A Fast Ransac Based Approach for Computing the Orientation of Obstacles in Traffic Scenes

Florin Oniga and Sergiu Nedevschi
Computer Science Department, Technical University of Cluj-Napoca, Romania
Florin.Oniga@cs.utcluj.ro, Sergiu.Nedevschi@cs.utcluj.ro

***Abstract* — A low complexity approach for computing the orientation of 3D obstacles, detected from lidar data, is proposed in this paper. The proposed method takes as input obstacles represented as cuboids without orientation (aligned with the reference frame). Each cuboid contains a cluster of obstacle locations (discrete grid cells). First, for each obstacle, the boundaries that are visible for the perception system are selected. A model consisting of two perpendicular lines is fitted to the set of boundary cells, one for each presumed visible side. The main dominant line is computed with a RANSAC approach. Then, the second line is searched, using a constraint of perpendicularity on the dominant line. The existence of the second line is used to validate the orientation. Finally, additional criteria are proposed to select the best orientation based on the free area of the cuboid (on top view) that is visible to the perception system.**

Keywords— obstacle detection; oriented cuboids; autonomous driving; lidars

I. Introduction and Related Work

In the field of autonomous driving, the sensorial perception system is one of the key elements. Common sensors used are cameras, mono or stereo, radars, or lidars, often in combination to sum up the strengths of each sensor type.

Lidar sensors have one or multiple layers used for scanning the surrounding environment. A number of accurate 3D measurements are provided for each layer, with a certain angular resolution. Recently, lidar sensors with 16 or 32 layers have been introduced, as an intermediate solution between the common 1-, 2-, 4- layer and the expensive 64-layer lidar.

Various obstacle models are proposed in the literature, the cuboid (parallelepiped) being the most common one. It allows a compact representation and fast higher level processing (such as tracking). Other potential models are simplifications of the cuboid model, such as lines or rectangles (on the bird's eye view of the environment), or more complex such as poly-lines and clusters of grid cells/voxels. The cuboid is a good compromise for autonomous driving applications, as the interest obstacles (vehicles, urban structures, pedestrians etc.) can be delimited with this simple geometric shape. The cuboid model is less appropriate for obstacles with irregular shape, and, when the obstacle axes are not aligned with the reference system. For the latter issue, the solution is to find the right orientation of the cuboid in order to align the cuboid sides with the obstacle boundaries.

Most lidar obstacle detection systems rely on the L-shape model for detection, and this model is usually applied in the polar coordinates representation, along the same layer of a lidar. In [1], lidar 3D measurements are first clustered by scanning along the same layer and grouping points in the same cluster, if they are close in angular distance. Then, for each cluster the L-shape is extracted by initially fitting a line to the extreme points of the cluster. Iteratively, the line is split until an L-shape is obtained, which provides both the location and the orientation of the obstacle. Tracking is further employed to smooth the results and to estimate dynamic features. 4-layer lidars are used, that are placed in the front bumper, and oriented quasi parallel with the road.

The approach from [2] proposes the fitting of the L-shape as an optimization problem, without using the polar representation. A measure is proposed, in a least-squares sense, for how good the L shape fits a cluster of points representing an obstacle. The best fit of the L-shape is then searched by computing the fitness measure for each possible orientation. The method required about 4 ms of processing time for each obstacle, mainly due to the exhaustive search needed.

Points are clustered in potential obstacles in [3] by using angular and range thresholds, directly in polar coordinates. Each cluster is fragmented into edges by applying the iterative end point algorithm, initially proposed in [4]. For each segment, the line is computed with generalized least-squares fitting and used as primitives for tracking.

Either a line or a rectangle is fit to the cluster of points from an obstacle, with least-squares, in [5]. The rectangle fit is preferred if available. Tracking is then used for temporal filtering and smoothing. 1-layer lidars are used.

In [6] the lidar scan data, from one layer, is first segmented into obstacle clusters using the adaptive point break algorithm

978-1-5386-8445-0/18/$31.00 ©2018 IEEE

proposed earlier [7]. For each cluster, two perpendicular lines are computed with least-squares minimization. Each point from the cluster is considered as pivot (the common point of the two lines) and the best pair of lines is found. The point scan order must be known, and the average processing time is 1.5 ms for each obstacle cluster.

