AUTONOMOUS EMBEDDED NAVIGATION SYSTEM - DETECTION, CLASSIFICATION AND TRAJECTORY PREDICTION OF OBSTACLES IN URBAN ENVIRONMENTS BASED ON LASER SENSOR DATA AND KALMAN FILTER

Breno Almeida Esteves, brenoae@hotmail.com
Poliane Torres Megda, politorresmegda@hotmail.com
Marcelo Becker, becker@sc.usp.br

Universidade de São Paulo – Escola de Engenharia de São Carlos – Departamento de Engenharia Mecânica, Grupo de Mecatrônica, Laboratório de Robótica Móvel (USP – EESC – SEM – GrMk – LabRoM), Av. Trabalhador são-carlense, 400, Pq. Arnold Schimidt, São Paulo/SP – Brazil – CEP 13566-590

Abstract. *Every day the quantity of vehicles on the roads around the world is increasing. This growth combined with the negligence of drivers and some external factors such as road and weather conditions result on a huge rise on the accidents and fatalities statistics. Since the beginning of the 21st century, many research groups and automotive companies are developing and adapting technologies that can be embed on the vehicles in order to reduce these dramatic numbers. One interesting example of these technologies is the detection and classification of mobile obstacles (vehicles, people, etc.) in urban environments. This work presents the development of algorithms for identification, classification, tracking, and prediction of moving obstacles,in special pedestrians. We used laser sensor data for monitoring the environment that surrounds our test vehicle (a modified passenger car). Based on these data we performed a computational treatment in order to classify all detected obstacles into two main classes: static and mobile obstacles. Then, the algorithm tracks and predicts the mobile obstacles positions for a certain time window by applying a Kalman Filter and a simply linear model for obstacle velocities. Even if the mobile obstacle is out of the sensor range (or occluded by other obstacles), the Kalman Filter used can predict its estimated position and trajectory for the time window. Another benefit of using a simple and more generic model is the fact that we are dealing with obstacles that may have different dynamic characteristics (e.g.: cars, motorcycles, bicycles, pedestrians, etc.). Based on the prediction of the obstacle positions, the vehicle navigator (an embedded navigation algorithm) can generate the best path taking into account all detected and hidden obstacles.*

Keywords*: autonomous embedded navigation system, obstacle detection, obstacle classification, trajectory prediction, laser sensor.*

1. INTRODUCTION

Every day, the quantity of vehicles traveling on the roads around the world is increasing. This fact, combined with negligence of the drivers and some external factors such as road and weather conditions, has increased drastically the statistics of accidents and fatalities. Due to this, today many research groups and automotive industries are studying and developing new technologies to improve the vehicles safety. Recently, some events and competitions took place around the world whose main goal was to demonstrate that researchers in the field of mobile robotics could be transferred and applied on autonomous passenger cars. Events like the DARPA Great Challenges (in 2005 the challenge in the Mojave Desert and in 2007 the urban challenge) in the United States and ELROB (yearly since 2006) in Europe are highlighting the use of these technologies into military and civilian vehicles. Thus, it is expected that in the near future passenger cars will have onboard systems that will assist the driver, for example, warning the driver if there is a danger situation. In addition to this assistive technology that increases the safety, there are also other technologies that can improve the driver comfort. An example of this technology that is becoming quite popular among costumers is the Intelligent Parking Assist (IPS), which is found in the Toyota Prius.

Many of the problems found in this research area are still open. The dream scenario of intelligent cars moving on intelligent roads all connected and sharing data by using some wirefire technology is still far away. But, many of the technologies developed for the mobile robotics field can be adapted and used on vehicles today. So, it is important to emphasize that in order to have an assistive or an autonomous passenger car it is necessary to have several different sensors and computers onboard the vehicle. These sensors would collect data about the current vehicle state and monitor its surrounds and give these data for the embedded computers that would analyze them and decide which is the best vehicle behavior taking into account the current scenario and the vehicle goal. This is not a simple task because urban environments are extremely dynamic. Depending on the sensor type used to monitor the environment, it is possible that when it detects an obstacle, it creates a blind region behind the detected obstacle. This can become a huge problem for the vehicle navigation procedure, because moving objects can suddenly appear and the vehicle may have no time enough to avoid a collision. In order to prevent this issue, tracking techniques are used for moving obstacles. So, it is possible to keep moving obstacles data for a certain time window even if they are occluded by other obstacles

and use these data to generate a safer path for the vehicle. In this paper we present the tracking algorithm developed. We used laser sensor data to detect and classify objects, and a Kalman Filter to track their movements and predict their paths and velocities.

2. METHODOLOGY

2.1. Segmentation Procedure

The laser reading data consists of distances measured between the sensor and the obstacle at fixed angular resolution (e.g.: 1°, 0.5°, etc.) and a quasi constant time interval. So, it is very likely that two sensor data points that are close may be part of the same object. If these points remain together after some readings, one may conclude that they belong to the same object. On the other hand, if after some readings their distance increases, one may conclude that they do not belong to the same object. Therefore, it is very important to calibrate the algorithm in order to use adequate threshold values for the maximum distance between reading to fuse them into a single group that forms an object.

