RoadGraph: High level sensor data fusion between objects and street network

Kai Homeier and Lars Wolf

***Abstract*— The RoadGraph is a graph based environmental model for driver assistance systems. It integrates information from different sources like digital maps, onboard sensors and V2X communication into one single model about the vehicle's environment. At the moment of information aggregation some function independent situation analysis is done. In this paper we look at techniques for lane-precise map-matching even with moderate GPS reception using distinct information sources. We also analyze the concepts of aggregating objects from different sources with a-priori knowledge of the street-layout. Results of this novelty approach are shown.**

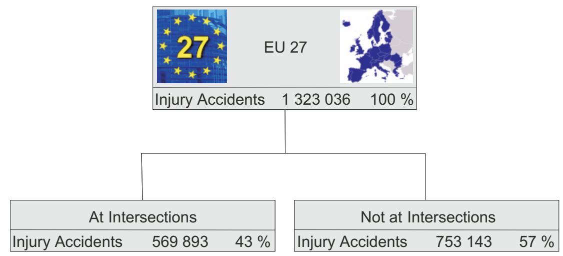


Fig. 1: Distribution of accidents in EU27 [1]

I. Introduction

In the current decade environment perception became more and more important for automotive applications such as driver assistance systems, autonomous driving, pre-crash, etc. After addressing various applications for highway assistance, support at intersections came into the focus of research activities nowadays.

This paper describes a graph based environmental model which combines information of digital maps, onboard sensors and cooperative data, which brings sensor data to a higher level of abstraction than what is typically done.

Object fusion is done on this high level, allowing new possibilities in complex urban scenarios. The knowledge is then represented in a concentrated model.

To reduce false warnings of an intersection assistance function we need to know the exact lane each road user is on. The developed lane-hypotheses are introduced in this paper.

II. Motivation

Driver assistance systems for intersection scenarios are not available nowadays. Nevertheless road junctions are an accident hotspot in the EU27 countries as can be seen in fig.1. 43% of all accidents with injuries happen at intersections, mostly due to distraction of the driver, occluded field of view or misinterpretation of safe gaps.

Driver assistance systems consist of the three main parts information sources, environmental models and functions. For realizing reliable driver assistance systems for complex situations like intersection-related scenarios, the systems have to deal with a lot of information. Sources for information are for example onboard sensors, vehicle-to-x communication or digital maps. The quality and environment representation of this information sources differs for example in used description models, coordinate systems, measurement errors or latencies dramatically (see [2]).

The advantage of more precise sensor data, complex data-fusion and better prediction is however limited if we don't understand the scenario surrounding us and therefore give more meaning to the modelled environment. In simple words:

> *"I don't need 20 objects with perfect sigmas, I just want to know if it's safe what the driver is doing."*

Our environment model does not only need to provide all objects around the car, it has to answer abstract questions. Therefore the already introduced RoadGraph[3] was equipped with more functionalities like multi-lane hypotheses and object-lane fusion.

Functions that stand to benefit from this environmental model are for example traffic light assistance, right-of-way assistance, left and right turning assistance (see fig. 3).

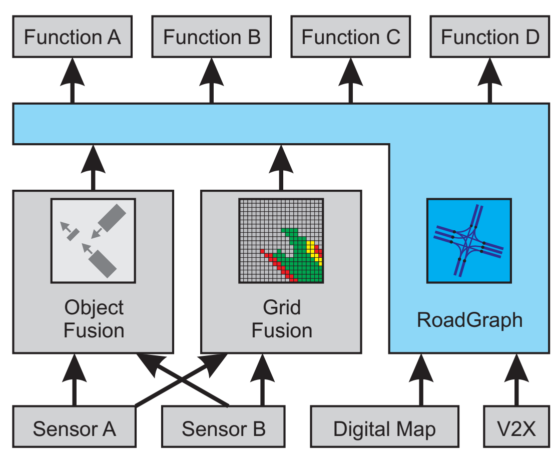
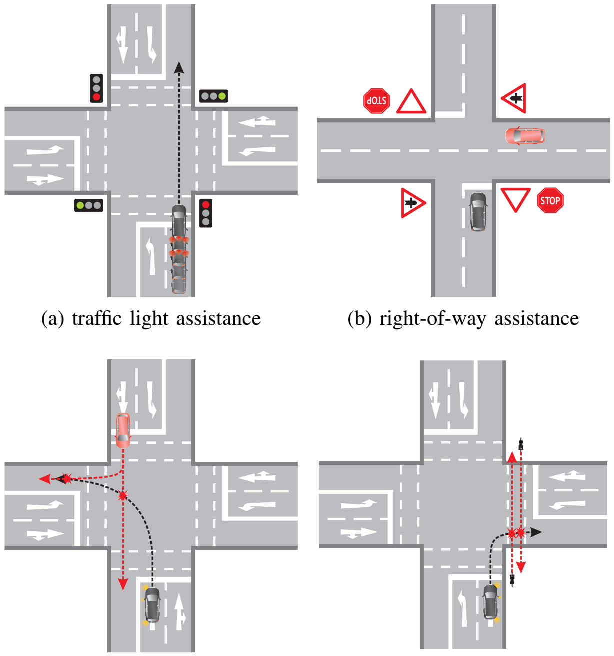


