LiDAR-based SLAM algorithm for indoor scenarios

Felipe Jiménez, Miguel Clavijo and Javier Juana
University Institute for Automobile Research (INSIA)
Universidad Politécnica de Madrid (UPM)
Madrid, Spain
e-mails: {felipe.jimenez, miguel.clavijo}@upm.es, javier.juana.serrano@alumnos.upm.es

Abstract— **Simultaneous Localization and Mapping (SLAM) algorithms are one of the elements that have great relevance for autonomous driving in order to locate the vehicle, even in areas in which other methods have difficulties, or to improve the positioning given by these other systems. It also offers knowledge of the scenario in which the vehicle moves, information that can have multiple uses. There are several solutions to the SLAM problem using Light Detection and Ranging (LiDAR), but these algorithms require a high computational cost. However, certain environments with a specific structure allow the use of more simplified algorithms. Specifically, this paper shows a SLAM algorithm where only the LiDAR signal is used and vertical planes are taken as reference (perpendicular to the ground plane). This solution is quite effective in some scenarios, such as indoor parking areas. In addition, various alternatives are explored to increase the robustness of the results of positioning and mapping reconstruction. The algorithm has been tested in real scenarios with satisfactory results.**

Keywords - SLAM; LiDAR; Autonomous vehicle; Detection; Algorithm.

I. INTRODUCTION

One of the biggest difficulties of autonomous vehicles is the ability to react correctly and safely to the eventualities that may occur during driving. Consequently, these vehicles must be equipped with sensors capable of responding in a very short time and a computer capable of interpreting all this information in real time. Among these sensors are cameras, radars, Global Positioning System (GPS), LiDAR sensors, infrared sensors, etc. Specifically, LiDAR has advantages that make it particularly suitable for autonomous driving [1]. One of the main advantages of LiDAR sensors versus computer vision is that the latter are very sensitive to light changes or environmental conditions [2]. In addition, cameras are directional and only detect objects in the direction in which they are placed, and LiDAR sensors offer most of the time a 360º field of view, obtaining a complete point cloud from the vehicle's surroundings. On the other hand, a drawback of the LiDAR sensors is that the density of points decreases with distance due to the divergence between the laser beams.

Solving the SLAM problem consists in the estimation of the movement of the ego-vehicle and the mapping of the environment in which it is located simultaneously. Among the possible applications of this technique, we can highlight its use for autonomous vehicle guidance systems and the creation of three-dimensional models of the environment in a fast and accurate manner. The growing interest in self-driving cars has caused researchers to once again seek a solution to the problem of SLAM, a problem that appeared in robotics more than 25 years ago. This concern to achieve an accurate SLAM solution is fundamentally due to the need for an error positioning of centimeters and an understanding of the environment for a proper path-planning and decision making. As far as the SLAM techniques are concerned, there are many branches to tackle this problem. From the classical techniques used in robotics, or new approaches using range sensors, or techniques based on artificial intelligence, to new trends, such as multivehicle SLAM [3].

When talking about the problem of vehicle localization, the solution is not trivial using a GPS. GPS has shown great performance for the location, however, cannot guarantee an error below centimeters in all cases, even when using D-GPS with a positioned base station. A classic approach would be to make use of inertial units, encoders on the wheels, etc. However, all of them produce an incremental error in time, which makes a correct localization impossible. Another classic approach could be the localization of the ego-vehicle using the line markings on the road, but these are not enough when the vehicle approaches complex scenarios, like a crossing or merging lanes. Finally, we always look for a trade-off, merging different approaches into a fusion.

Other approaches would be techniques based on Visual-SLAM or LiDAR-based SLAM, both make use of the extraction of characteristic points of the environment, or landmarks. After the extraction of these landmarks, a matching and calculation of the relative movement are carried out. The prediction of the new position of the vehicle can be estimated, on one hand, by filter techniques, e.g., Extended Kalman Filter (EKF) [4]–[6] or Particle Filter [7]–[9]. On the other hand, we could also find another approach for the same task, this time using optimization methods, such as Bundle Adjustment [10] among others.

