as the window. If the size of the object varies, we should scale the object to adapt it to the window. This approach is called an image pyramid, since the image is scaled descendingly in different levels. Fig. 6 shows an example of scaling an image with a factor α. Clearly, the height h and the location (u, v) of an object (here corresponds to its bottom left corner) are scaled with α in the meanwhile. So we can derivate a linear equation for scaled images by multiplying factor α with both sides of (3) and obtain

$$h_\alpha = \begin{bmatrix} u_\alpha & v_\alpha & \alpha \end{bmatrix} \cdot \begin{bmatrix} a \\ b \\ c \end{bmatrix} \tag{6}$$

with

$$\begin{bmatrix} h_\alpha & u_\alpha & v_\alpha \end{bmatrix} = \alpha \cdot \begin{bmatrix} h & u & v \end{bmatrix}.$$

Here we can rewrite (6) as

$$h_\alpha = \begin{bmatrix} u_\alpha & v_\alpha & 1 \end{bmatrix} \cdot \begin{bmatrix} a \\ b \\ \alpha \cdot c \end{bmatrix}. \tag{7}$$

Since h_α is the new height of the object and vector $[u_\alpha, v_\alpha, 1]$ represents the new location, (7) depicts exactly their dependence for scaled images. In comparison to (3), it only differs in the last element of the parameter matrix. In the same way, we can get the simplified equation for zero roll angle

$$h_\alpha = \begin{bmatrix} v_\alpha & 1 \end{bmatrix} \cdot \begin{bmatrix} b \\ \alpha \cdot c \end{bmatrix}. \tag{8}$$

And the valid position of an object in a scaled image is given by

$$v_{obj} = \frac{h_{win} - \alpha \cdot c}{b}. \tag{9}$$

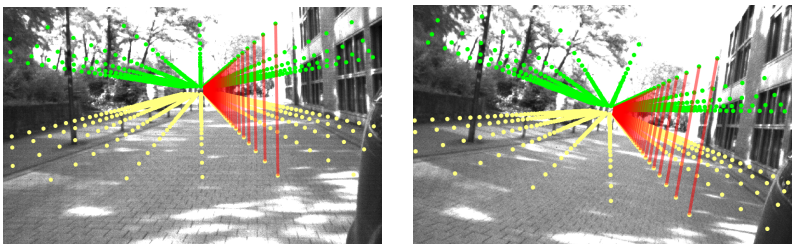
Therefore, the geometric constraint can be extended to multi-scale detection. Compared to [SL11], our approach only needs to scale one parameter instead of the whole intrinsic matrix.

Actually, this geometric constraint based detection method is not valid in every case. According to our experiments, the camera should be installed higher than the ground. If the origin of the camera coordinate system and the viewing direction fall into the ground plane, all the ground points will share the same vertical coordinates in the image, so that they cannot be distinguished from each other and the size of an object is no longer relevant for its position in the image.

Moreover, only when the image is rectified, the linear function (3) is valid. Thus, distortion compensation should be done at first. Otherwise, the boundary lines of the ROI turn into parabolas. In [SL11], it is also pointed out that if the camera viewing direction is exactly parallel to the ground, the ROI can be bounded by two horizontal lines. But in our experiments, a non-zero pitch angle of the camera is also allowed, which means that the optical axis of the camera intersects the ground plane. In this case, the ROI is only vertically shifted by some pixels if a small pitch angle exists (Fig. 7 (a)).

To maintain the ROI in horizontal direction, the roll angle of the camera should be zero, which is already addressed before. While [SL11] is based on the assumption that the object is vertically aligned in the image, our approach can handle images with rotations. Even with a non-zero roll angle, (3) still works, because it only considers the distance between point pairs and their coordinates (Fig. 7 (b)). To detect objects in such an image, we either need detectors trained for rotated objects or to rotate image objects to align them to the vertical axis.

Up to now, we are only concentrated on constant installation height and orientations of the camera. For dynamical cases, such as on a moving vehicle, this geometric constraint is also applicable if the fluctuation in driving is mild or the variation of camera poses can be on-line estimated.



