IEEE Intelligent Vehicles Symposium, Parma, Italy, 2004

Vision-Based Pedestrian Detection: The PROTECTOR System

D.M. Gavrila, J. Giebel and S. Munder

Machine Perception, DaimlerChrysler Research, Ulm, Germany
{dariu.gavrila,jan.giebel,stefan.munder}@DaimlerChrysler.com

***Abstract*— This paper presents the results of the first large-scale field tests on vision-based pedestrian protection from a moving vehicle. Our PROTECTOR system combines pedestrian detection, trajectory estimation, risk assessment and driver warning.**

The paper pursues a "system approach" related to the detection component. An optimization scheme models the system as a succession of individual modules and finds a good overall parameter setting by combining individual ROCs using a convex-hull technique. On the experimental side, we present a methodology for the validation of the pedestrian detection performance in an actual vehicle setting. We hope this test methodology to contribute towards the establishment of benchmark testing, enabling this application to mature. We validate the PROTECTOR system using the proposed methodology and present interesting quantitative results based on tens of thousands of images from hours of driving. Although results are promising, more research is needed before such systems can be placed at the hands of ordinary vehicle drivers.

I. Introduction

Initiatives have been started to improve the safety of vulnerable road users, namely pedestrians and bicyclists. European Commission-funded research projects PROTECTOR ("Preventive Safety for Unprotected Road User", 2000-2003) and SAVE-U ("Sensors and System Architecture for VulnerablE road Users protection", 2002-2005 [9]) are two examples. Both projects are aimed towards the development of sensor-based solutions for the detection of vulnerable road users, in order to facilitate the use of warning or preventive measures to avoid or minimize the impact of collisions. This paper describes the vision-based pedestrian system developed within PROTECTOR and the progress made since within SAVE-U.

Many interesting approaches for the visual recognition of pedestrians can be found in the literature (e.g. [1], [2], [6], [7], [11]). For a recent survey, see [4]. However, meaningful quantitative data on overall system performance is virtually non-existent. Most previous work illustrate their approach by means of a few pictures. A few [6], [7], [11] do show quantitative results, but only related to system sub-components (i.e. classification) and not to overall obstacle detection and object classification. Few if any, list performance after temporal integration, i.e. on the trajectory level. Finally, many important test criteria remain nebulous (e.g. intended coverage area, localization tolerances, data assignment rule, processing cost at the preferred ROC point).

In this paper, we pursue a "system approach" to pedestrian detection. We first describe the modules of our current system (Section II). A system optimization scheme finds an overall good parameter setting by combining the ROCs of the individual modules (Section III). We furthermore introduce a test methodology for the evaluation of overall detection performance in an actual vehicle setting (Section IV); this methodology can facilitate benchmark testing. Finally, we validate the PROTECTOR system using the proposed methodology and present quantitative results based on tens of thousands of images, derived from hours of driving on the test track and in real urban traffic (Section V).

II. The PROTECTOR System

For an overview of the modules of our system, see Figure 1. *Stereo pre-processing* performs obstacle detection and provides an initial area of interest. A depth map is computed in real-time by hierarchical feature-based stereo [3]. The depth map is multiplexed into N different discrete depth ranges, which are subsequently scanned with windows related to minimum and maximum extents of pedestrians, taking into account the ground plane location at a particular depth range and appropriate tolerances. The locations where the number of depth features exceeds a percentage of the window area are added to the ROI point list of the template hierarchy of the Chamfer System.

The *Chamfer System* [5] performs shape-based pedestrian detection: a hierarchy of pedestrian templates is matched with distance-transformed images in a tree traversal process. This method efficiently "locks onto" desired objects in a coarse-to-fine manner. A maximum chamfer distance is given as a threshold for each hierarchy level which determines whether child nodes are to be traversed, or whether a detection was made at the leaf level. We only consider the leaf level threshold for system optimization.

Texture classification involves a neural network with local receptive fields [10] to verify the Chamfer System detections. An image patch extracted from the bounding box of a detection is scaled to a standard width and height and fed into the neural network. Detections for which the output of the neural network is below a user-defined confidence are discarded.

Stereo verification is a second verification approach to filter out false detections onto the background. The shape template masks out background pixels for a dense cross-correlation between both stereo images within a certain

disparity search range. A threshold is enforced on both height and spread of the resulting correlation function.

