Real-time pedestrian detection and pose classification on a GPU

Alexander Gepperth [1] and Michael Garcia Ortiz [1] and Bernd Heisele [2]

***Abstract*— In this contribution, we present a real-time pedestrian detection and pose classification system which makes use of the computing power of Graphical Processing Units (GPUs).**

The aim of the pose classification presented here is to determine the orientation and thus the likely future movement of the pedestrian. We focus on the evaluation of pose detection performance and show that, without resorting to complex tracking or attention mechanism, a small number of safety-relevant pedestrian poses can be reliably distinguished during live operation. Additionally, we show that detection and pose classification can share the same visual low-level features, achieving a very high frame rate at high image resolutions using only off-the-shelf hardware.

I. INTRODUCTION

This article presents a real-time system for combined pedestrian detection and pose classification, aiming at safety products in future generations of vehicles. It uses only standard computer hardware and makes massive use of the parallelization capabilities of graphics processing units (GPUs) to accelerate the system and make it real-time capable (see [21], [18]).

Applications on GPUs become more and more relevant as they are now available at a relatively low price. Even the latest generations of smartphones are equipped with this technology, allowing possible mobile applications to emerge. The fact that we can now develop these algorithms using standard hardware leaves no doubt about the possibility to apply them to embedded products. The necessary technology is now affordable and widely available.

A. Motivation

Accidents involving pedestrians in inner-city are frequently fatal, even at a relatively low driving speed. Indeed, pedestrians have no protection in case of impact, they are highly vulnerable. The goal behind pedestrian detection by intelligent vehicles is, for the most part, inspired by safety considerations: if pedestrians can be detected in time, collisions might be avoided.

The inner-city scene can be extraordinary complex, and it requires the driver to focus his attention on the parts of the scene he (subconsciously) finds relevant. This prioritization has its drawback: the driver can simply miss something. If the driver should react to the appearance or to the movement of a pedestrian with a certain behavior, but does not or is late to do so, a Driver Assistance System could warn him about the situation. However, this requires that the system is able to perceive its environment. In our case, we need to be able to reliably detect pedestrians in inner-city scenes.

Pose classification takes this consideration even further as it allows, under certain conditions, to estimate a pedestrian's next actions. For this, even a small number of pose categories may be sufficient ("front view", "back view", "facing right" and "facing left"). A reliable pose classification system can be used to focus attention on a pedestrian that might cross the road even if the pedestrian is not, at the moment, in the vehicle's path.

In this article, we will focus on pedestrian pose classification performance, as our pedestrian detection method is largely similar to methods presented in the literature (see, e.g., [8], [10], [12] for an overview). Additionally, we will show that pose classification is possible using features already computed for pedestrian detection, minimizing the use of computing resources. Numerous contributions showed the advantage of using tracking for improving the detection (see [1], [19], [9]). However, we chose to avoid using tracking at this stage of detection because even if the movement gives a strong indication about the pose of a pedestrian, it would not be beneficial in multiple inner-city scenarios. For example, if a pedestrian stands on the sidewalk near to a zebra crossing and does not move, it is not possible to estimate its facing direction with tracking.

B. Related work

The issue of pose classification has been raised by several authors (e.g. [11], [8], [14], [20], [4]), mainly in the context of road traffic and surveillance. Due to the real-time nature of our approach, we are interested in the distinction of a small number of behaviorally relevant pose *categories* (see [11], [8]) that allow a guess at a pedestrian future behavior. This is different from the determination of a precise geometric pose, i.e. the heading in a 3D space, as described in [14], [13] which is, in addition, hard to reconcile with real-time constraints. For the time being, our approach makes no use of tracking as demonstrated in [23], [20], as we want to achieve first a sufficient performance on single-frame pose classification. It is nevertheless planned to build algorithms making use of the temporal consistency of the scene on top of the current system.

