The 2010 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 18-22, 2010, Taipei, Taiwan

Describing the environment using semantic labelled polylines from 2D laser scanned raw data: Application to autonomous navigation.

Pavón N., Ferruz J., and Ollero A., *Member, IEEE*

Abstract— **This paper describes a real-time method that obtains a hybrid description of the environment (both metric and semantic) from raw data perceived by a 2D laser scanner. A set of linguistically labelled polylines allows to build a compact geometrical representation of the indoor location where a set of representative points (or features) are semantically described. These features are processed in order to find a list of traversable segments whose middle points are heuristically clustered. Finally, a set of safe paths are calculated from these clusters. Both the environment representation and the safe paths can be used by a controller to carry out navigation and exploration tasks. The method has been successfully tested in simulation and on a real robot.**

I. Introduction

SERVICE robots are becoming part of our world every day. Those performing cleaning, surveillance or entertainment tasks can be already bought in electrical appliance stores. Assistant robots able to interact with people are also being tested and improved [1].

Autonomous vacuum cleaners, for example, have become very popular. This kind of mobile robots needs to navigate in unknown indoor furnished environments, distinguish between different rooms and build a map to get a complete cleaning coverage. However, commercially available models normally use simplistic random path-planning algorithms and very limited sensors [2], so their operation is quite inefficient. Finding low-cost solutions for this kind of problems is currently a challenge, and a subject of research. An interesting comparative between different cleaner robots is carried out in [3]. An approach for navigation of cleaning robots in an unknown workspace using a triangular cell map representation is presented in [4]. A technique that estimates the size of the cleaning area using only the information of encoders and collision detectors can be found in [5].

Surveillance mobile robots also need to navigate safely. Apart from exploring a place, these vehicles must be capable of recognizing intruders and properly interacting with the user. A general purpose home service robot called 'ISSAC' applicable for vacuum cleaning or home security tasks, among others, was tested in an apartment in [6].

Assistant robots for the elderly and disabled people also have to navigate and explore the environment. Furthermore, in this case, human interaction is a very important issue that should be taken into account ([7], [8]).

Independently of the specific goal of a service mobile robot, all of them share similar control and navigation problems. The set of needed or highly desirable interrelated functions includes low level control of dynamic and kinematic behaviours, path tracking, path planning, geometric and topological map building, map-based localization, autonomous exploration and finally, at the highest level, the capability of extracting a cognitive representation of the environment in order to interact with it.

Sensors used in the perception process determine not only how successful the result will be, but also what kind of techniques can be applied. Monocular and stereo cameras are a good solution to recognize real life objects according to their texture, colour and shape; however, adverse environmental conditions, such as lighting and contrast changes can compromise the results. Range sensors such as ultrasonic, infrared and 2D or 3D laser scanners are also very common. Ultrasonic and infrared devices are usually cheap but their performance may be poor in real environments if compared to laser scanners. However, small, low-cost versions of the latter are increasingly available, so they have become a valid option for the design of affordable service robots. Furthermore, well-known simulation environments, such as *Player/Stage* [9], include robot models with 2D laser sensors very similar to the real devices.

When a robot should understand and extract information from the environment using only the raw data obtained from a 2D laser range finder, it is necessary to process those data and obtain a compact representation of the information. Techniques based on line extraction have been found to be very useful in indoor environments [10]. An experimental evaluation of different line extraction algorithms is presented in [11].

Recognizing natural landmarks in the environment from the extracted lines is the next step. Landmarks and lines can be combined to create metric maps [12], useful for localization tasks. Polylines (or polygonal approximations), are useful when a set of line segments must be clustered. From this linear representation of the environment, it is possible to generate topological maps [13], used by the robot to find known locations or carry out specific tasks proposed by a higher level planning module [14].

A number of research works addressing the building of an environment representation from 2D laser range finder data by means of line extraction have been presented ([10], [11], [12]). In general, they are only focused on obtaining a geometrical description of the environment.

In this paper, a new method for line extraction is

This research work was partially supported by the projects ROBAIR (DPI2008-03847), CONET (SI-0190/2008) and RURBAN (P09-TIC-5121).

described. The proposed method generates a hybrid representation of the environment where metric and semantic representations are combined. A set of linguistically labelled polylines is extracted from the raw data perceived by a 2D laser range finder. It provides a compact geometrical representation of the indoor environment where the representative points are semantically described. On the other hand, some interesting labelled points are selected and processed in order to find a list of traversable segments. Finally, a set of safe paths is computed from the middle points of the traversable segments. Higher level planning modules use this particular representation to generate specific behaviours in the robot.