In our case, we used a SICK LMS 291 laser sensor. It provides a complete reading of 180° range up to 80m as an ordered sequence of N points (P). Each point can be defined in Cartesian coordinates (x_n, y_n) or polar (r_n, α_n), that is:

$$\mathrm{P}=\{P_n=\begin{pmatrix} r_n \\ a_n \end{pmatrix}\}\ ,n\ \in [1,N] \tag{1}$$

Following the representation presented in Fig. 1 and in accordance with (1), a segment (S_i) can be described as:

$$S_i=\{(x_i,y_i)\ \text{ or } (r_i,\alpha_i), i=k{:}n\}, \text{ where } 1\le k<n<N. \tag{2}$$

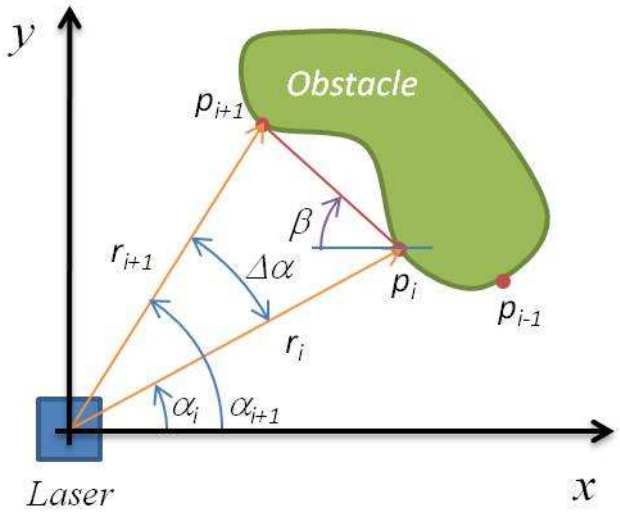


Figure 1: Representation of the laser readings and some of its parameters used on the segmentation procedure. In this figure, p_{i-1}, p_i, p_{i+1} represent a sequence of laser readings.

The segmentation algorithm takes as input data a complete sequence of laser readings (P) and delivers as output, pairs of values that represent the start and end of each segment detected. According to Premebida and Nunes (2005) it is possible to separate the segmentation methods into two main groups:

- Based on the distance between points (Point-Distance-Based - PDBs);
- Based on the use of Kalman Filters (Kalman Filter Based - KFB).

In this paper, we focused the use of Segmentation based on the distance between points. Basically, the algorithms belonging to this category has the following form: IF $D(r_i, r_{i+1}) > D_{lim}$ THEN the points do not belong to the same segment, ELSE the points are part of the same segment. Where D_{lim} is a threshold value for the maximum distance allowed between two points belong for them be classified as belonging to the same segment and $D(r_i, r_{i+1})$ is the Euclidean distance between two consecutive points, which is given by the following equation:

$$D(r_i,r_{i+1})=\sqrt{r_i^2+r_{i+1}^2-2r_ir_{i+1}\cos\Delta\alpha} \tag{3}$$

where $\Delta\alpha$ is the angular resolution of the laser sensor.

In the literature, one may find several methods used to determine D_{lim}. Dietmayer et al. (2001) proposed the following definition for this threshold value:

$$D_{lim} = C_0 + C_1 min\{r_i, r_{i+1}\} \tag{4}$$

Where $C_1 = \sqrt{2(1-\cos\Delta\alpha)} = D(r_i, r_{i+1})/r_i$, and C_o is a constant used in order to reduce noise.

Santos et al. (2003) included a new parameter (β – the angle between two consecutive readings, p_i and p_{i+1}, as presented in Fig. 1) in the previous equation, in order to reduce the segmentation dependence on the distance between the sensor and the object detected. So, Eq. (4) is changed and the new representation for the D_{lim} resulted in:

$$D_{lim} = C_0 + \frac{C_1 min\{r_i, r_{i+1}\}}{cotg(\beta)[\cos(\Delta\alpha/2) - \text{sen}(\Delta\alpha/2)]} \tag{5}$$

On the other way, a simpler method to define D_{lim} was proposed by Lee (2001):

$$D_{lim} = \left|\frac{r_i - r_{i+1}}{r_i + r_{i+1}}\right| \tag{6}$$

Finally, Borges and Aldon (2004) proposed a method called Adaptive Breakpoint Distance (ABD), which sets the threshold value as:

$$D_{lim} = r_i \frac{\text{sen}\,\Delta\alpha}{\text{sen}(\lambda - \Delta\alpha)} + \sigma_r \tag{7}$$

where λ is an auxiliary parameter and σ_r is the variance that addresses the random behavior of the sequence of sensor data points and the associated noise.

2.2. Obstacle detection

When it comes to obstacle detection, one may found in literature many ways to detect them in different kinds of environment based on laser sensor data. The biggest challenge in this procedure is the classification of the detected obstacles into different categories, for instance, static and moving obstacles. For fixed sensors, this task is quite easy, since the simple comparison between two consecutive readings can indicate if the objects are static or moving ones. However, the use of laser sensors onboard vehicles moving on roads becomes this task more complex. In order to solve this problem it is necessary to relay on an as much precise as possible localization procedure of the vehicle. So, the vehicle must have embedded a set of sensors composed by GPS, IMU, encoders, etc. Then, these data will be combined by applying some data fusion algorithm in order to estimate the vehicle position. As this is not the focus of this work, we decided that this subject will not be discussed here.