A low complexity approach for computing the orientation of cuboidal obstacles will be presented in this paper. The proposed contributions are as follows: a model consisting of two perpendicular lines representing the two potential visible sides of rectangular (or quasi-rectangular obstacles), model that is fitted to the obstacle boundaries using a RANdom Sample Consensus approach (RANSAC, [9]); a strategy to validate/choose the best orientation based on the occupied area of the cuboid in the bird's eye view (top view).

The method that will be presented does not require the original scan order of the points and the processing of each lidar layer individually for computing the orientation. Thus, it is a more general approach, suitable for lidars with many layers. Instead of fitting the two line model with least squares, iteratively for each pivot point, as in [6], a random consensus approach will be used: the first line is computed, and then, the second line is searched on the perpendicular direction. This allows a lower computational complexity for the method, as it will be later discussed in the results section.

In the next sections, an overview of the proposed approach will be presented, the main steps will be detailed, and, finally, results and evaluation of the performance are presented and analyzed.

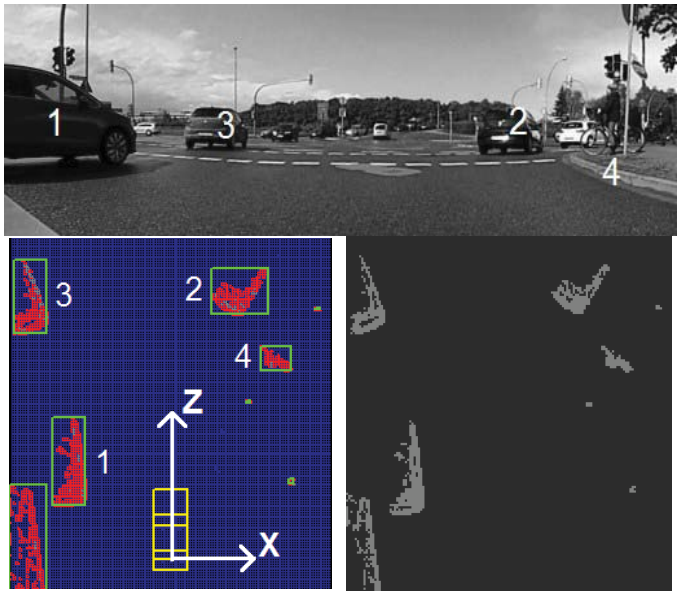


Fig. 1. Input of the proposed approach: a set of detected obstacles, with un-oriented cuboids (green). Top image: a camera view in front of the ego vehicle (yellow in the top view), with four relevant obstacles marked, 1,2,3 - vehicles, 4 - bicyclist. Bottom-left: the result of obstacle detection (red areas), a bird-eye view, bottom-right: occupied grid cells are shown with a medium grey shade.

II. Overview of the Proposed Approach

The perception system is similar to the one described in [8] consisting of multiple lidars of 32 layers, each placed close to one of the four corners of the ego vehicle roof. Obstacle are detected using a grid based approach derived from [10].

The approach for computing the orientation of the obstacles starts from a set of un-oriented cuboids, each cuboid having its set of occupied grid locations in the horizontal plane (top view of the 3D environment, Fig. 1). A grid location/cell represents a square of 10 x 10 cm in the horizontal plane.

A. The model for computing the orientation

The reference system used to represent the top view will have Z as longitudinal axis and X as lateral axis (relative to the ego vehicle). For all the images in this paper that show a top view of the scene, or the occupancy grid, the vertical axis corresponds to Z, and the horizontal to X.

The model consists of two perpendicular lines, as in Fig. 2. The first line L (without any relation with the L-shaped model, just a notation) is considered the dominant orientation (on the side with the most boundary cells), while the second line L_P is used to validate that the obstacle has two visible sides for the perception system.

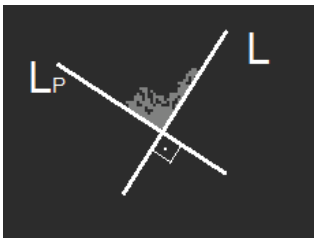


Fig. 2. The two lines used to model oriented objects: L will be fitted to the dominant visible side (the longest visible side of the obstacle), while L_P will model the other visible side, if any.