The paper is organized as follows. Section II explains how the hybrid description of the environment is obtained. Next, section III shows how to calculate traversable segments from the set of labelled polylines and subsequently generate potential exploration paths according to the location. Section IV presents some experiments, including real-time testing on a mobile robot. Finally, Section V summarizes the conclusions, highlighting the advantages of the proposed method. Future research developments are also enumerated.

II. SEMANTIC DESCRIPTION OF THE ENVIRONMENT

This section describes the algorithms that obtain a hybrid (metric and semantic) description of the environment.

A. From raw data to polylines

In the experiments, a Hokuyo 2D laser scanner is being used. Specifically, raw data obtained from this sensor consist of the distances to objects in a 240º measuring area with an angular resolution of 0.36º approximately. The laser scanner can *see* objects within the distance interval of [0.006, 5] meters. Fig. 1 shows an example of scanned raw data.

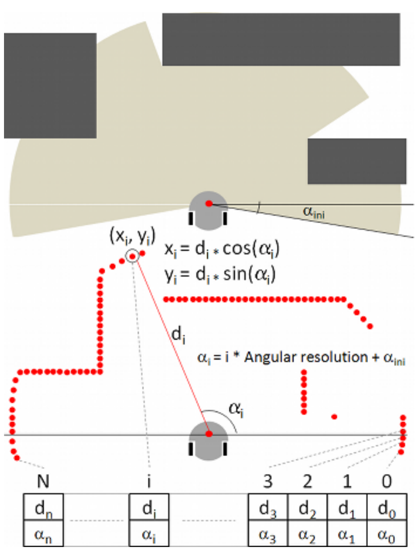


Fig. 1: Scanned data at a given time.

The data structure shown in Fig. 1 stores the set of raw data using a polar representation of each point with respect to the reference system of the vehicle. The first processing step involves generating a polygonal representation of the environment that fits such input data. As mentioned in Section I, several methods have been used to extract lines from a 2D laser scanner raw data. Fig. 2 shows a typical polygonal representation obtained by standard techniques.

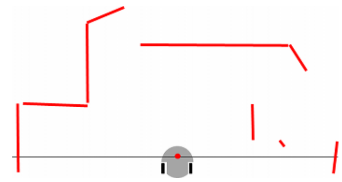


Fig. 2: Polygonal representation of the environment using a standard lines extraction method.

The goal of standard line extraction algorithms is limited to the generation of a geometric representation of the environment, and no specific meaning is assigned to features. In contrast, the proposed method carries out the line extraction by taking into account heuristically defined semantic properties based on relations between the raw points obtained by a 2D laser scanner. The extracted line segments are connected by means of semantically labelled representative points. Features are clustered into labelled polylines according to their specific meaning.

B. Semantic labeling of features

First of all, it is necessary to define which semantic properties will be taken into account. Since the robot navigates through an indoor furnished environment, representative scanned points should include features usually found in this kind of locations such as:

- *Occlusions (OC)*: A point will be labelled as *occlusion* when it defines one of the ends of a polyline.
- *Inside angle corners (IAC):* This label will be applied to a point when it is the vertex of an inside angle.
- *Outside angle corners (OAC):* This label will be applied to a point when it is the vertex of an outside angle.
- *Lateral wall proxy points (LW)*: If one of the points defining an extracted line segment on the right or on the left of the vehicle is located in front of the robot, and the other is located behind the robot, the line defines a lateral wall. A representative point of this line will be labelled as a *lateral wall proxy point*.
- *Frontal wall proxy points (FW)*: If one of the points defining an extracted line segment in front of the vehicle is located to the right and the other is located to the left, the line defines a frontal wall where there is a representative *frontal wall proxy point* inside the line.
- *Saturated points (SP)*: Points that define an interval where the laser scanner returns the maximum value are called *saturated points*. Lines delimited by saturated points are called *saturated lines*.

Fig. 3 illustrates an example of semantic features extraction using the raw points showed at Fig. 1.

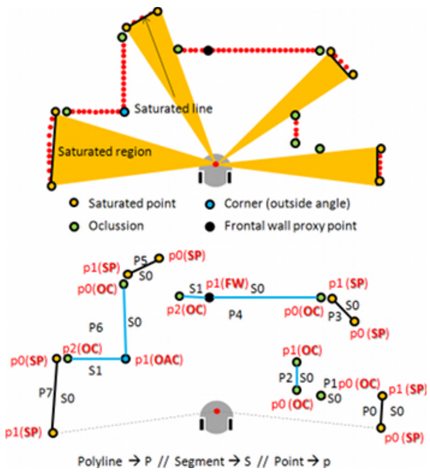


Fig. 3: Lines extraction process using semantic labelled points.