C. Messages and structure of the article

This article aims to demonstrate the following things:

This work was supported by Honda Research Institute USA

[1]A. Gepperth and M. Garcia Ortiz are with ENSTA ParisTech, UIIS division, 858 Blvd des Maréchaux, 91762 Palaiseau, France. Alexander Gepperth is a member of INRIA FLOWERS. alexander.gepperth,migarcia at ensta-paristech.fr

[2]B. Heisele is with Honda Research Institute USA, Inc., 400 National Ave, Mountain View, CA 94043, USA bheisele at hra.com

- **Show the enormous potential of off-the-shelf GPUs for object detection applications.** This will be mainly shown by measuring frame rates, while verifying of course that detection and pose classification performances are equivalent to CPU-based approaches which are not real-time capable.
- **Show that pedestrian detection and pose classification can operate on the same feature basis.** This is especially important as it avoids the computation of dedicated pose classification features, again with the aim of increasing processing speed.
- **Prove that pose classification works robustly under realistic outdoor conditions.** As pose classification depends crucially on the detection method to provide candidates, it is complicated by all noise that is introduced by that detection method. It is therefore important to show that pose classification has good performance under realistic outdoor conditions.
- **Determine which pose classes can be reliably distinguished.** Some pose classes like "front view" or "back view" are visually very similar. We investigate whether performance can be increased by grouping these two classes into a single one.

To deliver these messages, the article proceeds as follows: in Sec. II, the training, evaluation and the component parts of the real-time system are described in detail. Subsequently, we will present experiments validating the previous points in Sec. III and discuss the significance of the results in Sec. IV. In Sec. V, we will conclude this contribution by providing an outlook of our future works.

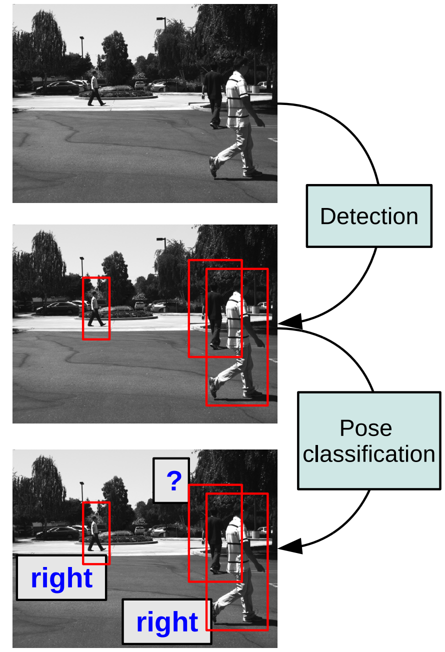

Fig. 1. Block diagram of the real-time pedestrian detection/pose classification system.

II. Methods

A. Images features for classification

All classification experiments are based on the computation of Histograms of Oriented Gradients (HOG) features using the open source OpenCV library. This technique describes localized patches of an image by counting the amount of gradient orientations in multiple directions. Adopting the terms presented in [5], we use an image size of 800x600, a cell size of 8x8 pixels, a block size of 16x16 pixels, a border of 0 pixels, and a window size of 32x64 pixels.

When computing HOG features on a single image of dimension 32x64, we obtain a HOG feature vector of 756 entries. We use the module "gpu" for all calculations and access the internal OpenCV data structures so as to get hold of the references in GPU memory. This is important since we want to perform the sliding window SVM search on the computed features without copying them to the CPU memory, which is very costly in terms of processing time.

B. Description of the real-time system

The presented system consists in a cascade of pedestrian detections followed by a pose classification, as can be seen in Fig. 1. Both pedestrian detection and pose classification operate on the same basic HOG features, and use the same classification method, namely linear and non-linear support vector machines implemented on GPU. As explained in Sec. II-A, we use the GPU based implementation contained in the OpenCV library ([2]) to speed up the feature computation process, and an own GPU implementation of the sliding-window SVM classification, where we took care that the entire feature computation and classification tool-chain is conducted in GPU memory.

1) Pedestrian detection: While common approaches use linear SVMs for detection, our approach makes use of more powerful (but slower) non-linear SVMs as well, arranging SVMs in the form of a detection cascade as outlined in Fig. 2. This allows us to circumvent the speed disadvantage of non-linear SVMs as they are only applied to the (few) detections given by the linear SVM stage. The training and application of SVMs is further described in Sec. II-C.2. We use linear and non-linear SVMs for implementing the detection cascade presented previously. We consider a detection window to contain a pedestrian if the outputs from both the linear and the nonlinear-SVM, s_{lin} and s_{nonlin} exceed their respective thresholds, θ_{lin} and θ_{nonlin}. To save computation time, we apply the non-linear SVMs only to windows for which $s_{\text{lin}} > \theta_{\text{lin}}$. There will usually still be overlapping detections: we do not perform non-maxima suppression in order to keep a maximum of correct detections to be passed to the pose classification stage.

