Stationary Urban Environment Modeling using Multi-Layer-Grids

Richard Matthaei, Gerrit Bagschik, Jens Rieken, and Markus Maurer
Technische Universität Braunschweig
Institute of Control Engineering
38106 Braunschweig, Germany
Email: <matthaei,bagschik,rieken,maurer>@ifr.ing.tu-bs.de

***Abstract*—Future advanced driver assistant systems and autonomous driving put high demands on the environmental perception especially in urban environments. Today's on-board sensors and on-board algorithms still do not reach a satisfying level of robustness and availability. In this paper we focus on the perception of the stationary environment which mainly consists of raised obstacles and lane-markings. We propose to fuse both the raised obstacles as well as the markings in a grid-based representation for a higher robustness against partial occlusion and false detections. The algorithms are designed to work with both research and close-to-production sensors. The algorithms run in real-time on a standard PC and are evaluated with real sensor data.**

I. Introduction

The research on future advanced driver assistant systems (e.g. an inner-city intersection assistant) and autonomous vehicles for urban environments points out the importance of a complete scene representation for a robust and predictive vehicle guidance. The scene's elements can be divided into two groups: stationary and movable elements. The movable elements, like other traffic participants, are not considered in this paper, but we focus on the stationary elements of a scene. The stationary elements, which are relevant for the previously mentioned systems, are given by raised obstacles (e.g. buildings, walls, bushes) on the one hand, and on the other hand lane markings and road boundaries. The raised obstacles give a rough impression of the environmental structure (e.g. the road course), whereas lane markings and road boundaries provide more detailed information especially on wide places where less raised obstacles are available (e.g. intersections and expansive roads). Hence, the road course and the lane course are defined by both: lane markings and raised surroundings.

The big variety of the urban environment involves a special challenge for perception systems. For example, the number of lanes varies frequently, intersections consist of lanes with different driving directions and the lanes are not parallel in all cases. Furthermore, there are gaps in the building-lines and abrupt changes in the facades of the buildings as well as streets without any lane markings. Especially in urban scenarios, partial occlusion is another big issue for perception systems due to the high amount of other traffic participants.

Today's on-board perception systems are not able to detect lane markings and to estimate the course of lanes in urban scenarios with the required availability and robustness [1]. Some research efforts focus on using detailed a-priori map data to compensate this lack of perceptional performance [2][3][4][5][6]. In our opinion more research has to be done on detecting and understanding the entire vehicle environment because we do not expect map data to be up-to-date and available in the required level of detail in the near future.

So a central task for inner-city scenarios is to robustly detect the course of the road and in more detail the course of the lanes. An algorithm for a robust extraction of the road-course has been previously described in [7]. In this paper we now focus on how to obtain a complete grid-based image of the surroundings as a reliable data-base to improve the road-extraction algorithm. One issue we could identify during our past research is the high amount of incomplete structures in inner-city scenarios which makes it difficult to estimate the road course over long distances just based on a snap-shot of environmental data.

The incompleteness is caused by the aforementioned characteristics of the stationary environment: incomplete building lines, missing lane markings and partial occlusion. Some of these issues can be solved by a mutual support of the raised objects and the lane markings (see Section VI). In the case of partial occlusion of the lane markings, for example, the information about the raised surroundings can help to estimate the further direction of the lane markings. Or, the other way around, if the stationary environment is interrupted on intersections, the information about the lane markings can support the estimation of the road course across the intersection. Furthermore, if parts of the stationary environment or lane markings are occluded by moving obstacles, these parts may have been perceived before or may be visible some moments later. As they are stationary, we can add them to the known environmental data with the knowledge about our motion.

All of these aspects have to be taken into account in order to robustly extract the course of the lanes and the roads in inner-city scenarios. For that reason we propose to fuse the raised and the ground structures into one consistent grid-based representation for a subsequent environment modeling algorithm considering both environmental feature groups at the same time. In this paper, we will use the Velodyne HDL-64 laser scanner as a proof-of-concept to detect raised structures, as well as lane markings. The algorithms are designed to be ported to close-to-production and series sensors as well, e.g. a combination of camera systems and single-layer laser-scanners. First results are shown in [7].

The aforementioned fusion of multiple grid-based representations leads to a new technical challenge: the grid-based processing becomes a bottleneck because all sensor data is fused into one single representation and all reading modules (e.g. lane-estimator, road-estimator or visualization) need an asynchronous access to this data. That is why a highly parallel access is required. A solution for this challenge is proposed in this paper as well.

II. RELATED WORK

Usually, information about the road course or the course of the own lane is directly extracted from image data by detecting the lane markings. A typical approach is to extract relevant features from the image and to estimate the course in a subsequent filter process. In this way, a certain amount of false-detections can be compensated and the noise of the measurement data can be reduced. A brief survey is given by [1].