Fig. 2: Architecture for driver assistance systems using the RoadGraph and arbitrary sensors and functions

K. Homeier and L. Wolf are with the Institute of Operating Systems and Computer Networks, Technische Universität Braunschweig, Germany {homeier|wolf}@ibr.cs.tu-bs.de



(a) traffic light assistance (b) right-of-way assistance

(c) left turning assistance (d) right turning assistance

Fig. 3: Functions using the RoadGraph

III. RoadGraph

The RoadGraph is a directed graph composed of nodes and edges, based on the boost-graph c++ extension. This classic graph model is a well-established research field giving the advantage of fast access to elements by using standard programming techniques. The edges of the graph start and end at nodes which interconnect the edges, but hold no attributes. Edges represent single lanes of the streets and hold two or more positions to describe their trace. All additional attributes describing the scene are modelled to the edges as so-called lane side descriptions. Different IDs describe the relationship between edges, like if they belong to the same way (see fig. 4). Modelling streets on lane level gives the benefit of interpreting the scene in a very detailed level. Objects of the vehicle environment sensed by onboard sensors are a good example for such additional lane side descriptions. Since only objects which can be matched to a lane by their position and direction are relevant to overlying functions, it is important to model the lanes in high accuracy. This ensures a high rate of successful associations of environment information and lanes.

If a more detailed description of the street-network (e.g. at intersections) is needed, it is feasible to replace the existing description of an intersection with a more precise one. If an intersection is equipped with V2X technology it can communicate such a detailed geometry to replace the RoadGraph content in the supported area. For example additional turning lanes, positions of stop lines or other details can be added.

Intersections are defined objects holding information about the right-of-way situation like 'right-before-left' or 'traffic light controlled'. The edges on intersections connect lanes approaching and leaving the intersection. Their existence describes the possibility of going from one lane to another, which corresponds to the turning possibilities. The positions are modelled with the most probable way of traversing the intersection.

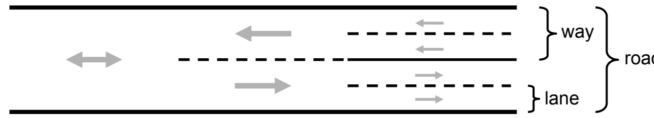


Fig. 4: Definition of road, lane and way

Lane side descriptions are implemented as template classes. They hold start- and end-parameters (where on the edge they are) and a specialized template parameter. This architecture is easily extendable with new attributes. Lane positions are given in a world-centered coordinate system (UTM - Universal Transverse Mercator), so everything associated with the RoadGraph has to be in these coordinates as well. This demands a relative high positioning accuracy of the ego vehicle to minimize errors in coordinate transformations.

IV. Object Sources

The RoadGraph is able to integrate information from different sources. Current data of the environment is measured by sensors. Additional to the position some sensors are able to measure the velocity. Object state is described by a state vector $\overrightarrow{x} = [x, y, w, l, v_x, v_y, ...]^T$. Using tracking algorithms the objects and their parameters can be followed over time (see [4]). Object sensors are for example laser scanners, radars or stereo vision systems. Besides using UTM as coordinate system as common base of all sources, the clocks of the different systems need to be synchronized to combine their data and to compensate transmission delays by predicting the objects to the same timestamp.

A. *Onboard Sensors*

Onboard sensors are able to measure the relative position of objects in respect to the ego vehicle. Laserscanners and radar-systems in our vehicle track the moving objects within an object-fusion module. The global position of the objects is determined, using the ego-position matching the object-time.

