Project no.:	**IST-FP6-FET-16276-2**
Project full title:	**Learning to emulate perception action cycles in a driving school scenario**
Project Acronym:	**DRIVSCO**
Deliverable no:	**D6.2**
Title of the deliverable:	**Algorithm for Disambiguating and Linking SVEs and SAEs**

Date of Delivery:	14.06.2008		
Organization name of lead contractor for this deliverable:	KUL		
Author(s):	M. Van Hulle, N. Chumerin (KUL)		
Participant(s):	KUL,BCCN,AAU,UMU,VMU		
Work package contributing to the deliverable:	WP6		
Nature:	R		
Version:	2.0 (revised 10.06.2008)		
Total number of pages:	18		
Start date of project:	1 Feb. 2006	**Duration:**	42 months

Project Co-funded by the European Commission		
Dissemination Level		
PU	Public	X
PP	Restricted to other program participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Summary: We report on a procedure for disambiguating Independently Moving Objects (IMOs), one of the 3 SVE types that we consider. However, the IMO maps provided by WP4 are not yet of sufficient quality to warrant for an automatic detection of the IMOs. Hence a disambiguation of these maps is required using an object recognition procedure that at the same time labels the objects by their types. It also facilitates the listing of the object's visual properties such as relative speed, distance and so on. We also perform disparity-based ground plane estimation so as to be able to detect obstacles (SVEs). The disambiguation of the Structured Action Event (SAE) is fortunately much less problematic since several actions are recorded via the CANBUS. Finally, we develop a concept of the SVE/SAE linking and also report on the first promising recordings that were recently made in support of the concept.

Contents

1 Executive Summary

As stipulated in the TA, the objective of WP6 is to: 1) disambiguate the Structured Visual Events (SVEs), 2) disambiguate the Structured Action Events (SAEs), and 3) link SVEs to SAEs. In the first year, it became clear that the disambiguation of the Independently Moving Objects (IMOs, a particular type of SVE), detected in WP4, posed more serious problems than anticipated. Indeed, the IMO maps detected in WP4 often appeared as disconnected and sparse clouds of pixels representing the same object. Furthermore, false positives often occurred, thus, pixels falsely labeled as IMOs (see Fig. 1). This, and also the difficulty to fully include prior visual domain knowledge, made the initially suggested text mining approach (based on weighted word frequency histograms) not suited. Hence, it was decided to spend much more time on IMO disambiguation as well as tracking. In this report, we show our progress on combining the IMO maps of WP4 with object recognition. In a later stage, also LIDAR data will be fused with visual data for disambiguating the IMOs and for differentiating them from (still) obstacles. The disambiguation of the SAEs is fortunately much less problematic than anticipated, according to the analysis of the CANBUS data in year 2 in WP5. Finally, we show here a concept of SVE/SAE linking and mention the recordings that were specifically made for it.

Status of the Task:
The disambiguation of the IMOs is an ongoing task which requires much more attention than foreseen. The disambiguation of the SAEs is much less of a problem since the CANBUS data can be readily used for that purpose. A concept of SVE/SAE linking is developed and recordings were accordingly made. Part of this work has been published (Chumerin and Van Hulle, 2008).

Revision notes:
As asked by the reviewers, we have included: 1) a more elaborate state of the art on fusion, a technique that can be used for disambiguating IMOs (see *Introduction* next), 2) a benchmark test to prove the advantage of disambiguation, 4) a concept of SVE/SAE linking, and 5) the description of the recordings recently made in support of the concept. Apart from these requirements, we have implemented a disparity-based ground plane detection algorithm (assuming planarity of the road) so that we can estimate obstacles by looking at the deviation of their stereo disparities from the “planar” disparity of the ground plane.

2 Introduction

The detection of the independently moving objects (IMOs) can be considered as an instantiation of the obstacle detection problem, which plays a crucial role in traffic related computer vision. The problem of independent motion detection from stereo video data acquired by static cameras differs dramatically from the same problem in the case of moving cameras. The problem is complicated by ego-motion, camera vibrations, imperfect calibration, complex outdoor environments, insufficient camera resolution and other limitations. The fusion of information obtained from multiply cues and sensors can dramatically improve quality of information perception (Handmann et al., 1998; Stiller et al., 2000; Becker & Simon, 2000; Bertozzi et al., 2000; Bertozzi et al., 2002; Fang et al., 2002; Kato et al., 2002; Steux et al., 2002; Laneurit et al., 2003; Hofmann et al., 2003; Kastrinaki et al., 2003; Blanc et al. 2004; Sole et al., 2004; Labayrade et al., 2005; Gandhi & Trivedi, 2006; Sun et al., 2006). There are a number of approaches to fusion characterization (Hall & Llinas, 1997; Dasarathy, 1997; Pohl, 1998; Wald, 1999) but, most frequently, fusion is characterized by the abstraction level:

1. Low (signal) level fusion combines raw data provided directly from sensors, without any preprocessing or transformation,
2. Intermediate (feature) level fusion aggregates features (e.g. edges, corners, texture, optic flow) extracted from raw data before aggregation,
3. High (decision) level fusion aligns decisions proposed by different sources.

Depending on the application, several different techniques are used for fusion. Matching of the targets detected by different sensors is often used for obstacle detection. Extensions of the Kalman filter (KF) (Kalman, 1960) (e.g. extended Kalman filter (EKF) and unscented Kalman filter (UKF) (Julier & Uhlmann, 1997) are mostly involved in estimation and tracking of obstacle parameters, as well as in ego-position and ego-motion estimation.

Furthermore, successful tracking of IMOs requires that a solution is found for the disconnected and sparse cloud of pixels representing the same object. Also, false positives are often detected, thus, falsely labeled as IMOs (see Fig. 1).

The approach we will follow is based on an intermediate level of fusion: the IMO maps of WP4 will be disambiguated by means of object recognition. The latter is performed with a convolutional neural network. Next, a simple algorithm is proposed for tracking the disambiguated IMOs and, finally, a number of properties of the disambiguated and tracked IMOs are estimated such as distance, speed, acceleration.

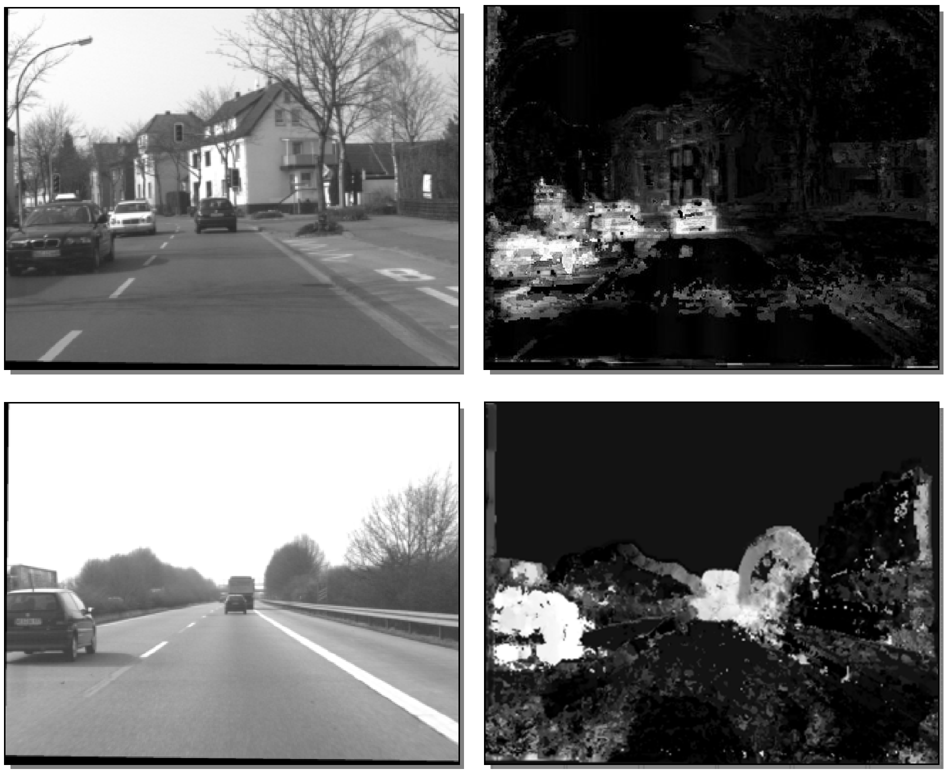


Fig. 1. Left images are rectified versions of original (left) frames taken from sequences city3 (upper) and motoway3 (lower). Right images are independent motion maps corresponding to left images.

