Multi-layer occupancy grid mapping for autonomous vehicles navigation

Simone Mentasti
Dept. of Electronics Information and Bioengineering
Politecnico di Milano
Milano (MI)
simone.mentasti@polimi.it

Matteo Matteucci
Dept. of Electronics Information and Bioengineering
Politecnico di Milano
Milano (MI)
matteo.matteucci@polimi.it

***Abstract*—Perception of the surrounding is a crucial task in most of the autonomous driving scenarios. For this reason most vehicles are equipped with a broad range of sensors like lidar, radar, cameras and ultrasound to sense the space around the car. On the other end, planning algorithms need a simple and usable representation of the obstacle around. One of the biggest drawbacks of such a wide range of sensors is the need to resolve conflicting information and identify false positives. What we propose in this paper is an effective framework for sensor fusion and occupancy grid creation capable of retrieving a uniform representation of the ambient around the vehicle and able to handle conflictual information from different sensors.**

***Keywords*—Occupancy gird, mapping, sensors funsion**

I. Introduction

Map building has been a research topic for many years in the robotic field [1] [2] [3]. With the uprising interest on autonomous driving most of the robotics approaches are being transferred to the automotive field. However vehicles present nevertheless some differences; they are generally bigger than mobile robots, move faster, and can carry different type of sensors to perceive the environment. In particular almost all autonomous cars use lidar sensor as their main data source [4], [5]; while also integrating cameras [6], sonars and radars [7].

Different sources provide heterogeneous representation of the surrounding, at different data rate and also with different accuracy. Lidar point clouds have low frame rate but offer a 360 degrees information while solid state lasers offer only a few points at high speed. Stereo cameras also provide a 3D representation of their field of view, but they are generally less accurate and error prone than laser and radar.

When multiple sensors are available informations need to be fused into a uniform representation of the world around the car before they can be feed to the planning algorithms. In particular it is important to properly weight different information according to their reliability and resolve conflicting measures. It is also mandatory to properly filter the street surface but also include low height obstacles which needs to be avoided by the car.

978-8-8872-3743-6 ©2019 AEIT

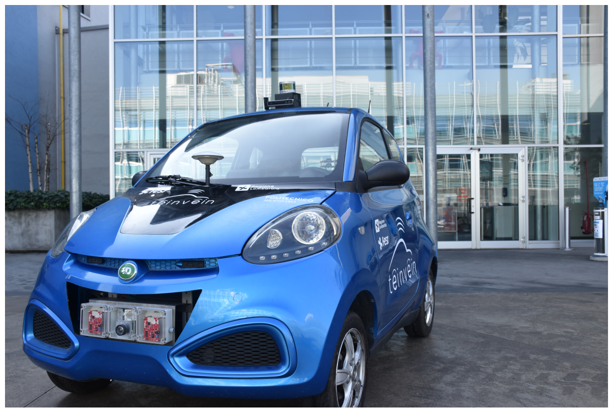

Fig. 1. Picture of the testing car. Perception sensors are visible on the roof (Velodyne and cameras) and on the front bumper (radars and laser).

The goal of this paper is to show an implementation of a toolchain for an occupancy grid creation leveraging multiple sensors. In particular, we describe how we managed to fuse different sources like lidar, cameras and radar in a uniform grid used by the planning system. The paper is structured as follow: we first analyze the recent works in the field of sensor fusion and grid mapping, next we proceed with a description of our test setup and the proposed method for integrating data coming from different sources. Last we will show some experimental results achieved on our testing vehicle.

II. Related works

As stated in the introduction the map creation problem has its roots in the robotics field [1]; due to the robot size and limited costs this process is generally performed using only a few sensors [8], [10], [11].