Tracking consolidates detection results over time, discarding spurious detections; a rudimentary α-β tracker is used based on a 2.5 D bounding box object representation. Finally, the *Risk assessment and driver warning* module computes a risk level for each detected pedestrian based on its position and time-to-collision and issues an acoustical driver warning if it exceeds a certain limit.

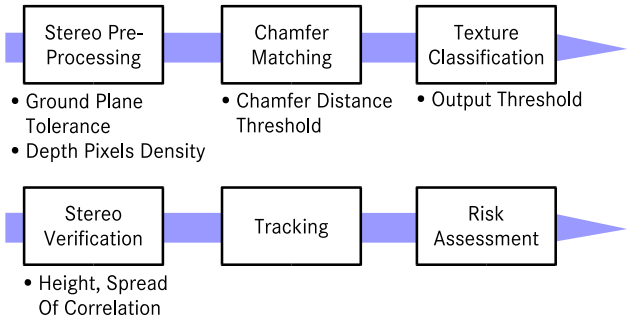

Fig. 1. System modules with parameters to be optimized

III. System Optimization

System parameters are typically tuned by optimizing an objective function using gradient descent. In our case, such an approach is inappropriate for two reasons. Firstly, we prefer to remain flexible regarding the ROC point used in a particular application, so we choose as optimization objective an entire ROC curve; this non-scalar entity lacks a straightforward ordering criterion. Secondly, if P parameters are involved, gradient descent requires iterative computation of $P+1$ runs over the image database, which exceeds our computing resources. Instead, a sequential approach inspired by Dynamic Programming is employed that successively optimizes each module of the system by computing its optimal ROC curve and recording the "path", i.e. parameter vector, to get to each point of this curve. See Figure 1 for an overview of the system modules along with their parameters due for optimization.

One optimization step, which optimizes module number $m+1$ under the assumption that optimization is completed up to module number m, is done as follows. Let

$$\widehat{\mathrm{ROC}}^{m}(\nu) = (\mathrm{FP}^{m}(\mathbf{P}_{\nu}^{m}), \mathrm{TP}^{m}(\mathbf{P}_{\nu}^{m}))\ ,$$

$\nu = 1 \ldots n$, denote the already optimized ROC curve up to module m consisting of n pairs of false positive (FP^m) and true positive rates (TP^m) obtained for n corresponding optimized parameter vectors $\mathbf{P}_{\nu}^{m}$ covering parameters for modules $1 \ldots m$. A number of (yet non-optimal) ROC curves

$$\mathrm{ROC}_{\nu}^{m+1}(\iota) = \left(\mathrm{FP}^{m+1}(\mathbf{P}_{\nu}^{m}, \mathbf{p}_{\iota}^{m+1}), \mathrm{TP}^{m+1}(\mathbf{P}_{\nu}^{m}, \mathbf{p}_{\iota}^{m+1})\right)$$

$\iota = 1 \ldots j$, for module $m+1$ is determined by selecting a (fixed) parameter vector $\mathbf{P}_{\nu}^{m}$ and varying the parameters of module $m+1$, denoted by $\mathbf{p}_{\iota}^{m+1}$. Instead of all n possible,

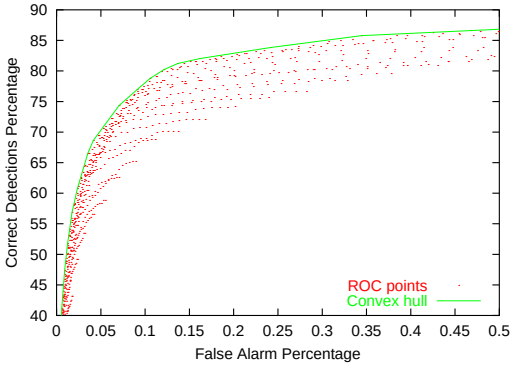

Fig. 2. Convex hull of ROC points taken for different thresholds on Chamfer distance and neural net output.

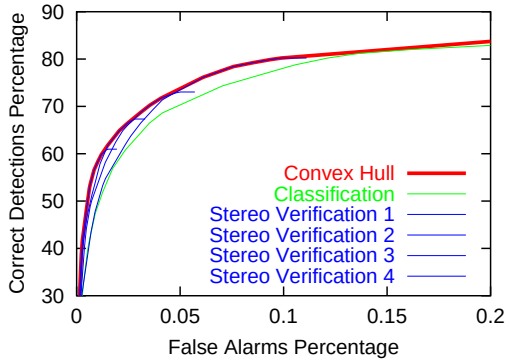

Fig. 3. Combination of ROC curves. *Classification* denotes the curve derived in the previous step (Figure 2), from which 4 ROC curves for *Stereo Verification* are computed and combined to an optimized one.

only $k < n$ such curves are computed for processing time reasons.