(a) (b)

Figure 7: Geometric constraint for rotated cameras, each with a pitch angle of -6° (a) and a roll angle of 11° (b). The used coordinate system is the same with [GLU12].

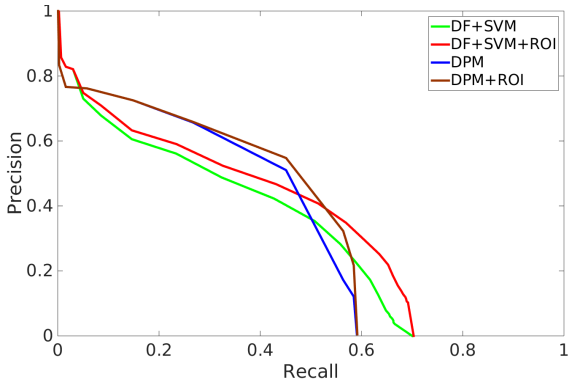


Figure 8: Precision-recall-curves for our detectors with (red) and without ROI extraction (green). And the DPM detector with and without ROI extraction are shown in brown and blue respectively.

4 EXPERIMENT RESULTS

To explore the classification performance of our detector empirically, we have captured a video sequence both on the KIT campus and in nearby areas. This sequence consists of $45,000$ images with a size of 1312×1082 pixels. Lots of objects are included, such as pedestrians, cyclists and cars. For comparison, we also trained a cascaded DPM detector as the baseline with the code provided by [FGM10]. We both test our detector and the DPM detector on a computer with 8 cores of 2.7 GHz and a memory of 8 GB. But no multi-threading is carried out. For evaluation, we use the same criteria as [FGMR10], i.e. an object is considered as detected if the detection bounding box has an overlap of more than 50% with the ground truth. The precision-recall-curves of both detectors are plotted in Fig. 8. Additionally, we list the average processing time of each detector in Table 2. The image rectification takes about 50 milliseconds and is not included in the list. Some detection examples are showed in Fig. 9 and Fig. 10.

In total, we compared the performance of four kinds of detectors: our detector without (DF+SVM) and with ROI extraction (DF+SVM+ROI), the original DPM detector and the one

Table 2: Average processing time for our detector (DF+SVM) and the DPM detector. +ROI means that ROI extraction is integrated.

	DPM	DPM+ROI	DF+SVM	DF+SVM+ROI
time (s)	2.2	0.7	0.28	0.09

integrated with ROI extraction (DPM+ROI). As can be seen, with the help of ROI extraction, both our detector and the baseline detector can achieve an up to 10% increase in precision. Because the detection only focuses on the valid image region of objects, some of the false positives outside the ROI, e.g. on the ground or in the trees (Fig. 4 (b)) can be directly removed. In the meanwhile, we obtain a factor of about 3 for both methods (DF+SVM and DPM) on computational efficiency if the ROI extraction is integrated.

Another point to be mentioned is that the precision-recall-curves of both methods intersect each other. In the high precision ranges the DPM detector achieves a higher recall value than our detector. An explanation is that part filters are utilized by the DPM detector and the final score is estimated over all individual filters. Therefore, some partially occluded or truncated objects can also be detected due to the high scores of the visible parts (Fig. 9 (a)). However, such a case is difficult for our method to handle, since only whole body detectors are available (Fig. 9 (b)). As the precision decreases, our detector can also outperform the DPM detector by a higher recall. The reason is that all the part filters of the DPM detector are applied on images with doubled resolution as the ones for the root filters. This characteristic can be a drawback in recognizing small image objects. Because their appearance are usually with low resolution and can be easily distorted by overlapped noise, the detections by part filters may fail, e.g. the small cyclist is not detected in Fig. 9 (c). In comparison, our detector performs better on this point without demanding high resolution images (Fig. 9 (d)). Considering the computational time, our detector runs at a speed of less than 0.1 second per image, while the DPM detector, even if integrated with the ROI extraction, still takes 0.7 second for one image, which is the result of utilizing part filters and high resolution images. In our case, we have a real time requirement of 10 fps, which is consistent with the frame rate of our camera, so that no delay occurs. Obviously, our detector is more advantageous for real time application.