Starting from the DARPA Grand Challenge in 2005 it has become obvious that autonomous driving cars cannot rely on such a simple configuration [12]. But with the increased number of sensors we experienced also a growth in the complexity of the task of map creation. From the results of the first years of the century different approaches to the problem have been proposed, in particular many researches are focused on achieving the best possible results on a specific task like parking [13], driving in off road scenarios [17], or using only

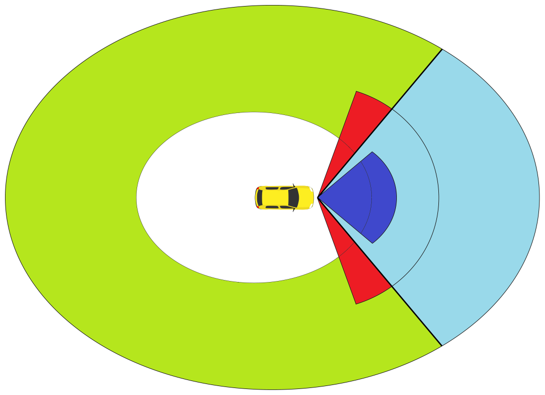

Fig. 2. Schema showing the different areas covered by the sensors, only in the forward direction for simplicity. In light green the velodyne, il light blue the laser, in blue the stereocamera and in red the radar.

specific sensors like radar [14]. Nevertheless, during past years some more complex and complete systems has been proposed, [15], [16].

The general approach to the problem is to use occupancy grid based map [18] where the space surrounding the vehicle is divided in a grid and each box is labelled as empty or occupied. More complex representations use a multiple layer map [19] and consider also inflation area of the surrounding objects, creating a danger zone around them. Recent approaches based on Bayesian modelling [22] improve the current state of occupancy grid computation, trying to guarantee an accurate representation of obstacles, weighing in a more accurate way the different sources and the car kinematics.

III. Experimental setup

Our experimental setup is built around a ZED One electric car, shown in Figure 1. The vehicle is equipped with a Velodyne VLP16 lidar mounted on the roof for 360 degrees awareness. Other sensors are mounted around the car; we use a symmetric configuration for front and rear detection, showed in Figure 2. The sensors array consists of a Continental Radar, a Leddartech M16 solid state laser and a Texas Instruments AWR 1642 Radar. On top of the vehicle, facing forward, is also mounted a Stereolabs ZED stereo camera.

A. Data types

All listed sensors provide information on the surrounding of the car, but they differ in precision, field of view, and quality of data. In particular, the lidar is the only sensor retrieving information at 360 degrees, ranging from 3 meters to 100 meters with centimetre precision. The returned data from the velodyne is a dense point cloud of the environment. Both radar

Partially supported by project TEINVEIN: TEcnologie INnovative per i VEicoli Intelligenti, CUP (Codice Unico Progetto - Unique Project Code): E96D17000110009 - Call "Accordi per la Ricerca e l'Innovazione", cofunded by POR FESR 2014-2020 (Programma Operativo Regionale, Fondo Europeo di Sviluppo Regionale Regional Operational Programme, European Regional Development Fund).

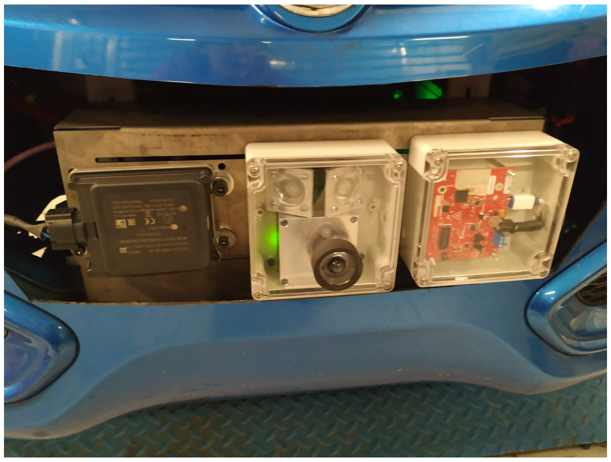

Fig. 3. Detail of the sensors mounted on the front bumper, from the left to the right, Continental radar, Leddartech M16 solid state laser and Texas instrument radar

sensors also provide point cloud data, but in a smaller field of view: while the AWR 1642 has a range of ± 70 degrees horizontally and ± 40 degrees vertically, the Continental has a smaller window, ranging from ± 40 degrees horizontally and ± 10 degrees vertically. The advantages of those sensors, thanks also to the mounting position, is the ability to retrieve data at a minimum distance of 4 cm for the Texas Instruments and 30 cm for the Continental. This, as shown in Figure 2 covers most of the blind spots of the Velodyne. The difference between the two devices is the maximum range; the first one can cover only 15 meters, while the second ranges up to 100 meters.