Another approach using image data is presented in [8]. The image data is projected by an inverse perspective mapping (IPM) into a bird's eye view and the gray-scale values are accumulated over time in a grid-based structure. The possible detection range seems to be about 10 m. Unfortunately, the limits of this approach are not discussed in detail in [8], but one can imagine that the performance of subsequent applications would be strongly limited due to a short detection range. This approach is described in [8] as an additional possibility to generate a grid-based representation beside the known approaches of accumulating laser or radar data in a grid.

The grid-based processing of sensor data is already well known (e.g. [9], [10], [11]) and still part of current research activities. Especially the so called occupancy grid with the Bayesian update for raised obstacles is widely used (e.g. [12],[13],[14]). In [3] a grid-based structure is filled by the reflectance values of a laser-scanner. The result is very similar to the approach of [13], but the sensor-data is much more precise. Additionally, the gray-scale grid is augmented to a probabilistic representation [3].

Many of the approaches extracting the road course from laser or radar data work only on occupancy grids (e.g. [14], [15], [16], [17]). Some of them are using multiple grid-based representations ([15], [16]), but only [15] is fusing both representations into one solution. The lane markings are not considered in any of these solutions. In [18] measurements of a laser-scanner are used to fill an occupancy grid with ground-reflections, but the raised surroundings are not taken into account.

Based on our experiences in estimating the road course as presented in [7], it would be beneficial for further processing to first fuse the results of the lane information and the raised obstacles. One problem for the perception system is that – due to partial occlusion, gaps in building-lines or missing lane-markings – the road course in urban environments is often only partially represented. A fusion of both information sources, the raised obstacles and the lane markings, combined with an accumulation of the measurements over time would give a more continuous low-level description of the entire road course. This makes it easier to extract the relevant data. To the best knowledge of the authors, such an approach has not yet been presented.

III. LIDAR PROCESSING CHAIN

To obtain satisfying results for the grid representations some preprocessing of the sensor data has to be done. The entire processing of the Velodyne data is illustrated in Figure 1.

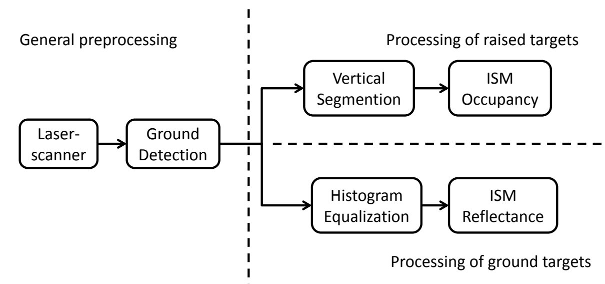

Fig. 1: The lidar processing at a glance.

First of all, we have to distinguish between laser reflections from the ground and reflections from raised objects. After separating ground targets from the raised ones the processing chain splits into two chains: one for the raised targets and one for the ground reflections. These three resulting processing sequences are explained in more detail in the following.

A. General Preprocessing

We separate the Velodyne sensor data into the 64 layers and the channels with their typical horizontal discretization. Afterwards, all data of a total 360°-scan is collected taking the sensor calibration into account. This means especially that the vertical order of the layers is sorted by their physical angle and the horizontal misalignment of the laser diodes is compensated as well. Ground targets are then detected by comparing the distance of adjacent layers within each channel, as described in [19]. The key feature is the distance between two targets of the same channel in two adjacent layers. We define an expected distance between both targets assuming they are on the ground level ($z = 0$) which is then compared to a calculated distance derived from real measurements. We use the given sensor mounting height and a tolerated vehicle pitch and roll movement for the calculation of the expected distance.

If the distance of the measurements is significantly smaller than the previously mentioned expected distance, the targets seem to be above each other and thus do not belong to the ground level. The resulting classification of the laser targets is illustrated in the upper part of Figure 2.

B. Processing of Ground-targets

If the applied sensor gives enough information about the reflectance (like the Velodyne laser-scanner), this information can be used to create a gray-scale grid, similar to [3].

To compensate the different characteristics of the laser diodes we apply a layerwise histogram expansion within the ground targets of each layer as known from image-processing. The histogram equalization yields a contrast maximization of the reflections. So, the required information for the lane detection becomes clearer. The histogram expansion runs online and adapts itself to local changes. We currently do not apply an overall intensity calibration as proposed by [3]. Due to the histogram expansion a certain adjustment between the diodes can already be noticed which seems to be sufficient in a first step. The results of this preprocessing steps is shown in the lower part of Figure 2.

