

# Learning Appearance in Virtual Scenarios for Pedestrian Detection

Javier Marín, David Vázquez, David Gerónimo and Antonio M. López

Computer Vision Center and Computer Science Dpt. UAB, 08193 Bellaterra, Barcelona, Spain

{jmarin, dvazquez, dgeronimo, antonio}@cvc.uab.es

## Abstract

Detecting pedestrians in images is a key functionality to avoid vehicle-to-pedestrian collisions. The most promising detectors rely on appearance-based pedestrian classifiers trained with labelled samples. This paper addresses the following question: can a pedestrian appearance model learnt in virtual scenarios work successfully for pedestrian detection in real images? (Fig. 1). Our experiments suggest a positive answer, which is a new and relevant conclusion for research in pedestrian detection. More specifically, we record training sequences in virtual scenarios and then appearance-based pedestrian classifiers are learnt using HOG and linear SVM. We test such classifiers in a publicly available dataset provided by Daimler AG for pedestrian detection benchmarking. This dataset contains real world images acquired from a moving car. The obtained result is compared with the one given by a classifier learnt using samples coming from real images. The comparison reveals that, although virtual samples were not specially selected, both virtual and real based training give rise to classifiers of similar performance.

## 1. Introduction

Advanced driver assistance systems (ADAS) aim to improve traffic safety by providing warnings and performing counteractive measures in dangerous situations. Pedestrian protection systems are specialized in vehicle-to-pedestrian collisions. They consist in vehicles equipped with a forward facing image acquisition and processing system able to detect pedestrians on the road. Accordingly, research on image-based pedestrian detection for this task has been a very relevant topic for the Computer Vision community [8, 10, 11]. The challenge lies in the fact that pedestrians are very difficult to detect: they are articulated and imaged from a mobile platform in cluttered scenarios and present high variability in clothes, pose, distance to the camera, background and outdoor illumination. Moreover, the nature of the addressed application requires real-time and a demanding tradeoff between misdetections and false alarms.

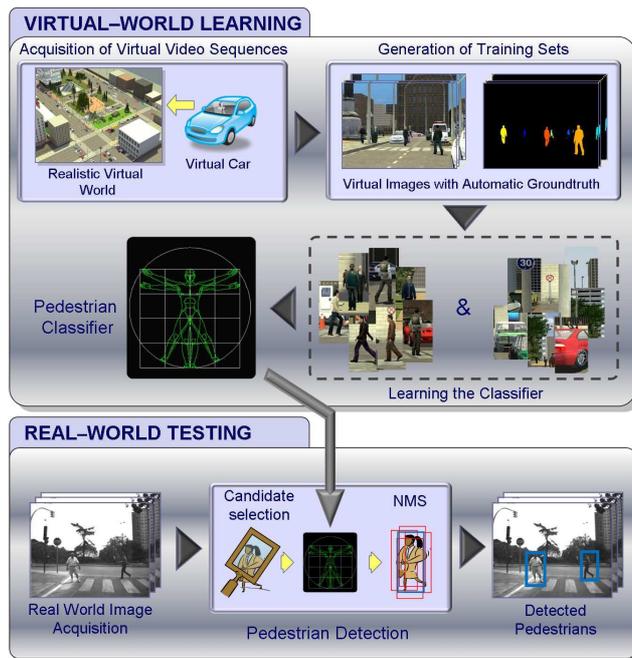


Figure 1. Can a pedestrian appearance model learnt in virtual scenarios work successfully for pedestrian detection in real images?

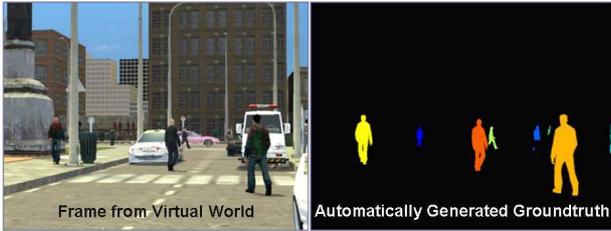


Figure 2. Virtual image with corresponding automatically generated pixel-wise groundtruth for pedestrians.

large number of labelled examples is expensive in the sense that many video sequences must be recorded on-board and a large amount of manual intervention is required. Moreover, just subjectively adding more examples does not guarantee higher variability, *i.e.*, it can happen that we are just adding pedestrians too similar to the ones we already had.