B. *Infrastructure Sensors*

An intersection equipped with infrastructure laser scanners, mounted at raised positions is able to monitor vehicles and vulnerable road users at the road-sectors and the area inside the intersection. The intersection-control itself propagates the results along with traffic-light statuses and road condition information via IEEE 802.11p.

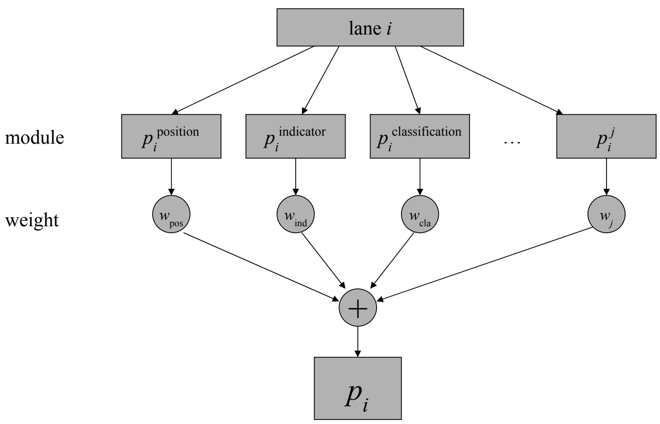
C. *CAM*

Vehicles following the SIM^{TD} standard[5] exchange a CAM (Cooperative Awareness Message) via IEEE 802.11p to distribute their position and status. This information is used as another external source.

V. Lane-Hypotheses

For many assistance functionalities at intersections it is important to know on which lane the own, as well as the other road-users are. We give some examples to demonstrate this:

- Stoplines on parallel lanes can be at different positions
- A right-turn assistance function can stay inactive, if the car is on a lane not going right
- A right-of-way warning does not have to warn of vehicles on lanes not interfering



$$p_i^j = \text{probability of lane i by module j}$$

$$\sum_i p_i^j = 1$$

$$w_j = \text{weight of module j}$$

$$\sum_j w_j = 1$$

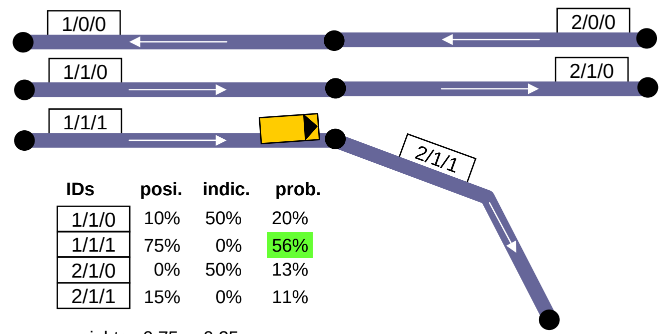
$$p_i = \sum_j w_j \cdot p_i^j = \text{overall-probability of lane i}$$

Fig. 5: calculation of lane-hypotheses

For each possible lane a road-user can be matched on, a hypothesis is calculated. An overall-probability p_i combined by several modules all given a probability from their point-of-view, along with some extra information forms this hypothesis. This is illustrated in fig 5. Typically all lanes of the same way along with the succeeding lanes are analyzed.

Finally the lane with the highest probability is selected. Objects are added as attributes to this lane and the ego-position is enriched with this information. If needed, functions can always access the less probable lanes later, since this information is not lost, but also stored with the objects and ego.

In fig. 6 a simplified example of the calculation is given. Our vehicle is mainly an lane 1/1/1, but also somewhat close to other lanes. Considering the localization-sigmas the position module calculates probabilities for the lanes. The second module in the example is the indicator-module, which distributes the possibility to all straight lanes, because the vehicles indicator in our example is off. (Note: on the lanes at the bottom, we expect the indicator to be on, because an intersection is modeled.) For lane 1/1/0 the overall-probability is $0.75 * 10\% + 0.25 * 50\% = 20\%$.