While radar and velodyne cover almost all the surrounding of the car, the need for a faster and easily interpretable sensor for low level safety is supplied by the Leddartech M 16 laser. This sensor returns only a sixteen points array spread on a 100 degree horizontal field of view, with centimeter precision. This device provides less information but can be used as a validation system for the radar data, being more precise and reliable. The data coming from the stereo camera, retrieved using 3D reconstruction from two aligned images, are the less accurate and error prone. This information provides nevertheless a dense representation of the area in front of the vehicle, and is the only devices with easily accessible semantics information. For this reason data from the zed camera are fused with the more precise coming from other sources.

IV. Proposed method

The process of occupancy grid creation uses data coming from all the sensors previously listed. Before starting the effective computation of the surrounding we need to compute the relative position between all the sensors. To achieve this task we use the lidar data as the reference sensor, thanks to the 360 degrees field of view and the high precision, and align the other pointclouds to this source. This process needs to be performed only once, than the computed value could be hard

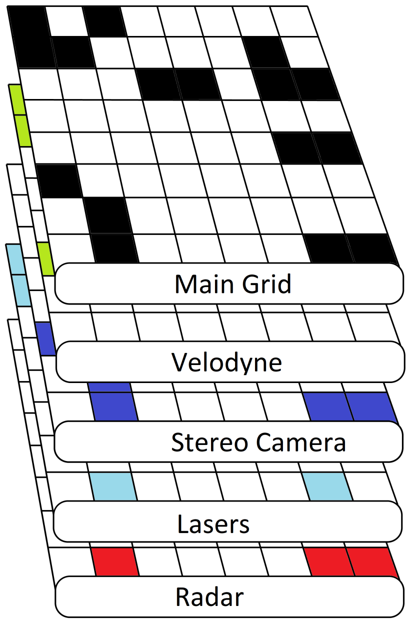

Fig. 4. Picture of the different layer used for the occupancy grid creation. The first layer consists of the output layer created from all the below level.

coded in the occupancy grid algorithm which will shift each sensor's data accordingly.

The next phase, showed in Figure 4 consists in the creation of one occupancy grid for each sensor. All those grids will be translated and rotated using the previously computed transformation and finally combined in one main grid, which can be easily used by the planning algorithms.

A. *Point cloud alignment*

To compute a coherent occupancy grid we need the respective position of each sensor. To retrieve those values we compute the relative position from the Velodyne sensor. This process can be done for all sensor, in all position due to the 360 degree field of view of the lidar. The alignment process consists, for all sensors, in a pointcloud alignment.

To compute the position of a sensor we recorded a set of pointclouds from both the sensor and the Velodyne in a controlled environment. In particular, we choose surfaces which guarante the best reflection for sensors like radar and lasers, while for stereo cameras we used a feature rich scene where the 3D reconstruction algorithm could easily find triangulation points. For each device we recorded approximately 50 pointcloud on different ambient configuration.

For each set of pointclouds we processed offline the data computing the alignment between each pointcloud from the lidar to the pointcloud of the sensor. From all those transformations we compute the mean root square error and remove outliers. Finally, we used the mean of the remaining values as the roto-traslation from one sensor to the velodyne.