2) Pose classification: Hypotheses who have been approved both by linear and non-linear SVMs are subjected to pose classification using a set of K pose-specific non-linear SVMs. After training, pose classification is conducted using pedestrian images provided by the real-time system. We employ a one-against-all approach (see [15]) to differentiate between pose classes, a method which has been verified to

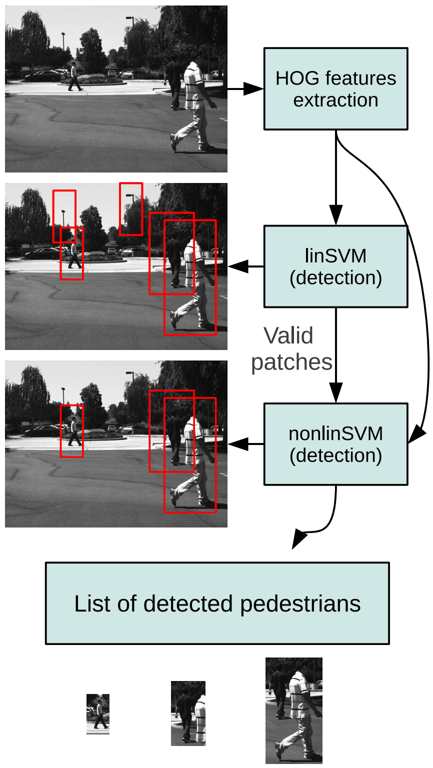

Fig. 2. Internal classifier cascade of the real-time pedestrian detection system.

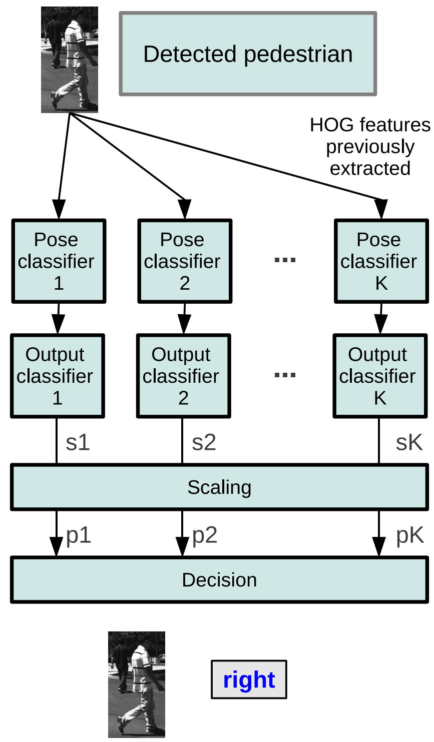

Fig. 3. Step-by-step pedestrian pose classification system.

be competitive to other approaches like pairwise one-against-one (see [16], [22]). It has the further advantage of scaling linearly with the number of classes, which facilitates training and real-time application. If we assume K pose classes, a total of K pose classifiers needs to be trained, always labeling examples of a single pose class as positive examples during training, and the remaining examples as negative ones (hence the name "one-against-all"). Platt-Scaling [17] is used to compute two sigmoid parameters $\mu_i, \kappa_i, i \in [0, P-1]$, which allows us to convert SVM outputs to approximate probabilities. For testing, given the HOG feature vector of a pedestrian of unknown pose, all K SVMs are simultaneously fed this vector, generating K pose scores $s_i (i \in [0, K-1])$, which are converted to probabilities p_i using the sigmoid parameters μ_i, κ_i as $p_i = \sigma_{\mu_i,\kappa_i}(s_i)$. We then search for the SVM with the highest output probability p_i. If it exceeds the threshold θ, we assign the corresponding pose c to the example. Else we assign a constant value which means that the system was not able to take a decision:

$$p_{\text{max}} = \max_i \ p_i \tag{1}$$

$$c = \begin{cases} \text{argmax}_i \ p_i & \text{if } p_{\text{max}} > \theta \\ -1 & \text{else} \end{cases} \tag{2}$$

$$\tag{3}$$

By varying θ, we determine how certain the system must be in order to validate a classification of the pose. The pose classification is illustrated in Fig. 3.