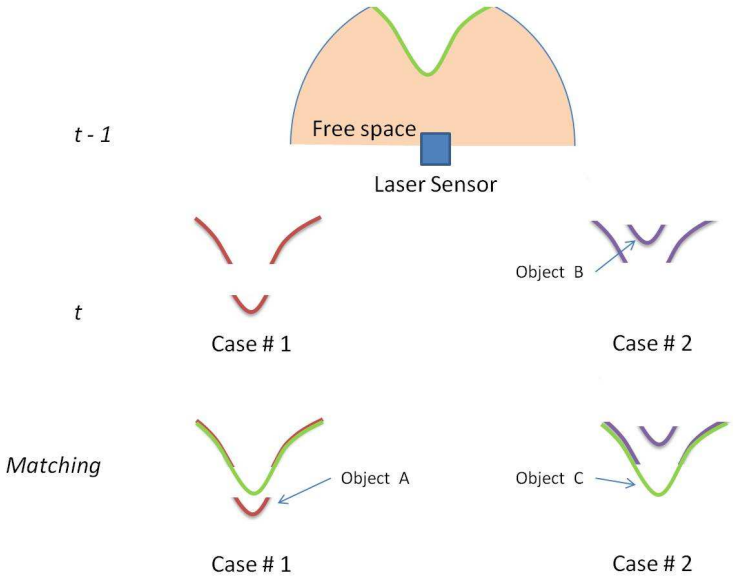


Figure 2: Illustration of the rules proposed by Wang and Thorpe (2002) for obstacle detection.

Wang and Thorpe (2002) presented in their work an obstacle detection method based on two rules:

- *Rule #1: Object detection*
  From previous readings, we know that some space is not occupied (free space). If an object appears in this space, this object should be moving. One may visualize an example of this case in Fig. 2.
- *Rule #2: Detection of objects that deviate*
  In Fig.2, based on two consecutive sensor readings (at time *t* and *t-1*), we can not conclude that the object B is moving, because it may be a new static obstacle that was hidden by the object C. However, we can conclude that the object C is moving. In order to really know if the object B is static, we need more than two consecutive sensor readings. The previous readings will help us to figure out the characteristics of the objects.

2.3. Local Map

A mobile robot needs to know as much information as possible about an environment in order to navigate in this environment. One way represent the environment data is the use of a grid representation and it was firstly described by Elfes (1989). This technique is widely used in applications of SLAM (Simultaneous Localization And Mapping), however, in this project we will use it in order to detect dynamic obstacles close to the robot. Essentially, the method consists of creating a two-dimensional map of the environment. This map is represented in the form of grid where each cell contains a value that indicates the probability that an object exists or not in that position. As the map is a discrete representation of the environment, it is very important to choose the grid size. This parameter is crucial because as small is the grid value, the greater the computational cost. In addition to this, large grid values cause loss of environment features. So, it is necessary to find a balance between the need of environment detailed representation and the computational cost associated. It is also necessary to highlight the need of a precise auto-localization of the map features and the mobile robot.

2.4. Obstacle Tracking Procedure

When it comes to dynamic environments (e.g.: urban like-environments), it is not enough to detect obstacles, it is also necessary to know their dynamic characteristics. Knowing these characteristics, the mobile robot control system can plan collision-free paths. Siegwart and Nourbakhsh (2004) present some methods that are used for path planning. Some methods act in a simplistic way and just ban the movement in a direction that leaves them closer to obstacles without worrying about the obstacles they are moving. But, for an efficient path planning is necessary to take into account the moving obstacles. This can be carried out by monitoring the dynamic characteristics of objects that are around the vehicle. In order to obtain even more efficient paths, one can estimate the obstacle future states, and add this information in the planning phase. So, the risk of collisions with obstacles decreases.

In urban-like environments it is quite common that obstacles close to the vehicle sensors hide other obstacles behind them. In this situation, the vehicle navigation system can become inefficient, disconsider hidden obstacles, and plan unsafe paths. So, in this scenario, the use of a tracking procedure that estimate the hidden obstacles state for a certain time window is more than welcomed.

In our research, each detected obstacle is characterized by the center of gravity of the occupied cells perceived by the laser sensor. As the laser provides us an environment reading on two dimensions, each obstacle requires two coordinates for its position in space. As we are also interested in how the obstacles move, we need to know the velocity components in *x* and *y*. These characteristics are united and stored in a vector that determines the state of the obstacle, as follows:

$$\vec{v}_{state} = \begin{pmatrix} x \\ v_x \\ y \\ v_y \end{pmatrix} \tag{8}$$

For simplicity, from now on this vector will be represented only by *x*. The vector containing the coordinates *x* and *y* measurements will be represented by the vector *z*. The state vectors and measurement estimated will be represented, respectively, by $\hat{x}$ and $\hat{z}$.

2.5. Kalman Filter

When it is necessary to predict future states of a dynamic system based on noisy measurements, the Kalman filter is a widely used technique (Kalman, 1960). The filter works recursively, i.e., it has the ability of calling itself. So, it is possible to solve problems by repeatedly processing the output of the same process. The Kalman filter consists basically of two steps. Firstly, an estimate is carried out based on the dynamic characteristics of the system based on the previous step data. Then, it performs an update of the estimate based on data from a new measurement. It is necessary to

highlight that Kalman filters are recommended only for linear systems. If the system is not linear, the use of more complex filters like the Extended Kalman Filter (EKF), or the Unscented Kalman Filter (UKF), is recommended.