The described process results extremely efficient for sensors like cameras, where the pointcloud covers a wide field of view and is extremely dense. Radar still provides a pointcloud which can be aligned with the lidar data, but is more sparse and sensible to the reflective surface. To increase the number of points from radar we used high reflective surfaces and we also feed the algorithm of pointcloud alignment with an initialization value retrieved by manual measurement. In this way the system only needed to do a minor adjustment, in the order of 20 centimetres and a few decimals of a degree. With this assumption the system is able to find a consistent alignment in all the sets. We performed all the step previously described removing outliers and using the mean value as the position of the sensor.

LeddarTech data are more complex to handle, having only sixteen points it's not possible to use classical point cloud alignment. First we converted the sixteen value in a set of points in a 3D environment using Equation 1. The height is estimated to be constant thanks to the horizontal mounting of the system.

$$\begin{cases} x_i = sin(40 + ray_i * (100/16)) * distance \\ y_i = cos(40 + ray_i * (100/16)) * distance \end{cases} \tag{1}$$

To align this sparse sets of points with the Velodyne we moved the car in front of a flat surface at 10 meters. Next we interpolated the LeddarTech points creating a surface which is than aligned with the lidar pointcloud. Similarly to the radar we feed the algorithm with initialization values, in this way we only did some minor adjustment to calibrate the system.

B. *Velodyne VLP 16 processing*

Data coming from Velodyne need to be processed and filtered to be used in computing the occupancy grid. This process is divided in different steps.

First the pointcloud needs to be straightened; the sensor is indeed mounted slightly oblique, facing forward. This choice guarantees a better field of view in the moving direction, covering a closer area also with the Velodyne rays. The first operation is than a rotation to have the groud plane perfectly horizontal.

The next step consists in ground plane filtering. To compute an occupancy grid we have first to remove the ground lines from the pointcloud; to perform this task we used an approach similar to the one described in [21]. Due to the different sensor, and the consequent smaller point cloud, we made some changes to the proposed method, in particular we reduced the number of segments used for the plane fitting process. Lidar with high resolution, i.e. with more than 16 rays, produce a more accurate plane on the street surface and the original approach can easily fit planes on short distances, while our model has considerably spaced lines and it is than more difficult to identify a surface. Our approach is then less reliable in a situation where the street surface presents considerable high changes, but performs better in traditional urban scenarios.

The ground points are then removed from the point cloud and before projecting it to the ground plane we also filter all the points above a certain height, in our case 3 meters, being

Algorithm 1 Costmap creation algorithm

```
procedure COSTMAP(pointcloud, normal, grid)
    for point in pointcloud do
        project (point, normal)
    end for
    for cell in grid do
        points = num points in cell
        if points > threshold then
            cell = occupied
        end if
    end for
    return grid
end procedure
```

relevant to the obstacle avoidance scenarios. The resulting pointcloud is considerably smaller and less subject to noisy data. It can be than feed to the system which computes the occupancy grid around the car.

C. *Occupancy grid computation*

Having computed the joint position of all the sensor it's possible to create the occupancy grid. This process, unlike the alignment, is done in real time while the car is moving. The process of conversion from pointcloud to occupancy grid is similar for each sensor mounted on the vehicle. The algorithms differ only for the parameters used, calibrated in function of the density of the input pointcloud.

Our algorithm, shown in Algorithm 1, starts converting the 3D pointcloud to a 2D object. This process is not done simply setting all the z axis value to zero, it projects each point using the normal to the ground plane preciously computed from the lidar data. After this conversion we apply a 2D grid to the flattened pointcloud and for each box we compute the number of points lying inside it. After accurate analysis we decided to use a rectangular grid with square boxes. The last phase of the costmap creation cycles through each cell and set the box as occupied by an obstacle if the number of points in that box is bigger than a threshold. This process allows us to filter noise from sensors which produces lots of 3D points but rich in false positives, as for stereo cameras. Due to the high precision and low number of data we do not filter the LeddarTech sensor.

After applying Algorithm 1 we have for each sensor a different occupancy grid, as previously shown in Figure 4. The next phase consists in the creation of a uniform representation of all those information. As shown in Figure 2 most of the area around the car is covered only by the Velodyne but there are some areas where we have data coming from multiple sources. In particular the most sensible zone, the space close in front of the car (the blue section in Figure 2), is covered by four different sources.