B. Main steps

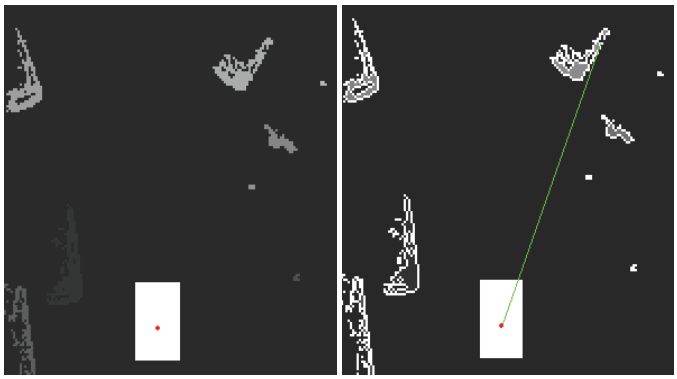
Each step of the approach was designed having in mind the need for low computational complexity, and most of the computation is done in the discrete grid representation where the obstacles were initially detected. The main steps of the approach for computing the orientation of each obstacle are:

- *Computing the visible boundaries*: the idea of visibility refers to those obstacle cells that are directly observable from the ego vehicle (for example, in Fig. 1, for vehicles 1, 2, and 3, only sides perceived directly by the perception system are representative for the rectangular shape of these obstacles). Only these cells will be used for fitting the two line model.
- *Fitting the two-line model to the boundaries*: a random sample consensus approach will be applied, initially for the first line, and then for the second line on the remaining visible cells.
- *Validation/selection of the best orientation*: based on the strength (consensus score from RANSAC) of the two lines and additional criteria, the best orientation will be selected.

III. DETAILED METHOD STEPS

A. Computing the visible boundaries

First, the boundary cells of the obstacle are selected by labelling all obstacle cell that are adjacent to at least one grid cell that is not obstacle.



a. b.

Fig. 3. a. Scenario from Fig. 1 with different labels for each obstacle, the ego vehicle is the white rectangle and the red dot represents the reference point for the perception system, b. Each obstacle patch has its boundary cells marked (white points), and an example of line of sight from the perception system to a boundary cell is shown as the green line.

Next, the visible cells are selected. The condition for visibility is checked with ray tracing. For each boundary cell, the line of sight is computed from the ego vehicle. The center of the car is considered the origin of the perception system (view point), as it is equally distanced in the horizontal plane from the four lidars that perceive the obstacle (depending on the obstacle position, it is likely that only some of the four lidars provide 3D measurements). A more accurate (but more costly) verification would be to consider each lidar as view point.

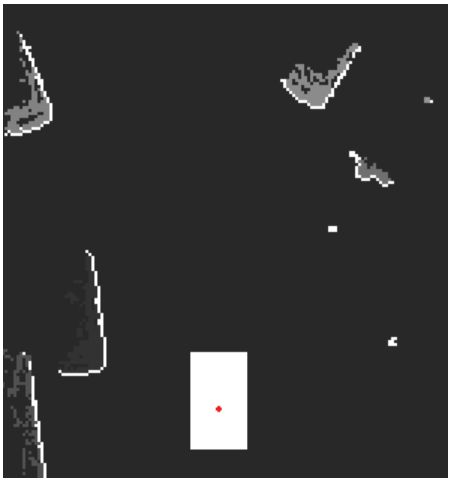


Fig. 4. Each obstacle patch has its visible boundary cells marked (white cells).

Normally, a boundary cell is visible if its line of sight does not intersect any other (current) obstacle cell. Given the discrete representation of the obstacle locations (grid cells), the line of sight for each boundary cell is approximated in the grid space using a Bresenham interpolation (between the perception system origin and the boundary cell). For obstacle sides that have a relative angle with the line of sight lower than 45 degrees, the discrete line of sight of some boundaries cells will likely intersect other boundary cells (such an example is shown in Fig. 3.b, the green line). Therefore, a boundary cell is considered visible (results in Fig. 4) if no obstacle cells are present on its line of sight, or if its line of sight intersects only some boundary cells from the obstacle. In this way, most of the visible sides of each obstacle are selected.

B. Fitting the two-line model to the boundaries

The first line, L, is represented with the general line model with 3 free parameters, as this allows representation of all possible orientations:

$$ax + bz + c = 0 \quad (1)$$

Given two points (x_1, z_1) and (x_2, z_2), the line parameters are computed as follows:

$$(z_1 - z_2)x + (x_2 - x_1)z + (x_1 z_2 - x_2 z_1) = 0 \quad (2)$$

The second line, L_p, given the perpendicularity constraint with L, will have the following equation:

$$bx - az + c_P = 0 \quad (3)$$

Once the main line L is computed, the direction of L_p will be perpendicular and only the free parameter c_P has to be estimated. If a point (x_3, z_3) belongs to L_p then:

$$c_P = az_3 - bx_3 \quad (4)$$

Computing the two lines for each obstacle, on the set of visible boundary cells, is described next.