Without an additional disambiguating procedure it is quite hard to remedy these errors. Further steps such as object tracking and visual properties extraction obviously are also affected. In this report, we propose an approach to disambiguate IMO maps, based on a separate processing and successive fusion of two information streams: *independent motion detection stream* and *disambiguating* (*recognition*) *stream* (see Fig. 2). We show some results as well as on the retrieval of an IMO's visual properties (IMO descriptors).

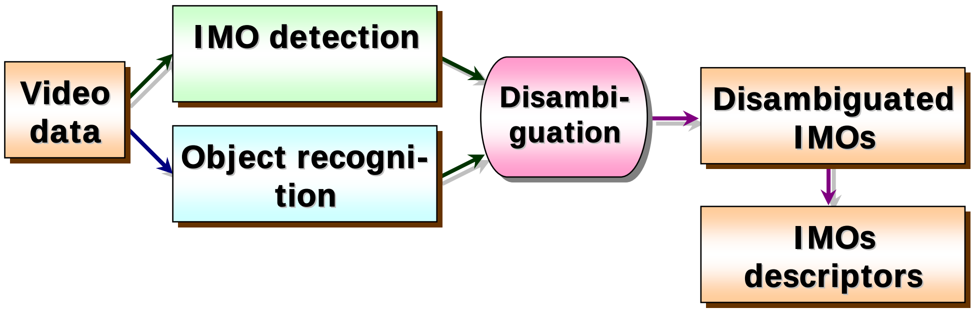


Fig. 2. Outline of proposed model. Upper branch represent *independent motion detection* stream and lower branch – *disambiguating* (*recognition*) stream.

3 Model outline

The main reason for exploiting two separate streams was the unsatisfactory quality of the final IMO segmentation based solely on the independent motion detection stream (IMO maps, see WP4). Without an additional disambiguation we cannot get rid of false positives and disconnected blobs in the IMO maps. On the other hand, we cannot use only object recognition alone, just because it works with static frames, and therefore does not take into account that the object might be moving. It means that recognition stream is not able to distinguish between *moving* and *static* objects.

4 Independent motion detection stream

We have used the IMO maps obtained in WP4 (Task 4.3 *Vision based organization of SVEs*) as a result of the independent motion detection stream.

5 Disambiguation (recognition) stream

For the recognition of vehicles and other potentially dangerous objects (such as bicycles, motorcycles and pedestrians), we have used a state of the art recognition paradigm – the convolutional network LeNet, proposed by LeCun and colleagues (1998). Modifications of LeNet were successfully exploited for generic object recognition (2004) and even for autonomous robot's obstacle avoidance system (2006).

We have used CSCSCF configuration of LeNet (see
Fig. 3) comprising six layers: three convolutional layers (C0–C2), two subsampling layers (S0–S1) and one fully connected layer (F). As input, LeNet receives a 64×64 grayscale image. Layer C0 convolutes the input with ten 5×5 kernels, adds (ten) corresponding biases, and passes result to a squashing function[1] to obtain ten 60×60 feature maps. In layer S0, each 60×60 map is subsampled to a 30×30 map in such a way that each element of S0 is obtained from a 2×2 region of C1 by a summation these 4 elements, multiplying by a coefficient, adding a bias and squashing. For different S0's elements corresponding C1's 2×2 regions do not overlap. S0 layer has ten coefficient-bias couples (one couple for each feature map). Computations in C1 are the same as in C0 with only difference in connectivity: each C1's feature map is obtained as a one single convolution, but as a sum of convolutions with a set of previous (S0's) maps (see Table 1).

[1] $f(x) = A \cdot \mathrm{tahn}(S \cdot x)$, where parameters were chosen $A = 1.7159$ and $S = 2/3$ according to LeCun and co-workers (1998).

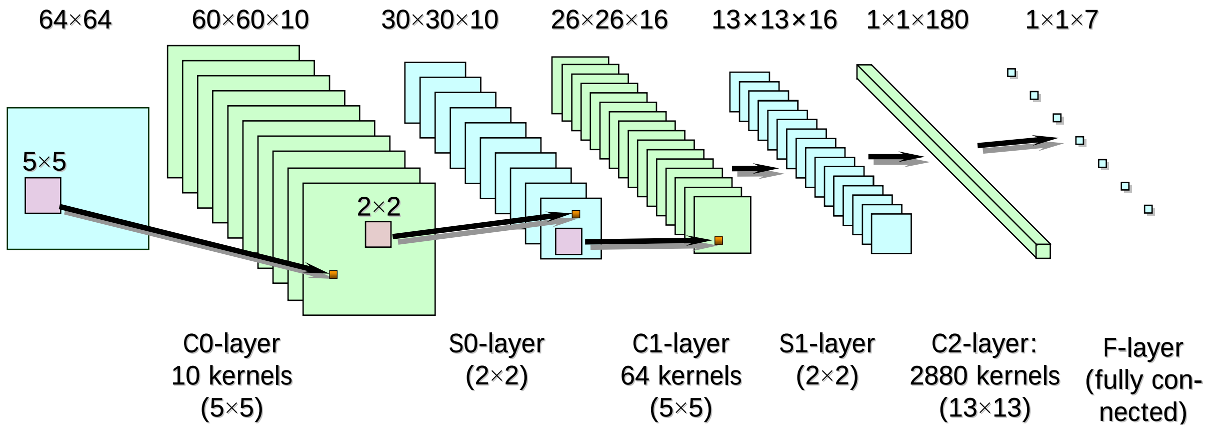

64×64
60×60×10
30×30×10
26×26×16
13×13×16
1×1×180
1×1×7
5×5
2×2
C0-layer 10 kernels (5×5)
S0-layer (2×2)
C1-layer 64 kernels (5×5)
S1-layer (2×2)
C2-layer: 2880 kernels (13×13)
F-layer (fully connected)

Fig. 3. LeNet – a feedforward convolutional neural network, used for object recognition.

Layer S1 subsamples C1's feature maps in the same manner as S0 subsamples the feature maps of C0. The final convolutional layer C2 has kernels sized 13×13 and 180 feature maps which are fully connected to all S1's 16 feature maps. It means that the number of C2's kernels is 16×180 = 2880 and correspondingly the connectivity matrix should have all cells shaded. Output layer consists of 7 neurons, which are fully connected to C2's outputs. It means that each neuron in F (corresponding to a particular class *background*, *cars*, *motorbikes*, *trucks*, *buses*, *bicycles* and *pedestrians*) just squashes the biased, weighted sum of all C2's outputs.

		C1 feature maps															
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S0 feature maps	0	■				■					■			■		■	■
	1	■					■				■	■			■		■
	2	■	■				■	■						■	■		■
	3		■				■	■				■	■			■	■
	4		■	■				■	■				■				■
	5			■				■	■			■	■				■
	6			■	■				■	■			■		■		■
	7				■				■	■				■	■		■
	8				■	■				■	■	■				■	■
	9					■				■	■			■		■	■

Table 1. S0-C1 connectivity matrix. In *i*-th column shaded cell corresponds to S0's feature map involved in C1's *i*-th feature map computation. For example: in order to get feature map in C1 indexed by 0, one have to convolve S0's feature maps indexed by 0, 1 and 2 (see first column) with corresponding kernels, compute sum of these convolutions, add bias to the result and squash it. Number of C1's kernels is equal to number of shaded cells in the table: 6×3 + 9×4 + 1×10 = 64.