5 CONCLUSION

The detection of vulnerable traffic participants is a key technique for autonomous driving and advanced driver assistance systems. However, while a couple of successful approaches for vision-based pedestrian detection have been developed, the state-of-the-art approaches for cyclist detection either do not provide accurate results or do not run on real time systems. In this paper we introduced a new approach for cyclist detection which yields comparable accuracy to the DPM model with a computational cost of only 0.09 second per image so that the new approach must run in real time. Two key ideas have contributed to this success, the usage of

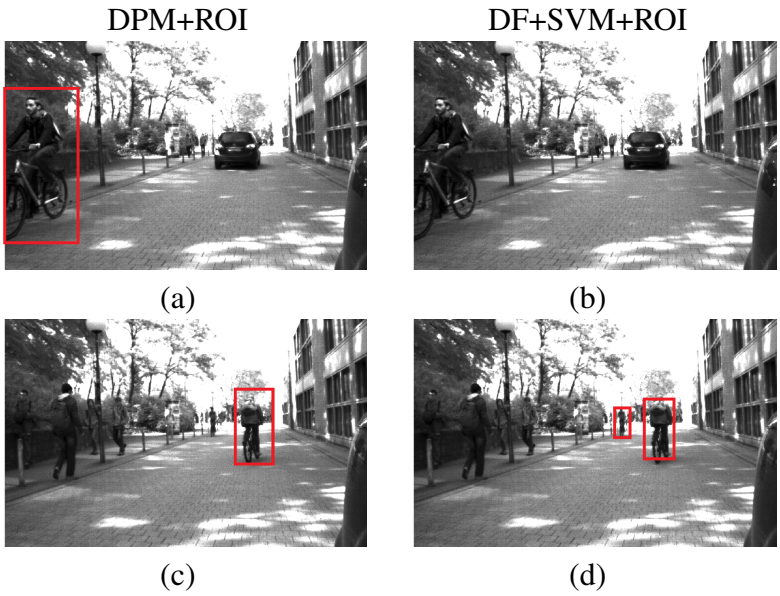


Figure 9: Examples (a) and (c) are detection results of the DPM detector and (b) and (d) are achieved with our detector. Both methods are integrated with the ROI extraction. All detected objects are presented in red rectangles.

a mixed cascaded detector that combines decision forests on the same HOG features with a final SVM decision stage, and an ROI selection technique by a geometric constraint, which is obtained by a regression method and can deal with camera rotations. Our experiments have shown that this detector is suitable for real time application in driver assistance and autonomous driving systems.

References

[BMTVG12] R. Benenson, M. Mathias, R. Timofte, and L. Van Gool, "Pedestrian detection at 100 frames per second," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2012.

[BS14] J. Beck and C. Stiller, "Non-parametric lane estimation in urban environments," in *IEEE Intelligent Vehicles Symposium*, June 2014.

[CRZ10] H. Cho, P. Rybski, and W. Zhang, "Vision-based Bicycle Detection and Tracking using a Deformable Part Model and an EKF Algorithm," in *IEEE Conference on Intelligent Transportation Systems*, July 2010.

Figure 10: Detection examples by our method integrated with the ROI extraction. All detected objects are presented in red rectangles.

[CRZ11] ——, "Vision-based 3D Bicycle Tracking using Deformable Part Model and Interacting Multiple Model Filter," in *IEEE Conference on Robotics and Automation (ICRA)*, May 2011.

[DNL02] R. Dharmaraju, D. A. Noyce, and J. D. Lehman, "An Evaluation of Technologies for Automated Detection and Classification of Pedestrians and Bicyclists," May 2002.

[Dol] P. Dollár, "Piotr's Image and Video Matlab Toolbox (PMT)." [Online]. Available: http://vision.ucsd.edu/~pdollar/toolbox/doc/index.html

[DT05] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2005.