In order to implement the Kalman filter it is necessary to adopt a dynamic model for the system in question. We used the model presented in Kohler (1997). The equations describing this model are linear, so we can apply the Kalman filter easily. So, the transition matrices of discrete time and the covariance of noise are given respectively by:

$$F(\Delta t) = \begin{pmatrix} 1 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \text{and} \quad Q(\Delta t) = \begin{pmatrix} \Delta t^3/2 & \Delta t^2/2 & 0 & 0 \\ \Delta t^2/2 & \Delta t & 0 & 0 \\ 0 & 0 & \Delta t^3/2 & \Delta t^2/2 \\ 0 & 0 & \Delta t^2/2 & \Delta t \end{pmatrix} \tag{9}$$

In the literature, it is possible to find other models that can be used. One of them is the constant angular velocity and it produces a more accurate estimative of the movement of vehicles while negotiating a curve. However, the system of equations used in this model is non-linear, so it would be necessary the use of the EKF in order to predict the vehicle states. As the EKF is more complex and computationally more expensive when compared to the Kalman filter, we decided to keep the use of a simpler dynamic and more generic model for the obstacles. So, the evolution of the dynamic system can be described by the following equations:

$$x(t + \Delta t) = F(\Delta t)x(t) + u(t + \Delta t) \tag{10}$$

$$z(t + \Delta t) = Hx(t + \Delta t) + w(t + \Delta t) \tag{11}$$

where *u* represents the white noise process, *w* the noise of the measurement *H* the measurement matrix, that performs the transformation from a state vector in a measurement vector and is given by:

$$H = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \tag{12}$$

Here starts the prediction step of the Kalman filter, where the state measured at the time interval, Δt, will be used to produce a new estimate. In addition, the state covariance, *P*, and the innovation covariance, *S* are also predicted. The innovation matrix is based on the error between the estimated and measured data. So, the prediction step is calculated based on the following equations:

$$\hat{x}(t + \Delta t) = F(\Delta t)\hat{x}(t) \tag{13}$$

$$\hat{z}(t + \Delta t) = H\hat{x}(t + \Delta t) + Q(\Delta t) \tag{14}$$

$$P(t + \Delta t) = F(\Delta t)P(t)F(\Delta t)^T \tag{15}$$

$$S(t + \Delta t) = HP(t + \Delta t)H^T + R \tag{16}$$

The matrix *R* represents the covariance of measurement noise and it is calculated multiplying the variance by the identity matrix, $R = I\sigma^2$. In order to improve the estimate done, a new sensor readings set is used. This is the updating step of the Kalman filter. The innovation can be calculated as the difference between the new sensor readings and the estimated value carried out in the previous step. So, the filter gain is obtained by using the covariance of the innovation state that was estimated in the previous step. Now, the prediction step can be updated based on the following equations:

$$v(t + \Delta t) = z(t + \Delta t) - \hat{z}(t + \Delta t) \tag{17}$$

$$W(t + \Delta t) = P(t + \Delta t)H^T S^{-1}(t + \Delta t) \tag{18}$$

$$\hat{x}(t + \Delta t) = \hat{x}(t + \Delta t) + W(t + \Delta t)v(t + \Delta t) \tag{19}$$

$$P(t + \Delta t) = P(t + \Delta t) - W(t + \Delta t)S(t + \Delta t)W^T(t + \Delta t) \tag{20}$$

2.6. Data Association

New laser sensor measures must be associated with a particular track, in order to detect obstacle movements. Yaakov and Fortmann (1988) described some methods used for data association in different conditions. One of these methods is known as "Joint Probability Method". Usually it is applied when the data acquisition is carried out in crowd

environments because it considers that sensor data that are in the same region not always belong to the same obstacle that is being tracked.

Among all the methods presented in Yaakov and Fortmann (1988), we considered that the most suitable one for urban-like environments is the "Nearest-Neighbor Standard Filter" (NNSF). In our case, some hypotheses were used to apply the NNSF method:

- Only a limited quantity of detected obstacles are considered at time *t*;
- In most cases, the barriers are sufficiently distant from one to another;
- If there is any doubt about the data association, the system must continue monitoring the data acquired in order to make sure it is tracking the same obstacle.

So, the system iteratively applies the NNSF and calculates all distances between the tracked obstacles and the new sensor readings. The distance is calculated using the Mahalanobis distance equation, $m = vS^{-1}v^T$. Thus, each track is associated with new reading set that is closer than a pre-defined threshold value.

2.7. Classification of Obstacles

To be able to monitor and predict obstacle movements, one must classify them into different classes. So, it is possible to choose appropriate dynamic models for them. As the most actors of the traffic in urban-like environments are vehicles (cars, busses, bikes, etc.) and pedestrians, the obstacles will be classified into these two classes.

