2004 IEEE Intelligent Vehicles Symposium
University of Parma
Parma, Italy • June 14-17, 2004

Multi-target Detection and Tracking with a Laserscanner

Abel Mendes, Luis Conde Bento and Urbano Nunes, *Member of IEEE*

Abstract— **In this paper we present a method of detection and tracking of moving objects (DATMO) using a Laser Range Finder (LRF). The DATMO system is able to classify several kind of objects and can be easily expanded to detect new ones. It is composed by three modules: scan segmentation; object classification using a suitable voting scheme of several object properties; and object tracking using a Kalman filter that takes the object type to increase the tracking performance into account. The goal is the design of a collision avoidance algorithm to integrate in a Cybercar vehicle, which uses the computed time-to-collision for each moving obstacle validated by the DATMO system.**

I. Introduction

The automotive industry has been concentrating efforts in the development of safety systems in order to decrease the danger caused by driver's faults or distractions. The active cruise control (ACC) is one of those systems that is being developed mainly for highways driving [1], in order to relieve the driver of the rather monotonous job of constantly fine-tuning the vehicle's speed. Our work is intended to develop an anti-collision system based in a laserscanner, to be integrated in a low speed vehicle, that runs in Cybercars scenarios (*www.cybercars.org*). The bigger difference between these two scenarios arises from the fact that the system that runs in the highways works as a velocity tracker, concerning to the velocities of the ahead vehicle, otherwise the system running on a Cybercars environment works more as a collision detection system and must take care about a large number of object types, like pedestrians, cycles, cars, posts.

Obstacle avoidance is one of the most critical technologies needed for autonomous vehicles, due to the safety reasons, both for occupants and other road users (like vehicles, pedestrians, etc...). Research is under way to find solutions to this critical issue, based on range data [2], [3], most often fused with visual data [4].

The research on the field of autonomous vehicles has been mainly oriented in lane detection and obstacle detection, mostly using either LRF or vision systems. In [5] and [6] the object detection is similarly performed using a laser scanner, but different approaches are used in order to lane detection and classification. Sparbert and co-authors [5] used for the lane detection the same laserscanner as for obstacle detection, taking the features of road margins into account, while in [6] the road is detected and classified through the white lane markers, using a vision system.

Institute for Systems and Robotics; University of Coimbra-Polo II 3030-290 Coimbra, Portugal; abfm, conde, urbano@isr.uc.pt

This work was partially supported by EU project CyberCars (www.cybercars.org); and by FCT (Portuguese Science and Technology Foundation) POSI project

The prediction of objects behaviour is an essential issue for intelligent vehicles in order to prevent accidents. To achieve a good object tracking and behaviour prediction, an object classification module is essential in the system. Common classification methods are based on multi-hypotheses, being the objects reclassified over time with the new processed data. Vision systems are more suitable in object classification since they provide more detailed shapes of objects [7], nevertheless object classification is being used with LRF sensors [8]. In order to increase the performance of classification, multi-layer LRF sensors are rising to minimize the problem of the lack of information noted in 2D laser scanners [9], [3].

A. Overview

Figure 1 shows the dataflow between the modules that constitute our DATMO system. The feedback loop means that information of past scans is incorporated in the classification and object tracking processes over time to improve its performances. In each sample period (after a scan), we make the prediction of the position of the tracked objects and their classification parameters are updated. The DATMO system computes and sends to the vehicle path-tracking controller, the time-to-collision and the estimated impact point on the vehicle associated to each validated moving obstacle.

The paper structure is the following. Sections II and III describe respectively the modules of scan segmentation and object classification. The object tracking and time-to-collision computation modules are summarized in section IV. Section V describes succinctly our prototype vehicle used in the experimental tests. Finally some experimental results, showing the effectiveness of the DATMO system and some concluding remarks are presented in sections VI and VII, respectively.

II. Scan Segmentation

The purpose of the scan segmentation is to look for segments defined by several lines, fitting the points that represent each object. To do that, the process is started by grouping all the measures of a scan, into several clusters, according to the distances between consecutive measures, followed by the line fitting of the points of each cluster.