In order to create the final occupancy grid we cycle again through all the cells of the costmap and check the occupancy value for each sensor which can perceive data in that position. We perform this check because some sensors can generate noise in positions outside their effective field of view, and it is easier to filter this noise in this phase of the algorithm. We than perform a weighted average on the values from each sensor in the cell and compare it to a threshold; if the value is greater than the threshold the cell on the final costmap is labelled as occupied. A pseudo code of this procedure is shown in Algorithm 2.

Algorithm 2 Costmap creation algorithm

```
procedure COSTMAP(costmap[6])
    for cell in costmap[0] do
        cellValue = 0
        usedSensors = 0
        for i=1 , i<=5 do
            if cell has $i$-th sensor then
                cellValue = cellValue+ cell[i]*weight[i]
                usedSensors = usedSensors + 1
            end if
        end for
        cellValue = cellValue / usedSensors
        if cellValue > threshold then
            costmap[0] = occupied
        end if
    end for
    return costmap[0]
end procedure
```

We decided to use a weighted average and a threshold to filter false positive data. Some sensors are considerably more reliable than other; the sixteen beam laser in particular is extremely precise, while camera's 3D reconstruction can generate artefacts, in particular in low light condition. For this reason we weight each sensor accordingly. Laser and lidar have height weight, while camera's pointcloud is smaller, in this way a false positive will be filtered by the threshold but a real measure, in particular in the area where the Velodyne data are not available, will reinforce radar and laser estimation.

V. RESULTS

Our framework has been tested in real urban scenarios to validate all of the step it performs, from sensors alignment to filtering to occupancy grid creation. The system has proved to be reliable in many scenarios, being able to properly identify all obstacles surrounding the vehicle. The whole toolchain resulted also extremely lightweight thanks to optimizations applied on the pointcloud, like filtering and decimation and was able to run on a laptop without excessive load to the cpu at 20 Hz.

In Figure 5 we show the result of the pointcloud filtering process in a real street scenario.

In particular we notice how the street lines on the left are correctly removed from the original data. The top centre of the figure shows how the system removes a large number of points generated by threes which otherwise could introduce noise in the resulting occupancy grid. The output of this first analysis is a smaller and more accurate pointcloud which can be easily handled by the costmap creation system.

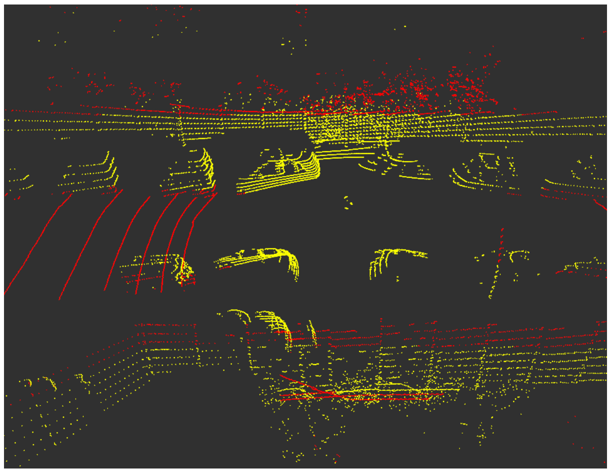

Fig. 5. Image showing the filtered points from the Velodyne sensor in red, and the points used for the occupancy grid creation in yellow. We notice the removed street lines in the left part of the image and the high number of points from the trees in the top which are also filtered. Obstacles like cars , pedestrians and walls are instead preserved.

In Figure 6 we show a demo setup in a small environment of the multilayer occupancy grid creation. For this test we used only the front sensors and the top lidar. We notice that most of the lateral area of the car is scanned only by one sensor, the 360 degree Velodyne. The front area, on the other hand, is mostly analyzed by multiple sensors. In this way we guarantee enough precision and redundancy on the most sensible area.