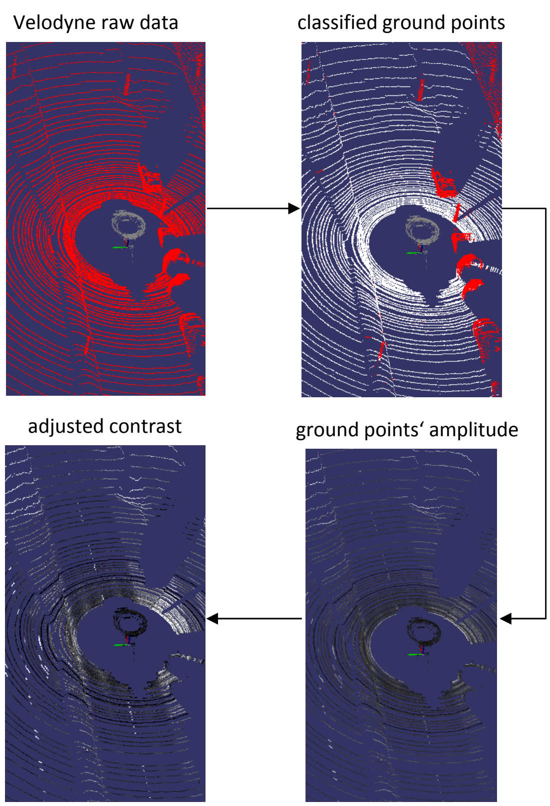

Fig. 2: Results of the Velodyne raw data processing steps: ground detection (upper right, ground: white dots) and contrast adjustment (lower left).

The laser reflections are then accumulated in the reflectance layer, using a simple low-pass filter as mentioned in section IV-B.

C. Processing of raised targets

After the ground points have been classified, we use the (r, ϕ)-part of the cylindric point coordinates to add the elevated points to an artifical polar grid, converting the single-target 3D point cloud into a multi-target 2.5D stixel representation (as introduced in [20]). This algorithm is applied in [19] as well. It can be applied to the Velodyne data as well as to close-to-production sensors. The applied polar-grid has an adaptable angular and radial resolution. Currently we use an angular resolution of 0.2° and a radial discretization of 0.2 m. Thus, it reduces the total amount of data. Additionally, only cells with more than one hit and with a difference in height greater 0.5 m between the associated points (in the current state of development) will be marked as valid targets. By this, we remove flat structures which were not correctly classified as ground points. Although, theoretically some measurements get lost applying this strategy, we did not notice any significant errors in the representation of the environment for our applications: Curbs and other structures of medium height are currently handled either as ground points or as raised targets, but in both cases the structure of the environment is represented. In future work we will focus on the correct handling of curbs and other structures of medium height as raised targets because they are relevant for a collision-free driving.

The resulting data is processed as a virtual scan and used for updating the occupancy grid. Furthermore, this data is clustered by a segmentation algorithm and is used for object tracking. The problem of moving object detection for cleaning up the occupancy grid has been studied by past work (e.g. [21], [22], [7]) and is not part of this paper.

IV. Grid-Based Environment Modeling

A. Multi-Layer-Concept with Fusion

Derived from mobile robots, systems based on different sensor technologies (e.g. ultrasonic, mono- and stereovision and lidar) are applied in production vehicles nowadays. The challenge of low-cost sensor systems is to fuse the information in such a way that the overall system has a higher level of robustness and detection rate. As proposed in [11] a separate occupancy grid-layer has to be used for each sensor technology. Additionally, each state vector requires a separate grid-layer, as well. To handle different layer types and fuse them to a single layer representation we propose a concept as illustrated in Figure 3. Each layer with its state vector will be abstracted to a so called *tristate* representation which allows us to compare the layers. In this paper we use the well-known *occupancy* and the *reflectance* layer. The layer types, the tristate and an example of a fused layer are shown in Figure 5.

B. Layer types for the Ground Representation

Figure 4 shows examples for possible ground representations as they were introduced in Section IV-A.

In addition to the well known occupancy layers for raised stationary obstacles, we use two different ways of representing the ground by a grid-based approach. The first one (left image in Figure 4) is similar to the occupancy grid. This approach has already been introduced in [18] and [7]. The incoming laser reflections classified as ground-targets are accumulated in a Bayesian grid. This allows us to collect data of the road markings even with a close-to-production laser scanner, which does not provide sufficient data about the reflection's intensity and gives only a sparse set of marking measurements.