More recently, the use of artificial intelligence techniques, specifically deep learning, has gained great interest due to the great advance of computing capacity and the development of several quality databases (e.g., KITTI dataset [11], Ford Campus [12] or Málaga Urban data [13]). The use of Convolutional Neural Networks (CNNs) is very suitable for the recognition of images and for the extraction of characteristics. On the other hand, CNNs have not only been used to extract image characteristics but also to estimate continuous signals when dealing with regression problems. Therefore, we can find several developments

 ISBN: 978-1-61208-643-9

where they take advantage of these benefits to apply to the SLAM problem [14]–[17].

Although the SLAM problem is not new and there are many developments in the field, it still presents some issues. The first one is the drift that occurs when the trajectory increases, and on the other hand, the construction of the map in any weather condition, traffic or time of year [18].

In indoor scenarios, the GPS signal is not available or is not sufficiently reliable and its accuracy is low, so it cannot be used as the only reference for the positioning of autonomous vehicles. Autonomous vehicles are equipped with perception systems, being the LiDAR one of the best ones regarding robustness and information provided. This sensor has been used in common SLAM problems, although these algorithms require a high computational cost. However, certain environments with a specific structure allow the use of more simplified algorithms. Specifically, this paper shows a SLAM algorithm where only the LiDAR signal is used, and vertical planes are taken as reference (perpendicular to the ground plane). In addition, various alternatives are explored to increase the robustness in the result of positioning and scenario reconstruction.

The rest of the paper is structured as follows. In Section 2, the SLAM algorithm developed in this paper is described. In Section 3, results from several tests both indoors and outdoors are depicted. Finally, in Section 4, conclusions and future works are presented.

II. SLAM ALGORITHM

The proposed algorithm for scenarios where the vertical planes can be taken as reference includes 3 phases:

- Detection of the characteristic elements of the environment (vertical planes)
- Determination of the trajectory (the planes detected are projected on the horizontal plane and the lines resulting from those projections and their intersections are used)
- Reconstruction of the environment (three-dimensional reconstruction by superimposing point clouds)

A. Characteristic elements detection

The characteristic elements are those elements that have certain properties that allow them to differentiate themselves from other elements of the environment. Therefore, the repetitiveness with which they are detected will be very important to ensure that each of them is detected in successive time intervals.

In this way, they can be tracked over time and the trajectory of the vehicle can be determined from the relative movement of the detected characteristic elements.

Therefore, the detection of these elements is the starting point of the algorithm, and the precision in their detection determines the accuracy of the trajectory obtained.

In the urban and industrial environment, for which this algorithm is designed, walls and columns of rectangular section can be found abundantly. Therefore, the characteristic elements to be detected are vertical corners and vertical planes.

However, due to the difficulty of detecting the vertical corners directly, only the vertical planes will be detected, and the corners will be extracted from them.

Method 1: Lines detection in each laser layer

For the detection of characteristic elements, a coefficient has been implemented, proposed in [19], which evaluates the smoothness of a surface, with the aim of detecting vertical corners. However, this coefficient has finally been discarded because all those rough objects that were detected in the environment could be detected as possible landmarks.

As an alternative, for the detection of sudden changes of curvature in the different sections, the angle formed by the segment between two consecutive points with the horizontal axis of the XY plane has been analyzed.

A more accurate way to detect these lines is to calculate a local line for each point. Then, those points that are placed a certain distance away from the line are removed, and the equation of the line is obtained with the others and the average square error is recalculated. If the error is less than a certain threshold value, the line is accepted. This process is repeated for all laser layers. Once all the lines have been extracted, they are compared with each other to detect possible matches between one layer and the others.

Method 2: Planes detection from points clouds

Another method involves extracting the planes in a direct way from the point cloud in 3D. For this purpose, the M-estimator SAmple Consensus (MSAC) algorithm has been used [20]. The MSAC algorithm is a variant of the RANdom SAmple Consensus (RANSAC) algorithm. Once the model that we want to fit is known, in this case a plane, this algorithm optimizes according to the number of inliers and outliers with the cost function (1):

$$Cost = \sum_i \rho\left(e_i^2\right) \tag{1}$$

$$\rho(e^2) = \begin{cases} e^2 & e^2 < T^2 \\ T^2 & e^2 \geq T^2 \end{cases} \tag{2}$$

Where T is the threshold for considering inliers and e^2 provides the error for the point data.

The adjustment is executed and the points of the extracted plane are eliminated for the next execution. From the set of all the extracted planes, only those perpendicular to the horizontal plane of the vehicle are of interest for the calculation of the trajectory. Therefore, planes that do not meet this condition or those whose average error is too high are removed.