C. Training and evaluation

1) Data used for training: For training the linear and non-linear pedestrian detectors, we use the training set from the Daimler Monocular Pedestrian Detection Benchmark (DM-PDB, [7]), as well as from the Daimler Stereo Pedestrian Benchmark [6]. Each benchmark contains cut-out training images and annotated test videos from which images could be cropped. We do not use these images, but instead use the full training images.

To easily obtain training and test data for the pose classification system, we recorded a set of outdoor videos from a car, during daytime, on a parking lot in California. In these monochrome videos of resolution 800x600, only a single pedestrian is ever visible whose pose (belonging to one of the categories given in Sec. I) is constant throughout the video, saving the trouble of annotating poses. All object detected by the basic detector of the system are cut-out and passed as training data to the pose classifier training. We recorded a total of 52 video sequences of around 300 images containing a single pedestrian walking in front of the camera.

2) Details on SVM training: All training is performed using the libSVM library and tools [3]. We assume that training data exist as set of images for which the semantic pose category is known, all having a size of 32x64 pixels. From these images, we compute HOG features according to II-A and store the resulting feature vectors, along with suitably assigned class memberships, in a libSVM training file.

3) Evaluation of pedestrian detection: We train the detector using the Daimler Benchmark Datasets, and evaluate it on streams recorded in inner-city scenarios, as previously described. The detection thresholds are set so that the system provides no false positives on the streams we used, were only one pedestrian is present. We perform non-maximal suppression to obtain one detection per image.

We propose two simple evaluation measures to estimate the quality of the pedestrian detection. For the first one, we count all the images where a pedestrian is detected, and divide by the amount of images where a pedestrian appear.

For the second one, we start counting the detections at the first detection of a pedestrian, and finish the counting at the last detection, which gives us the amount of detected pedestrians in the range of the detection algorithm. We obtain a quantity of detected pedestrians in the range of the detection algorithm by dividing the amount of detected pedestrian in the range of the detection algorithm by the amount of images where a pedestrian appear.

4) Evaluation of pose classification : As we recorded a set of videos containing only one pedestrian for training and evaluation purposes, we can use a modified version of the N-fold cross-validation approach. We isolate one video containing one pedestrian, and train our pose classifiers on all the other video sequences. Then we use the trained system to evaluate the pose classification using the isolated video.

We repeat this procedure, which is analogous to leave-one-out cross-validation, for every pedestrian instance (video) and evaluate all the results. This procedure allows us to thoroughly test and train on the whole dataset, without testing on data which has been used to train the classifiers.

We obtain a set of classifier outputs and labels that will be used to evaluate the quality of the pose classification. In order to evaluate the pedestrian pose classifier, we use two measures to estimate the quality of the system.

On the one hand we plot the misclassifications (examples that were not classified correctly) depending on the percentage of discarded images. This percentage depends on the threshold θ as can be seen in Eqn. 1. This plot will represent the possible compromise that can be found between the quality of the classifier (a low percentage of misclassification) and the selectivity of the classifier (a high amount of discarded examples).

On the other hand, after the selection of a suitable threshold θ, we can present the results as a confusion matrix c_{ij}:

$$c_{ij} = \frac{\#(\text{pose} = i, c = j)}{\#(\text{pose} = i)}, j \neq -1. \tag{4}$$

This represents the result of the classification for individual classes. It helps us identify which classes are recognized easily, and which ones are too similar to be discerned.

III. Experiments

All experiments are conducted using the HOG feature representation (see Sec. II-A) of the databases described in Sec. II-C.1, either for training for testing purposes. Using the techniques described in II-C.2, we train the pedestrian detector cascade and $K = 3$ or $K = 4$ pose classifiers for the semantic pose categories “facing left”, “facing right”, “front/back view”. Based on the continuous outputs of the classifiers, decisions are taken as described in Sec. II-B.2 and evaluated as detailed in Sec. II-C.4.

It is noteworthy that, due to the use of GPU techniques, the combined pedestrian detection and pose classification system achieves a frame rate of 20Hz using an off-the shelf graphics card (nVIDIA GeForce FX 570) and a standard 2GHz, 4 core PC running Linux, in the conditions presented in Sec. II-A.

A. Pedestrian detection performance

Even if the pedestrian detection is not the main topic of this paper, it is important to present its performance, as the pose classification is based on images provided by this system. We do this using the parking lot-scenes presented in Sec. II-C.1 which are used to train and evaluate pose classification.