In order to face this problem, Broggi *et al.* [3] use synthesized examples for pedestrian detection in far infrared images, *i.e.*, images capturing relative temperature. In particular, a rough 3D pedestrian model encoding the morphology of a person is captured from different poses and viewpoints. The background is just roughly modelled since it is mainly dark in the used images. Each combination of pose and viewpoint constitute a kind of grayscale template of *human relative temperature*. Then, instead of following a learning-by-examples approach to obtain a single model (classifier), a set of templates is used by a posterior pedestrian detection process based on template matching. However, the authors admit poor results, since it is difficult to handle variability due to different clothes, person size, more complex background and, in addition, computational time increases with the number of templates to be considered.

Enzweiler *et al.* [7] enlarge the set of examples by transforming the shape of pedestrians (labelled in real images) as well as the texture of pedestrians and background. The pedestrian classifier is learnt by using a discriminative approach (NNs with LRFs and Haar features with SVM are tested). Since these transformations encode a generative model, the overall approach is seen as a generative-discriminative learning paradigm. The generative-discriminative cycle is iterated several times in a way that new synthesized examples are added in each iteration by following a probabilistic selective sampling to avoid redundancy in the training set. The reported results show that this procedure provides classifiers of the same performance than when increasing the number of training examples with new manually labelled ones. However, the authors show that much of the improvement comes from enlarging the training set by applying jittering to the pedestrian examples as well as by introducing more counterexamples. Notice that jittering does not involve synthesizing

pedestrians since it only requires shifting them inside their framing window, *i.e.*, it is introduced to gain certain degree of shift invariance in the learnt classifiers. Besides, for applying the different proposed transformations the overall pedestrian silhouette must be traced, which requires a manual labelling much more labour intensive than standard bounding box framing of pedestrians. Examples mirroring is also used in all cases to induce invariance against such a geometrical transformation. Certainly, jittering and mirroring are recommendable for enlarging the training set.

The reviewed proposals are appealing because if we are able to use a set of automatically generated samples for learning, then we would have an easier control of its variability and cardinality, avoiding human labelling for the learning phase (but not for testing). However, rather than using rough morphological models or synthesized real examples, we propose to explore the synergies between modern Computer Graphics and Computer Vision in order to *close the circle*: the Computer Graphics community has modelled real world by building increasingly realistic virtual worlds (*e.g.* video games). Thus, can we now learn our models of interest in such virtual worlds and use them successfully back in real world? In this paper we focus this question on the visual appearance of pedestrians. In particular, we want to learn such appearance using virtual samples in order to detect pedestrians in real images (Fig. 1).

The experiments we conduct here suggest a positive answer to the previous question, which we think is a new and relevant result for research in pedestrian detection. In particular, we record training sequences in realistic virtual cities (Fig. 2) and train appearance-based pedestrian classifiers using HOG and linear SVM, a baseline method for building such classifiers that remains competitive for pedestrian detection in the ADAS context [6, 8]. We test such classifiers in a dataset for pedestrian detection benchmarking that was recently made publicly available by Daimler AG [8]. The obtained results are evaluated in a per-image basis and compared with the classifier obtained when using real samples for training. The comparison reveals that virtual and real based training give rise to similar classifiers. Furthermore, given that at this time we do not fine tune virtual training sets, the obtained outcome opens the possibility of a more custom design of these sets to obtain better classifiers, *e.g.*, following active learning approaches as proposed in [7, 9]. In addition, virtual based training can be also used for appearance-based pedestrian pose recovery as it was done in [16, 1] (assuming human detection before pose recovery).

The remainder of the paper is organized as follows. Section 2 introduces the datasets used for training and testing. Section 3 details the conducted experiments while Sect. 4 presents the results and corresponding analysis. Finally, Sect. 5 summarizes the conclusions and future work.



Figure 3. Examples and counterexamples taken from real images (Daimler’s dataset) and from virtual ones.

## 2. Datasets

### 2.1. Real images

The lack of publicly available large datasets for pedestrian detection in the ADAS context has been a recurrent problem for years [6, 8, 11]. For instance, INRIA dataset [5] has been the most widely used for pedestrian detection. However, it contains photographic pictures in which people are mainly close to the camera and in focus. Moreover, there are backgrounds that do not correspond to urban scenarios, which are the most interesting and difficult ones for detecting pedestrians from a vehicle.