B. Trajectory calculation

The landmarks that are used to calculate the trajectory are the vertical planes of the environment. These planes are projected on the horizontal plane and, in order to determine the trajectory we work with the lines resulting from that projection.

 ISBN: 978-1-61208-643-9

In addition to the lines that result from the projection of the vertical planes, the points of the intersections of these lines are also used for the calculation of the trajectory. Some of these points correspond to the real corners of the planes that share a corner in the field of vision of the LiDAR, while the rest correspond to the virtual corners that result from the prolongation of those planes.

The mathematical representation of intersecting lines is presented with 2 notations based on their orientation:

$$\begin{cases} \text{Type 0:} & y = m_0 x + n_0 \\ \text{Type 1:} & x = m_1 y + n_1 \end{cases} \quad (3)$$

Once the straight lines have been calculated, for the calculation of the intersections (Figure 1), the four possible pairings have been considered according to the type of line. The number of intersections for a number n of lines is determined by the following expression:

$$n_{\text{intersections}} = \frac{n}{2} \cdot (n-1) \quad (4)$$

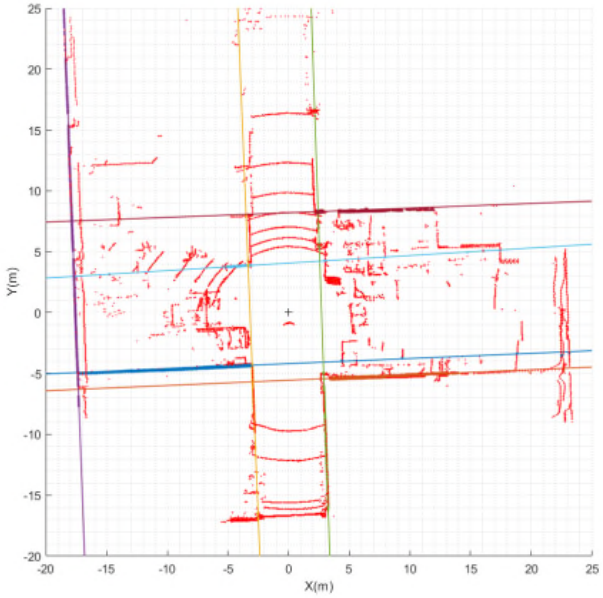


Figure 1. Lines intersections

Intersections located further than 100 m are eliminated.

To determine the trajectory of the vehicle, three different methods have been developed to obtain the values of displacement and rotation.

Method A:

The first one uses a function called *estpos* which calculate the displacement and the rotation of the vehicle simultaneously, from the relative movement of the intersections previously calculated, by solving a linear system of equations.

First, those points that have been detected in two consecutive frames are identified and matched. Therefore, the points of the previous instant, called *points0*, are displaced and rotated with the values of dx_0, dy_0 and φ_0 calculated at the previous instant. Then, the nearest point of the current frame is searched for each of the points already displaced. If the distance is less than a threshold, both points are considered to correspond to the same characteristic element and the pair of points is saved.

Once the points detected in the two frames have been determined, the longitudinal and lateral displacements and the yaw angle are calculated. For this, the following system of equations is solved.

$$\begin{bmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{bmatrix} \cdot \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} + \begin{bmatrix} dx \\ dy \end{bmatrix} = \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} \quad (5)$$

Where x_1 and y_1 correspond to the coordinates of the points in the current frame, and x_0 and y_0 correspond to the coordinates of the points in the previous frame. The unknowns of this system are: φ, dx and dy.

This system is a non-linear system and for its resolution, it would be necessary to resort to iterative calculation methods. However, since the time interval between two consecutive frames is very small, it can be considered that the angle φ will also take small values.

In this way, simplifications can be considered and the system becomes linear. The simplified system is as follows.

$$\begin{bmatrix} 1 & 0 & -y_0 \\ 0 & 1 & x_0 \end{bmatrix} \cdot \begin{bmatrix} dx \\ dy \\ \varphi \end{bmatrix} = \begin{bmatrix} x_1 - x_0 \\ y_1 - y_0 \end{bmatrix} \quad (6)$$

Method B:

The second method is a variation of the previous one, and instead of solving the linear system of equations, it uses the Iterative Closest Point (ICP) algorithm to obtain the displacement and rotation values. The pairing of the points occurs identically.