[DWSP09] P. Dollár, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: A benchmark," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2009.

[DWSP12] ——, "Pedestrian Detection: An Evaluation of the State of the Art," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 34, 2012.

[EG10] M. Enzweiler and D. Gavrila, "Integrated pedestrian classification and orientation estimation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2010.

[FGM10] P. Felzenszwalb, R. Girshick, and D. McAllester, "Discriminatively Trained Deformable Part Models, Release 4," 2010. [Online]. Available: http://people.cs.uchicago.edu/~pff/latent-release4/

[FGMR10] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan, "Object Detection with Discriminatively Trained Part-Based Models," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 32, pp. 1627–1645, Sept 2010.

[FMR08] P. Felzenszwalb, D. McAllester, and D. Ramanan, "A discriminatively trained, multiscale, deformable part model," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2008.

[GLU12] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[GT07] T. Gandhi and M. Trivedi, "Pedestrian Protection Systems: Issues, Survey, and Challenges," *IEEE Conference on Intelligent Transportation Systems*, Sept 2007.

[KB08] J. V. Krogmeier and D. M. Bullock, "Inductive Loop Detection of Bicycles and Inductive Loop Signature Processing for Travel Time Estimation," *Statewide Wireless Communications Project*, vol. 2, 2008.

[LCX10] T. Li, X. Cao, and Y. Xu, "An effective crossing cyclist detection on a moving vehicle," in *World Congress on Intelligent Control and Automation (WCICA)*, July 2010.

[OBT14] E. Ohn-Bar and M. M. Trivedi, "Fast and Robust Object Detection Using Visual Subcategories," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[QYZ+03] Z. Qui, D. Yao, Y. Zhang, D. Ma, and X. Liu, "The study of the detection of pedestrian and bicycle using image processing," in *IEEE Conference on Intelligent Transportation Systems*, vol. 1, Oct 2003, pp. 340–345.

[RP00] S. Rogers and N. Papanikolopoulos, "Counting bicycles using computer vision," in *IEEE Conference on Intelligent Transportation Systems*, 2000.

[SL11] P. Sudowe and B. Leibe, "Efficient Use of Geometric Constraints for Sliding-Window Object Detection in Video," in *Computer Vision Systems*. Springer Berlin Heidelberg, 2011, vol. 6962, pp. 11–20.

[TKM10] K. Takahashi, Y. Kuriya, and T. Morie, “Bicycle detection using pedaling movement by spatiotemporal gabor filtering,” in *TENCON 2010 - IEEE Region 10 Conference*, Nov 2010, pp. 918–922.

[TL15] W. Tian and M. Lauer, “Fast and Robust Cyclist Detection for Monocular Camera Systems,” in *International joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP)*, March 2015.

[VJ04] P. Viola and M. Jones, “Robust Real-Time Face Detection,” *International Journal of Computer Vision*, no. 2, pp. 137–154, 2004.

[YLWL14] J. Yan, Z. Lei, L. Wen, and S. Li, “The Fastest Deformable Part Model for Object Detection,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

Algorithm 1 Train cascaded detector.

Input:

- D: Dataset of images for training.
- T: Dataset of images for validation.
- D_P: Positive samples generated from D.
- D_N: Negative samples, which are initialized by random selection from background images in D.
- n: Number of DFs of cascaded detector.
- l: Number of iterations to train SVM.
- p_0: Predefined precision.

Part (a)
    for $i = 1$ to n **do**
        Train one DF for stage i on D_P and D_N.
        Clear D_N and run current detector on D to collect false positives in D_N.
    end for
End

Part (b)
    for $i = 1$ to l **do**
        Train one SVM and add it to the detector.
        Calculate the current precision p on T.
        if $p \geq p_0$ **then**
            Break.
        else if $i < l$ **then**
            Run the detector on D to collect false positives
            and add them to D_N.
            Delete SVM from the detector.
        end if
    end for
End

Output: Cascaded detector.