A. Clustering

The readings are subdivided into sets of neighbour points (clusters), taking the proximity between each two consecutive points of the scan into account. A cluster is hence, a set of measures (points of the scan) close enough to each other, which due to their proximity, probably belong to the same object. The segmentation criterion is based on the one

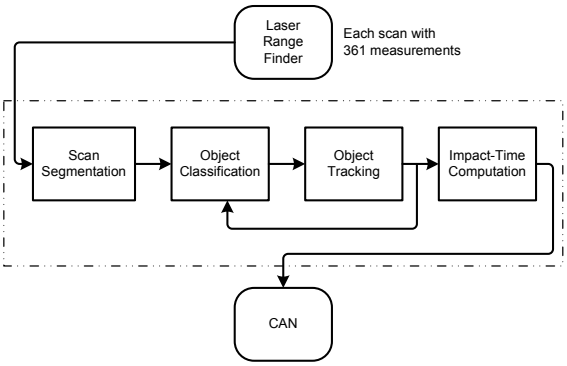

Fig. 1. DATMO system

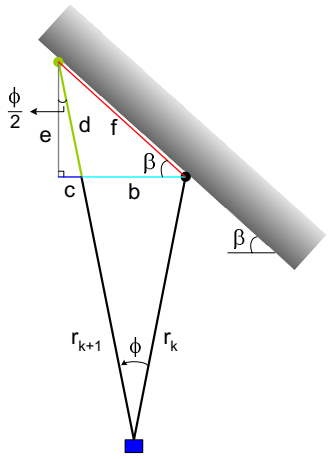

Fig. 2. Schematic of clustering method

proposed in [8]: two consecutive points, distant from the LRF, r_k and r_{k+1}, belong to the same segment as long as the distance between them fulfils the following expression:

$$r_{k,k+1} \leq C_0 + r_{min} \cdot \frac{\tan\beta \cdot \sqrt{2 \cdot (1 - \cos\phi)}}{\cos\left(\frac{\phi}{2}\right) - \sin\left(\frac{\phi}{2}\right)} \quad (1)$$

where $r_{min} = min\{r_k, r_{k+1}\}$, $r_{k,k+1} = |r_k - r_{k+1}|$ and ϕ is the angular resolution of the LRF. β was introduced to reduce the dependence of the segmentation with respect to the distance between the LRF and the object and C_0 to handle the longitudinal error of the sensor. If $C_0 = 0$, then β represents the maximum absolute inclination that an object's face can have to be detected as an unique segment (Figure 2). The distance d, represents the maximum difference allowable between two consecutive measures in order to consider them as belonging to the same object.

B. Line fitting

Assuming that the surrounding environment can be approximated by poligonal shapes, then line fitting is a suitable choice for object faces approximation. Thus, a good object characterization and data reduction can be achieved, resulting on a set of line segments per cluster, defined as:

$$L = \{l_i = [P_{ini}, P_{end}, m, b]^T : 0 \leq i < n\} \quad (2)$$

where the cartesian points P_{ini} and P_{end} are respectively the initial and the end point of the line, m and b are the straight line parameters. For each cluster, our process starts with a recursive line fitting [10] with the purpose of detecting the break points on the set of points of the cluster. The process starts connecting the first and last point of the cluster by a straight line (Figure 3.a)), and for all points in between the perpendicular distance to the line $d_{\perp,k}$ is calculated. If the maximum of these distances $d_{\perp,max}$ exceeds a given threshold, the line is splitted in the point of $d_{\perp,max}$ (Figure 3.b)) and two new sets of points are created. Afterwards, new tests of set splitting are made recursively for each set. The process ends, when no more splits take place (Figure 3.c)), which means that all sets can be approximated to a line with an error less than the threshold.

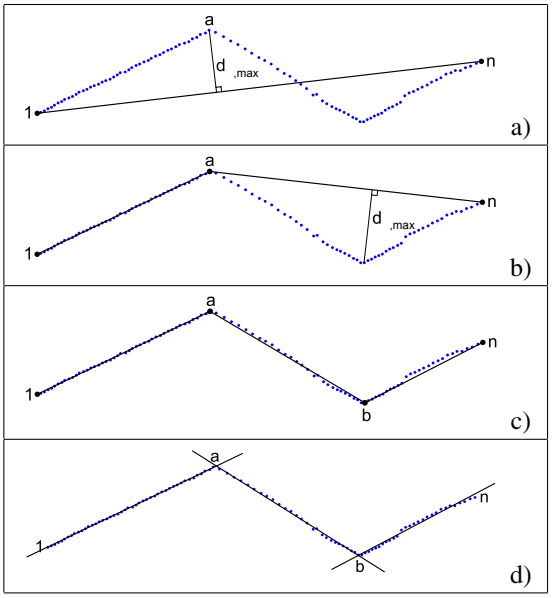

Fig. 3. Recursive line fitting example

The above procedure is used simply to detect break points, since the line fitting results can be improved with an additional data processing (see Figure 3.c, where the approximation line between points b and n is not the best one to fit all points in between). Thus, after splitting all the points of a cluster in subsets, an orthogonal regression [11] is applied to determine the straight line equation that minimizes the quadratic error (Figure 3.d).