Taking into account the sensor data, one may notice that there are remarkable differences between vehicle and pedestrian sizes. In all the readings in which pedestrians were identified, they appeared most often as two or three readings data. This happened because the sensor height was approximately 0.5m, so it can detect only the pedestrian legs. So, the obstacle classification can be carried out by calculating the standard deviation of the reading data that belong to the same obstacle:

$$\mu_X = \frac{1}{n}\sum_{i=1}^{n} x_i \quad \text{and} \quad \mu_Y = \frac{1}{n}\sum_{i=1}^{n} y_i \tag{21}$$

$$\sigma^2{}_X = \frac{1}{n}\sum_{i=1}^{n}(\mu_X - x_i)^2 \quad \text{and} \quad \sigma^2{}_Y = \frac{1}{n}\sum_{i=1}^{n}(\mu_Y - y_i)^2 \tag{22}$$

where: *x* and *y* are the Cartesian coordinates of the reading data of a specific obstacle, *n* is the quantity of reading data that were associated with the obstacle during the segmentation procedure, μ is the mean value, and σ the standard deviation of the reading data.

By calculating the standard deviation (Eq. 23), we can compare it with predefined values (that were obtained experimentally) in order to determine if the object is a vehicle or a pedestrian.

$$\sigma_{norm} = \sqrt{\sigma_X{}^2 + \sigma_Y{}^2} \tag{23}$$

3. RESULTS

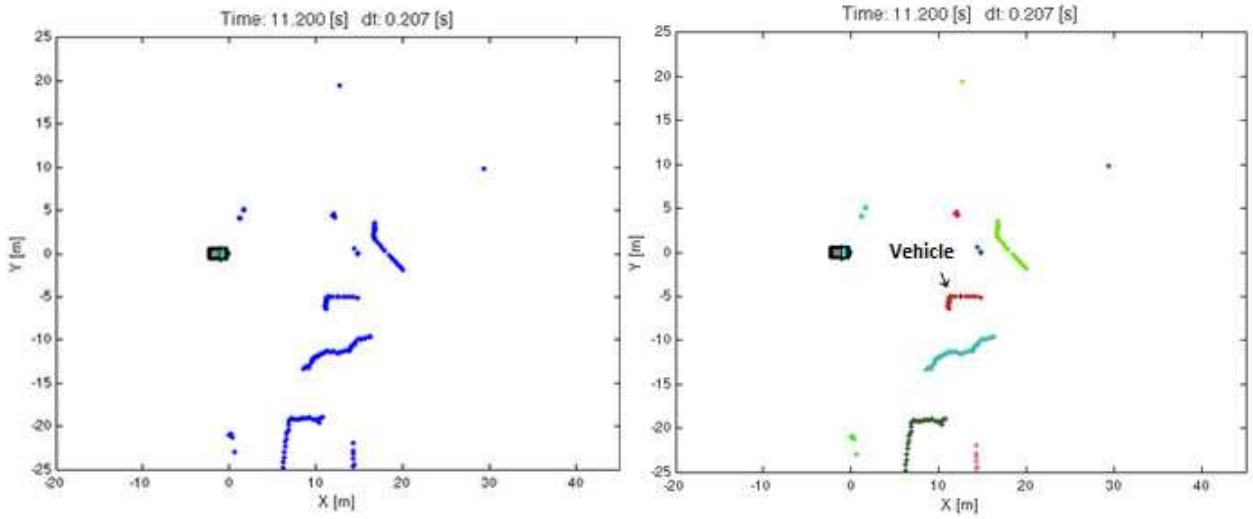
The experimental tests were carried out using our test vehicle, an adapted Fiat Stilo Dual Logic. This test vehicle has as embedded sensors a set of laser scanners, IMU, and GPS. It is also possible to access the odometry and other vehicle data through the vehicle CAN network. In Fig. 3 a photo of the vehicle is presented. In this configuration, 2 SICK LMS 291-S05 laser sensors were used in order to monitor the environment (one in the front of the vehicle and the other one, on its top). During the experiments for this article, we used only the data of the front sensor.

3.1. Segmentation Procedure

As previously described, after acquiring the laser data, it is necessary to carry out the segmentation procedure in order to select and group them into detected obstacles. Each sensor reading is a set of 181 points containing the distances between the sensor and the environment features detected. Initially the data is converted from polar to Cartesian coordinate system. Then, we assumed that the origin of this Cartesian coordinate system is in the sensor position, where the *x*-axis is placed at 90° (sensor local system), and the *y*-axis, perpendicular to the *x*-axis (at 180° in the sensor local system). The next step is the use of the "Point-Distance-based" (PDB) method to identify which sensor readings data belong to an object. We choose this method because it is very simple to implement. The results obtained were satisfactory, taking into account that different objects were mostly well distinguished. Figure 4 presents a reading of the laser and the associated segmentation procedure result. One may observe in Fig.4-b that each obstacle detected is represented by a different color.

Figure 3: Experimental test vehicle used – SENA Project (www.eesc.usp.br/sena).

The drawback of this method is the need of a threshold distance value between the laser data to segment them. If one chooses a value that is too large, two distinct objects can be fused as one. On the other hand, if the value is too small, the same object can be divided into two or more objects. Due to this, the selection of this threshold value becomes a trials and error task and the result is very affected by the kind of the robot/vehicle operating environment. Fig.5 shows the results obtained for two different trials. In the first one (Fig. 5-a) the segmentation algorithm fused several objects into a single one, and in the second one, (Fig. 5-b), a single object was divided into two distinct ones.