IDs	posi.	indic.	prob.
1/1/0	10%	50%	20%
1/1/1	75%	0%	56%
2/1/0	0%	50%	13%
2/1/1	15%	0%	11%
weights	0.75	0.25	

Fig. 6: example for lane hypotheses calculation

The modules used so far:

- Position: For the probability of each lane the overlapping area of the lane-width and the normal-distribution-curve is calculated. (fig. 7) This ensures strong results, when having high position accuracy with low standard-deviation and more distributed votes if the positioning results are bad, e.g. due to low number of satellites. [6]
- Classification: Size and speed are some inputs leading to a classification result like truck, car or bike. Cars will not be matched to bikeways and vice versa.
- Indicator: The indicator module votes for all lanes matching the indicator. The indicator is assumed to be off, when the comfort-indicator function for lane change is used. A lazy driver not using the indicator and therefore getting false warnings will have to accept this as his fault. Nevertheless the weight of this module should not be crucial.
- (Stereo) camera: This system detects arrows painted on the road. This is another independent information on which lane we are. If the system detects a left-arrow in front of the car, all lanes turning left at the next intersection are selected.

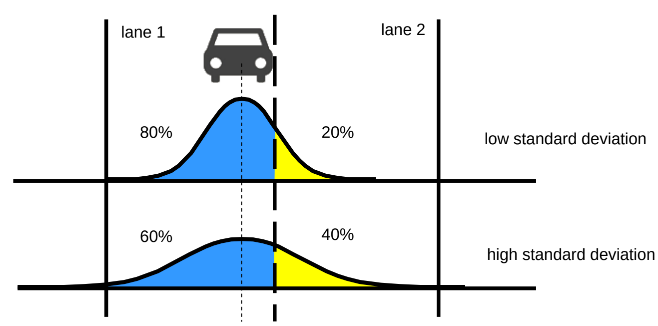


Fig. 7: calculation of p_i^{position} with low or high standard deviation

Lanes in the RoadGraph can have a reserved lane ID to mark them as bikeways. Especially at intersections this additional information is very helpful, to match and purposeful predict the taken paths of bikes. Because a bike much more often takes the wrong direction on its track, than a car would do this, we were forced to extend all algorithms to also match and predict these cases on bikeways. Besides that, all implemented RoadGraph functionality could be used for the bike extension.

This allows the easy use of the RoadGraph for the 'right turning assistance' presented in fig. 3d.

VI. Object-Merge

The challenge in merging objects from different sources (see section IV) lies in the disparity of used sensors, technologies, sample rates and latencies.

One solution would be to use a global tracking, working on raw-data from all sources. Since not all sources provide raw-data and synchronization of external objects is not feasible this solution was not considered. Another solution would be to add a second global tracking to process the objects that were already tracked independently for each source. Having two filter stages would result in lower reaction rates of the system. Therefore we decided to fuse (or merge) the different objects instead of track them without feedback of previous merged objects. This allows minimal jumping objects, but brings a very acceptable latency.

Another decision was to perform the RoadGraph matching step before the merging. The benefits of this are described in the following detailed description of the merge steps:

A. RoadGraph Matching

Objects are always primarily matched to the edges of the RoadGraph using the most probable lane hypothesis as described earlier. Assuming that cars belong to a distinct lane eliminates a lateral measure or model error, reducing the prediction and merging task to a one-dimensional problem.

B. Object Prediction

To merge these objects, first of all they have to be at the same timestamp. Therefore a prediction of objects is necessary, which we assume to only happen along edges of the RoadGraph. Without this assumption objects could be calculated to leave the lanes due to imprecise velocity-vectors.

All objects in the RoadGraph are predicted to the timestamp of the newest object in the system. Typically received objects are newer than those already in the RoadGraph, but due to higher latencies of external sensors, objects can also be older - than these objects have to be predicted.

C. Object Selection

Multiple objects are selected for merging if their similarity is above a threshold. The similarity of two objects is defined as inversely proportional to the difference of the position and the velocity along the edges. Not all objects have to be compared pairwise, only objects on the same and succeeding edges.

D. Merge Details

To allow easy top-down programming a merge routine was added to the objects functionalities. This allows easy changes in the used methods to raise performance or adapt to new sensor or value types later.