C. Joining broken objects

During the grouping and line fitting processes, none effort is made to adjust the lines to the most common objects that can be found. For that reason, the segmentation can be very faithful regarding to the measure points, however not

corresponding to the related objects, specially for objects like pedestrian that often result in two segments, or big objects like walls with smaller ones in between them and the LRF, resulting in the division of the big one in two or more objects. In order to avoid those problems, two algorithms are proposed:

Legs detection Observing a sequence of LRF scans, we can easily notice that the shape of a pedestrian walking, alternates between one and two small objects, representing the legs. Then, when two small segments separated by a distance less than $50cm$ are found, our algorithm join both in one segment.

Broken lines The detection of broken walls or other big obstacle faces, is an iterative process that goes through the current segment set searching for lines bigger than a given threshold, and check for each one if there are other ones that might fit in the same line. Afterwards, for each pair of small lines that can fit the big one, another test must be done to make sure that there's nothing behind.

III. Object Classification

A new entity is introduced on this section named *object*, which is similar to segment, but classified in a sort of predefined types, tracked from previous scans and owning special features, like: type, velocity, size. The segments become objets after correspondence with objects detected on previous scans, or in case of lack of correspondence new objects are created.

The object classification module is made up by three submodules, with the following purposes: 1) making correspondence between objects and segments detected in the segmentation; 2) approximate the objects to geometrical figures like circles and rectangles, in order to decrease the amount of needed information to describe the objects; 3) classify each object in a set of possible ones.

A. Segment-Object correspondence

The segment-object correspondence can be subdivided in three types: the first one, where the correspondence is immediately assumed due to the small distance between the centre of geometry of the actual segments and of the predicted position of previous tracked objects; the second one, when the object tracking is not properly adapted to real movements, originating undesirable estimations of the object movements and consecutively bigger distances of the corresponding objects in consecutive scans; the third type, representing segments without correspondence with older objects.

The process uses a table of segment-object distances similar to that one shown in Table I, containing distances less than δ_0. So, the implementation runs through the set of actual segments and calculates per each one, the distance to every object previously detected and filling the table using equation 3.

$$\lambda_{i,j} = \begin{cases} |o_i - s_j| & if\ |o_i - s_j| \leq \delta_0 \\ 0 & if\ |o_i - s_j| > \delta_0 \end{cases} \quad (3)$$

	s_1	s_2	$\dots$	s_n	
o_1	λ_{11}	λ_{12}	$\dots$	λ_{1n}	$\Sigma\lambda o_1$
o_2	λ_{21}	λ_{22}	$\dots$	λ_{2n}	$\Sigma\lambda o_2$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
o_m	λ_{m1}	λ_{m2}	$\dots$	λ_{mn}	$\Sigma\lambda o_m$
	$\Sigma\lambda s_1$	$\Sigma\lambda s_2$	$\dots$	$\Sigma\lambda s_n$	

TABLE I

Table of distances between segments and objects, used in the correspondence process

with $i = 1..m$ and $j = 1..n$.

The auxiliary parameters $\Sigma\lambda s_j$ and $\Sigma\lambda o_i$ are respectively the number of elements in column j bigger than 0 and the number of elements in row i bigger than 0.