Fortunately, two more adapted datasets for the ADAS context have recently been made publicly available. One of them is presented by Caltech [6] and the other one by Daimler AG [8]. In the future we plan to work with both, but to start our study we decided to use the Daimler’s dataset since it comes from one of the most relevant automotive companies worldwide, thus, we can expect the images to be quite representative for ADAS. Next, we summarize the main characteristics of Daimler’s dataset. In fact, it consists of a training set and a different testing set.

#### 2.1.1 Training set

The images of this set are grayscale and were acquired at different times of day and locations (Fig. 3).

**Examples.** The original training frames with pedestrians are not publicly available, but cropped pedestrians are. From 3915 manually labelled pedestrians, 15660 were obtained by applying small vertical and horizontal random shifts (*i.e.*, jittering) and mirroring, and then put publicly available. The size of each cropped example is  $48 \times 96$  pixels, which comes from the  $24 \times 72$  pixels of the contained pedestrian plus an additional margin of 12 pixels per side. All the original labelled pedestrians are at least 72 pixels high, thus, some of the samples come from downscaling

but none from upscaling. All the samples contain pedestrians that are upright and not occluded.

**Counterexamples.** 6744 pedestrian-free frames were delivered. Their resolution is  $640 \times 480$  pixels. Thus, to gather cropped counterexamples these frames must be sampled. Conceptually, the sampling process we use can be thought as follows. We need counterexamples of the same dimensions than the cropped pedestrian examples, *i.e.*,  $48 \times 96$  pixels. Therefore, we can select windows of size  $48k^i \times 96k^i$  pixels, where  $k$  is the scale step (1.2 in our case) and  $i \in \{0, 1, 2, \dots\}$ , provided that they are fully contained in the image we are sampling. Then, we can downscale the counterexamples by a factor  $k^i$  using, for instance, bi-cubic interpolation. In practice, we implement this sampling idea by using a pyramid of the frame to be sampled and then by cropping windows of size  $48 \times 96$  pixels at each layer [4], which is closely related to the scanning strategy used by the final pedestrian detector (Sect. 3.1).

#### 2.1.2 Testing set

The testing set consists in a sequence of 21790 grayscale frames of  $640 \times 480$  pixels. The sequence was acquired on-board while driving during 27 minutes through urban scenarios. Moreover, this testing set does not overlap the training set. The testing set includes 56492 manually labelled pedestrians. The labels contain also an additional information indicating whether they are of *mandatory* detection or not. Basically, the pedestrians labelled as non-mandatory are those either occluded, not upright, or smaller than 72 pixels high. There are 2459 mandatory pedestrians in total. Frames of the training set can be seen with overlaid results in Sect. 4 (Fig. 5).

## 2.2. Virtual images

In order to obtain virtual images, the first step consists in building virtual scenarios. We have done it by using the video game Half-Life 2 [15]. This game allows to include

maps created with an editor named *Hammer*, as well as to add modifications (*a.k.a. mods*).

We use *Hammer* to create realistic virtual cities with roads, streets, buildings, traffic signs, vehicles, pedestrians, different illumination conditions, etc. Once we start to *play*, the pedestrians and vehicles move through the virtual city but respecting physical laws (*e.g.*, pedestrians do not float and cannot be at the same place than other solid objects at the same moment) as well as by following their artificial intelligence (*e.g.*, vehicles move on the road).

In order to acquire images in virtual scenarios we use the *mod* created by the company ObjectVideo. Taylor *et al.* [17] show the usefulness of such a *mod* for designing and validating people tracking algorithms for video surveillance (static camera). A relevant functionality consists in providing pixel-wise groundtruth for human targets (Fig. 2). However, since the aim in [17] is to test algorithms under controlled conditions, all the work is done with virtual scenarios, without considering real world images. Thus, the work we present in this paper is not actually related to [17] except for the use of the same *Half-Life 2 mod*.