The objects are composed of different value types, which are handled as follows:

- Single value: The mean value is calculated.
- Optional values with a *valid* flag: Combining two sources of which one does not have a specific type of information will result in a completed object. If for example one sensor measures the length of an object and another sensor from a different position the width, the combined object will have both informations.
- Values with variance: Each value is weighted by a factor inversely proportional to its variance [7]:

$$\hat{v} = \sum_i \frac{v_i}{\sigma_i^2} \Big/ \sum_i \frac{1}{\sigma_i^2}$$

VII. Results

A. Simulation

To verify and estimate the allowed maximum variances of localisation and sensors the map-matching algorithms were tested in several simulation runs. One of the scenarios is illustrated in figure 8, showing three parallel lanes. The simulation task is to correctly map a vehicle with distance d to the middle lane. The values for localisation error y and yaw, as well as the sensor error yo are picked randomly matching their pre-defined σs for the test run, which is repeated several thousand times. We do not have to consider an error in x direction, though an error in the direction of the lanes would not change the matching result. Yaw (or heading) measurement of the ego vehicle on the other hand needs to be very precise to not match far away objects to their neighbouring lanes.

The results of this scenario can be seen in figure 9, where two of the three variances are fixed to typical values and the last one and d are altered. Reading this figure we can say for example, that if we would want a positive matching in 90%

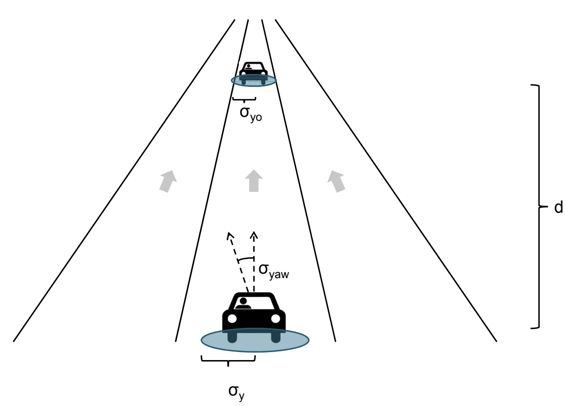


Fig. 8: simulation scenario

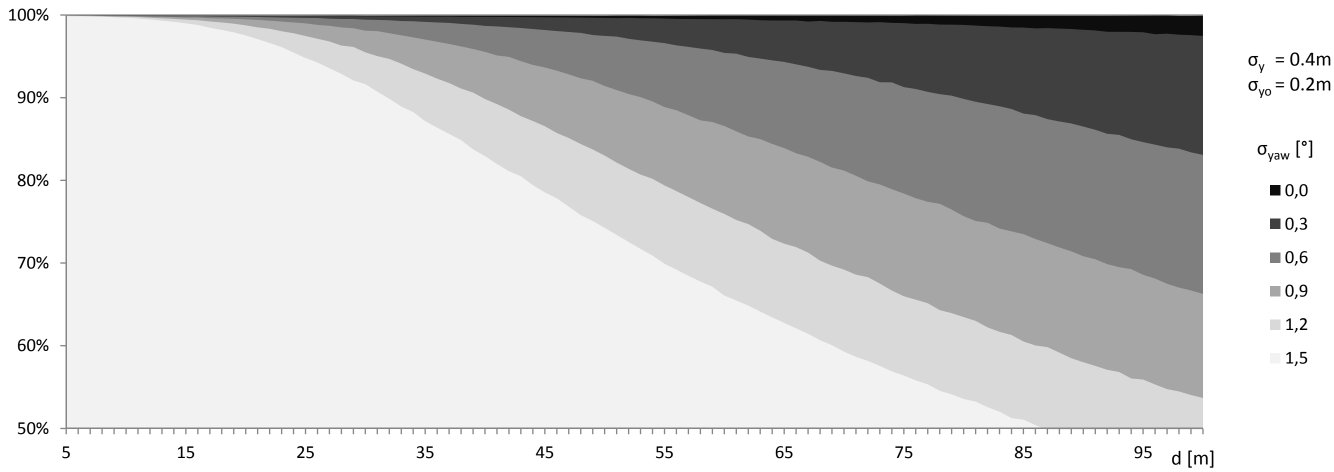


Fig. 9: correct hypotheses of a simulation run with varying σ_{yaw} and d

of all cases in a distance of 50m, the variance σ_{yaw} has to be bellow $1°$. These values can be easily reached by a high end localisation module. Our first experiences show that even the low cost variant is already getting close to reach these goals.

This scenario with a parallel lane on each side is among the most difficult scenarios. Without parallel lanes, the matching success would be at 100% even with great variances, because there are no alternative lanes the vehicle could belong to.

B. External Real Life Tests

A real life test with 30 runs covering six scenarios using the RoadGraph, conducted by an independent organization tests if a reference vehicle in an intersection is sensed and mapped to the correct lane [8]. The overall result is a proper matching in 98.6% of the time. In fig. 10 an example-run is given, that shows that even if one of the sources fails, the overall result is still positive.