These k curves are combined to an optimal ROC curve by means of a ROC Convex Hull technique [8]. When regarded as a set of points $\{\mathrm{ROC}_{\nu}^{m+1}(\iota) | \nu = 1 \ldots n, \iota = 1 \ldots j\}$ in ROC space, those that span their convex hull are selected; let the indices selected be denoted by $(\nu_\lambda, \iota_\lambda)$, $\lambda = 1 \ldots l$. The optimized ROC curve is then given by

$$\widehat{\mathrm{ROC}}^{m+1}(\lambda) = (\mathrm{FP}^{m+1}(\mathbf{P}_{\lambda}^{m+1}), \mathrm{TP}^{m+1}(\mathbf{P}_{\lambda}^{m+1}))\ ,$$

$\lambda = 1 \ldots l$, where $\mathbf{P}_{\lambda}^{m+1} = (\mathbf{P}_{\nu_\lambda}^{m}, \mathbf{p}_{\iota_\lambda}^{m+1})$ are the concatenated parameter vectors that lead to the optimized ROC curve. See Figure 2 and Figure 3 for an example.

The number k is successively increased until no significant performance gain is achieved any longer; in practice, 3 or 4 are sufficient. In order to compute a single ROC curve, $\mathrm{ROC}_{\nu}^{m+1}(\iota)$, $\iota = 1 \ldots j$, it is not normally necessary to run j times over the image database. If intermediate results are recorded during processing, then a ROC curve can be determined from one run only. For example, the texture classification module records the output of the neural network for each sample processed, so that the threshold parameter can be applied afterwards. This leads to a total of 13 runs over the image database only: 1 for the first module

and about 4 for each of the three remaining modules under consideration.

IV. Test Methodology

The proposed test methodology is illustrated in Figure 5. At the core, our aim is to compare entries from ground truth and from system output, related to 3D object position relative to the vehicle (we prefer to evaluate the system in 3D rather than in image space, because it is in 3D where we can more easily incorporate application-specific considerations).

There are two possibilities for obtaining 3D ground truth data. The first involves designing a test set-up where by means of auxiliary measurement equipment vehicle- and object position over time is determined in a world coordinate system. Synchronization and transformation into the vehicle coordinate system leads to the desired 3D ground truth data. This procedure was followed for the PROTECTOR field tests on the test track. The other possibility is for a human operator to label objects in monocular images and using some world knowledge to back-project into 3D. For the case of pedestrians, the latter means making the "flat world" assumption coupled with the reasonable conjecture that the pedestrian feet stand on the ground plane. This option had to be taken for the PROTECTOR field tests in real traffic.

When comparing ground truth and system entries, the following items need to be specified.

Sensor Coverage Area. The sensor coverage area represents the space surrounding the vehicle where the defined object detection capability is required. Outside the sensor coverage area, we consider detection capability optional in the sense that the system is not rewarded/penalized for correct/false/missing detections. The PROTECTOR sensor coverage area is shown in Figure 4.

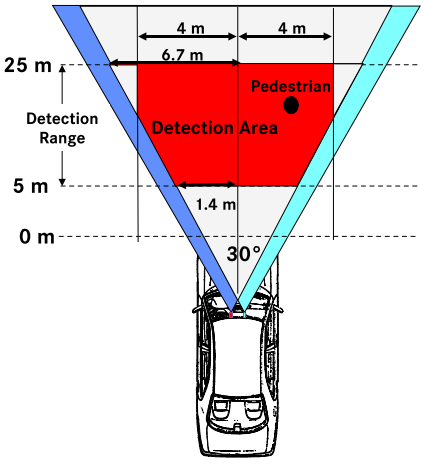

Fig. 4. PROTECTOR system coverage area

Localization Tolerance. Given an object detected by the system at a certain location ("alarm"), and given a true object location ("event"), the localization tolerance is the maximum positional deviation that still allows us to count the alarm as a match. This localization tolerance is the sum of an application-specific component (how precise does the object localization have to be for the application) and a component related to measurement error (how exact can we determine true object location).