The proposed algorithm based on successive refinements uses the set of raw data as input and provides a list of labelled polylines (see Fig. 3), by means of these steps:

(Step 1) Extracting saturated regions (see Fig. 4).

> ***while*** *i* < *n*
> ***if*** d(p_i) is maximum ***then***
> ***if*** ∀p ***in*** [p_{i-k}, p_{i-1}] / d(p) is not maximum ***then***
> label p_i as SP;
> ***while*** d(p_{i+1}) is maximum and *i* < *n* ***do*** *i*++;
> ***if*** (*i* == *n*) label p_{n-1} as SP;
> **else**
> ***if*** ∀p ***in*** [p_i, p_{i+k}] d(p) is not maximum ***then***
> label p_i as SP and *ADD Saturated Line*;
> ***else*** *i*++;
>
> Where *d(p_i)* *is the distance to point p_i*, *k* is heuristically selected and *n* is the number of points.

Fig. 4. Extracting saturated regions.

(Step 2) Extracting non-saturated regions (see Fig. 5).

> ***for*** each pair of consecutive saturated lines (p_{ini1}, p_{end1}) and (p_{ini2}, p_{end2})
> ***for*** i ***in*** [end1, ini2]
> d1 = d(p_i); d2 = d(p_{i+1});
> ***if*** |d1 – d2| > *th1* ***then*** label p_i as OC;
>
> Where *th1* is heuristically fixed.

Fig. 5. Extracting non-saturated regions.

(Step 3) Extracting regions bounded by pairs of occlusion points.

The number of occlusion points between two consecutive saturated regions is always even. Every pair of OC points are the ends of a new region where a new polyline will be detected. In Fig. 3, the polyline P1 consists of only one point. In this singular case the first and last OC points are the same (p0). This kind of polylines will be filtered out by high level software components that use this representation of the environment.

(Step 4) Labeling representative features of a polyline (see Fig. 6).

> ***for*** each region bounded by OC points
> ***for*** each p_i
> ***if*** (∀p ***in*** [p_{i-k}, p_{i-1}] → d(p) > d(p_i)) **AND**
> (∀p ***in*** [p_{i+1}, p_{i+k}] → d(p) > d(p_i)) **AND**
> (∀p_j ***in*** [p_{i-k}, p_{i-1}] → d(p_j) > d(p_{j+1})) **AND**
> (∀p_j ***in*** [p_{i+1}, p_{i+k}] → d(p_j) < d(p_{j+1}))
> ***then***
> angle = calculate_angle((p_{i-k}, p_i), (p_i, p_{i+k}));
> ***if*** angle < *th2* ***then***
> label p_i as OAC;
> ***else if*** (∀p ***in*** [p_{i-k}, p_i] → d(p) < d(p_i)) **AND**
> (∀p ***in*** [p_{i+1}, p_{i+k}] → d(p) < d(p_i)) **AND**
> (∀p_j ***in*** [p_{i-k}, p_{i-1}] → d(p_j) < d(p_{j+1})) **AND**
> (∀p_j ***in*** [p_{i+1}, p_{i+k}] → d(p_j) > d(p_{j+1}))
> ***then***
> angle = calculate_angle((p_{i-k}, p_i), (p_i, p_{i+k}));
> ***if*** angle < *th2* ***then***
> label p_i as IAC;
>
> Where *k* and *th2* are heuristically fixed.

Fig. 6. Labeling representative features of a polyline.

Finally, when the angle is higher than *th2*, the point can be labelled as FW or LW according to the definition described at the beginning of this section.

Fig. 7 illustrates the data structure that stores the list of polylines consisting of semantically labelled points according to the example shown in Fig. 3.

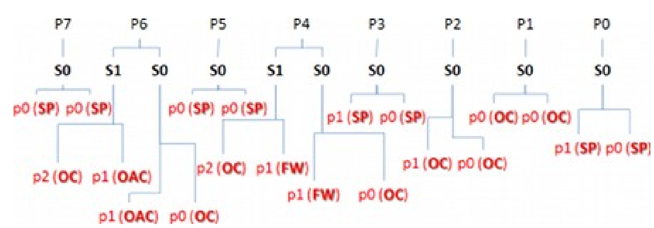


Fig. 7. Semantic representation of the environment using labelled polylines.

III. CALCULATING EXPLORATION PATHS

A set of feasible exploration paths can be calculated by taking into account the relation between labelled features detected using the algorithms described in Section II. This section addresses the description of the method used to solve this problem.