LeNet scans the input image (left frame) in two scales, 320×256 and 640×512, with a 64×64 sliding window and in 8 and 16 steps respectively. For each position of the sliding window we add the class' output of LeNet to the corresponding (window) range in a 320×256 matrix which, after normalization, is considered as a probability map for the entire class.

6 LeNet Training

For the disambiguation stream training, we used two rectified stereo video sequences, each consisting of 450 frames. Using specially developed software, we labeled the left frames of the sequences. These labels were used for preparing the training dataset for LeNet.

For the LeNet training, we prepared a huge training dataset of 64×64 grayscale images (see table hereunder). The images were taken not only from the training sequences but mainly from publicly available recognition databases (LabelMe[2], VOC[3]). All kernels, coefficients and biases described in the previous section are trainable parameters.
LeNet was trained with a stochastic version of the Levenberg-Marquardt algorithm with diagonal approximation of the Hessian (LeCun et al., 1998). The training and preliminary test performances are listed in the next Table. Trucks are often confused as cars; backgrounds are recognized very well even for test sets (so it can be used as a mask).

Classes	Training set				Testing set			
	Number of samples	Ratio	Correctly recognized	Recognition performance	Number of samples	Ratio	Correctly recognized	Recognition performance
Backgrounds	390490	78,10%	389490	99,74%	1000	76,92%	973	97,30%
Cars	65806	13,16%	63916	97,13%	200	15,38%	145	72,50%
Trucks	7972	1,59%	6826	85,62%	100	7,69%	23	23,00%
Buses	4954	0,99%	3455	69,74%	-		-	-
Motorbikes	15286	3,06%	14233	93,11%	-		-	-
Bicycles	11210	2,24%	10891	97,15%	-		-	-
Pedestrians	4282	0,86%	4013	93,72%	-		-	-
Total	**500000**	**-**	**492824**	**98,56%**	**1300**		**1141**	**81,50%**

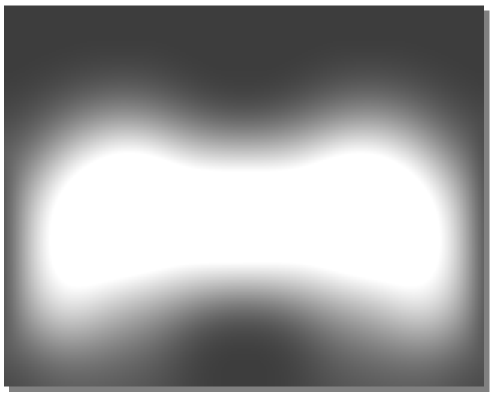


Fig. 4. Mask of most probable IMO appearance in a frame.

7 Disambiguation

IMO disambiguation we have done in three steps. The first step – mixing (multiplying) he independent motion map with a mask of most probable locations of IMOs in a frame (see Fig. 4). The second step – the intersection of the previous result with the non-background map (these maps we have used further in tracking procedure). The third step – the intersection of the previous result with the (normalized) likelihood map of each class, which results in six maps $L_1,\ldots,L_6$ (one for each class, except the background). In Fig. 5 one can see an example of the procedure.

8 Tracking and properties retrieval of IMOs

As soon as we are able to disambiguate IMOs, it becomes much easier to track IMOs and to retrieve their properties (size, absolute speed, relative speed, time to contact, absolute acceleration *etc*). For IMO tracking we have used a simple tracking technique. As input we used maps, ob-

[2] http://labelme.csail.mit.edu/
[3] http://www.pascal-network.org/challenges/VOC/

tained on second step of disambiguation procedure (intersection of masked independent motion maps with corresponding non-backgrounds). Then we have performed a spatio-temporal filtering (i.e. for *i*-th map we apply smoothing of three-dimensional array – concatenation of (*i*–2)-th, (*i*–1)-th, *i*-th, (*i*+1)-th and (*i*+2)-th two-dimensional maps along third time-dimension). Then we search for local maxima in the entire (*i*-th) filtered frame and consider them as IMO centers for this frame.
For a more robust tracking of each IMO, we have introduced a parameter called *tracking score*. For a particular IMO we increase this parameter when in the next frame, only in a small neighbourhood of the entire IMO there is a new IMO with the same class label, and approximately with the same properties (size, distance). Otherwise the tracking score is decreased. An IMO survives while the tracking score is above a fixed threshold. The tracking score works as a momentum and allows the system to keep tracking IMO even when there is no sufficient data to track IMOs in the next few frames.

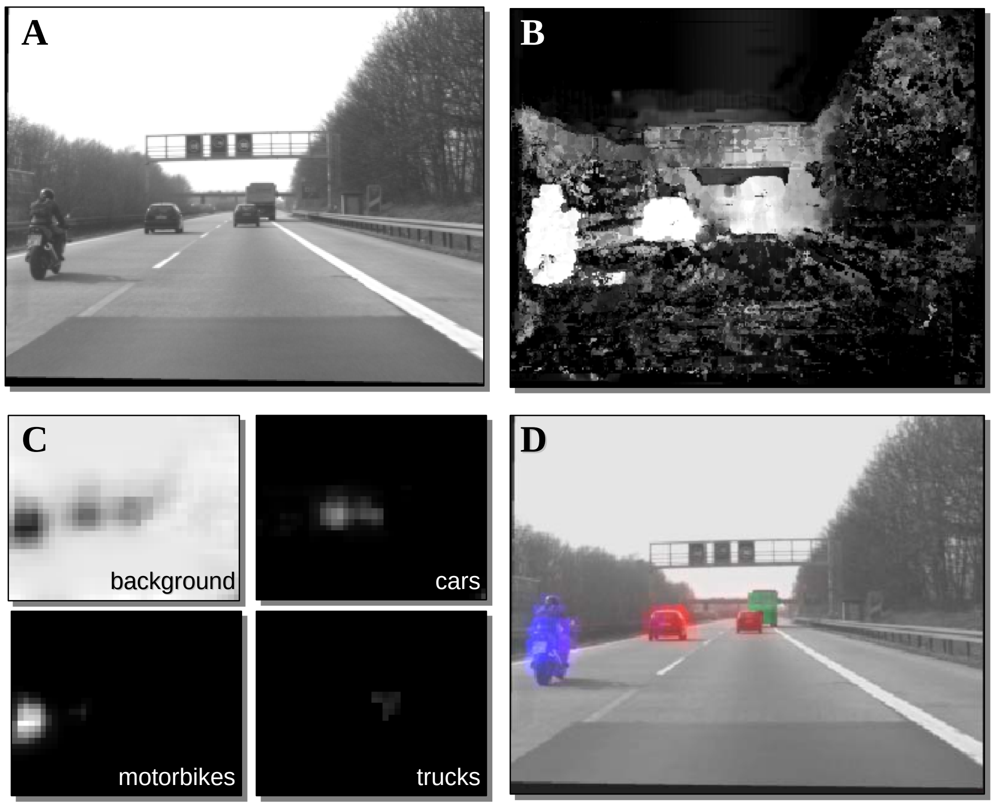


Fig. 5. Results of IMOs disambiguation: *A*, an original (left) frame of testing video sequence. *B*, output of the independent motion detection stream: intensity of each pixel means probability of being part of the IMO. *C*, output the disambiguation stream: each subfigure contains the likelihood map for one of the classes: background, cars, motorbikes, trucks, buses, bicycles and pedestrians (we present here only four classes because the other maps contain values very close to zero). *D*, result of the IMO disambiguation. Here, we used different colors to represent the different classes.

One of the most important tasks of WP6 consists of the retrieval of an IMOs properties.
The *Class* c_k of the *k*-th IMO we define as:

$$c_k = \arg\max_{1 \le c \le 6} \left\{ L_c(\mathbf{x}_k) \right\}, \qquad (1)$$

where $\mathbf{x}_k = (i_k, j_k)$ center of *k*-th IMO (in image domain *I*) and L_c – maps, defined in third step of disambiguation procedure.