We train the detector using the Daimler Benchmark Datasets, and we evaluate its performance on streams recorded in parking lot-scenes, using evaluation measures described in Sec. II-C.3. We can then verify if our approach is portable to other technical settings, as the cameras used are not the same. Moreover, we can verify if the approach is flexible relative to the driving environment, which were not the same in the Daimler datasets and in or own datasets.

If we take into account all the images where a pedestrian appear, the detection system detects 38% of the pedestrians. However, once a maximal-suppression algorithm is used, it makes no false positive detections, which means that it does not detect a pedestrian where there is none. This detector is really selective in order to minimize the amount of incorrect detections that will be send to the pose classifier, the drawback being that it does not have a high detection rate. Another explanation for this low detection rate is that when a pedestrian is too far away, he is out of range for the detector which works on image patches of a certain size. If we start counting the detection from the first detection to the last detection, which means if we focus our evaluation on the valid range, we attain a performance of 85% pedestrians correctly detected.

We can conclude by saying that once the pedestrian enters the range of detection, it is successfully detected. Also, the range can be increased by using cameras of better quality and smaller detection windows. Given the fact that we currently perform the detection and classification at 20Hz, using standard hardware, it is reasonable to think that we can maintain a real-time capable system while improving the range of detection.

Using the pedestrian detection system, we provide the following pedestrian images to the pose classifier:

- 1665 pedestrians “facing right”
- 7258 pedestrians “front view”
- 1307 pedestrians “facing left”
- 3933 pedestrians “back view”

These images are not filtered by non-maximal suppression algorithm, so one pedestrian in one image can be detected multiple time by the SVMs and multiple example of the same pedestrian can be send to be classified by the system, in order to evaluate their pose.

B. Pose classification with 4 categories

In this experiment, we train and evaluate pose classification performance in the real-time system using the data described in Sec. II-C.1. We use $K = 4$ pose categories, namely “facing left”, “facing right”, “front view” and “back view”. Training is conducted on all of the pedestrian examples apart from one pedestrian sequence, as explained in Sec. II-C.4. The pedestrian images are extracted by the pedestrian detection module of the real-time system when running on the video sequences. The remaining examples are used to benchmark pose classification performance using the measures of Sec. II-C.4.

The effect of discarding examples can be seen in Fig. 4. With no discarding of example, we reach 71% of poses correctly classified.

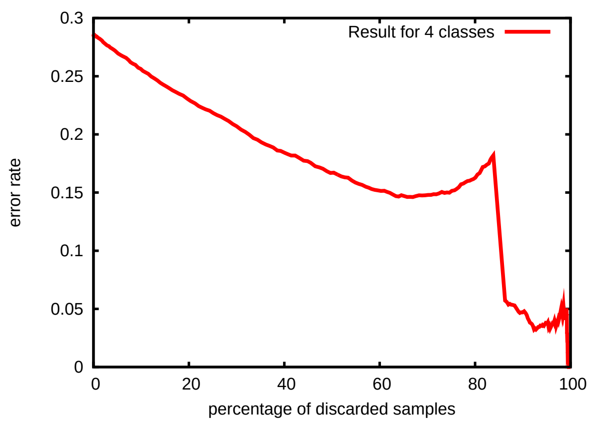

Fig. 4. Results for the pose classification using 4 classes. Overall pose classification error depending on the applied threshold θ.

In order to have a good class-by-class quality estimate, we provide the confusion matrix of the classifier in Fig. 5. In order to have a fair estimation of the quality of the system, we did not discard any unreliable prediction. All examples are classified and evaluated.

	Predicted Classes			
	60	10	23	7
Real	5	83	0	12
class	13	5	82	0
	10	37	0	53

Fig. 5. Experimental results of pose classification using 4 pose categories. From left column to right column (or first row to last row), the categories are “facing right”, “front view”, “facing left”, “back view”.

We can see that the pose classifier does not discriminate well between front and back view, because they are visually similar. Also, the main cause of misclassification for the left or the right classifiers are their geometric counterpart (respectively right and left). In the following section, we will group together the “front view” and “back view” classes.