Method C:

Finally, the third method calculates the rotation and displacement of the vehicle independently. On the one hand, the angle rotated is calculated using the equations of the straight lines and later, the displacement of the vehicle is calculated from the intersections.

To do this, the first step is to match each line with its line equivalent to the previous frame. This process is identical to the pairing of points, with the difference of using the equation of the line instead of the coordinates of the point.

C. Scenario reconstruction

In this step, it is necessary to transfer and rotate the point cloud obtained in each of the frames, and in this way generate a single three-dimensional image that includes all the points in absolute coordinates.

This reconstruction is of special interest because it is used to quantify the precision of the trajectory calculation and, in this way, to be able to compare the different

 ISBN: 978-1-61208-643-9

algorithms used and the influence of the improvements introduced in them.

D. Algorithm improvements

On the previous algorithm, 2 improvements have been proposed:

- Weighting of points and lines.
- Correction of the possible inclination of the laser with respect to the ground.

The first correction is based on the fact that not all data is extracted with the same reliability. Therefore, a confidence value is assigned to each of the points or the lines, depending on certain characteristics that may affect the accuracy with which they have been calculated.

This weighting is carried out by increasing the number of points based on a confidence value. In this way, those points with a higher confidence coefficient will be repeated a greater number of times than those with a lower coefficient.

For the definition of the value of the confidence of the points, two factors have been taken into account: the distance of the points from the vehicle and the angle formed by the lines with which the point has been calculated.

The distance of the points has been quantified by the inverse of the distance. Since x_p and y_p are the coordinates of the points, the weighting coefficient has been defined as follows:

$$C_1 = \frac{1}{\sqrt{{x_p}^2 + {y_p}^2}} \quad (7)$$

In this way, the points near the vehicle will have a greater weight compared to those further away.

On the other hand, because the points have been obtained by intersecting the lines corresponding to the planes detected, the greater the angle formed by these lines, the greater the precision with which the point has been calculated. To quantify this effect, the angle formed by these lines has been calculated and the following coefficient has been used.

$$C_2 = 1 - \frac{2 \cdot \left(\frac{\pi}{2} - \alpha\right)}{\pi} \quad (8)$$

where α is the angle formed by the lines with which each point has been calculated.

Finally, a third coefficient that combines the previous two has been implemented.

$$C_3 = C_1 \cdot C_2 \quad (9)$$

On the other hand, the lines resulting from the projection of the detected planes are weighted taking into account two factors: the number of points contained in the plane and the error in obtaining it.

In this way, a confidence value has been defined that is a function of the value of the mean square error when obtaining the plane and the number of points of the same.

$$C = n_{points} \cdot (1 - \varepsilon) \quad (10)$$

Where n_{points} are the points of each of the planes and ε is the mean square error calculated by the pcfitplane function.

Regarding the second improvement, it is intended to correct the lack of parallelism between the plane of the laser and the ground. Generally, in garages, placing the laser level on the roof of the car, the planes detected are perpendicular to the horizontal plane of the laser. However, this inclination is not negligible. To avoid this effect, a plane is defined that remains constant regardless of the possible rolling and pitching movements of the vehicle, or the possible inclination of the terrain. Therefore, a reference plane is defined as perpendicular to the detected vertical planes.

To sum up, the flowchart of the implemented algorithm is depicted in Figure 2.

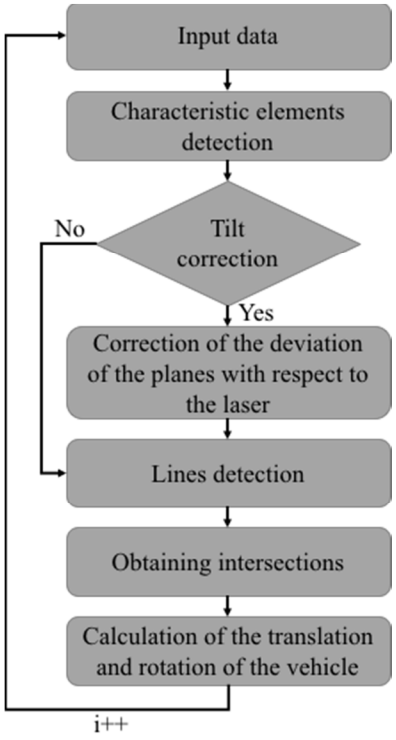


Figure 2. Algorithm flowchart

III. TESTS AND RESULTS

Different tests have been carried out not only in outdoors scenarios, but also in indoor scenarios using an instrumented vehicle that includes a 3D laser scanner Velodyne VLP-16. The LiDAR sensor was placed on top of the car in order to acquire a 360º field of view.