In fact, we created an application to augment the functionalities of such a *mod* with the possibility of moving a virtual camera as if we were driving. In particular, in order to *drive* through a virtual city we introduced a camera with a given height as well as pitch, roll and yaw angles, and then we move it keeping these parameters constant. The only constraint that must be ensured is that these parameters are compatible with a camera forward facing the road from inside a vehicle, for instance, as if it were placed at the rear view mirror behind the windshield. Finally, in order to emulate the dataset of Daimler, we set the resolution of our virtual camera to  $640 \times 480$  pixels.

We created four virtual cities which, in fact, correspond to a single one in terms of graphical primitives, *i.e.*, we only changed some building textures so that they look different as well as the overall illumination to emulate different times of day. In order to introduce pedestrians in these cities, we use the 18 virtual people and 19 sets of clothes directly available from *Half-Life 2*. Only 81 combinations person-clothes are possible, so this is the number of *different* pedestrians available in our virtual scenarios. However, note that since they are articulated moving models seen from a moving camera, each virtual pedestrian can be imaged with different poses and backgrounds. Figure 3 plots samples of the virtual training set that we describe in the rest of this section.

**Examples.** We recorded five video sequences by driving through the virtual cities. In total we obtained 26046 frames. The virtual car was driven without any preferred *plan of route*. Along the way we captured images containing pedestrians in different poses and with different back-

grounds. Since we can obtain the groundtruth of the virtual pedestrians automatically, we consider only those upright, non-occluded, and with a height equal or larger than 72 pixels in the captured images (pedestrians *taller* than 72 pixels require further down scaling as we will see) like in the training set of Daimler. This gives us 7973 pedestrians to consider in order to construct the set of examples for training. It is worth mentioning that for having automatically labelled examples analogous to the manually labelled ones of Daimler’s training set (*i.e.*, with the torso centered with respect to the horizontal axis), we cannot just take the bounding boxes corresponding to the pixel-wise groundtruth. Instead, we apply the following process to each virtual pedestrian:

1. For some pedestrian poses, the bounding box obtained from the pixel-wise groundtruth is such that the torso is not well centered in the horizontal axis, so we automatically correct this. More specifically, we project the pedestrian groundtruth into its horizontal axis. Then we take the location of the maximum of the projection as the horizontal center of the torso. Finally, we shift the initial pixel-wise bounding box so that its horizontal center matches the one of the torso.
2. Then, we modify the location of the sides of the pedestrian bounding box preserving the previous re-centering, but enforcing the same aspect ratio and proportional background margins than the pedestrians in the training set of Daimler (*i.e.*,  $24/72$  and  $12/72$ , respectively). This is automatically achieved by simply applying standard rule of proportionality.
3. The bounding box at this point can still be larger than the canonical bounding box of the pedestrian examples of Daimler’s training set, *i.e.*, larger than  $48 \times 96$  pixels. Thus, the final step consists in performing a down scaling using bi-cubic interpolation.

**Counterexamples.** In order to collect the counterexamples for training, we used the same four virtual cities than for obtaining the examples, but now without pedestrians inside, *i.e.*, we drove through these *uninhabited* cities to collect pedestrian-free video sequences. We collect frames from these sequences in a random manner but assuring a minimum distance of 5 frames between any two selected frames, which is a simple way to increase variability. In fact, it is also possible to use pedestrian-free scenarios created by *Half-Life 2* sympathizers and made publicly available through the internet. Thus, we collected some more frames from them to augment the number of counterexamples in our training set. In total we have 2049 frames without virtual pedestrians, so they can be sampled to gather virtual counterexamples. The sampling process is, of course, the same than the one previously described for Daimler (Sect. 2.1.1).

Table 1. Training and testing settings for our experiments. Virtual datasets can be found at [www.cvc.uab.es/adas](http://www.cvc.uab.es/adas).

	Training Set	Training process	Testing sets
	Cropped pedestrians (jitter and mirroring included) & Background frames	1st round: cropped pedestrians / cropped background & Bootstrapping: additional cropped background	
Daimler	15660 & 6744	15660 / 15560 & 15660	Full set: 21790 frames
Virtual	3200 & 2049	3200 / 15560 & 15660	Mandatory set: 974 frames

### 3. Experiment design

#### 3.1. Pedestrian detector components

In order to detect pedestrians we need a pedestrian classifier learnt from the training set by using specific *features* and a *learning machine*. With this classifier we *scan a given image* looking for pedestrians. Since multiple detections can be produced by a single pedestrian, we also need a mechanism to *select the best detection*. The procedures we use for features extraction, machine learning, scanning the images, as well as selecting the best detection from a cluster of them, are the same no matter if the classifier was learnt using virtual images or real ones (*i.e.*, from Daimler). Let us briefly review which are these components in our case.