C. Pose classification with 3 categories

In this experiment, we train and evaluate pose classification performance similarly to III-B, except that we group the “front view” and “back view” classes together. Examples are visually really similar, so it tends to merge these poses together. Additionally, if we focus on applications for safety systems, detecting the difference between a pedestrian facing front and a pedestrian facing right is not so important, because most dangerous situations come from pedestrians crossing in front of the vehicle, involving the “left” and “right” poses.

The result of the pedestrian pose classification can be seen in Fig. 6. With no discarding of example, we reach 91% of poses correctly classified. So by grouping together front and back views of the pedestrian, we improved the score of the front/back detector. We redefined our problem to improve the discriminative power of the classifiers.

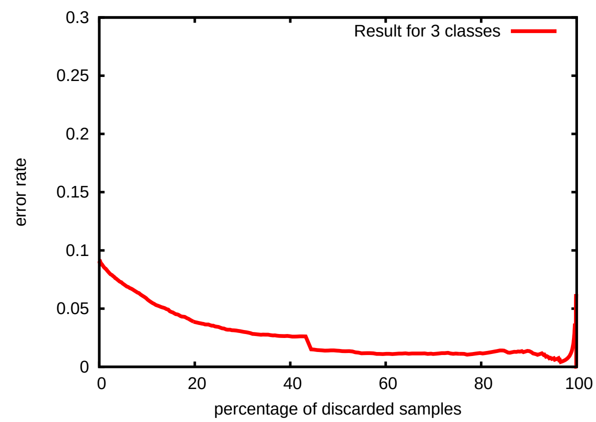

Fig. 6. Results for the pose classification using 3 classes. Overall pose classification error depending on the applied threshold θ.

In order to have a good class-by-class quality estimate, we provide once again the confusion matrix of the classifier in Fig. 7. We can once again observe the gain in quality by merging “front view” and “back view” classes.

	Predicted Classes		
	60	20	20
Real	3	97	0
class	13	5	84

Fig. 7. Experimental results of pose classification using 3 pose categories. From left column to right column (or first row to last row), the categories are “facing right”, “front or back view” and “facing left” (The results in the last row do not add up to 100 because of rounding errors).

The “front/back view” classification performs better compared to the classification with 4 poses. There are still confusions between the “facing right” and the “facing left” poses. We will discuss possible approaches to improve this result in Sec.V.

IV. Discussion

In this contribution, we presented a pedestrian detection and pose classification system implemented on a Graphical

Processing Unit (GPU). It allows the system to perform in real-time using standard hardware. The detection and classification are done using Histograms of Oriented Gradients (HOG) features fed to a cascade of linear and non-linear Support Vector Machines.

We presented a pedestrian pose classification which uses the same features as the pedestrian detection. Consequently, we reduce the computational power needed for our system. We showed that by selecting the right amount of classes, we can tremendously improve the quality of the classification. By grouping together classes that are visually similar, while maintaining the relevance of the class for safety applications, we can estimate the pose of a pedestrian based solely on their visual appearance.

Of course, these results are not final. The system does not perfectly distinguish the left and right classes yet. This can be explained by the fact that they were far less data to train the "facing right" (1665 examples) and "facing left" (1307 examples) classes compared to the data available to train the "front/back view" (11191 examples) class. Another reason explaining why the results can be improved is the fact that we do not perform non-maxima suppression after the detection stage, so even detections with low confidence are used for training and testing the pose classification.

V. Future Works

Numerous possibilities exist to improve the current system and to use it for Advanced Driving Assistance System applications. In this last section, we will propose several implementations that we plan to incorporate in our system in the following months.

First, it is obvious that we have to make use of the temporal consistency of the data to improve the detection. On the one hand, if a pedestrian is tracked in a video sequence, his pose can also be tracked. Moreover, if he is moving, the direction of the pedestrian can be a good estimate of his pose. Not only can it be used to improve the pose classification, but it can also help to generate additional training data. Indeed, the detection system currently misses 15% of the pedestrians within the range of detection, which can be recovered using tracking. On the other hand, if a pedestrian is detected and his pose is estimated, this pose can serve as a prior for the next detection. One of our point of interest is to explore if pedestrian pose classification and tracking can benefit from each other.

Secondly, we want to explore the possibility of pedestrian behavior prediction using the pose estimation (improved by a possible tracking) and other scene elements. For example, the position and the pose of a pedestrian relative to the road holds a lot of information about what he will possibly do in the near future. Related to this possible orientation of our work, we plan to explore the possible optimization of our system exploiting the fact that pedestrian can only be found at certain locations in the driving scene.