The tests have been carried out in 3 different scenarios:

- INSIA laboratory (Figure 3a). It is a particularly suitable environment for the calculation of the trajectory since it is an environment with a large number of vertical walls and some columns. In this scenario, 2 types of maneuvers have been carried out: straight manoeuvre and L-shape manoeuvre.

 ISBN: 978-1-61208-643-9

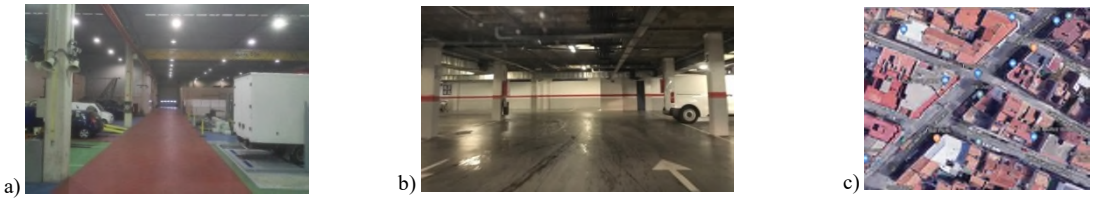


Figure 3. a) INSIA laboratory, b) Parking area, c) Narrow urban street

- Indoor parking area (Figure 3b). This type of environment also has columns and some vertical walls. However, there is not much quantity of them and being an environment with a low ceiling, a lot of points are lost in it. In addition, parked cars are also an obstacle to detection.
- Narrow urban street (Figure 3c). this method will work correctly in those streets where there are a certain number of intersections.

A. *Final method selection*

The first of the scenarios has been used to determine the most appropriate method among those presented for the calculation of the character elements, the determination of the turn and the translation, as well as in the alternatives of improvements over the algorithm. To do this, the bag is taken with L-shape trajectory and 7 points have been chosen from the main walls of the ship in which the maximum distance between the same wall detected in different frames is measured. The chosen points are those marked with a yellow square in Figure 4.

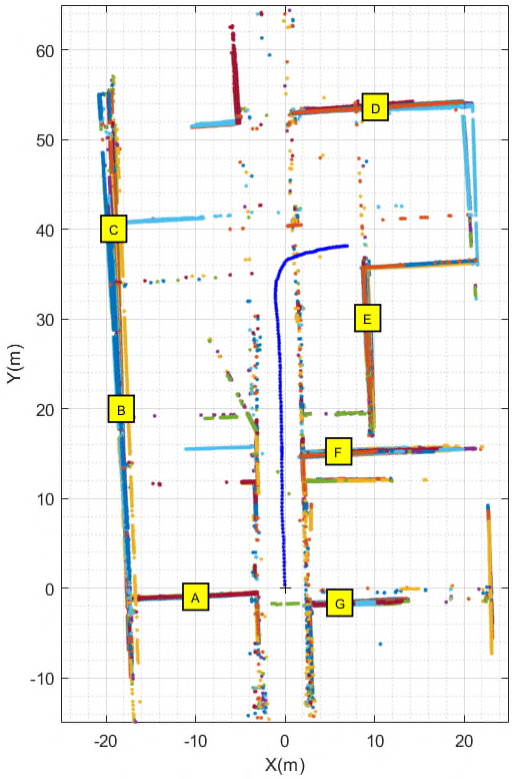


Figure 4. Control points for error calculation

First, we analyze the 2 methods to obtain the landmarks. It is observed as the first option of calculation of lines by layers has limitations in the detection of the planes from a certain distance. However, the function based on the plane-fitting is able to identify them and with a much lower computational time. Therefore, Method 2 is chosen from those described in section 2A.

To calculate the displacement and rotation of the vehicle, the three methods described above were tested, verifying that only method C provides null errors in ideal environments and, therefore, the best results in real environments.

Finally, the improvements proposed for the algorithm have been analyzed. The results are shown in Table I.