For the PROTECTOR field tests, we define object localization tolerance as percentage of distance, for lateral and longitudinal direction (X and Z), with respect to the vehicle. Regarding the application-specific component, values of $X_a = 5\%$ and $Z_a = 15\%$ appear reasonable; for example, this means that, at 20m distance, we tolerate a localization error of $\pm$1m and $\pm$3m in the position of the pedestrian, lateral and longitudinal to the vehicle driving direction, respectively. Regarding the measurement-specific component, $X_m = 5\%$ and $Z_m = 15\%$ appear necessary (with the larger Z_m value to account for non-flat road surface and/or vehicle pitch in case of ground truth by monocular image labeling). For the PROTECTOR field tests, we then use overall tolerances of $X = 10\%$ and $Z = 30\%$.

Data Assignment. For the PROTECTOR application we allow many-to-many correspondences. An event is considered matched if there is at least one alarm matching it. In practice, this means that in the case a group of pedestrians walking sufficiently close together in front of the vehicle, the system does not necessarily have to detect all of them in isolation, it suffices if each true pedestrian is within the localization tolerance of a detected pedestrian.

Finally, having established rules for matching ground truth and system entries, we need to specify what statistics to collect to describe detection performance. We consider performance at two levels, at individual frame level and at the trajectory level. Among the latter, we distinguish two trajectory types: "class-B" and the higher quality "class-A" trajectories that have at least one entry or at least 50% of their entries matched, respectively. We consider established performance ratios such as sensitivity and precision. See Table I for the terminology used in the remainder of this paper.

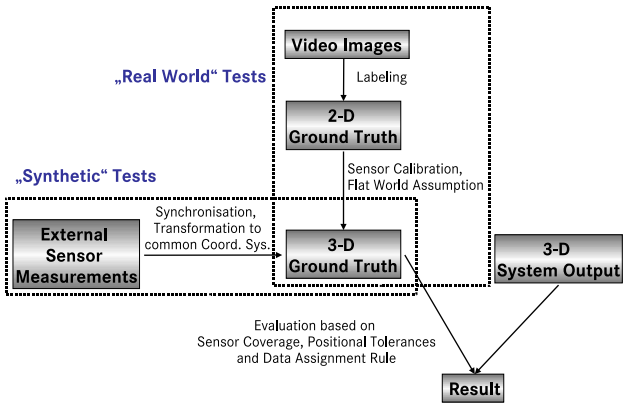

Fig. 5. Test Methodology

TABLE I

TERMINOLOGY

event	an object according to ground truth
alarm	an object according to detector system
required event	an event within the detection area
optional event	an event outside the detection area
good event	a required event with at least one matching alarm
good alarm	an alarm with at least one matching event (either required or optional)
event trajectory	a sequence of events with the same object id
alarm trajectory	a sequence of alarms with the same object id
class-B event/ alarm trajectory	an event/alarm trajectory with at least one good event/alarm
class-A event/ alarm trajectory	an event/alarm trajectory with at least 50% of good events/alarms
object sensitivity	number of good events divided by the total number of events
object precision	number of good alarms divided by the total number of alarms
trajectory sensitivity	number of trajectories with at least x hits divided by the total number of trajectories according to ground truth
trajectory precision	number of trajectories with at least x hits divided by the total number of trajectories generated by the system

V. FIELD TESTS

The PROTECTOR field tests were performed in both test track and in real traffic, according to the methodology described in previous section. We distinguish two types of results: those obtained from online vehicle processing during the field tests Fall 2002, with the then available system (i.e. implementing solely Section II, without texture classification), and secondly, the results obtained offline with our current system (i.e. fully implementing Sections II and III). We denote these by "PROTECTOR" and "PROTECTOR+", respectively. In both cases, test and training were strictly separated; the system had not previously seen *any* test track or urban scenes of the field tests. Processing involved a 2.4 GHz Intel Pentium PC.

A. Test Track

The test track experiments were performed at the Institut für Kraftfahrwesen Aachen (IKA) in Germany. 29 different traffic scenarios were enacted, involving a vehicle at 30 km/h approaching one or two pedestrians crossing laterally at various walking speeds, with or without additional road side objects (e.g. cars, panels). Figure 6 illustrates two scenarios. In the top scenario, two pedestrians are crossing the street in opposite direction. The closest pedestrian just enters the sensor coverage area of the approaching vehicle, when he starts crossing the road. In the lower scenario, a pedestrian suddenly appears behind a parked car. Figure 7 shows a third scenario, in which the ability of the PROTECTOR system to discriminate between pedestrians and other road side objects is tested; several rectangular wooden panels are placed next to the road to potentially confuse the system.