The first line is computed in a RANSAC standard fashion: k samples of two random cells are selected from the set of visible boundary cells.

The number of samples $k=\log(1-p)/\log(1-w^s)=52$ was computed by considering a success probability $p=0.9999$ and the ratio of good data of $w=0.4$, sample size $s=2$. The percentage of good data is a conservative value: assuming a vehicle with two of its sides visible, the side corresponding to the length will contain more than half of the visible boundary cells.

For each sample, the parameters of L are estimated as in equation (2). The consensus set is computed for each sample: the visible boundary cells that verify the line are counted (within a maximum distance threshold $T_d=0.75$ cells, this allows a band of 1.5 cells width around the line position). The line having the largest consensus score (number of inliers) is considered as solution for L if its number of inliers is at least the number of expected good cells (w times the number of visible boundary cells). If selected as solution, the parameters of L are recomputed with least squares fitting on the consensus set.

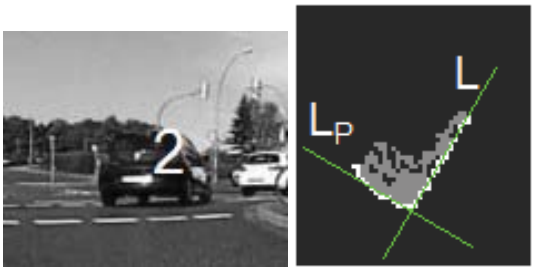


Fig. 5. The two lines computed for the vehicle 2 are shown (the scenario presented in Fig. 1).

The outliers, those cells that do not belong to the dominant line L, are selected for computing the second line L_p. In this case, one point is enough to compute the free term c_P. The approach is also based on selecting random samples: each sample consists of one cell, and the line with the largest number of inliers is selected. In this case, a larger ratio of good data is assumed (cells from the first line are not taken into account) w=0.6. As a consequence, the number of random samples is reduced to k=10 (beside a greater *w*, the sample size is *s*=1). The results for L and L_p for one of the vehicles from Fig. 1 are shown in Fig. 5.

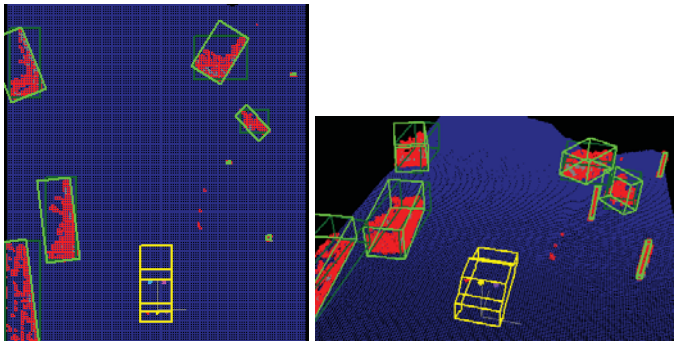


Fig. 6. The oriented cuboids for the scenario presented in Fig. 1 are show with bright green, in top-view (left) and a perspective view of the scene (on the right).

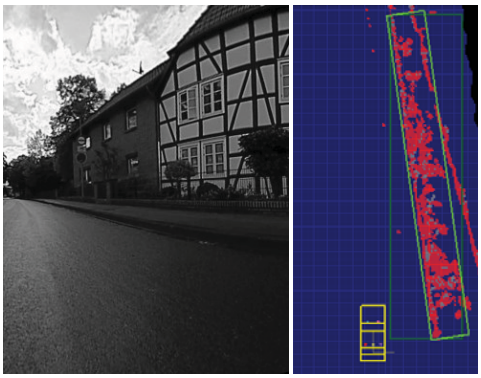
Based on the dominant line, the oriented cuboid is computed for each obstacle. This is done by projecting the obstacle cells onto the two lines and finding the extreme projections on each line as the limits for the oriented cuboid. The scenario presented in Fig. 1 is a typical example where the proposed approach is efficient, with the oriented cuboids shown in Fig. 6. However, there are many situations where additional validation is required for the orientation, and this issue will be discussed next.