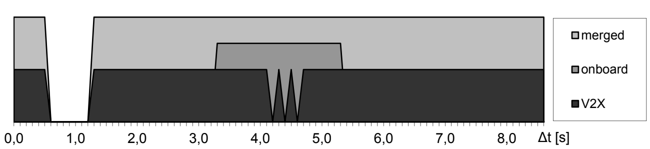


Fig. 10: one of the real-life test runs: A value different to zero means that the object is matched correctly

C. One Situation in Detail

Finally in fig. 11 the steps of processing incoming objects are shown. Objects from different sources with asynchronous data rates are received and preprocessed to the same data type. (11a). These objects are then matched to the edges of the RoadGraph, if possible (11b). This reduces the number of objects: Those not close to any edge are not of interest and those moving in directions not corresponding to RoadGraph-edges are possible 'ghost'-objects and dropped as well. In fig. 11c the merge step described in VI is performed, reducing the number of objects again. Finally what a function would receive from the RoadGraph are only the three objects shown in fig. 11d. Because our own vehicle is on a left-turn lane on an intersection with traffic lights (which are not shown, to keep the screenshots simple), only oncoming traffic is important for us at the moment. This example shows that we can reduce the amount of information from 'many object from different sources' to 'a few objects on conflicting trajectories'.

Another benefit is the similar representation of objects from different sources. Objects recognized by infrastructure-sensors can fill blind spots of our car, without notice of a function.

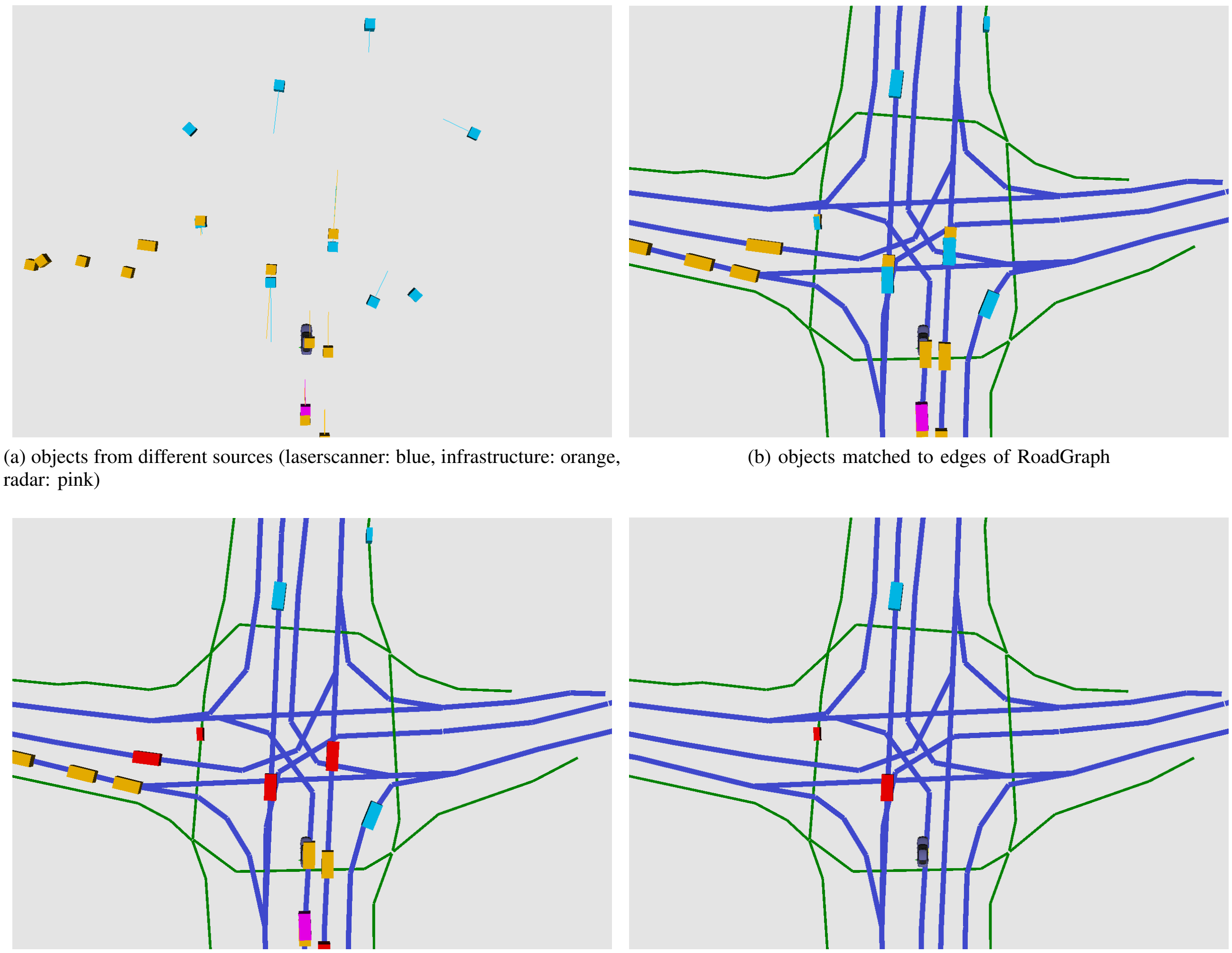
The development of new functionalities reacting on other road-users is reduced to asking the right question to the RoadGraph and then work on the few resulting objects, if any.