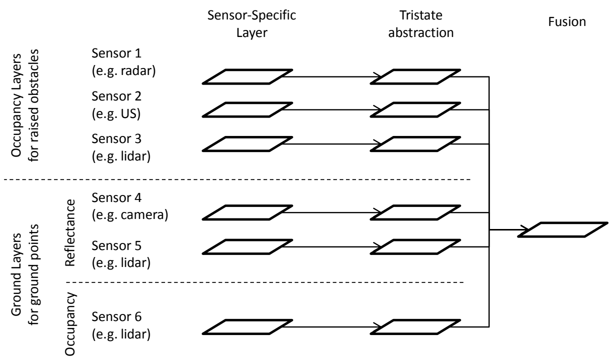

Fig. 3: Multi-layer-concept for fusion of the entire stationary environment with a sensor-specific threshold-decision.

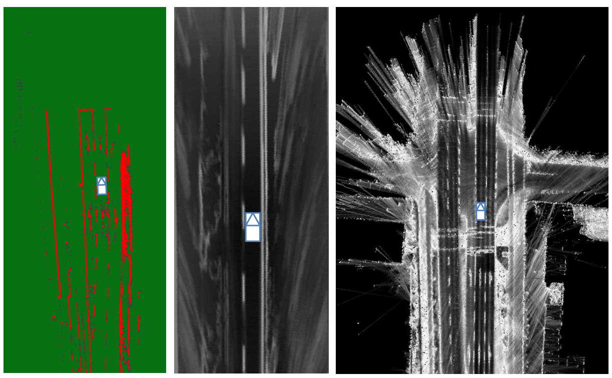

Fig. 4: Our two representations of the ground filled by three sensors. Left: Occupancy grid filled by close-to-production laser-scanner. Center: Reflectance grid filled by camera data. Right: Reflectance grid filled by Velodyne laser-scanner.[23]

For filtering gray-scale values as well, we introduce a second layer: the reflectance layer. This reflectance layer can be filled by laser data (e.g. [3] or right image in Figure 4) or camera data (e.g. [8] or center image in Figure 4). The reflectance layer accumulates the gray-scale values of a laser-scanner or gray-scale camera. The initial update of a cell is just set by the measurement value v_{meas}, in the further processing the cell's value v_{cell} is fused with the incoming data over time using a low-pass filter with the weight $k = [0..1]$. The filter equation is given by:

$$v_{cell} = k \cdot v_{cell} + (1 - k) \cdot v_{meas} \tag{1}$$

C. *Fusion of Grid-Layers*

As motivated in Section I we propose to fuse the entire stationary environment into one single representation (see Figure 5). Due to different state vectors of every single layer we have to abstract the layer as a *tristate* representation where the cells contain the state *unknown*, *free* or *occupied*. This step makes it easy to change the state vector of the original layer (e.g. from reflectance value to occupancy value or from Bayesian representation to a Dempster-Shafer representation). Additionally, only with this abstraction step it is possible to fuse different state vectors of the grid (e.g. reflectance with occupancy) for a complete image of the stationary surroundings.

In the resulting fusion layer we can distinguish between areas which are occupied by raised obstacles and areas which are 'occupied' by markings (see Figure 5).

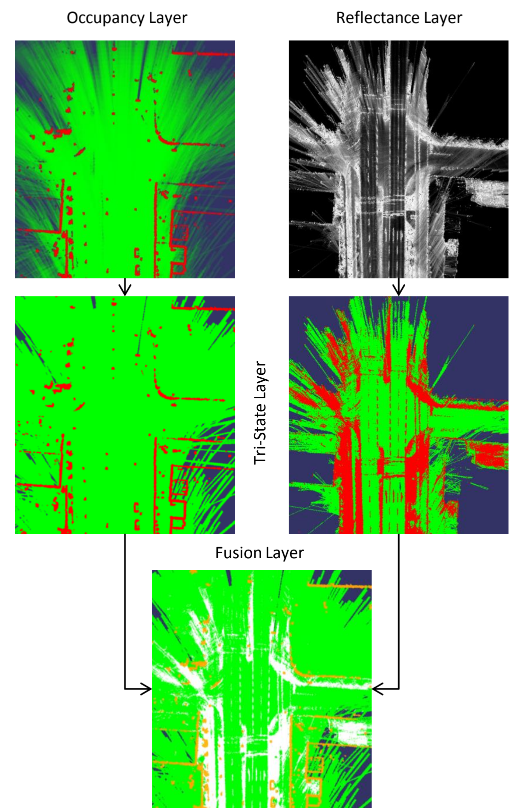

Fig. 5: Processing-Steps for grid fusion. In this case the Velodyne laser-scanner is used.[23]

V. Implementation of Multi-Layer-Grid

The data stored in the grid is used by a number of instances, for example the tristate-fusion modules, inverse sensor models and visualization modules. Our grid implementation is based on a block structure as already proposed in [24, pp. 107 - 109] to be able to fuse serveral possible data inputs in the future. This structure allows multiple write accesses to a layer instance, as long as there is only one sensor updating one block at the same time. The grid is being moved (shifted) with the movement of the vehicle, keeping the orientation from its initialization and thus not requiring a rotation of the cells. The shifting is done block-wise in a separate thread.