TABLE I. COMPARISON OF IMPROVEMENTS ALTERNATIVES

Improved method	Average error	Relative error
Base case (without improvements)	0,76 m	1,65 %
Points weighing. Distance	0,61 m	1,32 %
Points weighing. Angle	0,60 m	1,30 %
Points weighing. Combined	0,59 m	1,28 %
Lines weighing	0,86 m	1,86 %
Horizontal plane correction	0,57 m	1,24 %
Combined correction (except Lines weighing)	0,54 m	1,17 %

B. *Results in real scenarios*

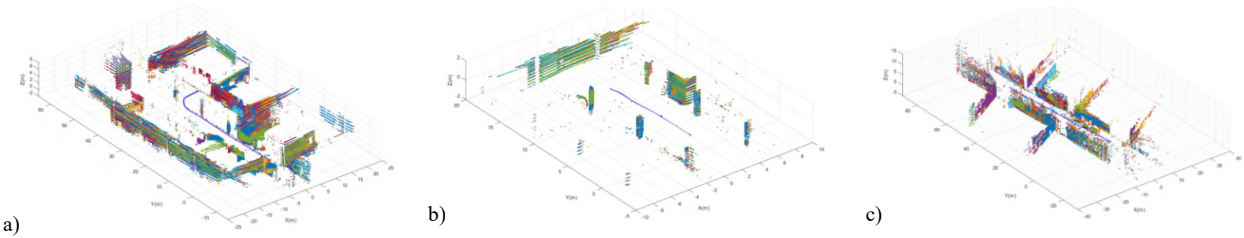
Finally, the 3 test environments were reconstructed using the selected method. Similarly to the preliminary trials, a set of control points have been defined to evaluate the accuracy of the reconstruction. Table II shows the results of this quality indicator and Figure 5 shows the reconstructions.

TABLE II. QUALITY INDEX IN RECONSTRUCTION

Scenario	Average error	Distance	Relative error
INSIA lab (straight trajectory)	0,35 m	64, 99 m	0,54 %
INSIA lab (L-shape trajectory)	0,54 m	46,13 m	1,17 %
Parking area	0,15 m	11, 94 m	1,24 %
Urban area	0,27 m	70,91 m	0,38 %

The lowest relative errors have been obtained in the INSIA laboratory when the trajectory is straight and in the urban circulation. Both environments are characterized by having large vertical planes, corresponding to the high walls of the workshop and the facades of the buildings, respectively. In this way, these planes when detected with a great number of points, the accuracy is higher, and thanks to this the calculation of the trajectory is also more precise.

 ISBN: 978-1-61208-643-9



a) b) c)

Figure 5. a) INSIA laboratory, b) Indoor parking area, c) Urban area

It is noted that, although in all cases the measurements of a Trimble BX935-INS multi-constellation GNSS receiver have been available, its signal has not been reliable due to the specific scenarios considered, so it cannot be taken as a reference.

IV. CONCLUSION

As conclusion, the developed SLAM technique requires specific characteristics of the environment in order to function correctly. Specifically, it requires the presence of a sufficient number of vertical planes, which will be used to calculate the relative movement of the vehicle. This fact happens in several real scenarios.

As an advantage, we could emphasize that it is a totally autonomous method, since it uses only the data obtained from the LiDAR, and therefore, it is not subject to the need to receive external signals as it happens to other positioning systems, such as GPS. Thanks to this, this algorithm can be used anywhere, and regardless of whether the environment is underground or outdoors, as long as it is an environment with the characteristics described above. The SLAM algorithm can be used to calculate the position of the vehicle in those areas where the GPS signal does not arrive or is weak. In addition, thanks to the simplifications introduced, its computational load is much lower than conventional SLAM algorithms, favoring its execution in real time.

The main limitation presented by the algorithm, as indicated, is the inability to calculate the displacement and rotation of the vehicle in those moments of time when the algorithm does not detect a sufficient number of characteristic elements. To solve these problems, it would be interesting to incorporate additional positioning systems such as INS, to compare the results with those obtained by the SLAM technique. Another alternative consists in incorporating other types of characteristic elements, such as, for example, the edges of the vertical planes, the poles of the traffic signs or the trunks of the trees.

Also, to facilitate detection, it could be interesting to incorporate a second LiDAR sensor. Thus, objects would be detected with a greater number of points, which would facilitate their detection.

ACKNOWLEDGMENT

This work has received the support of the MINECO CAV project (TRA2016-78886-C3-3-R), and Excellence Network SEGVAUTO-TRIES.