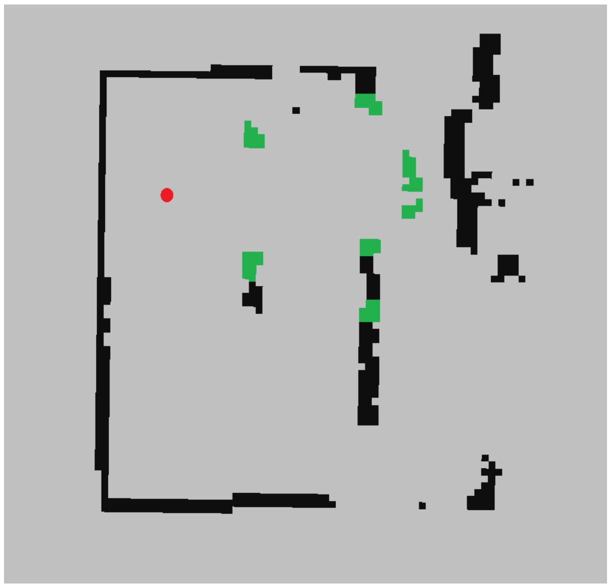

Fig. 6. Image showing the computed occupancy grid in a testing area. The car position, which coincides with the Velodyne, is represented as a red dot and is facing to the right. The black areas are obstacles retrieved only by one sensor, in our case the lidar, while the green areas are obstacles retrieved by multiple sensors. For this test only the forward direction sensors are used

VI. Conclusion

In this paper we showed a simple but efficient framework for computing a uniform representation of the environment surrounding an autonomous driving vehicle fusing data from different sensors into an occupancy grid. The proposed method is performed in real time on the car without a significant load to the control computer unit. The system showed to be considerably reliable in filtering noisy data and false positives, while preserving real obstacles.

Further improvement of this framework will focus on the usage of inertial units to predict the occupancy grid from one frame to the next one, and also use this prediction to improve the filtering mechanism by estimating motion vectors.

References

[1] Giorgio Grisetti, Cyrill Stachniss, and Wolfram Burgard: Improved Techniques for Grid Mapping with Rao-Blackwellized Particle Filters, IEEE Transactions on Robotics, Volume 23, pages 34-46, 2007

[2] Giorgio Grisetti, Cyrill Stachniss, and Wolfram Burgard: Improving Grid-based SLAM with Rao-Blackwellized Particle Filters by Adaptive Proposals and Selective Resampling, In Proc. of the IEEE International Conference on Robotics and Automation (ICRA), 2005

[3] Einhorn Erik , Schrter Christof , Gross Horst-Michael. (2011). Finding the adequate resolution for grid mapping - Cell sizes locally adapting on-the-fly. 1843 - 1848. 10.1109/ICRA.2011.5980084.

[4] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. N. Clark et al., Autonomous driving in urban environments: Boss and the urban challenge, Journal of Field Robotics, vol. 25, no. 8, pp. 425-466, 2008.

[5] M. Montemerlo, J. Becker, S. Bhat, H. Dahlkamp, D. Dolgov, S. Ettinger, D. Haehnel et al., Junior: The stanford entry in the urban challenge, Journal of Field Robotics, vol. 25, no. 9, pp. 569-597, 2008.

[6] C. Laugier, I.E. Paromtchik, M. Perrollaz, M.Y. Yong, J-D. Yoder, C. Tay, K. Mekhnacha, and A. Negre, Probabilistic analysis of dynamic scenes and collision risks assessment to improve driving safety, IEEE Intelligent Transportation Systems Magazine, vol. 3, no. 4, pp. 4-19, 2011.

[7] T-D. Vu, J. Burlet, and O. Aycard, Grid-based localization and local mapping with moving object detection and tracking, Information Fusion, vol. 12, no. 1, pp. 58-69, 2011.

[8] M. Labb and F. Michaud, RTAB-Map as an Open-Source lidar and Visual SLAM Library for Large-Scale and Long-Term Online Operation, in Journal of Field Robotics, accepted, 2018.