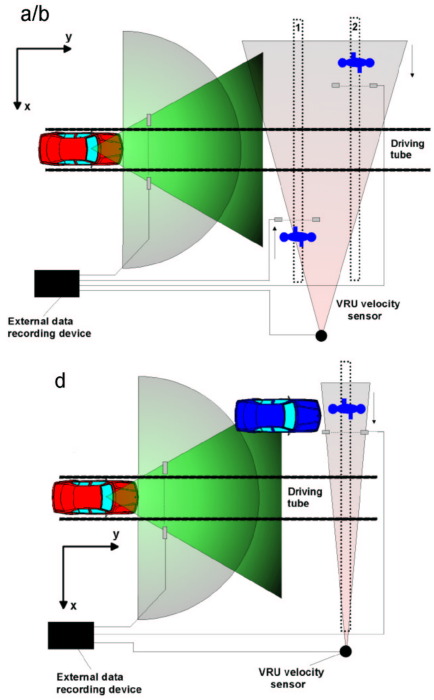

Fig. 6. Two IKA test track scenarios (out of 29 total)

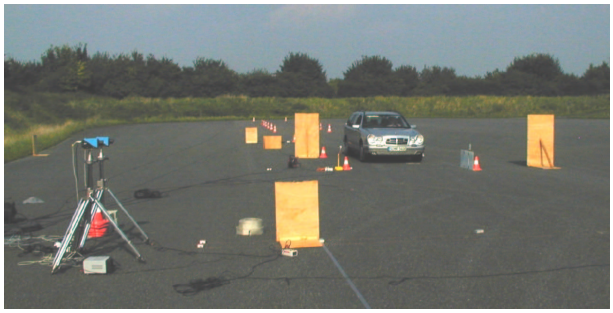

Fig. 7. IKA test track scenario dealing with object discrimination

Figure 7 also provides an impression of the auxiliary measuring equipment used for ground truth determination. Two laser scanners, shown left in the figure, were used to determine distance to the pedestrians. Not visible is the so-called "Correvit" sensor attached to the vehicle for measuring vehicle speed. Auxiliary equipment is triggered at the moment the vehicle passes through a light barrier in the driving corridor.

The results of the 29 scenario runs are summarized in Table 2. The first row shows average performance over scenarios of low-to-moderate complexity (i.e. un-obstructed views of pedestrians walking at a normal pace, without road side objects). The second row shows average performance over the more challenging cases (i.e. pedestrians partially obstructed and/or running, additional road side objects). Final row shows aggregated performance over all 29 sce-

narios. As apparent, overall performance is quite good: a sensitivity of 1.0 and 0.95 for "class B" and "class A" trajectories, and a precision of 0.97 and 0.96, respectively. Considering the second row, it was observed that the system experienced mainly problems in those scenarios where the pedestrian was running from behind a vehicle, for which we blame the shape detection module (i.e. incomplete shape training set). The system also experienced a small number of false detections on the measuring equipment placed on the test track.

B. Real Traffic

The biggest challenge of the PROTECTOR field tests was undoubtedly the pre-selected "Real-World" route through suburbia and inner city of Aachen, Germany. Two runs (*Run1* and *Run2*) on the same route were performed in close temporal succession, lasting 27 min and 24 min, respectively. On the route, ten pedestrian "actors" awaited the system, either standing or crossing at various walking speeds, according to a pre-defined choreography (for both runs the same). In addition, there were the "normal" pedestrians which happened to be on the road. The vehicle driver was requested to maintain 30 km/h, traffic conditions permitting.

Statistics for the both runs are shown in Table III. Rows relate to the total number of images, and the number of objects and trajectories (partially) within the sensor coverage area. A more restrictive area in front of the vehicle was derived from the sensor coverage area by restricting the lateral positional offset from the vehicle medial axis to lie within 1.5m. Objects and trajectories entering this area were labeled "risky". Columns "Ground Truth" relate to quantities labeled by a human operator, whereas the others relate to quantities processed by the two system versions: PROTECTOR and PROTECTOR+. The two runs were performed at different system parameter settings for the PROTECTOR version: (*Run1* "minimize false detections" and *Run2* "maximize the correct detections"). The PROTECTOR+ used *Run1* for parameter optimization, therefore, only results on *Run2* are reported.