**Features and learning machine.** The combination of the histograms of oriented gradients (HOG) features and linear SVM learning machine, proposed by Dalal *et al.* in [5], has been proven as a competitive method to detect pedestrians in the ADAS context [8]. Similar conclusions are also obtained when using a large ADAS-inspired dataset for testing in [6]. In fact, recent proposals that outperform HOG/linear-SVM when using the INRIA dataset include both HOG and linear SVM as core ingredients [18]. Thus, we think that HOG/linear-SVM stands as a relevant baseline method for learning pedestrian classifiers, so we use it in our experiments. In particular, we follow the settings suggested in [5] for both HOG and linear SVM, as it is also done in [8]. A minor difference comes from the fact that in Daimler’s datasets the images are grayscale while the virtual images are RGB. This issue is easily handled by just taking at each pixel the gradient orientation corresponding to the maximum gradient magnitude among the RGB channels (as in [5] for INRIA dataset).

**Scanning strategy.** As in [8] we use the widely extended *sliding window* strategy implemented through a pyramid to handle different detection scales [4]. We could consider the sliding window parameters found in [8] as the best in terms of pedestrian detection performance for the so-called *generic pedestrian detection* case with Daimler’s testing set. However, it turns out that a single experiment with such parameters takes about a month with our current computational resources. As will be seen, since we run several experiments we would need several months to obtain the results. In short, the reason is that with such settings there

are much more windows and it is difficult to share HOG features among different testing windows. Accordingly, at this stage of our research we decided to follow the settings proposed in [4].

**Selecting the best detection.** In order to group multiple overlapped detections and (ideally) provide one single detection per pedestrian we follow an iterative confidence- and overlapping- based approach, *i.e.* a kind of *non-maximum-suppression*. This technique, used by I. Laptev in [13], consists of four basic steps: 1) create a new cluster with the detection of highest confidence; 2) recompute the cluster with the mean of the detections overlapping the new cluster; 3) iterate to step 2 until the cluster position does not change; 4) delete the detections contained in the cluster and iterate to 1 while there are detections.

**Pedestrian detector.** For us a *pedestrian detector* consists of a pedestrian classifier, plus the above seen techniques of sliding window and non-maximum-suppression. Therefore, we are not considering tracking of pedestrians, but we think this does not affect the aim of this paper.

#### 3.2. Training

##### 3.2.1 Training with Daimler dataset

We train the HOG/linear-SVM classifier with the 15660 provided examples and collect also 15560 counterexamples by sampling the 6744 provided pedestrian-free images as explained in Sect. 2.1.1, which is the approach followed in [8]. In addition, we also apply one *bootstrapping* step, *i.e.*, with the first learnt classifier we run the corresponding pedestrian detector on the 6744 pedestrian-free frames and collect false positives to enlarge the number of counterexamples and retrain. This technique is known to provide better classifiers [8, 14, 5]. In this case, again following [8], 15660 new counterexamples are collected during bootstrapping. Thus, the final classifier is trained using 15660 examples and 31220 counterexamples.

##### 3.2.2 Training with virtual dataset

Learning a classifier by using the virtual training set is basically analogous to the Daimler case, *i.e.*, HOG/linear-SVM and one bootstrapping are used. Thus, let us just note some differences. In this case we have 7973 examples, but we