For IMO's *size* σ_k estimation we search for $\sigma>0$, where the first minimum of function (2) take place.

$$\Delta_k(\sigma)=\int_I \left| L_{c_k}(\mathbf{x}_k)e^{-\|\mathbf{x}_k-\mathbf{x}\|^2/\sigma^2}-L_{c_k}(\mathbf{x})\right| d\mathbf{x}, \tag{2}$$

The IMO's *distance* estimation is a crucial point in the retrieval process. Using an averaged (in a small neighbourhood of the entire IMO's center) disparity, we have computed the distance to the IMO. In order to compensate for instabilities in the distance estimations, we have used averaging over the previous 5 estimates. The *Relative speed in depth*, we estimated as the derivative (w.r.t. time) of the distance using linear regression over the last 5 (averaged) estimations of the distance. In order to estimate the *time to contact*, we have divided the averaged distance by the averaged relative speed in depth. Using the precise value of the egomotion speed from the CAN-BUS data, and simply by adding it to the relative speed in depth we have also obtained the *absolute speed in depth*. The derivative of the absolute speed in depth can be considered as an estimation of the *acceleration* (it is true only in the case when the egoheading is collinear to the heading of the entire IMO). An example of IMO tracking and properties retrieval is shown in Fig. 6.

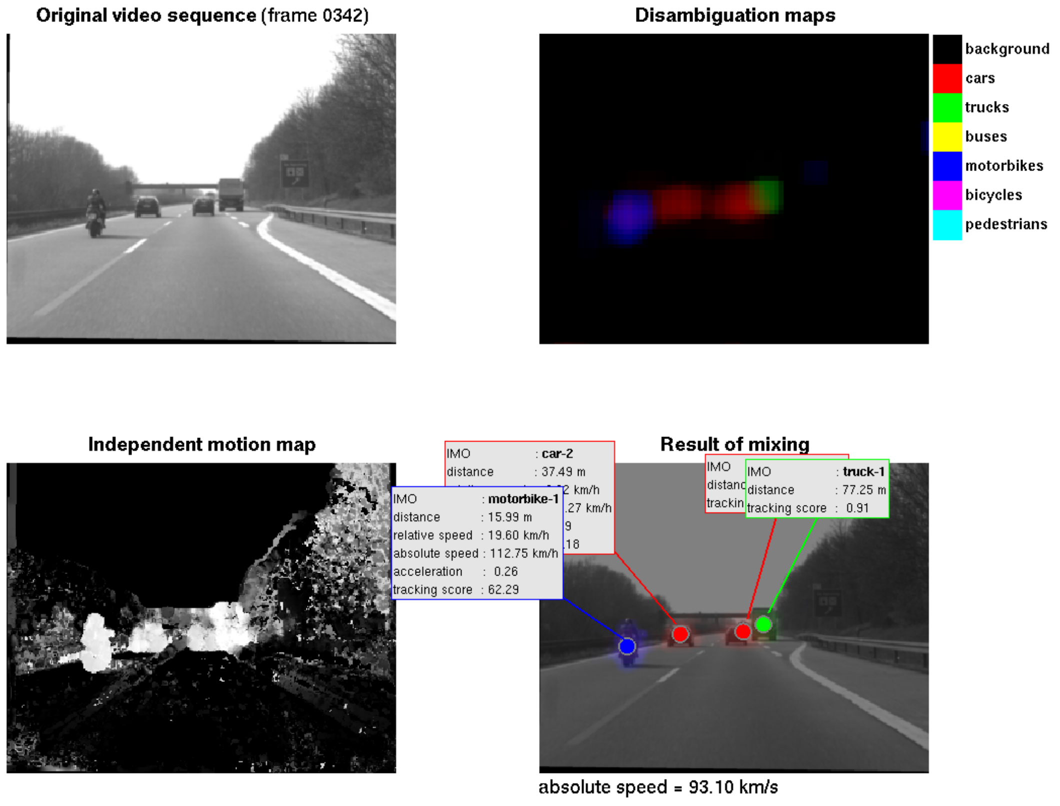


Fig. 6. Results of application IMOs disambiguation for IMO tracking and properties retrieval.

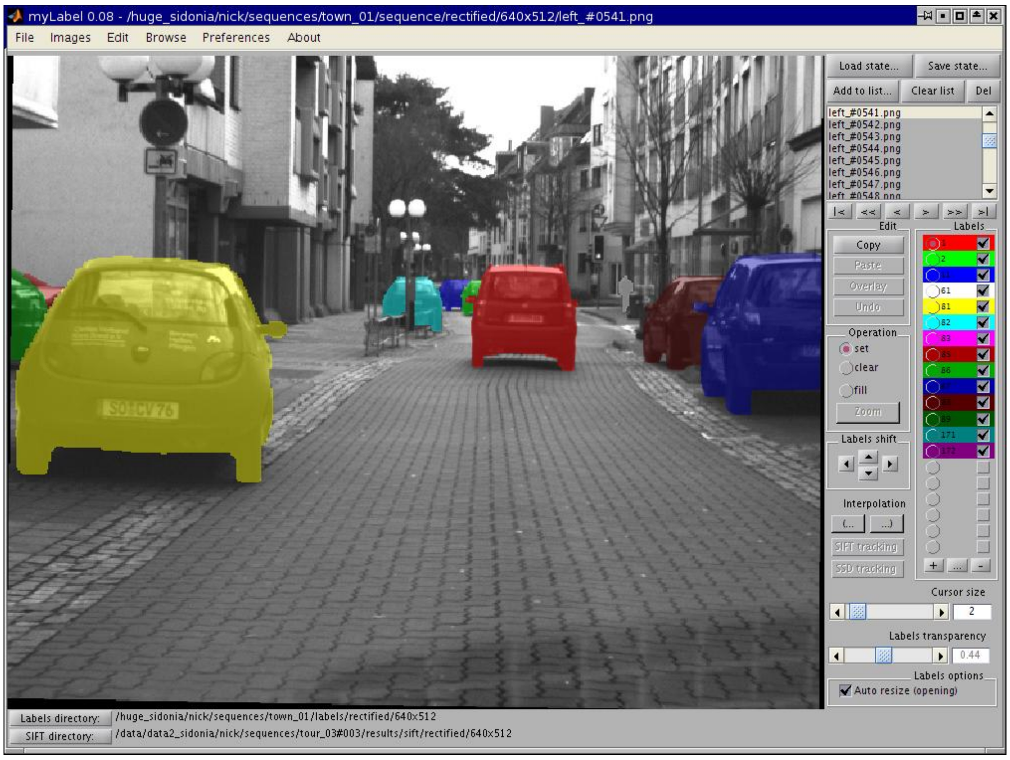
9 Benchmarking

9.1 myLabel tool

In order to assess the performance of our IMO disambiguation algorithm, and to compare it to the original IMO maps of WP4, we have developed a tool with which benchmark data sets of hand labeled IMOs can be generated. The tool, called myLabel, is a graphical tool for manual and semiautomatic pixel-wise labeling of image sequences in the MATLAB 7.0.4 (Linux/Windows).environment (Fig. 7). The main features of myLabel are:

- Project-like state support. Each state consists of an image list and all directory settings.
- Arbitrary image resolutions (even per state) and graphical formats can be handled.
- Labeling of up to 20 different objects in the same frame.
- Labeling according to the layers concept of standard graphical editors (e.g. Adobe Photoshop): one can add, delete, copy, paste, overlay and shift labels as well as toggle visibility, change transparency, color and value of each label.
- Semiautomatic object tracking/labeling using two simple techniques:
  - o SIFT keypoints tracking with warping of the corresponded label mask;
  - o best SSD (Sum of Squared Differences) matching.
- Most operations have been coded by key shortcuts.