Detailed performance statistics are shown in Table IV according to the terminology of Table I. Going from *Run1* to *Run2*, we see the expected effect of changed parameter settings for the PROTECTOR system: increased sensitivity but decreased precision. The integration of the texture classification module with the system optimization approach demonstrates its benefit in the PROTECTOR+ column. Precision increased from 10%/28% to 32%/75% for all/risky trajectories with sensitivity remaining approximately constant. Clearly, Table IV indicates that a lot of improvement needs to be made before a PROTECTOR-like system can reach commercial viability. Note however the large increase in performance when focusing on the more relevant "risky" objects/trajectories; application-specific constraints have the potential to improve matters considerably. Average processing rates over the entire runs were 12-13 Hz. In practice, rates fell to 4-10 Hz when pedestrians were actually present in the coverage area.

Finally, Figure 8 provides two screen shots of the PROTECTOR system in action. The top and lower image illustrates a test track and urban scenario, respectively. The left sub-images show the results of stereo-based preprocessing (the bounding boxes of shape templates activated by stereo are shown in grey, as discussed in Section II). Middle sub-images contain detection results superimposed. The right sub-images contain a top view of the scene in front of the vehicle. Shown is the sensor coverage area, with distance scale in meters. Detected pedestrians are denoted by red dots, (relative) velocity vectors by white line segments. The vertical "green-yellow-red" bar illustrates the associated risk level. Although in both scenes pedestrian trajectories were detected, only the top case resulted in a driver warning.

VI. Conclusions

We introduced a test methodology for the validation of a pedestrian detection system in a real vehicle setting; it brings benchmark testing on the pedestrian application a good step closer. We applied this methodology to the newly optimized PROTECTOR system and presented quantitative results from unique large-scale field tests, involving hours of driving on the test track and in real urban traffic. Although results are promising and considerable progress has been achieved over the past 1-2 years, more research is needed before such systems can be placed at the hands of ordinary vehicle drivers.

VII. Acknowledgments

The support of Mr. S. Deutschle and his colleagues from the Institut für Kraftfahrwesen Aachen (IKA) in performing the field tests is kindly acknowledged.

References

[1] A. Broggi *et al.* Stereo-based preprocessing for human shape localization in unstructured environments. In *Proc. of the IEEE Intell. Veh. Symp.*, pages 410–415, Ohio, USA, 2003.

[2] H. Elzein *et al.* A motion and shape-based pedestrian detection algorithm. In *Proc. of the IEEE Intell. Veh. Symp.*, pages 500–504, Ohio, USA, 2003.

[3] U. Franke. Real-time stereo vision for urban traffic scene understanding. In *Proc. of the IEEE Intell. Veh. Symp.*, Detroit, USA, 2000.

[4] D. M. Gavrila. Sensor-based pedestrian protection. *IEEE Intelligent Systems*, 16(6):77–81, 2001.

[5] D. M. Gavrila and V. Philomin. Real-time object detection for "smart" vehicles. In *Proc. of the ICCV*, pages 87–93, Kerkyra, Greece, 1999.

[6] A. Mohan *et al.* Example-based object detection in images by components. *IEEE Trans. on PAMI*, 23(4):349–361, 2001.

[7] C. Papageorgiou and T. Poggio. A trainable system for object detection. *IJCV*, 38(1):15–33, 2000.

[8] F. Provost and T. Fawcett. Robust classification for imprecise environments. *Machine Learning*, 42(3):203–231, 2001.

[9] SAVE-U homepage: http://www.save-u.org/.

[10] C. Wöhler and J. Anlauf. An adaptable time-delay neural-network algorithm for image sequence analysis. *IEEE Trans. on Neural Networks*, 10(6):1531–1536, 1999.

[11] L. Zhao and C. Thorpe. Stereo- and neural network-based pedestrian detection. *IEEE Trans. on ITS*, 1(3), 2000.

TABLE II

TEST TRACK RESULTS ("PROTECTOR" SYSTEM): "A"/"B" DENOTES A-CLASS/B-CLASS TRAJECTORY PERFORMANCE

scenario complexity	object sensitivity	object precision	trajectory sensitivity	trajectory precision
low-to-moderate (15)	**0.81**	**0.96**	**0.90/1.00**	**1.00/1.00**
moderate-to-high (14)	**0.78**	**0.92**	**1.00/1.00**	**0.92/0.94**
overall (29)	**0.80**	**0.94**	**0.95/1.00**	**0.96/0.97**

TABLE III

"REAL WORLD" STATISTICS: "GROUND TRUTH" RELATES TO QUANTITIES LABELED BY A HUMAN OPERATOR, OTHER COLUMNS RELATE TO QUANTITIES PROCESSED BY THE TWO SYSTEMS.