C. *Validation/selection of the best orientation*

First of all, the orientation is not validated for obstacles that have a small top view print, such as poles or pedestrians (normal standing pose). This is done by imposing a small absolute threshold (=8 cells) on the number of inliers of the first line L.

When dealing with obstacles that have two sides visible to the perception system, both lines are computed and the orientation is computed reliably (details in the Results section).

In real scenarios, obstacles have various positions and orientations, and often only one obstacle side is visible. This can apply not only for rectangular obstacles such as vehicles, but also for structures that exhibit only one visible side for the perception system: building, fences (see next figure) etc.



a. b.

Fig. 7. a. Scene with a large fence + building on the right side of the road, b. Top-view of the oriented (bright green) and un-oriented obstacle (dark green) detected from the 3D data of the fence and nearby vegetation.

Two different situations must be distinguished. The first one is when the dominant line L has a strong support and the orientation can be considered valid (example in Fig. 7), regardless of the second line. The second situation is when L is computed, but its score is not sufficient to rely only upon one side. These two situations are solved with hysteresis (double thresholding) and reasoning based on the strength of the second line L_p.

Fig. 8. Small passenger cars that are viewed from the rear (or front), without perceiving one of the lateral sides, often present curved visible boundaries that are less reliable to fit lines.

For the first situation, a strong threshold T_S is imposed upon the consensus score of L. If the consensus score is above T_S then the orientation is validated. The value of T_S was established by considering the width of common vehicles (passenger cars). If such a vehicle is seen only from the rear or from the front, then its visible side has a width of 1.6 to 2+ meters. If the dominant line locks onto the rear or the front of a small vehicle, then in order to have a reliable fit, T_S is set to 15 cells (recall that a grid cell has a size of 10x10 cm). This threshold is set rather high with the following justification: if the dominant line fits the rear or the front of a small passenger car, then a good support is

required for the line because this vehicle's side usually presents a curved profile (more or less depending on the model and shape of the bumpers, see Fig. 8).

If the consensus score of L is below T_S then the second line is considered. It might be either a vehicle with partially obstructed visible sides, or a smaller obstacle that can be represented with oriented cuboids. If line L_P has a consensus score of at least T_{SP} (= 10 cells), then the orientation is validated.

If both L and L_P have consensus scores below T_S, and T_{SP} respectively, then there are several ways to deal with the situation:

- Validate the orientation but assume it has a lower accuracy (it can be estimated from the consensus score of L), and rely on higher level processing (tracking) to get a better orientation through temporal filtering. From our experiments, for situations similar to Fig. 8, the orientation error is typically around several degrees (with extreme values of 10-12 degrees rarely observed), and tracking can smooth out these deviations.
- Use the context for reasoning: if only L was computed and with a poor score, then it's likely that the obstacle is (1) a vehicle without any lateral sides visible or (2) another obstacle that has only one significant dimension (example: a bicyclist, obstacle 4 in Fig. 1).

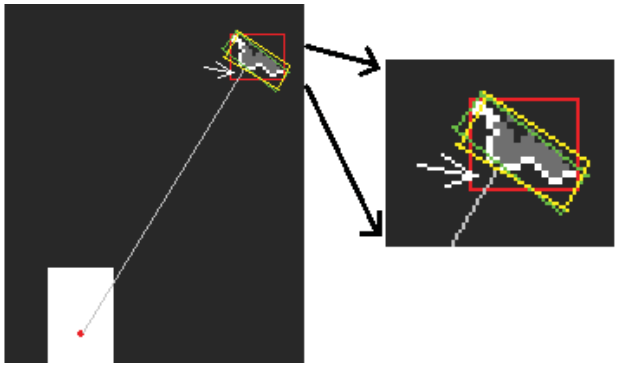


Fig. 9. When the dominant line L is less reliable, the line of sight can be used to infer the obstacle orientation, as the yellow cuboid. The un-oriented cuboid is shown with red, and the cuboid given by L with green. The line of sight to the obstacle center is shown. The white arrow depicts the free area, visible to the perception system, which is computed for each cuboid.