With this tool, several movies have been hand labeled and used for benchmarking.



myLabel 0.08 - /huge_sidonia/nick/sequences/town_01/sequence/rectified/640x512/left_#0541.png

left_#0541.png
left_#0542.png
left_#0543.png
left_#0544.png
left_#0545.png
left_#0546.png
left_#0547.png
left_#0548.png

Cursor size: 2
Labels transparency: 0.44

Labels directory: /huge_sidonia/nick/sequences/town_01/labels/rectified/640x512
SIFT directory: /data/data2_sidonia/nick/sequences/tour_03#003/results/sift/rectified/640x512

Fig. 7 Screenshot of the myLabel tool.

9.2 Evaluation of precision and recall

In order to evaluate the IMO maps, and the disambiguated ones, for segmenting IMOs, we consider two metrics: precision and recall. Precision is the number of correct detections divided by all detections; recall is the number of correct detections divided by the true IMO pixels. Since

detection depends on a threshold, as we vary the threshold, we obtain precision vs. recall plots. By plotting these curves for each method, we are in a position to compare their performances. The optimal threshold can be chosen as the harmonic mean of precision and recall: 2/(1/precision + 1/recall).
In addition to the IMO map, we consider three disambiguation cases: using object recognition (as reported here), using the deviation from the disparity-based ground plane (see further), and the fusion of the latter two. In the next figure we show our evaluation of tour_03#000. The small circle on each plot is the harmonic mean. We observe that the performance improves when disambiguating the original IMO maps.

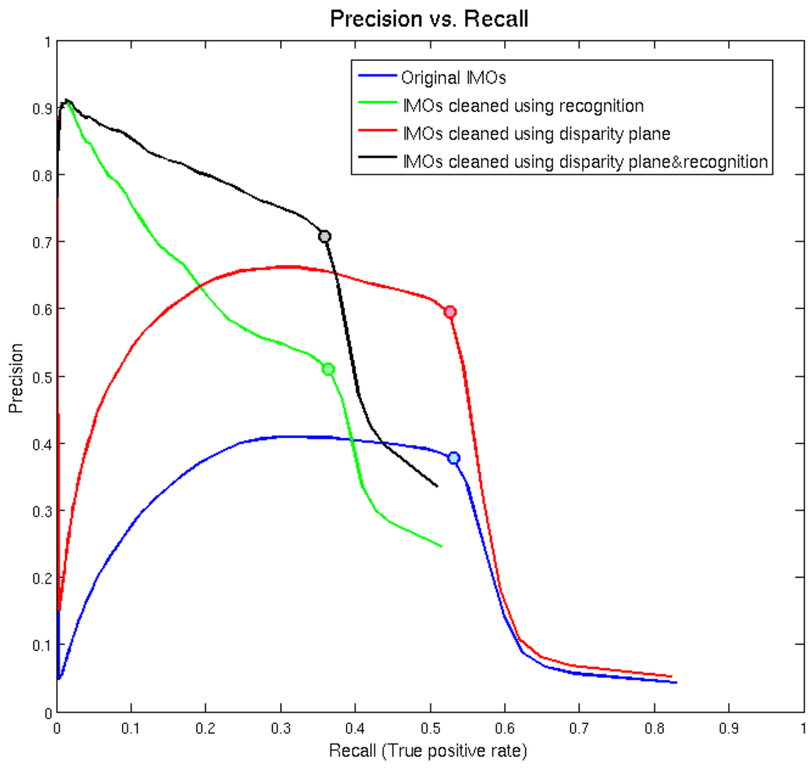


Fig. 8 Benchmarking of IMO maps and the disambiguated ones (see text).

10 Conclusion on IMO disambiguation

The disambiguation procedure we presented is successful in disambiguating IMOs (Fig. 5), and allows for an easy tracking and properties retrieval (Fig. 6). By mixing IMO maps and class likelihood maps we increase the reliability of the detected IMOs and automatically cleans up the false positives. This is a crucial issue when video streams obtained from moving cameras are used. Furthermore, we have benchmarked two disambiguation techniques.
Further improvements of the model's performance we see in the combination of other cues with the IMO maps, and by using the CANBUS range radar data (LIDAR data) to refine the distance estimations and for detecting (static) obstacles.
Also the two processing streams at earlier stages can be improved, namely, using a common bank of Gabor-like (fixed/non-trainable) filters in the visual cues extraction stage and in LeNet's C0 layer. This step definitely will reduce computations. Another way to reduce computations is to reduce the amount of data to process. In the disambiguation stream, we can build the object likelihood maps not for the entire frame, but only for regions containing motion information. The

last idea has obvious biological support: in real visual system recognition takes place with priority for moving objects.

11 Disparity-based ground plane estimation

There are a number of studies on ground plane detection based on lane markings (Zhang et al., 1994), *v*-disparity (Labayrade et al., 2002), geometry assumptions of urban scenes (Cornelis et al., 2006). Road modeling is a crucial issue in these studies. Assuming planarity of the road one can estimate obstacles by looking at the deviation of stereo disparity from a "planar" disparity.
Using the phase-based, multi-scale stereo disparity algorithm of WP4, we have first computed dense disparity maps. Then, we have applied a predefined road mask and fitted a disparity plane through the remaining disparities. An example of a disparity-based ground plane is shown in Fig. 9. The red bullets indicate a fixed set of nine points (3×3 lattice) The disparities for these points are computed in each frame using the estimated disparity plane model. Note also the horizon (yellow horizontal line). The actual plane fitting is done with a robust regression technique, called *Iteratively Reweighted Least-Squares* (IRLS) (Holland and Welsch, 1977). In order to track the ground plane over time, we verify whether the normal unity vector of the ground plane and the coefficient representing the distance from the camera nodal point to the plane do not differ much in two subsequent frames; if they do differ more than a preset threshold, then the previous frame's ground plane estimate is taken.
Given the disparity plane, all pixels for which the disparities that are larger than some preset value are considered as obstacles: $\delta(x,y) > \alpha x + \beta y + \gamma + \delta_0$ where $\delta(x,y)$ is the disparity value taken from the disparity map of the pixel at coordinate (x,y), parameters α, β and γ are the estimates of the disparity model of the ground plane, and δ_0 is a threshold which corresponds to the minimal height above the disparity plane for a pixel to be considered an obstacle. The result is shown in Fig. 10.

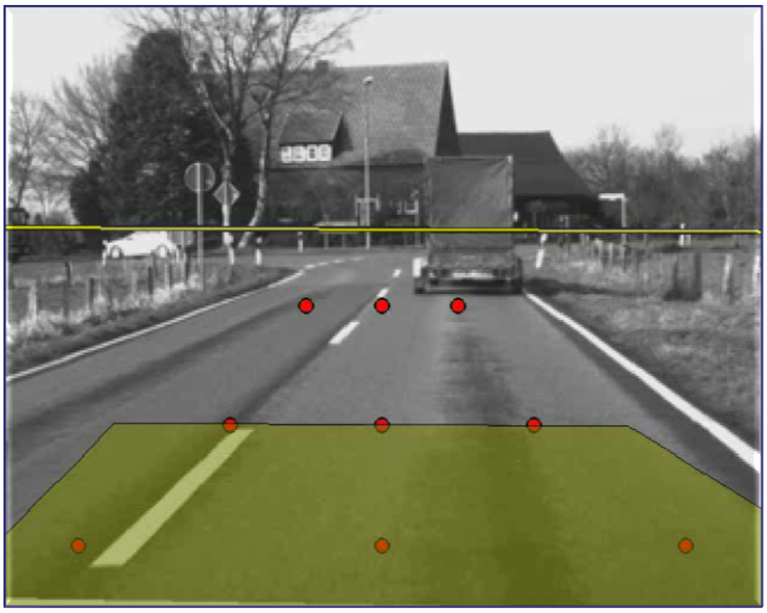


Fig. 9 Disparity –based ground plane estimation.