In Table I, the unambiguous correspondences are the (i, j) elements where $\Sigma\lambda s_j = 1$ and $\Sigma\lambda o_j = 1$, which means that, for segment j there is only one object i sufficiently close to be considered as correspondent. The problem becomes harder for bigger values of either $\Sigma\lambda s_j$ or $\Sigma\lambda o_i$, which represent more than one hypothesis of correspondence to the segment j or object i. In order to skirt that problem, other characteristics are taken into account, like: distance; dimension; orientation; occluded time and life time, through the following weight function:

$$P(o_i, s_j) = \begin{bmatrix} \rho_1 \\ \rho_2 \\ \rho_3 \\ \rho_4 \\ \rho_5 \end{bmatrix}^T \begin{bmatrix} e_{dist}(o_i, s_j) \\ e_{dim}(o_i, s_j) \\ e_{orient}(o_i, s_j) \\ T_{occlusion}(o_i) \\ T_{life}(o_i) \end{bmatrix} \quad (4)$$

with $i = 1..m$ and $j = 1..n$ and where each weight $\rho_u (u = 1...5)$ is obtained empirically in the range of $0..1$ and is related to the ability of segment-object correspondence making. The values of the right vector are normalized with their maximum acceptable values.

The occlusion time $(T_{occlusion})$ represents the time since the last segment-object successful correspondence. When this time exceeds $2s$ the object is deleted from the object tracking list.

The weight $P(o_i, s_j)$, is calculated for each segment-object pair (i, j) not yet affected. The correspondence process resumes starting by the bigger weight and decreasing until all those pairs have been corresponded, without duplicated correspondences. When the correspondence process is finished for both tables, the segments not affected will origin new objects.

B. Classification

Ideally, the object classification should give every object, a trustful classification on every scan. However, that is one of the biggest problems found on that subject, due to the LRF sensor characteristics, where the shape of the same object, for instance a car, can alternate from a small line to two big perpendicular lines. The way to surpass this problem is to take into account as many features as possible

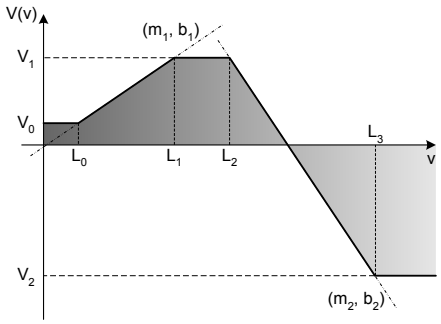

Fig. 4. Graphical representation of voting function

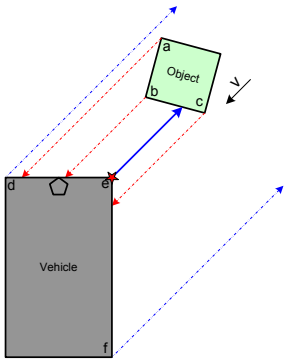

Fig. 5. Geometrical method of collision computation. (vector V represents the obstacle velocity relatively to the vehicle; edge e will be the impact point)

from previous object detections, in particular dimensions and dynamics.

Since we cannot classify an object with an high confidence immediately at the first scan when it appears, our classification method is based on a voting scheme considering every hypotheses over time, until have got an high classification confidence [12]. Each feature that characterizes the object, represent a voter actor, where the weight of the vote depends on the influence of the related feature to characterize the object and on the value ν of that feature (equation (5), graphically represented on Figure 4).

$$V(\nu) = \begin{cases} V_0 & \nu \leq L_0 \\ m_1\nu + b_1 & L_0 < \nu < L_1 \\ V_1 & L_1 \leq \nu \leq L_2 \\ m_2\nu + b_2 & L_2 < \nu < L_3 \\ V_2 & \nu \geq L_3 \end{cases} \quad \text{with } \nu \in R^+ \qquad (5)$$

The confidence level is achieved by adding the votes of every actor, and when some of the hypothesis reaches a reasonable value, we assume that the object is classified on the type of that hypothesis.

IV. Object Tracking and Time-to-collision

In this work, the object tracking is performed by a Kalman filter, assuming an object model with constant velocity and white noise acceleration [13], considering different maximum accelerations for each object type. For each detected object, the Kalman filter is independently applied to every hypotheses of object type while the related object has not been classified in an unquestionable class. Despite the high processing time needed, applying the filter in this way results in serious improvements on the tracking ability, once we are taking into account every hypotheses of object type, which is very useful in order to make easier the task of segment-object correspondence.

The time-to-collision computation module uses the results of tracking system to estimate the time-to-collision and position, for each one of the detected objects. The method consists on projecting all possible points of impact in the direction of the object velocity, assuming for each instant a constant object velocity relative to the vehicle. As we can see in Figure 5, these points are the edges of the car (d, e, f) and of the object (a, b, c). So, from the projection lines starting on the object, defined by the starting points and velocity vector, we select the shortest line that intercepts a line segment of the boundary of the vehicle. Applying the same method for the projection lines that start on the vehicle, we finish the process and achieve the colliding shortest distance. With the knowledge of the object velocity and the shortest distance, we can easily determine the time-to-collision.