For the second way, the position of the obstacle relative to the ego can be used to infer its orientation: if only one side is visible for a vehicle, then its orientation is likely along the line of sight with a certain freedom. However, if the obstacle is not a vehicle, but a thinner obstacle, then this "line of sight" orientation might be less accurate than the one given by L. To solve this issue, three possible orientations are considered: the one given by L, the orientation given by the line of sight of the center of the obstacle, and the default null orientation (un-oriented). For each cuboid hypothesis, the free area of the cuboid that is visible to the perception system is computed and used as a measure of how well the cuboid envelopes the obstacle. The cuboid with the smallest unoccupied area is selected.

IV. Results

The method for computing the orientation was implemented in C/C++ and integrated in the detection framework [8]. The approach is very fast: the running time for one obstacle (average vehicle sized) is 0.15-0.2 milliseconds. The processing speed is several times higher compared with existing methods. The processor used for testing was a Core I5, 2.5 Ghz. As comparison, for one obstacle, the processing time in [6] is about 1.5 ms on a similar processor (Core I7, C++ implementation).

The precision and accuracy of the orientation were measured on several scenarios were the real orientation of the vehicles in the scene could be estimated with a reasonable error.

The first scenario involved a sequence of several tens of frames where the ego vehicle was standing still, yielding the right of way to incoming traffic. The ego vehicle was positioned at an angle relative to the road axis, as it was leaving the main road. The angle between the ego vehicle reference frame and the road longitudinal axis was estimated manually to 14.4 degrees. The fixed road infrastructure (curbs) that is visible in the 3D lidar data was used to evaluate this angle.

Fig. 10. Scenario 1 used for evaluation. Incoming traffic (the direction of motion shown with the white arrow) at an angle of 14.4 degrees.

A total of 226 vehicle orientations, from 9 distinct vehicles, were kept for evaluation (some measurements were removed – vehicles that changed the lane, or small obstacles). The distance to the vehicles was up to 24 meters from the ego. The angle for each orientation was computed relative to the Z axis of the ego vehicle. The mean value obtained was 14.53 degrees, very close to the manual estimation of 14.4 degrees between the road axis and the ego vehicle. The standard deviation was 1.4 degrees, proving a robust evaluation of the orientation. The histogram of the angle values is shown in Fig. 11.

The second scenario used for evaluation was a sequence where the ego was driving straight on a multi-lane straight road, overpassed slower vehicles, and incoming vehicles were also present. A total of 375 orientations were computed for the vehicles (some up to 45 meters) that were moving straight (this was established from the video data). The mean angle value was 0.4 degrees (versus the expected value of 0 degrees) and the standard deviation 2.5 degrees.

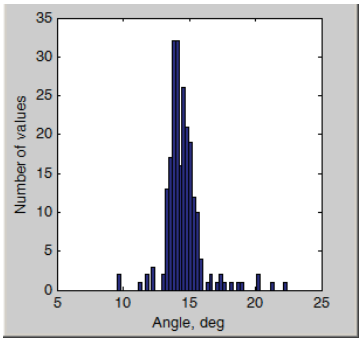


Fig. 11. Histogram of the angle values obtained for Scenario 1, mean value of 14.53 and standard deviation of 1.4 degrees.

Evaluation was also performed on vehicles that were visible strictly from the rear or front, here the standard deviation was larger (4-6 degrees), when using the dominant line L. Tracking can be employed to increase accuracy.

Other results are shown in Fig. 12. Limitations of the cuboid model can be seen in one of the scenes: some obstacles are less suitable for a cuboid representation (curved fences).

V. Conclusions

A low complexity method for estimating the orientation of 3D obstacles, detected with multiple lidars, was presented. The method is very fast and it has high precision and accuracy, as proven by the quantitative evaluation performed.

A potential future improvement is to use the semantic information provided by the target system [8] to have knowledge about the type of the obstacle. This would make the step of validation/selection of the best orientation more robust.

Acknowledgment

This work has been supported by the UP-Drive project (Automated Urban Parking and Driving), Horizon 2020 EU funded, Grant Agreement Number 688652.

References

[1] D. Kim, et al., "L-shape model switching-based precise motion tracking of moving vehicles using laser scanners," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 2, pp. 598–612, 2018.

[2] X. Zhang, W. Xu, C. Dong, and J. M. Dolan, "Efficient L-shape fitting for vehicle detection using laser scanners," in *Proc. IEEE Intell. Vehicles Symp. (IV 2017)*, Jun. 2017, pp. 54–59.

[3] M. Munz, K. Dietmayer, and M. Mahlisch, "A sensor independent probabilistic fusion system for driver assistance systems," *in Intelligent Transportation Systems*, ITSC'09, 12th International IEEE Conference on, 2009.