The classical approach to ensure data consistency while writing data into a layer in multi-threading environments uses a locking scheme, i.e. read-write-mutexes. These mutexes allow several parallel read-accesses, but only one write-access at a time. When a write-lock is requested, all read-accesses have to be finished and, even worse, no new read-access is given as long as the write-access is not finished.

The fusion module requires a fixed layout of all used layers when calculating the resulting cell values. With the mentioned locking-scheme, this implies that no new data could be added to any of the connected layers during the fusion process. Additionally, the movement of the layer has to be prevented. For complex fusion algorithms with several input layers and high computational costs, this will cause a lot of waiting-times and latencies in the data processing. In the worst case, these waiting-times may ruin the real-time capability of the entire perception system.

To avoid this issue, we have implemented a layer access scheme with block-based queues and reference-counting, as shown in Figure 6.

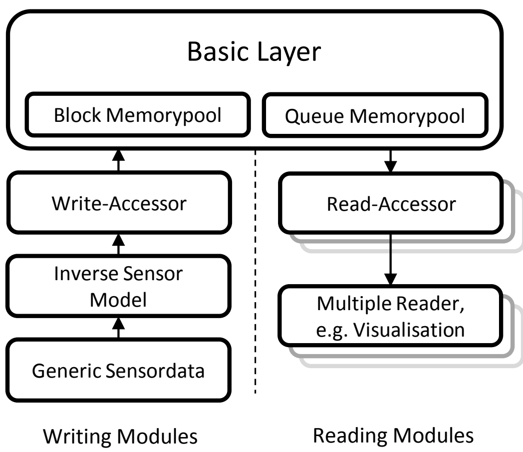

Fig. 6: Overview of grid-implementation.

This implementation does not rely on locking the underlying data, because it is based on a mark-and-sweep-algorithm. The memory-pool allocates a certain amount of block instances on start-up to prevent frequent computational expensive memory allocations and deallocations. The memory-pool also holds references to used and free blocks and performs an on-demand recycling of free blocks. Each block in the layer layout is represented by a block-queue. This block-queue manages the access to the block-data. As soon as the block is taken from the memory-pool the block's reference-counter is initialized with 1. This marks the block's data as *in use* and prevents the block from being recycled even if no module is currently working on this block.

The block-queue always provides a ready-to-read block which stores the current cell values and will be shared among all reader-instances. Upon requesting any read-access to a block, the reference counter is incremented. The reference counter is decremented whenever an access is terminated.

When a write-access is requested (e.g. by the inverse sensor model or by a cyclic fusion request), a new block is taken from the memory pool, initialized with the current contents of the readable block and given to the writer instance. The writer instance updates this copy while the original block is still available to other readers as shown in Figure 7.

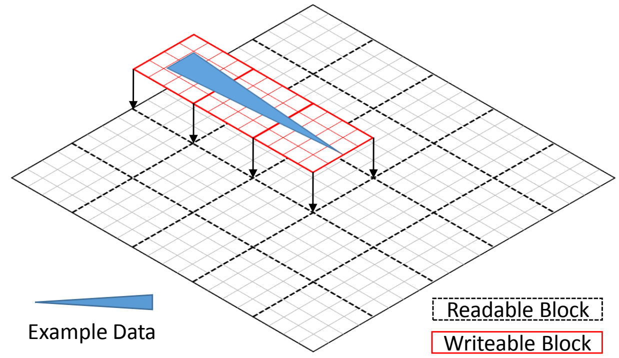

Fig. 7: Blockwise write-access concept.

After finishing the update, the writer instance returns the block to the block-queue. The block-queue exchanges the current ready-to-read block by the updated block. The new block is now available to all subsequent read-accesses. Pending readings to the last ready-to-read block will not be aware of the exchange and can finish their access. These blocks are preserved from being recycled by the reference counting mechanism.

This scheme allows a nearly-parallel read-write-access to the layer data. Parallel write-access to one block is still mutually excluded to prevent data corruption and it is ensured that every read-access gets the newest available data set.

The layer is defined by a certain layout of the block-queues. The access to the layer is abstracted by so-called accessor-classes. These accessor-classes copy the layer layout at the time of acquisition. Each time an accessor acquires the current layout, the reference counter on each queue is incremented to prevent the block-queue from being recycled by the memory-pool. So this approach to organize the block-queues is similar to the organization of the blocks themselves and is needed for shifting the entire layer.