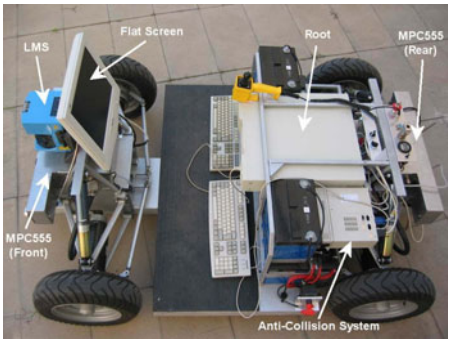

Fig. 6. Experimental testbed

V. Vehicle Prototype Testbed

As experimental testbed we use a four wheel-drive electrical vehicle (Figure 6) called “Robucar” [14], equipped

Fig. 7. Environment picture taken at first scan

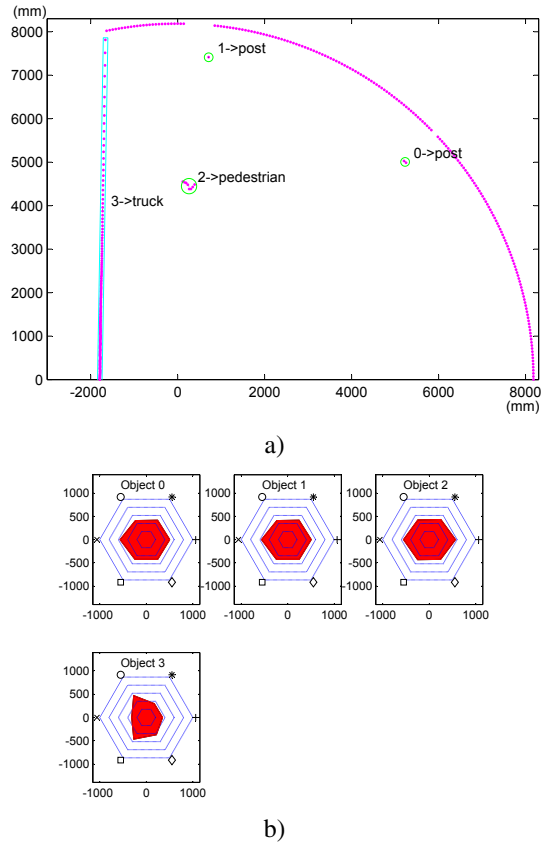

Fig. 8. Figure a), result of the entire process after the first scan of real data, and figure b), results of classification. Where: $+ \rightarrow$ pedestrian; $* \rightarrow$ car; $o \rightarrow$ truck; $\times \rightarrow$ post; $\square \rightarrow$ wall; $\diamond \rightarrow$ bush

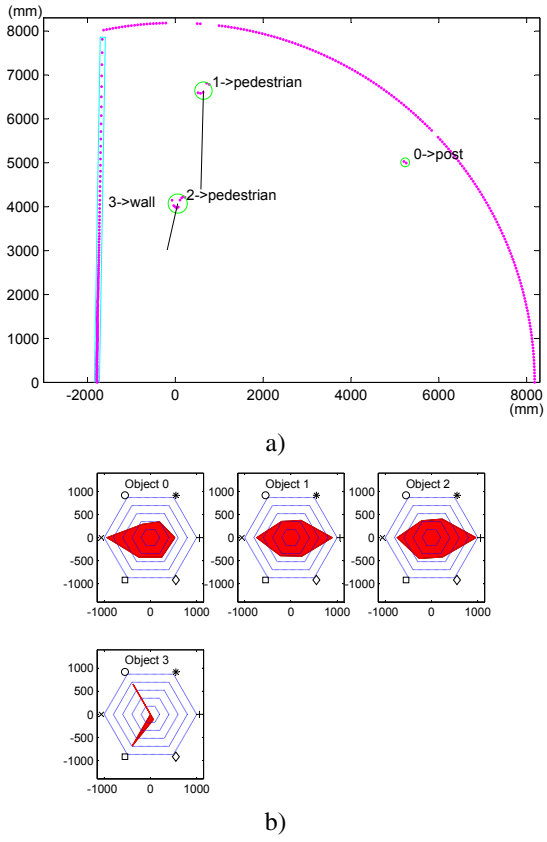

Fig. 9. The same as in figure 8, but after the 6^{th} scan. The lines associated with objects in image a), represent the object velocity

with a *LMS200-Sick* laser scanner [15].