REFERENCES

[1] F. Jiménez, *Intelligent Road Vehicles: Enabling Technologies and Future Developments*. Elsevier, 2017.

[2] F. Jiménez and J. E. Naranjo, "Improving the obstacle detection and identification algorithms of a laserscanner-based collision avoidance system," *Transp. Res. Part C Emerg. Technol.*, vol. 19, no. 4, pp. 658–672, 2011.

[3] E. Nettleton, S. Thrun, H. Durrant-Whyte, and S. Sukkarieh, "Decentralised SLAM with low-bandwidth communication for teams of vehicles," in *Field Service Robot*, 2006, pp. 179–188.

[4] P. Newman, D. Cole, and K. Ho, "Outdoor SLAM using visual appearance and laser ranging," in *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pp. 1180–1187.

[5] A. J. Davison, "Real-time simultaneous localisation and mapping with a single camera," in *Proceedings Ninth IEEE International Conference on Computer Vision*, 2003, pp. 1403–1410 vol.2.

[6] J. M. M. Montiel, J. Civera, and A. J. Davison, "Unified Inverse Depth Parametrization for Monocular SLAM."

[7] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM: A factored solution to the simultaneous localization and mapping problem," *Proc. 8th Natl. Conf. Artif. Intell. Conf. Innov. Appl. Artif. Intell.*, vol. 68, no. 2, pp. 593–598, 2002.

[8] M. Mohan and K. MadhavaKrishna, "Mapping large scale environments by combining Particle Filter and Information Filter," in *2010 11th International Conference on Control Automation Robotics & Vision*, 2010, pp. 1000–1005.

[9] D. Hahnel, W. Burgard, D. Fox, and S. Thrun, "An efficient fastslam algorithm for generating maps of large-scale cyclic environments from raw laser range measurements," in *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453)*, vol. 1, pp. 206–211.

[10] Z. Zhang and Y. Shan, "Incremental motion estimation through modified bundle adjustment," in *Proceedings 2003 International Conference on Image Processing (Cat. No.03CH37429)*, vol. 3, p. II-343-6.

[11] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets Robotics: The KITTI Dataset."

[12] G. Pandey, J. R. McBride, and R. M. Eustice, "Ford Campus vision and lidar data set," *Int. J. Rob. Res.*, vol. 30, no. 13, pp. 1543–1552, Nov. 2011.

[13] J.-L. Blanco-Claraco, F.-Á. Moreno-Dueñas, and J. González-Jiménez, "The Málaga urban dataset: High-rate stereo and LiDAR in a realistic urban scenario," *Int. J. Rob. Res.*, vol. 33, no. 2, pp. 207–214, Feb. 2014.

 ISBN: 978-1-61208-643-9

[14] K. Konda and R. Memisevic, “Learning Visual Odometry with a Convolutional Network,” in *International Conference on Computer Vision Theory and Applications*, 2015, pp. 486–490.

[15] M. Clavijo, F. Serradilla, J. E. Naranjo, F. Jiménez, and A. Díaz, “Application of Deep Learning to Route Odometry Estimation from LiDAR Data,” in *VEHICULAR*, 2017, pp. 60–65.

[16] A. Nicolai, R. Skeele, C. Eriksen, and G. A. Hollinger, “Deep Learning for Laser Based Odometry Estimation,” in *Science and Systems Conf. Workshop on Limits and Potentials of Deep Learning in Robotics*, 2016.

[17] S. Wang, R. Clark, H. Wen, and N. Trigoni, “DeepVO : Towards End-to-End Visual Odometry with Deep Recurrent Convolutional Neural Networks,” pp. 2043–2050, 2017.

[18] G. Bresson, Z. Alsayed, L. Yu, and S. Glaser, “Simultaneous Localization And Mapping: A Survey of Current Trends in Autonomous Driving,” *IEEE Trans. Intell. Veh.*, vol. XX, no. X, pp. 1–28, 2017.

[19] J. Zhang and S. Singh, “Low-drift and real-time lidar odometry and mapping,” *Auton. Robots*, no. October 2014, 2016.

[20] P. H. S. Torr and A. Zisserman, “MLESAC: A new robust estimator with application to estimating image geometry,” *Comput. Vis. Image Underst.*, vol. 78, no. 1, pp. 138–156, 2000.

 ISBN: 978-1-61208-643-9