While shifting the layer, the block layout of the layer will be changed according to the vehicle's movement since the last shift cycle. Some block-queues, which are then out of range will be removed, new ones coming in range will be added. The shifting algorithm ensures the correct physical alignment of the blocks, i.e. the block-queues' positions within the layer-layout are adapted, but not their position in the world.

VI. Results

We used our test vehicle *Leonie* of the research project Stadtpilot [4] for evaluation. It is equipped with a Velodyne laser-scanner and an IMAR iTrace as INS-DGPS-system. In this setup, we use the highlighted parts of the fusion concept shown in Figure 8.

For our tests we used a grid with a dimension of 150 m x 150 m and a cell resolution of 0.2 m for the bayes, reflectance and resulting tristate fusion layer. Each layer was separated in 15 blocks in each direction. Our implementation allows the real-time processing of the incoming Velodyne laser

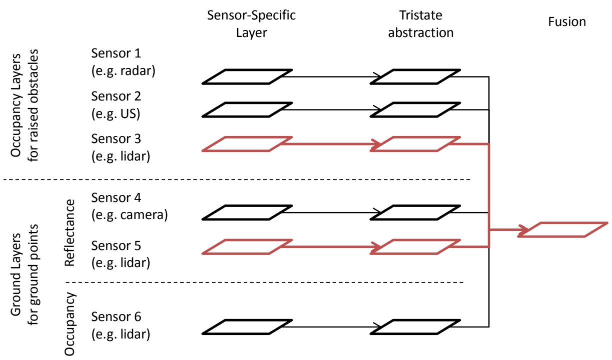

Fig. 8: Applied parts (red) of the fusion concept for the setup with the Velodyne laser scanner.

data with a scan-frequency of 10 Hz. We measured an average execution time of 35 ms for the raised targets and 25 ms for the ground reflections. The total processing time averaged about 45 ms, as both layers share some preprocessing modules. The detailed timings are shown in figure 9. All calculations were done on an Intel i7-4770 processor with 8 GiByte of RAM.

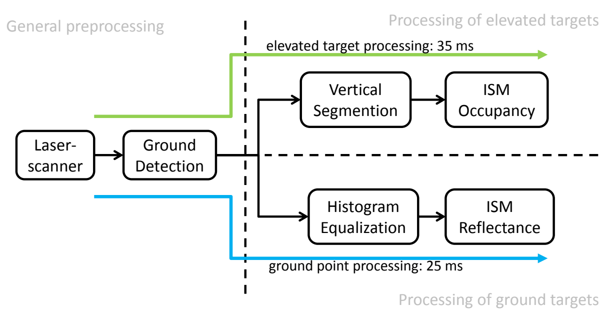

Fig. 9: Measured timings in the sensor data processing chain.

The grid implementation is capable of parallel read- and write-accesses. This flexibility, mainly required by the fusion module, comes with an extended amount of memory usage. The memory usage increases with the number of readers and the processing time of each module, because there is a *snapshot* of the grid from the reqeusted time. The used blocks can not be recycled before the modules finished their processing and according to that, new block have to be allocated. As each reading access to a grid-block prevents the block from being recycled by the underlying memory pool, the memory usage grows. In this configuration, we measured a memory pool usage up to four times the theoretical cell count of the layer, resulting in around 35 MiByte of memory usage for the bayes and reflectance layer when using a 64-bit double precision floating-point number as cell datatype. But – and this is more important for the real-time processing – the processing time for writing the sensor-data into the layers is not influenced. So all data will be processed in real-time from the sensor's point of view.

Due to the collection of an entire 360°-scan and the access to a layer, which consists of a mixture of cells updated in the last and in the current scan, some data might have a delay. As we assume the perceived data to be stationary and we know about the motion of the host vehicle, this delay is not an issue. The delay is only relevant for Bayes or reflection-value for the reflection layer and leads to a reduced sight distance, but does not yield a wrong position of the targets. This a major advantage to issues with latency in tracking algorithms of moving obstacles, where a delay in the filtering process has an influence on the object's position as well.

Figure 10 shows the benefit of the grid-based approach for road detection. Even though there are occluded regions due to other traffic participants (orange areas), the low-level information about the lane-markings is available for extraction (white triangles).