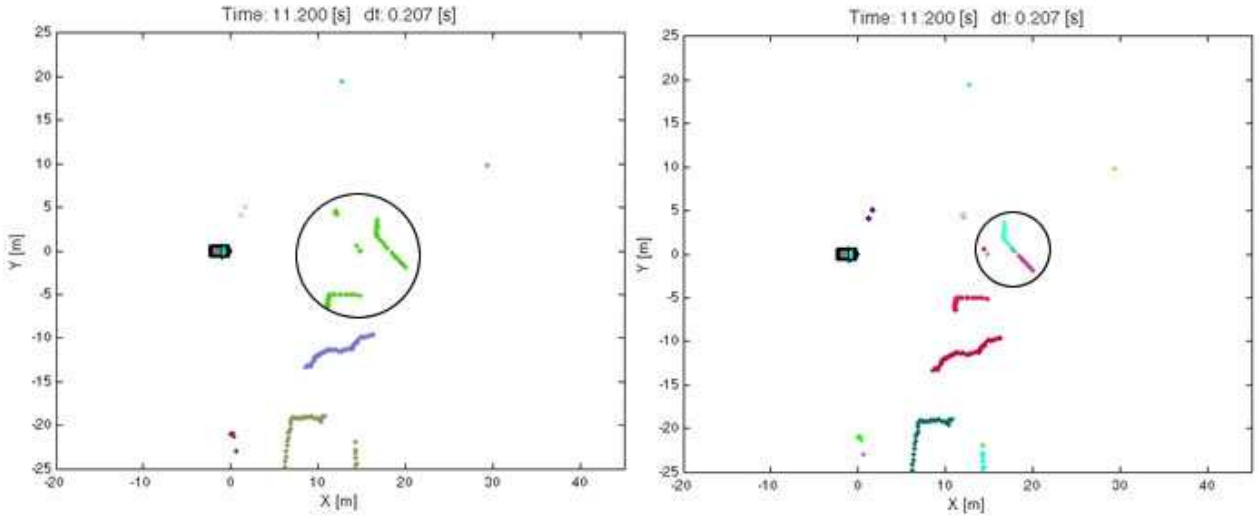


(a) (b)

Figure 4: Results obtained during the segmentation process. In (a): environment data acquired by the laser sensor (the blue dots represent the obstacles detected). In (b): the output of the segmentation procedure algorithm (each obstacle is represented by a different color - in detail, a vehicle that was detected).

3.2. Obstacle Detection Procedure

In order to detect the dynamic characteristics of the obstacles, we carried out comparisons between two consecutive laser readings. As they are separated by a known time interval, it is possible to detect possible movements of the obstacles and divide them into two classes, static and dynamic obstacles. The algorithm calculates the distance between the present and the previous reading data and evaluates the values obtained based on a threshold value. If a set of readings is within the limit of displacement, it is probably a static obstacle, if not, a moving obstacle (dynamic). However, there are situations where different parts of the same object overlap in two consecutive readings, giving the false idea that they are stopped (e.g.: when a car is detected moving perpendicularly to the sensor). This problem was solved by checking the previous classes of these obstacles. If they were previously classified as moving obstacles, they are classified as moving obstacles again.



(a) (b)

Figure 5: Results obtained during the segmentation process for two different values of the threshold distance between the laser data. In (a): the segmentation algorithm fused several objects into a single one (green dots inside the black circle). In (b): the segmentation algorithm divided a single object into two distinct ones (blue and violet dots inside the black circle).

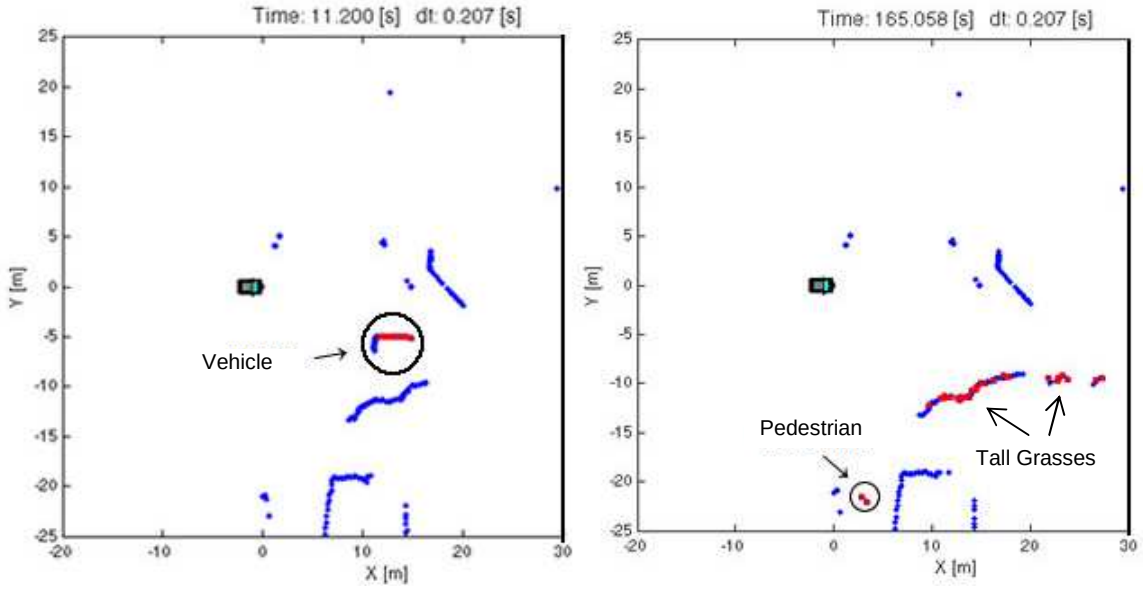
However, if no similarity between the data in the comparison phase is found, we can not guarantee that the object is moving. This can happen because the object may have been hidden by others obstacles on the previous scans or it could be out of the sensor range. In this case, we used a concept that determines a region of free area on the first reading. This area corresponds to a free region between the robot/vehicle and obstacles. If an object on the second reading (*t*) was not observed in the first one (*t-1*), but it is contained in the free area of the first reading, we can conclude that the object is moving. Aiming to facilitate the comparison of the sensor readings, we used a local map. The map size was chosen based on the sensor maximum range used. In our experiments we adopted a 60m maximum range, so we have a map size of 60m x 60m. The map was divided into cells small enough, so they could produce a good discretization of the environment, without becoming too computationally expensive (each cell has 0.2 m x 0.2 m). According to these dimensions, the local map is composed of 90,000 cells. Each cell has a value between 0 and 20. The higher is this value, the higher is the probability of the cell being occupied. Another reason for building a local map is to make easer the integration of the trackers data with a trajectory planner that is under development at our lab.