[4] A. Siadat, A. Kaske, S. Klausmann, M. Dufaut, and R. Husson, "An Optimized Segmentation Method for a 2D Laser-Scanner Applied to Mobile Robot Navigation," *Proceedings of the 3rd IFAC Symposium on Intelligent Components and Instruments for Control Applications*, 1997

[5] R. MacLachlan and C. Mertz, "Tracking of moving objects from a moving vehicle using a scanning laser rangefinder," in *IEEE Intelligent Transportation Systems Conference*, 2006, Canada, pp. 301–306.

[6] X. Shen, S. Pendleton, and M. H. Ang, "Efficient L-shape fitting of laser scanner data for vehicle pose estimation," in *IEEE Conference on Robotics, Automation and Mechatronics*, 2015, Cambodia, pp. 173–178.

[7] G. A. Borges and M.-J. Aldon, "Line extraction in 2d range images for mobile robotics," *Journal of Intelligent and Robotic Systems*, vol. 40, no. 3, pp. 267–297, 2004.

[8] R. Varga, A. Costea, H. Florea, I. Giosan, and S. Nedevschi, "Supersensor for 360-degree environment perception: Point cloud segmentation using image features," in *IEEE Intelligent Transportation Systems Conference*, 2017, pp. 126–132.

[9] M. Fischler, R. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography", Graphics and Image Processing 24(6) (1981), pp. 381–395.

[10] F. Oniga and S. Nedevschi, "Processing dense stereo data using elevation maps: Road surface, traffic isle, and obstacle detection," IEEE Trans. Veh. Technol., vol. 59, no. 3, pp. 1172–1182, Mar. 2010.

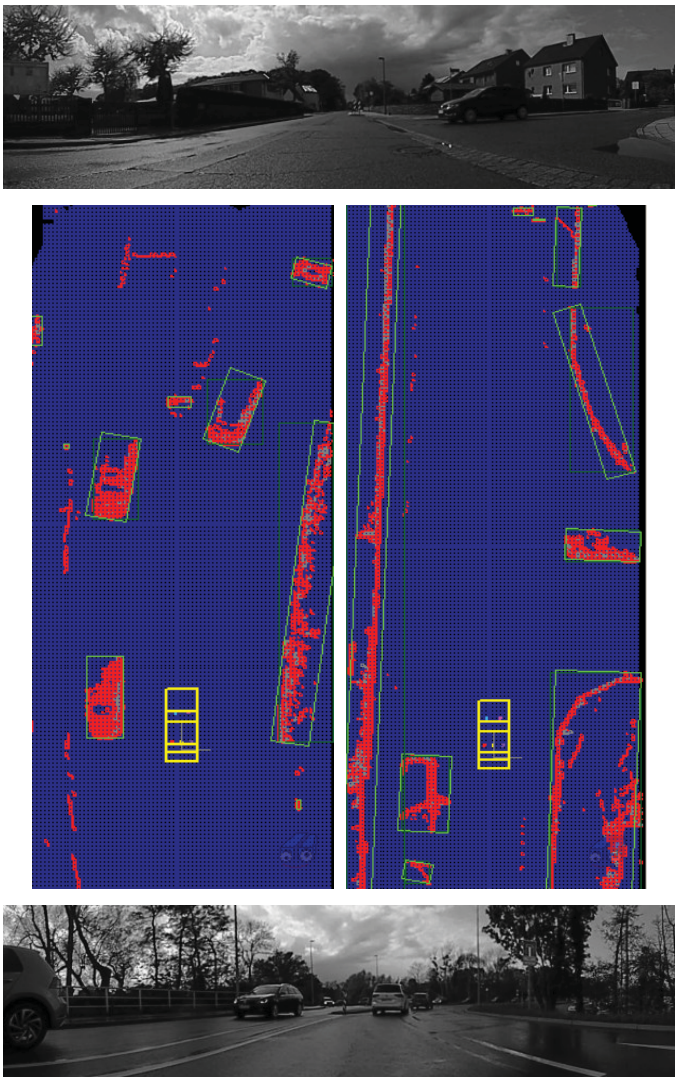


Fig. 12. Other results, including obstacles with small curvatures (fences). Top image shows the scene from the right top-view, bottom image shows the scene from the left top-view. The ego is drawn with yellow.