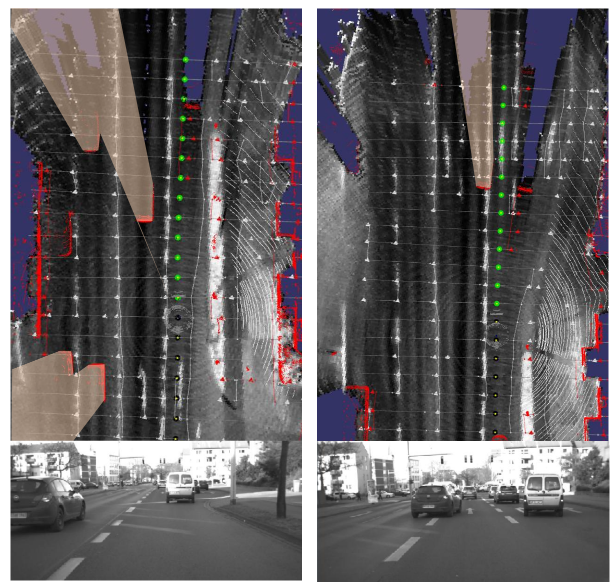

Fig. 10: Extracted road course based on markings and building lines. Background: reflectance-layer (for better visualization, even though the road course is extracted from the tristate-layer), foreground: sensor data of the current scan. Orange areas: occlusion for ground detection, white triangles: extracted road-markings, red triangles: extracted raised obstacles, green dots: online estimated reference-line for the piece-wise extraction (see [7] for more details).

The benefit of fusing the entire stationary environment into one single representation becomes clear in Figure 11. While the lane markings are not available due to occlusions or limited detection range, the range for a rough estimation of the road course can be extended by using the stationary environment. In this case, we are able to extract the course of the road up to 50 m ahead (see Figure 11, left). In another case, the building has a gap due to a crossing street. Just looking for continuous lines in an occupancy grid would lead to an interruption of the extraction. Due to the fusion of the stationary obstacles with the road markings the algorithm is able to span this gap and give a robust estimate of the road course even in this case (see Figure 11, right).

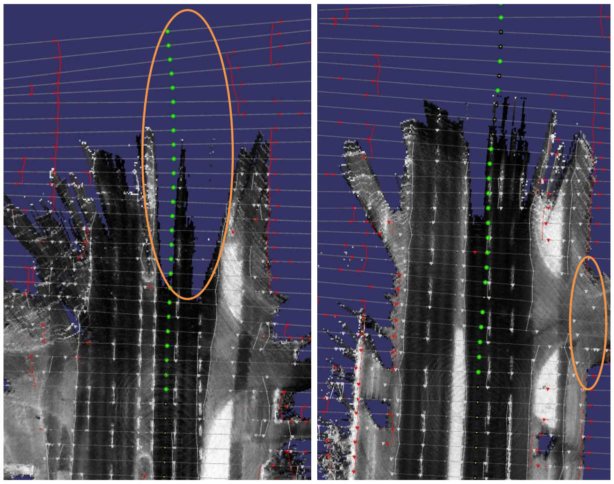

Fig. 11: Correct extraction of the road course even in cases of missing features. Left: Lane markings are missing, but the building line helps to estimate the rough course up to 50 m ahead. Right: A crossing street leads to a gap in the building line. This can be compensated by using the road-markings.

VII. Conclusion and Future Work

In this paper, we have presented a new approach for a complete low-level representation of the stationary environment with the focus on inner-city scenarios. We have shown that the fusion yields benefits for the extraction of the road course, especially in scenes with occlusion or incomplete and interrupted building lines or road markings. We have also presented a technical solution for a real-time processing of the computational expensive grid-processing. The identified bottleneck in the fusion-layer is solved using a quasi lock-free implementation of the entire grid processing, so that the algorithms run in real-time on a standard PC.

For our future work we will continue our activities towards environment modeling. A special challenge will be using close-to-production sensors like cameras or less expensive laser-scanners as well. First results of the usage of those sensors were already presented in [7]. The extracted road course or more detailed lane course can then be used for localization purposes [23] and scene-estimation.

Acknowledgment

The authors wish to thank Toni Günther and Jaebum Choi for their support.

References

[1] A. Bar Hillel, R. Lerner, D. Levi, and G. Raz, "Recent progress in road and lane detection: a survey," *Machine Vision and Applications*, pp. 1–19, 2012.

[2] J. Knaup and K. Homeier, "RoadGraph - graph based environmental modelling and function independent situation analysis for driver assistance systems," in *Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on*, Madeira, Portugal, Sep. 2010, pp. 428 –432.

[3] J. S. Levinson, "Automatic laser calibration, mapping, and localization for autonomous vehicles," Ph.D. dissertation, Stanford University, 2011.

[4] J. M. Wille, F. Saust, and M. Maurer, "Stadtpilot: Driving autonomously on braunschweig's inner ring road," in *Intelligent Vehicles Symposium (IV), 2010 IEEE*, San Diego, CA, 2010, p. 506–511.

[5] A. Geiger, M. Lauer, F. Moosmann, B. Ranft, H. Rapp, C. Stiller, and J. Ziegler, "Team AnnieWAY's entry to the grand cooperative driving challenge 2011," *IEEE Transactions on Intelligent Transportation Systems, to be published*, 2012.