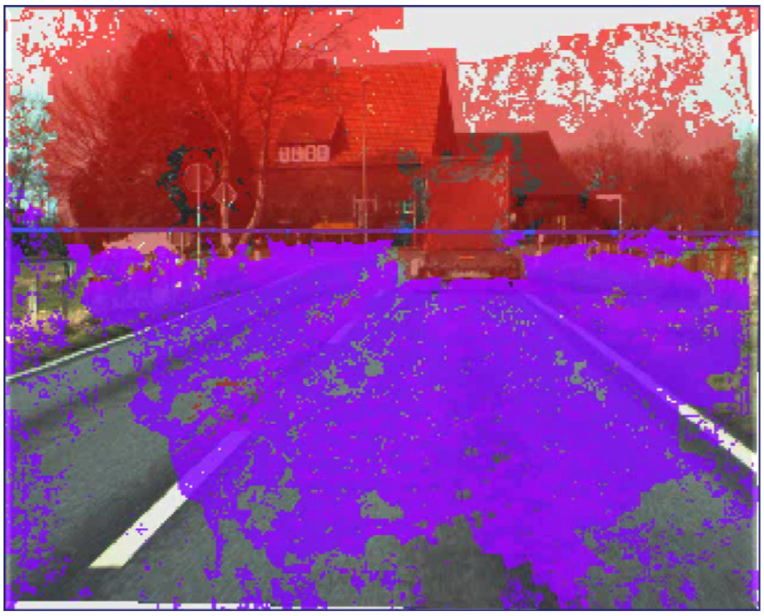


Fig. 10 Obstacle detection based on the estimated disparity plane. In red color we depict the obstacle pixels and in magenta we show the pixels near the disparity plane.

12 Concept of SVE/SAE linking

Based on the successful SAE road following of WP7, we decided to draft up a similar concept for SVE/SAE linking. In WP7 several runs of the same tour were made and used as a database from which driving actions were extracted for the current situation (i.e., a Look-Up Table (LUT) approach).

Here, we will also consider recordings of several runs of the same tour (see further), but we will issue warning signals if the driving state and actions are different from what is expected from the historical recordings. This illustrated in Fig. 11 for the case of lane following. Suppose that we have historical records that contain visual information about position on the road (differential GPS, line markings), orientation (heading), speed of the car, and also actions such as steering angle, acceleration/deceleration (braking) and indicator lights (the historical points are indicated by the green points in Fig. 11). Information on the line markings, speed, steering angle, braking, and indicator lights are available from the CANBUS. From this information, we can predict our current state based on an interpolation between the historical data points (LUT). We also have the information describing the current state, or even its predicted one (e.g. 1 s ahead), predicted from the current state (cf. heading, speed). We can compare the interpolated and the true or predicted state of the car to issue a warning if the current visual information and/or actions are anomalous compared to the historical records (which are regarded as "error"-free). Since the information is diverse (continuous as well as categorical, visual information as well as actions) we perhaps need to come up with several dissimilarity metrics and more than one threshold to detect anomalies. The thresholds need to be chosen empirically.

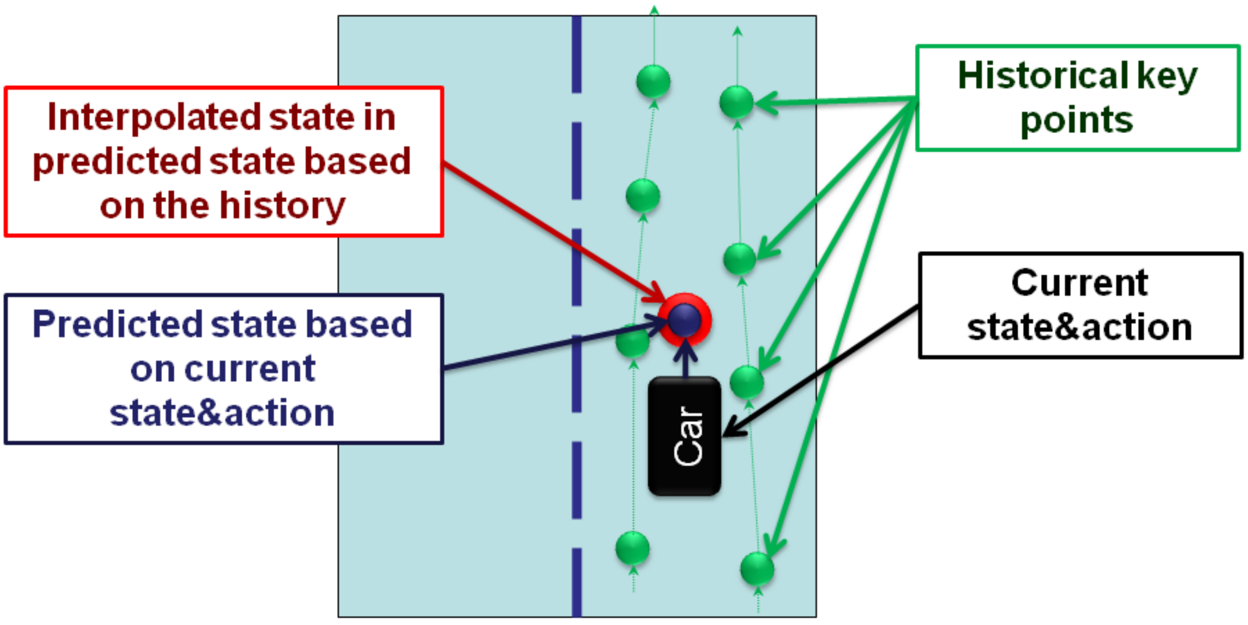

Interpolated state in predicted state based on the history
Predicted state based on current state&action
Historical key points
Current state&action
Car

Fig. 11: Warning signal is generated when the predicted state (including actions) deviates from the current state (also including actions).

A similar concept can be developed when IMOs are present. We assume that we can treat each IMO independently. The concept is depicted in Fig. 12. We can take into account the inaccurate estimates of the headings of our car and the IMO, and thus, their future positions, in our warning metric as follows. Assume that P1 is the area spanned by the uncertain positions over a certain time span, and likewise for our car, P2 (Fig. 13). The size of the conjunction of P1 and P2, $P1 \cap P2$, compared to the sizes of P1 and P2 are then an indication of danger, or, when normalized: danger= $2\, P1 \cap P2/(P1 + P2)$. If the danger level exceeds a threshold, we issue a warning.

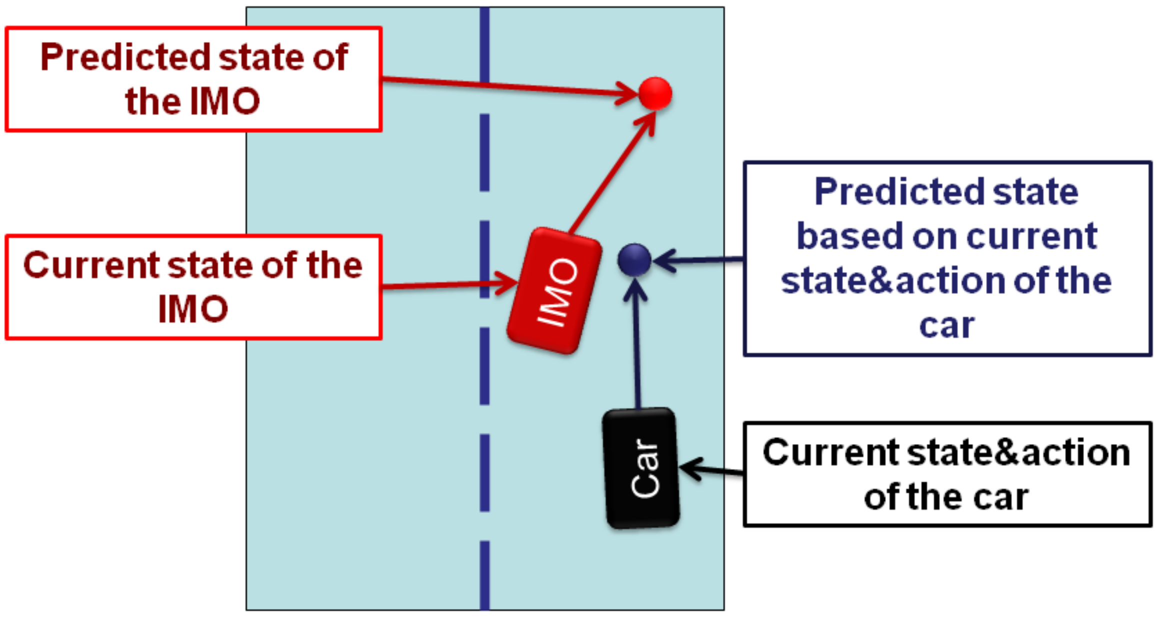

Predicted state of the IMO
Current state of the IMO
Predicted state based on current state&action of the car
Current state&action of the car
IMO
Car

Fig. 12 Warning signal is generated when the current and predicted state (including actions) of our car deviates from the current and predicted state (also including actions) of the IMO.