	Run1		*Run2*		
	Ground Truth	PROTECTOR	Ground Truth	PROTECTOR	PROTECTOR+
Images	1021	21239	855	17390	17390
Objects (all/risky)	485 / 71	637 / 68	370 / 47	1358 / 123	595 / 55
Trajectories (all/risky)	29 / 13	144 / 24	29 / 10	317 / 42	101 / 16

TABLE IV

"REAL WORLD" PERFORMANCE: "F" DENOTES FRAME-LEVEL PERFORMANCE, WHILE "A"/"B" DENOTE A-CLASS/B-CLASS TRAJECTORY PERFORMANCE, RESPECTIVELY.

	Run1			*Run2*					
	PROTECTOR			PROTECTOR			PROTECTOR+		
	F	A	B	F	A	B	F	A	B
Sensitivity (all)	31.5%	27.6%	44.8%	41.5%	55.2%	69.0%	52.7%	51.7%	75.9%
Precision (all)	28.4%	20.1%	20.1%	14.9%	9.8%	10.7%	43.0%	30.7%	31.7%
Sensitivity (risky)	43.7%	69.2%	69.2%	51.1%	80.0%	80.0%	62.0%	80.0%	90.0%
Precision (risky)	64.7%	62.5%	62.5%	33.3%	28.6%	28.5%	72.5%	75.0%	75.0%

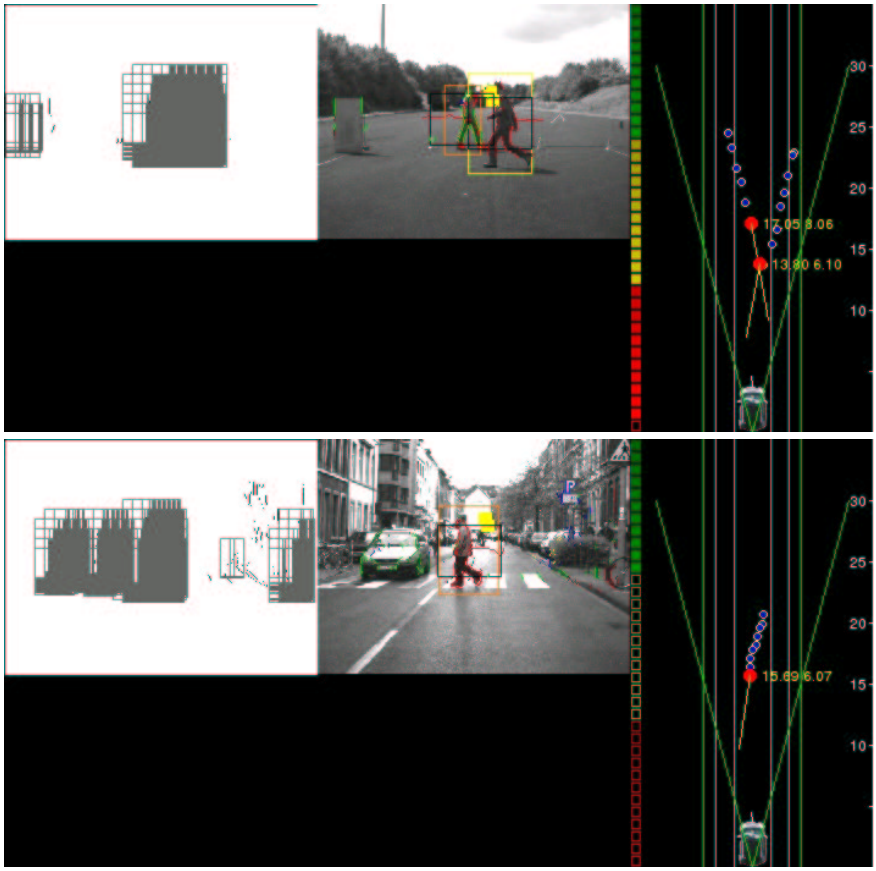

Fig. 8. PROTECTOR system results: stereo preprocessing, detections and trajectories, risk assessment.