The interconnections among all subsystems of our prototype, are made by two CAN buses, "Internal CAN" and "External CAN". The first one is used in the control motion of the car, making the connections among the motor controllers (Motorola micro-controller MPC555), the motor encoders and the central computer (*Root*). The "External CAN" is mainly used to interconnect different sensor systems like our anti-collision system and the central computer.

The hardware of DATMO system is composed by the laserscanner connected to an embedded computer with a Pentium III-1GHz, running RedHat 7.3 operating system. The laserscanner was setup with an angular range of 180^{o}, length range of $8m$, angular resolution of 0.5^{o} and a transfer rate of $500kbps$. To this configuration corresponds a longitudinal resolution of $1cm$ and a scan rate up to $37.5Hz$. The transfer rate of $500Kbps$ was made possible in our setup by developing a RS232 to CAN converter based on a PIC18F258.

VI. Results

This section presents results of outdoor real experiments, with two pedestrians, a tree (identified as post by the classifier; this tree is not shown in the picture of Figure 7) and a wall within the sight of view of the laserscanner. Figure 7 shows a picture at the beginning of our experimentation.

Each one of the eight classification graphs presented in Figures 8.b and 9.b, represents the amount of votes per type, of every detected object.

Figure 8.a presents the result of object detection and classification of two pedestrians, a wall and a post, just in the first scan of the experimentation. The insufficient information of the first scan lead the classifier to fail on object 1 and 3, but in the classification graphs (Figure 8.b) we can see that none of them owns a sufficient amount of votes to be considered as a valid classification. So, the confidence of the classification is very low at this time. This situation changes after some scans, as can be seen for example in Figure 9, showing the algorithm results for the 6^{th} scan.

The information acquired since the first scan, lead the classifier to more reliable decisions as can be observed in Figure 9.b, where the object 0 can already be classified as $post$, with high probability. The votes of object 1 and 2 have grown to $post$ and $pedestrian$, because both of them have small sizes (in the horizontal plane under analysis) and were stopped up to the 4^{th} scan. The classification graph of object 3 is related to the wall, and the reason of the two high probabilities shown, is due to the similarities between

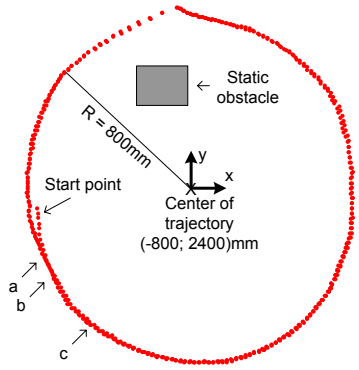

Fig. 10. Representation of the setup used to test the overall system, including the estimated robot trajectory. The laserscanner is attached to the middle of the front face of a static rectangular vehicle. The world coordinates system is defined by the position/orientation of the laserscanner

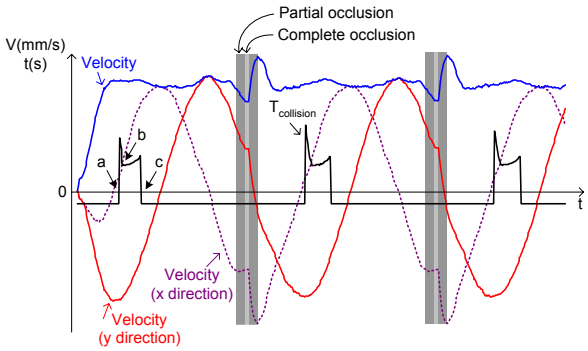

Fig. 11. Estimated velocities of the robot by a kalman filter with occlusion periods and estimation of time to collision. Note: the time less than zero means no predicted collision

a wall and the side of a stopped truck.

The overall system was tested on the setup shown in Figure 10, defined by a static vehicle with dimensions $1200 \times 2000(mm)$ integrating a laserscanner in the middle of its frontal face, that corresponds to the origin of the coordinates, a moving robot defining a circular movement with $800mm$ radius with constant velocity of $820mm/s$ and a static obstacle occluding the moving robot for some samples.