The algorithm results for obstacle detection and classification were encouraging. However, it is possible to optimize the algorithm in order to filter false classifications. Fig. 6 presents results of two different scenarios. In this figure, red dots represent sensor readings that were classified as belonging to moving objects, and the blue ones, as belonging to static objects. In the first situation, a moving car is detected, and the second a pedestrian is detected.

3.3. Tracking Obstacles and Predicting their Future States

Once it is possible to track the moving obstacles present around the robot/vehicle, it is possible to monitor them. This can be done by applying the Kalman Filter. For the first sensor reading, because there is no previous data, all detected obstacles are set as new trackers. The initial state vector is given by the measured position and all velocities are set as null. The covariance matrices are also initialized with convenient initial values. Each tracker has an associated variable that stores the quantity of scans in which it has not been updated by new sensor readings (this variable is called *unseen*). When a new tracker is inserted or updated, this variable is reset as null. In Fig. 7 one may visualize the detection of one pedestrians (the symbols used in this figure are presented in Tab. 1). The tracker algorithm is briefly described as follows:

1: FOR each tracker DO:
2: Estimate the state
3: Calculate the Mahalanobis distance for all new data
4: Check the smallest Mahalanobis distance (*m*)
5: IF $m \leq limit$ THEN:
6: Update the tracker with new data
7: $unseen = 0$
8: ELSE:
9: $unseen = unseen + 1$
10: IF no new measurements of a tracker for more than 1 sec:
11: Deletes the tracker
12: FOR each new measurement without correspondence:
13: Creates new tracker



(a) (b)

Figure 6: Results of the algorithm for obstacle detection.

Table 1. Description of the symbols used in figure 7.

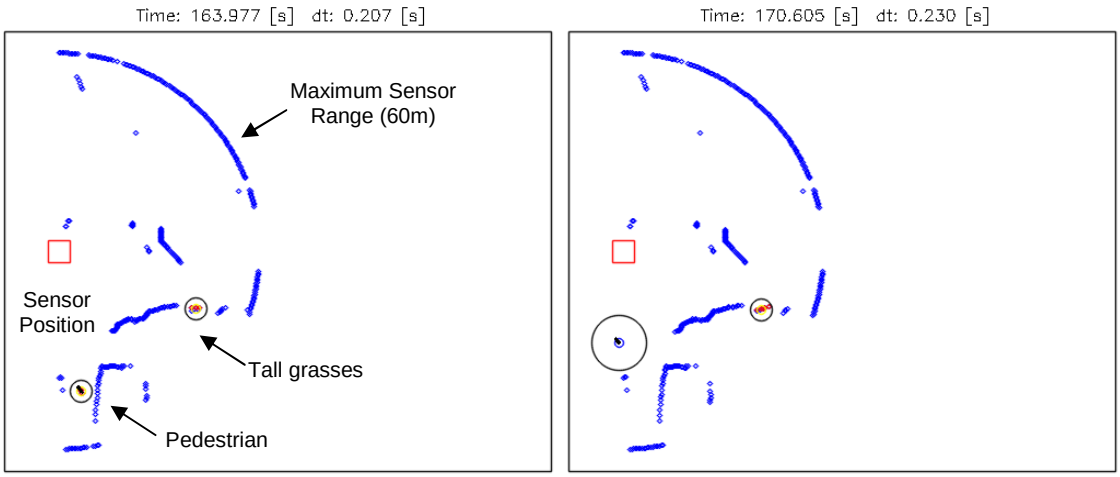
Symbol	Description
	Tracker being seen
	unseen Tracker
	Obstacle Speed
	Scan data
	Predicted Uncertainty

One can observe in this figure that the object position uncertainty is small while it is being detected by the sensor. When it is hidden by another obstacle or it leaves the sensor range area (Fig. 7-b), the uncertainty increases until the object is once more detected or it keeps out of the sensor range for more than 1 sec. In Fig. 7-b, it was possible to estimate the position and velocity of the pedestrian even while it was not detected by the laser sensor.