Finally, we plan to apply our detection and pose estimation to other traffic participants. We want to estimate the scalability of our approach to multiple detections (pedestrians, bicyclists and cars), in multiple driving environments (inner-city, rural road and highway), using different detection devices.

References

[1] M. Andriluka, S. Roth, and B. Schiele. People-tracking-by-detection and people-detection-by-tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[2] G. Bradski and A. Kaehler. *Learning OpenCV: Computer vision with the OpenCV library*. O'Reilly Media, Incorporated, 2008.

[3] C. Chang and C. Lin. Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.

[4] R. Cucchiara, C. Grana, A. Prati, and R. Vezzani. Probabilistic posture classification for human-behavior analysis. *IEEE Transactions on Systems, Man and Cybernetics, Part A*, 35(1):42–54, 2005.

[5] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 886–893. IEEE, 2005.

[6] M. Enzweiler, A. Eigenstetter, B. Schiele, and D. Gavrila. Multi-cue pedestrian classification with partial occlusion handling. pages 990–997. IEEE, 2010.

[7] M. Enzweiler and D. Gavrila. Monocular pedestrian detection: Survey and experiments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(12):2179–2195, 2009.

[8] M. Enzweiler and D. Gavrila. Integrated pedestrian classification and orientation estimation. In *Computer Vision and Pattern Recognition (CVPR)*, pages 982–989. IEEE, 2010.

[9] L. S. et al. Multiclass multimodal detection and tracking in urban environments. *Journal of Robotics Research*, 2010.

[10] T. Gandhi and M. Trivedi. Pedestrian protection systems: Issues, survey, and challenges. In *Intelligent Transportation Systems (IV)*, volume 8, pages 413–430. IEEE, 2007.

[11] T. Gandhi and M. Trivedi. Image based estimation of pedestrian orientation for improving path prediction. In *Intelligent Vehicles Symposium (IV)*, pages 506–511. IEEE, 2008.

[12] D. Geronimo, A. Lopez, A. Sappa, and T. Graf. Survey of pedestrian detection for advanced driver assistance systems. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(7):1239–1258, 2010.

[13] M. Hofmann and D. Gavrila. Multi-view 3d human pose estimation in complex environment. *International journal of computer vision*, 96(1):103–124, 2012.

[14] B. Leibe, N. Cornelis, K. Cornelis, and L. Van Gool. Dynamic 3d scene analysis from a moving vehicle. In *Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2007.

[15] J. Milgram, M. Cheriet, and R. Sabourin. Estimating accurate multi-class probabilities with support vector machines. In *International Joint Conference on Neural Networks (IJCNN)*, volume 3, pages 1906–1911. IEEE, 2005.

[16] J. Milgram, M. Cheriet, R. Sabourin, et al. "one against one" or "one against all": Which one is better for handwriting recognition with svms? In *Tenth International Workshop on Frontiers in Handwriting Recognition*, 2006.

[17] J. Platt et al. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74, 1999.

[18] V. Prisacariu and I. Reid. fasthog - a real-time gpu implementation of hog. Technical Report 2310/09, Department of Engineering Science, Oxford University.

[19] K. Schindler, A. Ess, B. Leibe, and L. V. Gool. Automatic detection and tracking of pedestrians from a moving stereo rig. *ISPRS Journal of Photogrammetry and Remote Sensing*, 2010.

[20] H. Shimizu and T. Poggio. Direction estimation of pedestrian from multiple still images. In *Intelligent Vehicles Symposium (IV)*, pages 596–600. IEEE, 2004.

[21] C. Wojek, G. Dorkó, A. Schulz, and B. Schiele. Sliding-windows for rapid object class localization: A parallel technique. In *DAGM-Symposium*, pages 71–81, 2008.

[22] T. Wu, C. Lin, and R. Weng. Probability estimates for multi-class classification by pairwise coupling. *The Journal of Machine Learning Research*, 5:975–1005, 2004.

[23] L. Zhang, B. Wu, and R. Nevatia. Detection and tracking of multiple humans with extensive pose articulation. In *International Conference on Computer Vision (ICCV)*, pages 1–8. IEEE, 2007.