[6] T. Weiss and K. Dietmayer, "Automatic detection of traffic infrastructure objects for the rapid generation of detailed digital maps using laser scanners," in *Intelligent Vehicles Symposium, 2007 IEEE*, 2007, p. 1271–1277.

[7] R. Matthaei, B. Lichte, and M. Maurer, "Robust grid-based road detection for ADAS and autonomous vehicles in urban environments," in *Proceedings of the 16th International Conference on Information Fusion (FUSION), 2013*, Istanbul, Turkey, Jul. 2013, pp. 1 –7.

[8] M. Konrad, M. Szczot, F. Schüle, and K. Dietmayer, "Generic grid mapping for road course estimation," in *Intelligent Vehicles Symposium (IV), 2011 IEEE*, 2011, p. 851–856.

[9] A. Elfes, "Sonar-based real-world mapping and navigation," *Robotics and Automation, IEEE Journal of*, vol. 3, no. 3, pp. 249–265, 1987.

[10] H. P. Moravec, "Sensor fusion in certainty grids for mobile robots," *AI magazine*, vol. 9, no. 2, p. 61, 1988.

[11] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents series)*, ser. Intelligent robotics and autonomous agents. The MIT Press, Aug. 2005.

[12] M. Bouzouraa and U. Hofmann, "Fusion of occupancy grid mapping and model based object tracking for driver assistance systems using laser and radar sensors," in *Intelligent Vehicles Symposium (IV), 2010 IEEE*, 2010, p. 294–300.

[13] M. Konrad, M. Szczot, and K. Dietmayer, "Road course estimation in occupancy grids," in *Intelligent Vehicles Symposium (IV), 2010 IEEE*, San Diego, CA, 2010, p. 412–417.

[14] T. Weiss, B. Schiele, and K. Dietmayer, "Robust driving path detection in urban and highway scenarios using a laser scanner and online occupancy grids," in *Intelligent Vehicles Symposium, 2007 IEEE*, Istanbul, Turkey, 2007, p. 184–189.

[15] M. Darms, M. Komar, and S. Lueke, "Map based road boundary estimation," in *Intelligent Vehicles Symposium (IV), 2010 IEEE*, 2010, p. 609–614.

[16] H. Loose, U. Franke, and C. Stiller, "Kalman particle filter for lane recognition on rural roads," in *Intelligent Vehicles Symposium, 2009 IEEE*, 2009, p. 60–65.

[17] A. Müller, M. Himmelsbach, T. Lüttel, F. v. Hundelshausen, and H.-J. Wünsche, "GIS-based topological robot localization through LIDAR crossroad detection," in *14th International IEEE Conference on Intelligent Transportation Systems (ITSC), 2011*, 2011, p. 2001–2008.

[18] F. Homm, N. Kaempchen, and D. Burschka, "Fusion of laserscannner and video based lanemarking detection for robust lateral vehicle control and lane change maneuvers," in *2011 IEEE Intelligent Vehicles Symposium (IV)*, Jun. 2011, pp. 969 –974.

[19] J. Choi, S. Ulbrich, B. Lichte, and M. Maurer, "Multi-target tracking using a 3D-Lidar sensor for autonomous vehicles," in *16th International IEEE Conference on Intelligent Transportation Systems (ITSC), 2013*. IEEE, 2013, p. 881–886.

[20] H. Badino, U. Franke, and D. Pfeiffer, "The stixel world-a compact medium level representation of the 3d-world," *Pattern Recognition*, pp. 1–10, 2009.

[21] R. Matthaei, H. Dyckmanns, M. Maurer, and B. Lichte, "Consistency-based motion classification for laser sensors dealing with cross traffic in urban environments," in *Intelligent Vehicles Symposium (IV), 2011 IEEE*, Baden-Baden, Germany, Jun. 2011, pp. 1136 –1141.

[22] R. Matthaei, H. Dyckmanns, B. Lichte, and M. Maurer, "Motion classification for cross traffic in urban environments using laser and radar," in *Proceedings of the 14th International Conference on Information Fusion (FUSION), 2011*, Chicago, Jul. 2011, pp. 1 –8.

[23] R. Matthaei, G. Bagschik, and M. Maurer, "Map-relative localization in lane-level maps for ADAS and autonomous driving," in *Intelligent Vehicles Symposium (IV), 2014 IEEE*, 2014.

[24] J. Effertz, "Autonome Fahrzeugführung in urbaner Umgebung durch Kombination objekt- und kartenbasierter Umfeldmodelle," Ph.D. dissertation, TU Braunschweig, Braunschweig, 2009.