[9] Zermas Dimitris , Izzat Izzat , Papanikolopoulos Nikolaos. (2017). Fast Segmentation of 3D Point Clouds: A Paradigm on lidar Data for Autonomous Vehicle Applications. 10.1109/ICRA.2017.7989591.

[10] I. H. Shanavas, S. A. Ahmed and M. H. Safwat Hussain, "Design of an Autonomous Surveillance Robot Using Simultaneous Localization and Mapping," 2018 International Conference on Design Innovations for 3Cs Compute Communicate Control (ICDI3C), Bangalore, 2018, pp. 64-68.

[11] Y. Deng, Y. Shan, Z. Gong and L. Chen, "Large-Scale Navigation Method for Autonomous Mobile Robot Based on Fusion of GPS and lidar SLAM," 2018 Chinese Automation Congress (CAC), Xi'an, China, 2018, pp. 3145-3148.

[12] Thrun Sebastian, Montemerlo Mike , Dahlkamp Hendrik , Stavens David , Aron Andrei, Diebel James , Fong Philip , Gale John , Halpenny Morgan , Hoffmann Gabriel, Lau Kenny , Oakley Celia , Palatucci Mark, Pratt Vaughan , Stang Pascal , Strohband Sven , Dupont Cedric , Jendrossek Lars-Erik , Koelen Christian , Mahoney Pamela. (1970). Stanley: the robot that won the DARPA Grand Challenge. 10.1007/978-3-540-73429-11.

[13] S. Park, K. Lim and J. Kim, "The research of global cost map building for Unmanned Ground Vehicle in parking area," 2013 13th International Conference on Control, Automation and Systems (ICCAS 2013), Gwangju, 2013, pp. 142-144.

[14] J. Steinbaeck, C. Steger, G. Holweg and N. Druml, "Next generation radar sensors in automotive sensor fusion systems," 2017 Sensor Data Fusion: Trends, Solutions, Applications (SDF), Bonn, 2017, pp. 1-6.

[15] M. Kang, S. Hur, W. Jeong and Y. Park, "Map Building Based on Sensor Fusion for Autonomous Vehicle," 2014 11th International Conference on Information Technology: New Generations, Las Vegas, NV, 2014, pp. 490-495.

[16] J. Koci, N. Jovii and V. Drndarevi, ”Sensors and Sensor Fusion in Autonomous Vehicles,” 2018 26th Telecommunications Forum (TELFOR), Belgrade, 2018, pp. 420-425.

[17] K. A. Redmill, J. I. Martin and U. Ozguiner, ”Sensing and Sensor Fusion for the 2005 Desert Buckeyes DARPA Grand Challenge Offroad Autonomous Vehicle,” 2006 IEEE Intelligent Vehicles Symposium, Tokyo, 2006, pp. 528-533.

[18] P. Tripathi, K. S. Nagla, H. Singh and S. Mahajan, ”Occupancy grid mapping for mobile robot using sensor fusion,” 2014 International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT), Ghaziabad, 2014, pp. 47-51.

[19] D. V. Lu, D. Hershberger and W. D. Smart, ”Layered costmaps for context-sensitive navigation,” 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, Chicago, IL, 2014, pp. 709-715.

[20] C. LAUGIER Embedded Bayesian Perception and Collision Risk Assessment IEEE ICRA 2017 WS on Robotics and Vehicular Technologies for Self-driving cars , Singapore, June 2nd 2017

[21] Zermas Dimitris , Izzat Izzat , Papanikolopoulos Nikolaos. (2017). Fast Segmentation of 3D Point Clouds: A Paradigm on lidar Data for Autonomous Vehicle Applications. 10.1109/ICRA.2017.7989591.

[22] Gindele, Tobias & Brechtel, Sebastian & Schroder, Joachim & Dillmann, Rdiger. (2009). Bayesian Occupancy Grid Filter for Dynamic Environments Using Prior Map Knowledge. 669 - 676