4. CONCLUSIONS

In this work we presented some of the key techniques for segmentation, detection, monitoring, and prediction of obstacle position and velocity. The segmentation procedure was implemented based on the distances between points method (PDB). The results were satisfactory and in a near future we plan to test the method based on Kalman filter (KFB) in order to compare both techniques. When it comes to obstacle detection, the algorithm that we implemented compares two consecutive scans. We concluded that it was efficient, but there are still cases where it did not work properly. For the purpose of solving this issue, a map site was created to improve the obstacle classification into the mobile and static classes. The use of a vision system would also help the algorithm to filter false mobile obstacles.

Reliable data from the obstacle detection system are vital in order to perform the prediction of object motions step. The obstacle detector algorithm ran efficiently most of the time, but some false classifications were produced due to the presence of tall grasses that were swinging because of the wing during the experiment. Once again, the use of a vision system can solve this problem and filter these results. The use of the Kalman Filter to predict the obstacle future states produced good results. Nevertheless, it is possible to improve the prediction using more complex dynamic models that can include motorcycles and bicycles. The main difference between them is related to their speeds. Finally, it is crucial to highlight the importance of detecting, monitoring, and predicting the movements of dynamic obstacles for the vehicle autonomous navigator. Based on these data, the system can figure out the best paths that would avoid collisions, and increase the traffic safety.



(a) (b)

Figure 7: Tracking Obstacles and Predicting their Future States. In (a) a pedestrian is tracked inside the sensor range, and in (b), the tracker is kept even while the pedestrian is out of the sensor range.

5. ACKNOWLEDGEMENTS

The authors would like to acknowledge FAPESP (Process No. 2009/04787-0), CNPq (Process No. 119523/2009-4), Fiat Automóveis, and the National Institute of Optics and Photonics (INCT-INOF) for their technical and financial support for this research.

6. REFERENCES

Becker, Marcelo; Hall, Richard; Kolski, Sascha; Macek, Kristijan; Siegwart, Robert; Jensen. Björn. 2D Laser-based Probabilistic Motion Tracking in Urban-like Environments. Journal of the Brazilian Society of Mechanical Sciences and Engineering, v. 31, p. 83-96, 2009.

Borges, G.; Aldon, M. Line Extraction in 2D Range Images for Mobile Robotics. Journal of Intelligent and Robotic Systems, p. 267–297, 2004.

Chieh-Chih Wang, C. Thorpe, Simultaneous localization and mapping with detection and tracking of moving objects. IEEE International Conference on Robotics and Automation (ICRA), 2002.

Dietmayer, K.; Sparbert, J.; Streller, D. Model based Object Classification and Object Tracking in Traffic scenes from Range Images. Proceedings of IV IEEE Intelligent Vehicles Symposium, 2001.

Elfes, A., Using Occupancy Grids for Mobile Robot Perception and Navigation, Computer, vol. 22, no. 6, pp. 46-57, IEEE Computer Society Press, 1989.

Hall, Richard. The use of Motion Tracking for Mobile Robot Collision Avoidance in Outdoor Environments. École Polytechnique Fédérale de Lausanne, 2006.

Lee, K. Reactive navigation for an outdoor autonomous vehicle. Master Thesis, 2001.

Kalman, R.E., 1960, “A new approach to linear filtering and prediction problems”, in: Trans. of ASME, J. Basic Engineering, Vol. 82, pp. 34 –45.

Kohler, M. (1997). "Using the Kalman Filter to track Human Interactive Motion – modeling and initialization of the KF for translational motion". In: TR 629. Informatik VII, Universität Dortmund, Germany.

Premebida, C.; Nunes, U. Segmentation and Geometric Primitives Extraction from 2D Laser Range Data for Mobile Robot Applications. In: Acta de. Coimbra: Universidade de Coimbra, 2005.

Robert MacLachlan, Tracking Moving Objects From a Moving Vehicle Using a Laser Scanner, Carnegie Mellon University, 2004, 18 de dezembro de 2009, < http://www.cs.cmu.edu/~ram/resume/datmo_report.pdf>.

Santos, S.; Faria, J.; Soares, F.; Araujo, R.; Nunes, U. Tracking of Multi-Obstacles with Laser Range Data for Autonomous Vehicles. Proc. 3rd National Festival of Robotics Scientific Meeting (ROBOTICA), p. 59–65, 2003.

SICK Product Database. 15 de dezembro de 2009, < http://www.sick.com/>.

Siegwart, R. and Nourbakhsh, I., "Introduction to Autonomous Mobile Robots", The MIT Press, Cambridge, Massachusetts, 2004.

Yaakov Bar-Shalom and Thomas E. Fortmann, "Tracking and data association", Academic Press Inc., London, 1988.

Wang, C.C. and Thorpe, C., "Simultaneous localization and mapping with detection and tracking of moving objects", Proc. of the IEEE Int. Conf. on Robotics and Automation - ICRA 2002, Vol. 3, pp. 2918-2924, 2002.

7. RESPONSIBILITY NOTICE

The authors are the only responsible for the printed material included in this paper.