The red path of Figure 10 shows the estimated position by the kalman filter along the circular trajectory described by the robot, even when the robot is occluded by the static obstacle. In the Figure 11, the kalman estimations are also shown by means of the estimated linear velocities of the robot, with the occlusion periods represented by the shaded regions. As final result of the system the black line denotes the time of collision of the moving robot with the vehicle. With the circular movement described and assuming constant velocity, the possibility of collision begins when the robot passes by point "a" in the direction of the rear left of the vehicle. Afterwards, the point "b" represents the transition of collision with lateral and frontal

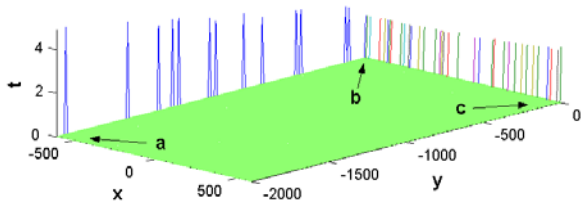

Fig. 12. Representation of estimated impact positions on the vehicle, with the respective time to impact. The points "a", "b" and "c" represent the estimated impact positions on the vehicle, when the robot was in the respective points "a", "b" and "c" shown in Figure 10

face and point "c" when the direction of the robot doesn't intercept the vehicle.

VII. CONCLUSION

A DATMO system integrating a time-to-collision computation is described in this paper. The system proved to be efficient on tracking multi-objects over time, resulting in good velocity estimates. Using the velocity estimates, reliable results in time-to-collision computation are achieved.

REFERENCES

[1] BMW, "Acc - active cruise control," BMW AG, Munich, Germany, Document de travail pour séminaire, 2000.
[2] C.-C. Wang, C. Thorpe, and A. Suppe, "Ladar-based detection and tracking of moving objects from a ground vehicle at high speeds," *IEEE Intelligent Vehicle Symp.(IV 2003)*, 2003.
[3] K. C. Fuerstenberg, K. C. J. Dietmayer, and V. Willhoeft, "Pedestrian recognition in urban traffic using a vehicle based multilayer laserscanner," *IEEE Intelligent Vehicle Symp., Versailles, France*, 2002.
[4] R. Labayrade, C. Royere, D. Gruyer, and D. Aubert, "Cooperative fusion for multi-obstacles detection with use of stereovision and laser scanner," *11th Int. Conf. on Advanced Robotics (ICAR 2003), Coimbra, Portugal*, pp. 1538–1543, 2003.
[5] J. Sparbert, K. Dietmayer, and D. Streller, "Lane detection and street type classification using laser range images," *IEEE Intelligent Transportation Systems*, August 2001.
[6] N. Shimomura, K. Fugimoto, T. Oki, and H. Muro, "An algorithm for distinguishing the types of objects on the road using laser radar and vision," *IEEE Transactions on Intelligent Transportation Systems*, vol. 3, no. 3, September 2002.
[7] A. J. Lipton, H. Fugiyoshi, and R. S. Patil, "Moving target classification and tracking from real-time video," *IEEE Image Understanding Workshop*, pp. 129–136, 1998.
[8] K. C. J. Dietmayer, J. Sparbert, and D. Streller, "Model based object classification and object tracking in traffic scenes," *IEEE Intelligent Vehicle Symp., Tokyo, Japan*, pp. 25–30, 2001.
[9] K. C. Fuerstenberg, "Pedestrian detection and classification by laserscanners," *9th EAEC International Congress*, June 2003.
[10] T. Einsele, "Localization in indoor environments using a panoramic laser range finder," Ph.D. dissertation, Technical University of München, September 2001.
[11] mathpages, "Perpendicular regression of a line," http://mathpages.com/home/kmath110.htm.
[12] T. Deselaers, D. Keysers, R. Paredes, E. Vidal, and H. Ney, "Local representations for multi-object recognition," *DAGM 2003, Pattern Recognition, 25th DAGM Symp.*, pp. 305–312, September 2003.
[13] M. Kohler, "Using the kalman filter to track human interactive motion - modelling and initialization of the kalman filter for translational motion," Informatik VII, University of Dortmund, TR 629, 1997.
[14] robosoft, www.robosoft.fr.
[15] SICK, "Proximity laser scanner," SICK AG, Industrial Safety Systems, Germany, Technical Description, 05 2002.