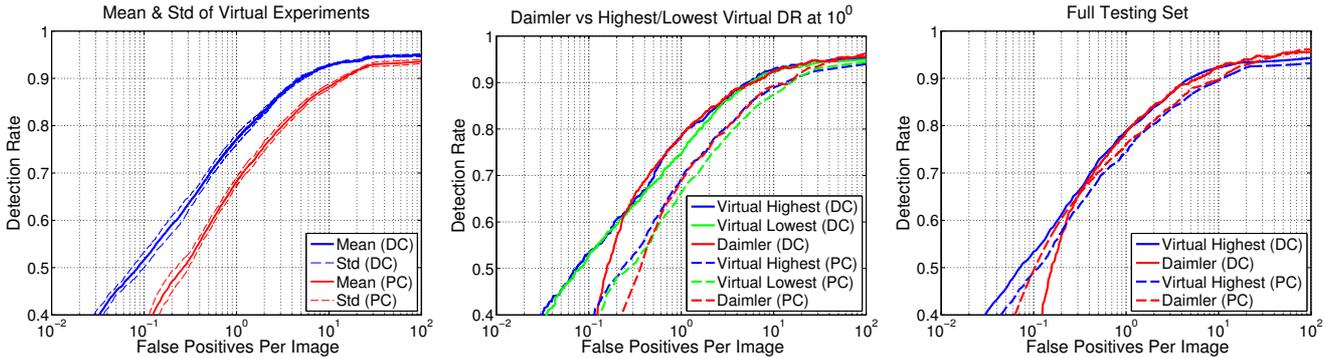


Figure 4. Per-image evaluation of the pedestrian detectors. DC stands for *Daimler criteria* and PC for *PASCAL criteria*. Left: mean performance and standard deviation obtained in the 10 experiments with virtual samples (mandatory testing set). Middle: curves with highest and lowest values for FPPI= $10^0$  of the training with virtual samples, and curve obtained with the training based on Daimler’s training set (mandatory set). Right: performance of virtual (highest at  $10^0$  FPPI) and Daimler trainings for the full Daimler’s testing set.

want to assess how robust the training process is to changes in this set. Therefore, rather than taking all the examples to compute a single training we perform a partition of this set into 10 subsets with the aim of conducting 10 trainings.

In order to ensure a certain variability on the subsets of examples, the partition is done so that examples of a given subset either come from different person-clothes combinations or are more than 10 frames apart (differences come from pose and background). To be more precise, since these conditions are not always achievable and in order to have 800 examples per subset, we allow the inclusion of some pedestrians in more than one subset. Altogether, the overlapping between two subsets is still less than 3%.

Given a subset of 800 virtual pedestrians, in order to emulate the content of Daimler training set, we apply two jitters per pedestrian plus a mirroring per jitter, which makes each subset end up with 3200 examples. As in the case of Daimler, we gather 15560 counterexamples from the virtual pedestrian-free training frames, and after learning the initial classifier we collect 15660 more using bootstrapping. Finally, we retrain with 800 examples and 31220 counterexamples per subset.

### 3.3. Testing

To reduce the computational time per experiment, rather than always using the 21790 testing frames from Daimler, we also rely on a representative but reduced testing set. In particular, we select first only those frames where there is at least a mandatory pedestrian to detect (2.1.2), then we take one out of each two frames. We term the final set of frames as *mandatory testing set*. There are 974 of such frames and they contain 1193 mandatory pedestrians.

We use the mandatory testing set for evaluating the pedestrian detector associated to the classifier learnt with Daimler’s training set and the same for the 10 detectors related to the virtual training set. Finally, we select the best

Table 2. Similarity detection overview. Common results between virtual and Daimler pedestrian detectors are shown, over Daimler and over virtual detectors, for Daimler and PASCAL criteria.

Daimler Criteria	TP(%)	FP(%)	FN(%)
Over Daimler	90.91	33.62	73.61
Over virtual	91.30	36.28	71.22
PASCAL Criteria	TP(%)	FP(%)	FN(%)
Over Daimler	90.30	45.77	70.65
Over virtual	86.61	32.55	82.28

virtual based detector and then we run a *full test*, *i.e.*, the 21790 testing frames from Daimler’s testing set are processed. We do the same full test for the pedestrian detector based on Daimler’s training set.

## 4. Results

Table 1 summarizes the settings explained in Sect. 3. In this section we draw the obtained results and discuss them. We assess the similarity of virtual and real world-based training, both in terms of the performance of the corresponding pedestrian detectors and the matching of the specific detection results.