The screenshots and results shown in this example are recorded in real-life scenarios at an urban intersection.

VIII. Conclusions and Future Work

This paper describes a new approach to perform a high level sensor data fusion between objects and a street network. Working on the introduced 'RoadGraph', a graph based environment model with higher abstraction than typical object- and grid-models, simplifies the interpretation of complex scenarios. Merging objects on this generalized level shows great benefits in our tests, especially in the task of combining information from external sources with those from internal sensors.

An accurate modelled street network on lane level is the base for our novel approach. Without that the RoadGraph will not be able to perform its tasks. We think that the acquisition of these detailed information will soon be precise enough for our goals. Current navigation solutions are already and permanently collecting data about their taken trajectories. Although the single device has low quality, the



(a) objects from different sources (laserscanner: blue, infrastructure: orange, radar: pink)

(b) objects matched to edges of RoadGraph

(c) RoadGraph after merge step (merged objects: red)

(d) reduction to objects that are potentially dangerous

Fig. 11: Screenshots of the RoadGraph in a 3D-visualization

huge amount of probe data allows high quality reference tracks and it will just be a matter of time until these data finds the way back into available maps.

Future steps are to validate the presented methods for lane-precise map matching without an inertial measurement unit, but with an odometry coupled differential GPS solution. Another topic is to establish a server for enhanced communication for sharing detailed and up-to-date maps with the vehicles. This is needed because nowadays map data is constantly growing, while the storage of the cars is limited. Pushing map updates further to realtime applications, attributes like accidents, free parking lots, traffic light statuses or even the position of other vehicles could be distributed via a server, aggregating these informations from vehicles in that area.

References

[1] *INTERSAFE-2 Deliverable D3.1 - User needs and requirements for INTERSAFE-2 assistant system*, available via http://www.intersafe-2.eu, 2009.

[2] J. Knaup, M.-M. Meinecke, M.-A. Obojski, J. Effertz, G.-P. Ostermeyer: *Latency and Error Propagation for Cooperative Environmental Modelling*, 13th Forum on Advanced Microsystems for Automotive Applications, Berlin/ Germany, 5-6 May 2009

[3] K. Homeier, J. Knaup: *RoadGraph: Graph Based Environmental Modelling and Function Independent Situation Analysis for Driver Assistance Systems*, 13th International IEEE Conference on Intelligent Transportation Systems, 2010.

[4] Y. Bar-Shalom, X.-R. Li, *Multitarget-Multisensor Tracking - Principles and Techniques*, 1995

[5] *SIM^{TD} Deliverable D21.4 - Definition der Kommunikationsprotokolle*, available via http://www.simtd.de, 2009.

[6] U. Haak, A. Sasse, P. Hecker: *On the Definition of Lane Accuracy for Vehicle Positioning Systems*, 7th IFAC Symposium on Intelligent Autonomous Vehicles, 2010.

[7] L. Lyons, D. Gibaut, P. Clifford: *How to combine correlated estimates of a single physical quantity*, Nuclear Instruments and Methods in Physics Research A270, 1988

[8] *INTERSAFE-2 Deliverable D8.2 - Test and Evaluation Results*, available via http://www.intersafe-2.eu, 2011.