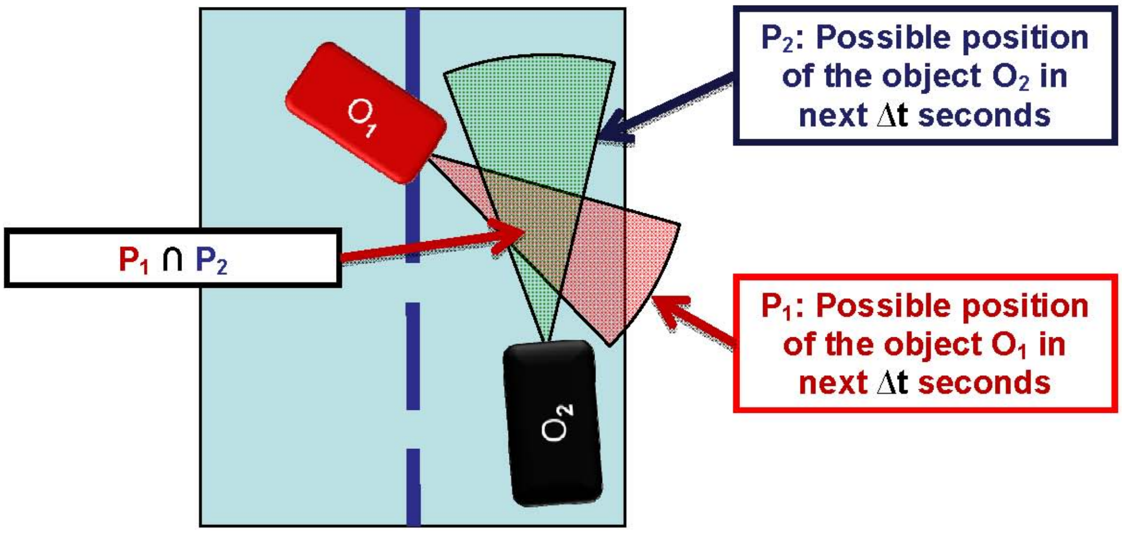

P_2: Possible position of the object O_2 in next Δt seconds
O_1
$P_1 \cap P_2$
P_1: Possible position of the object O_1 in next Δt seconds
O_2

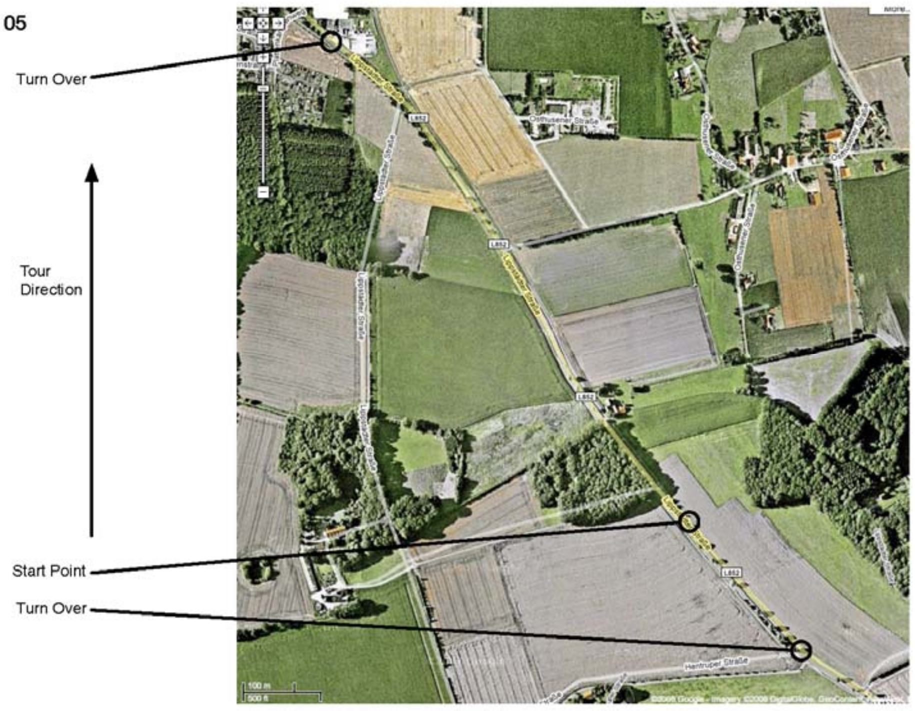
Fig. 13: Definition of the uncertainty of the position over a given time span of our car (O2) and an IMO (O1) (green and red circle segments, P2 and P1). The conjunction of the circle segments $P1 \cap P2$ is indicated and used in the metric for deciding if a warning signal needs to be issued (see text).

Finally, since we rely on historical recordings, since we are solving a regression problem with LUT, we can **generalize** beyond historical recordings and consider a more general regression problem instead, using different recordings. This is the subject of further research and will involve machine learning techniques (for which we have extensive experience).

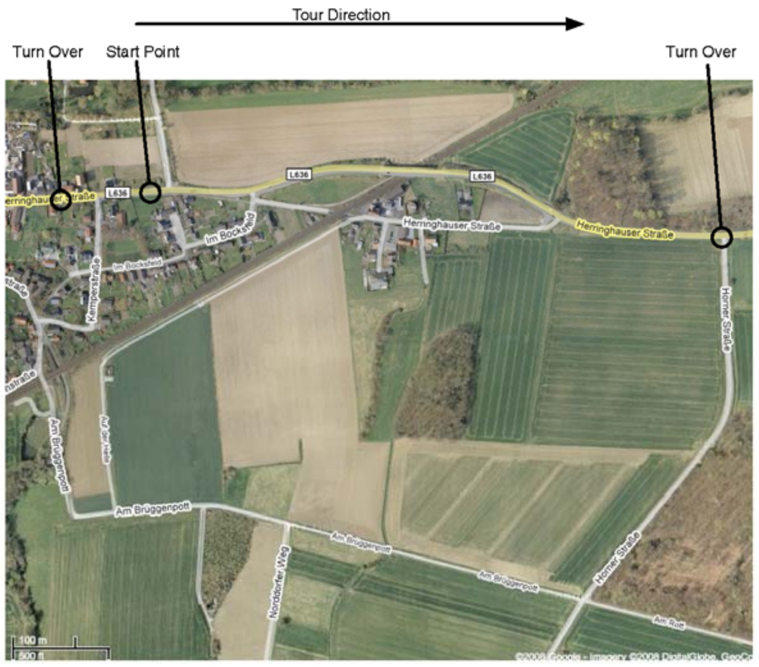
13 Description of first promising recordings

In order to test the SVE/SAE linking concept explained above (based on LUT), a number of sequences have been repeatedly recorded by UMU and SDU with a calibrated stereo rig. The sequences involve three different curves, characterized by soft bends (curve 5), medium bends (curve 6) and strong bends (curve 3). The trajectories are illustrated in the images below (fig. 14).
Three different conditions have been investigated: plain driving, car following and opposing traffic. For each combination of condition and sequence, a total of 15 runs have been performed (10 for training and 5 for testing).
In addition to the stereo video stream, the following data is available for each run: eye movements, curve radius, velocity, breaking, indicators, steering angle, LIDAR, lane information and GPS data.
These historical recordings will then serve as the **starting point of the SVE/SAE linking**, which is currently being studied, and for which we hope to report on in September 2008.