In order to assess the performance of the different pedestrian detectors we compute *per-image evaluation* (highly recommended in [6]). Since we are using Daimler’s datasets for testing, we follow the evaluation settings proposed by Enzweiler *et al.* in [8], who also employ per-image evaluation. In particular, we plot curves depicting the tradeoff between detection rate (*i.e.*, the percentage of mandatory pedestrians that are actually detected) and the number of false positives per image (FPPI) in logarithmic scale. The pedestrian detectors output detection windows, let  $W_d$  be one of them, and let  $W_l$  be a window labelled as mandatory pedestrian. Then, we define the ratio of areas  $r(W_d, W_l) = a(W_d \cap W_l)/a(W_d \cup W_l)$ . If there is a  $W_l$  for which



Figure 5. Qualitative results at  $10^0$  FPPI taken when following Daimler criteria. Top row: using the pedestrian detector based on Daimler’s training set. Bottom row: using the pedestrian detector corresponding to the training with virtual samples, in particular with the classifier of highest detection rate at  $10^0$  FPPI. Blue bounding boxes are right detections, green ones are false positives and red ones misdetections.

$r(W_d, W_l) > 0.25$ , then  $W_d$  is considered a true positive, otherwise,  $W_d$  is a false positive. Undetected mandatory pedestrians count as false negative, *i.e.*, those  $W_l$  for which there is no  $W_d$  with  $r(W_d, W_l) > 0.25$ . If given a  $W_l$  more than one  $W_d$  passes the true positive criterion (*i.e.*, multiple detections), only one of them is considered, the rest are ignored. We also perform the same experiments using PASCAL VOC criteria:  $r(W_d, W_l) > 0.5$  to accept  $W_d$  as true positive, and in case of multiple detections one counts as true positive and the rest as false positives. Note that such different criteria also affect training because of the bootstrapping. The overlapping threshold of the step *selecting-the-best-detection-2* (Sect. 3.1) is set to 0.25 when following Daimler criteria and to 0.5 for PASCAL criteria.

Figure 4 shows the obtained performance curves following both Daimler and PASCAL criteria. The standard deviation of the 10-experiments-based curves of the virtual based training reveal that the different sets of virtual examples (pedestrians) give rise to rather similar performance. For instance, a point of interest for ADAS applications is  $FPPI=10^0$ , since with one false positive per image in average, a further method doing some sort of temporal analysis has chances of discarding such isolated detections. At that point the standard deviation of the experiments is  $\leq 2\%$  of detection rate. The following plot comparing virtual and real world-based testing also reveals quite similar curves. The difference between the best virtual-world-based curve and the real-world-based one at  $FPPI=10^0$  is  $\leq 1\%$ , and comparing the worst virtual-world-based curve and the real-world-based one such difference is still  $\leq 4\%$ . Note that, although these results are for the mandatory testing set, they also extrapolate when using the full training set (27 min-

utes of video), as can be seen in the last plot. Altogether, the results allow us to conclude that the differences of the learnt virtual world based classifiers and the real world one are minimal in terms of performance.

Figure 5 shows some qualitative results from which we can see how similar are the particular detections coming from virtual and real world learning. These are just a small sample of the qualitative results, but we also perform a quantitative assessment. In order to decide whether a detection,  $W_v$ , coming from the virtual-world-based pedestrian detector and another, say  $W_r$ , coming from the real-world-based one are equivalent we need an overlapping criterion. Like for performance evaluation, we use the criterion  $r(W_v, W_r) > t$  to accept that these are analogous detections, with  $t = 0.25$  when applying Daimler criteria and  $t = 0.5$  for PASCAL criteria. The statistics of Table 2 suggest that virtual and real-world-based pedestrian detectors basically identify the same pedestrians, and also that there is a large number of matches regarding the misdetections. However, both detectors are quite different with respect to the false positives, *i.e.*, they are confused by different background clutter. These results are coherent with the fact that virtual and real pedestrians are similar to the *eyes* of the training, while background is different since we did not *copy* in the virtual world the backgrounds of the testing set and this is quite a more heterogeneous class.

Hence, we think that there is a high correlation regarding the performance of virtual and real world based training as well as the matching of the corresponding detection results. We argue that in part this is due to the high realism of modern computer graphics as well as what we could call *world invariance* of the HOG features. This invariance is

in fact just the consequence of being robust to illumination changes (gradient orientation is invariant under monotonic grayscale changes), which is mandatory for ADAS applications. Thus, other features with similar invariant properties (e.g., normalized Haar features or LBP) are also very likely to reach such *world invariance*, so their performance is also an interesting topic to research.