Curve 05



Curve 06



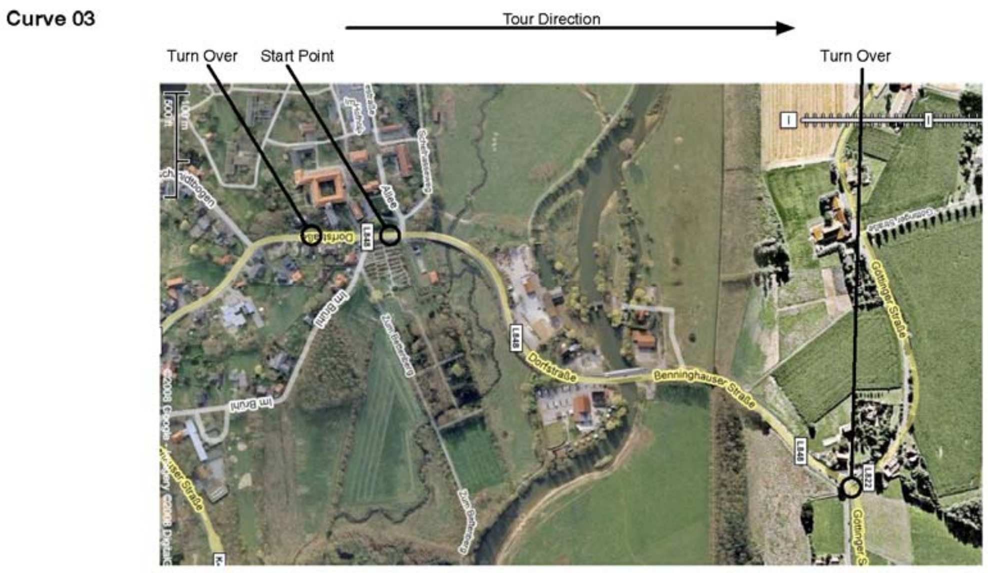


Fig. 14: (Upper, middle and lower panels) Trajectories (in yellow) of the tour that was repeatedly recorded for the purpose of SVE/SAE linking.

References

Becker J. and Simon (2000). A. Sensor and navigation data fusion for an autonomous vehicle. Intelligent Vehicles Symposium, 2000. IV 2000. Proceedings of the IEEE pp. 156-161.

Bertozzi M., Broggi A., Cellario M., Fascioli A., Lombardi P., and Porta, M. (2002). Artificial vision in road vehicles. Proceedings of the IEEE 90(7), 1258-1271.

Blanc C., Trassoudaine L., Le Guilloux Y. and Moreira R. (2004). Track to track fusion method applied to road obstacle detection. Procedings of the Seventh International Conference on Information Fusion.

Chumerin N. and Van Hulle M.M. (2008). Cue and Sensor Fusion for Independent Moving Objects Detection and Description in Driving Scenes. In: Signal Processing Techniques for Knowledge Extraction and Information Fusion, D.P. Mandic, M. Golz, A. Kuh, D. Obradovic, and T. Tanaka (Eds.), Springer, Boston, USA, pp. 161-180.

Cornelis N., Cornelis K. and Van Gool L. (2006). Fast compact city modeling for navigation pre-visualization. In CVPR 2006, volume **2**, 1339–1344.

Dasarathy, B. (1997). Sensor fusion potential exploitation-innovative architec- tures and illustrative applications. Proceedings of the IEEE 85(1), 24-38.

Fang Y., Masaki I., and Horn B. (2002). Depth-based target segmentation for in-telligent vehicles: fusion of radar and binocular stereo. Intelligent Transportation Systems, IEEE Transactions on 3(3), 196-202.

Gandhi T. and Trivedi M. (2006). Vehicle surround capture: Survey of techniques and a novel omni video based approach for dynamic panoramic surround maps. IEEE Transactions on Intelligent Transportation Systems, 7(3), 293-308.

Hall D. and Llinas J. (1997). An introduction to multisensor data fusion. Proceedings of the IEEE 85(1), 6-23.

Handmann U., Lorenz G., Schnitger T. and Seelen W. (1998). Fusion of different sensors and algorithms for segmentation. IV?98, IEEE International Conference on Intelligent Vehicles 1998 pp. 499-504.

Holland P.W. and Welsch R.E. (1977). Robust Regression using Iteratively Reweighted Least Square, Commun. Statist. Theor. Meth., A6, 813–827.

Hofmann U., Rieder A. and Dickmanns E. (2003). Radar and vision data fusion for hybrid adaptive cruise control on highways. Machine Vision and Applications 14(1), 42-49.

Julier S. and Uhlmann J. (1997). A new extension of the Kalman filter to nonlinear systems. Int. Symp. Aerospace/Defense Sensing, Simul. and Controls 3.

Kalman R. (1960). A new approach to linear filtering and prediction problems. Journal of Basic Engineering 82(1), 35-45.

Kastrinaki V., Zervakis M. and Kalaitzakis K. (2003). A survey of video processing techniques for traffic applications. Image and Vision Computing 21(4), 359-381.

Kato T., Ninomiya Y., Masaki I. (2002). An obstacle detection method by fusion of radar and motion stereo. Intelligent Transportation Systems, IEEE Transactions on 3(3), 182-188.

Labayrade R., Aubert D. and Tarel J. (2002). Real Time Obstacle Detection on Non Flat Road Geometry through `V-Disparity' Representation Proceedings of IEEE Intelligent Vehicle Symposium.

Labayrade R., Royere C., Gruyer D., Aubert D. (2005). Cooperative fusion for multi-obstacles detection with use of stereovision and laser scanner. Autonomous Robots 19(2), 117-140.

Laneurit J., Blanc C., Chapuis R., Trassoudaine L. (2003). Multisensorial data fusion for global vehicle and obstacles absolute positioning. Intelligent Vehicles Symposium, 2003. Proceedings. IEEE pp. 138-143.

LeCun Y., Bottou L., Bengio Y. and Haffner P. (1998). Gradient-Based Learning Applied to Document Recognition. In Proceedings of the IEEE, vol. 86, №11, Nov. 1998, pp. 2278–2324.

LeCun Y., Huang F.J. and Bottou L. (2004). Learning Methods for Generic Object Recognition with Invariance to Pose and Lighting. In Proceedings of CVPR'04.

LeCun Y., Muller U., Ben J., Cosatto E. and Flepp B. (2006). Off-Road Obstacle Avoidance through End-to-End Learning. In Advances in Neural Information Processing Systems, vol. 18.

Pohl C. (1998). Review article multisensor image fusion in remote sensing: con- cepts, methods and applications. International Journal of Remote Sensing 19(5), 823-854.

Sole A., Mano O., Stein G., Kumon H., Tamatsu Y. and Shashua A. (2004). Solid or not solid: vision for radar target validation. Intelligent Vehicles Symposium, 2004 IEEE pp. 819-824.

Steux B., Laurgeau C., Salesse L. and Wautier D. (2002). Fade: a vehicle detection and tracking system featuring monocular color vision and radar data fusion. Intelligent Vehicle Symposium, 2002. IEEE 2.

Stiller C., Hipp J., Rossig C. and Ewald A. (2000). Multisensor obstacle detection and tracking. Image and Vision Computing 18(5), 389-396.

Sun Z., Bebis G. and Miller R. (2006). On-road vehicle detection: a review. Pattern Analysis and Machine Intelligence, IEEE Transactions on 28(5), 694-711.

Wald L. (1999). Some terms of reference in data fusion. IEEE Transactions on Geoscience and Remote Sensing 37(3).

Zhang Z., Weiss R. and Hanson A. (1994). Qualitative obstacle detection Computer Vision and Pattern Recognition, 1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference on, 1994, pp. 554–559.