Enzweiler *et al.* [8] argue that the size and complexity of Daimler's testing set allows to draw meaningful conclusions. Therefore, we think that our current conclusions are trustworthy. Thus, coming back to the question that opened this research, *i.e.*, *can we learn a pedestrian appearance model in virtual worlds and then use it successfully back in real world for pedestrian detection?*, we answer *yes we can*.

## 5. Conclusions

In this paper we have explored how realistic virtual worlds can help in learning appearance-based models for pedestrian detection in the ADAS area. We have used the HOG/linear-SVM technique to learn a pedestrian classifier using only samples from virtual worlds. We have *plugged-in* such classifier in a standard pedestrian detection method and have evaluated how this detector works when applied to real images, *i.e.*, when the pedestrian classifier is used *out of its world*. The same procedure has been followed to obtain an analogous pedestrian detector that only differs from the virtual-world-based one in the sense that the *plugged-in* classifier was trained using real images (*i.e.*, manually labelled pedestrians). Comparison between virtual and real world based pedestrian detectors reveals a rather close performance. The size and complexity of Daimler's testing set allows for this conclusion to be reliable. Therefore, we think that the results presented in this paper are new and relevant for research in pedestrian detection. However, to provide a totally definitive conclusion we must test with more databases, more features and more learning machines. These experiments are part of our future work. Moreover, since we did not fine tune virtual training sets, we plan to do a more custom design of them to obtain better classifiers, *e.g.*, following active learning approaches. In addition, the detection of other targets (*e.g.* vehicles) can also be evaluated under the proposed framework.

## Acknowledgments

This work was supported by the Spanish Government (projects TRA2007-62526/AUT and Consolider Ingenio 2010: MIPRCV (CSD200700018)), Javier Marín's Grant BES-2008-007582, and by the Catalan Generalitat (project CTP-2008ITT00001).

## References

[1] A. Agarwal and B. Triggs. A local basis representation for estimating human pose from cluttered images. In *Asian Conf.*

*on Computer Vision*, pages 50–59, 2006.

[2] C. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[3] A. Broggi, A. Fascioli, P. Grisleri, T. Graf, and M. Meinicke. Model-based validation approaches and matching techniques for automotive vision based pedestrian detection. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, San Diego, CA, USA, 2005.

[4] N. Dalal. *Finding People in Images and Videos*. PhD Thesis, Institut National Polytechnique de Grenoble / INRIA Rhône-Alpes, 2006.

[5] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 886–893, San Diego, CA, USA, 2005.

[6] P. Dollár, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: A benchmark. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 304–311, 2009.

[7] M. Enzweiler and D. Gavrilă. A mixed generative-discriminative framework for pedestrian classification. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 1–8, Anchorage, AK, USA, 2008.

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

[9] P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, Anchorage, AK, USA, 2008.

[10] T. Gandhi and M.M. Trivedi. Pedestrian protection systems: issues, survey, and challenges. *IEEE Trans. on Intelligent Transportation Systems*, 8(3):413–430, 2007.

[11] D. Gerónimo, A.M. López, A.D. Sappa, and T. Graf. Survey of pedestrian detection for advanced driver assistance systems. *IEEE Trans. on Pattern Analysis and Machine Intelligence* (in press), 2009.

[12] A. Jain, R. Duin, and J. Mao. Statistical pattern recognition: A review. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(1):4–37, 2000.

[13] I. Laptev. Improving object detection with boosted histograms. *Image and Vision Computing*, 27(5):535–544, 2009.

[14] S. Munder and D. Gavrilă. An experimental study on pedestrian classification. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 28(11):1863–1868, 2006.

[15] Valve Software Corporation. <http://www.valvesoftware.com>.

[16] G. Shakhnarovich, P. Viola, and T. Darrel. Fast pose estimation with parameter sensitive hashing. In *Proc. IEEE Int. Conf. on Computer Vision*, pages 750–757, 2003.

[17] G. Taylor, A. Chosak, and P. Brewer. OVVV: Using virtual worlds to design and evaluate surveillance systems. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 1–8, Minneapolis, MN, USA, 2007.

[18] X. Wang, T. Han, and S. Yan. An HOG-LBP human detector with partial occlusion handling. In *Proc. IEEE Int. Conf. on Computer Vision*, Kyoto, Japan, 2009.