A. Obtaining traversable segments

A traversable segment (TS) is defined as a line segment that can be safely crossed by the vehicle; it is bounded by two labelled points that belong to different polylines.

The set of potential traversable segments is calculated by considering all pairs formed by OC and OAC points from different polylines. Polylines bounded by SP points are also added to this set. Only those segments that the vehicle can cross avoiding any collision with lines of the environment and whose size is bigger than the security diameter of the robot are considered safe. Unsafe segments are erased from the set together with those defined by too distant points. The following conditions are also taken into account: a) traversable segments do not intersect polylines and b) the triangle defined by the ends of a traversable segment and the reference point of the vehicle does not contain any labelled point.

Fig. 8 shows the traversable segments calculated from the data structure described at Fig. 7.

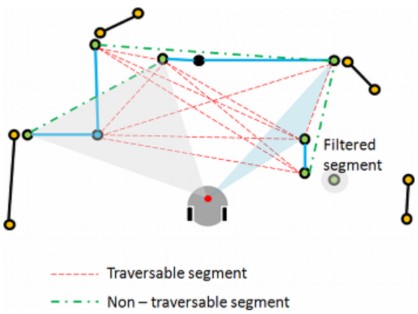


Fig. 8: Traversable segments calculated from the labelled polyline-based representation of the environment.

B. *Calculation a set of exploration paths*

Once the set of traversable segments is available, exploration paths can be calculated. The middle points of these segments are clustered according to five angular sectors: *Front*, *Left/Front*, *Right/Front*, *Left* and *Right* (see Fig. 9). A sector is defined as an angular interval or cluster [β_i, β_j), where β_i and β_j are heuristically fixed.

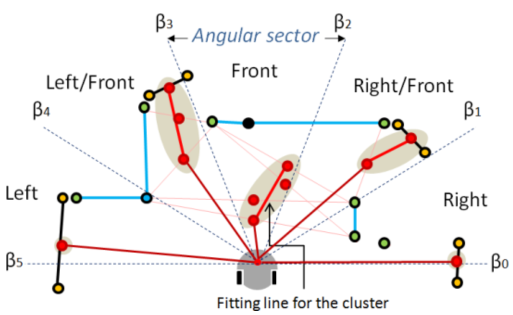


Fig. 9. Angular clusters and set of exploration paths.

A safe line fitting the points of a given angular interval is calculated:

- For each pair of middle points in the interval, a new line containing them and avoiding collisions with any of the environment polylines is obtained.
- The distance from each point in the interval to that line is calculated. The line fitting the maximum number of points with a distance lower than a given threshold is selected.
- Once the line is calculated, the middle points whose distance is higher than the threshold are erased from the cluster.

A higher level software component can select points inside a line located at a given distance and track them using any well-known path tracking method. The selection of the best path will depend on the objectives defined by a higher level process. Fig. 9 shows that there are five feasible paths to follow.

If all the angular clusters contain zero points, the robot is located in front of a closed convex space. In this case, the higher level controller can use FW, LW and IAC points to explore this kind of places and finally go out. If an object is detected at a distance lower than a threshold, a specific security algorithm is exceptionally executed: The robot decreases its velocity and properly turns around itself until the obstacle is avoided.

IV. EXPERIMENTS AND RESULTS

The experiments carried out to test the method are addressed in detail. The simulated and real robots are also briefly described in this section.

A. *The mobile robot BENDER*

The mobile robot BENDER is a low-cost platform designed and developed by the authors. Furthermore an *ad hoc* Player/Stage model has been also created to simplify testing (see Fig. 10).

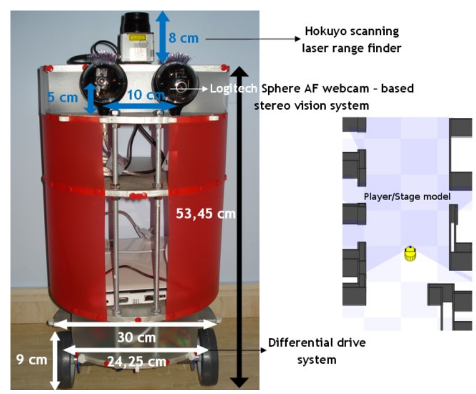


Fig. 10: The robot BENDER and the *ad hoc* Player/Stage model.

A full description of the software and hardware architectures of BENDER and the simulated environment based on the Player/Stage model can be found in [15] and [16].

BENDER is a wheeled differential-drive robot. Encoders installed on the wheels allow to implement odometry-based position and heading estimation. It is also equipped with a visual stereo system and a Hokuyo 2D laser scanner, accurate enough for the needs of an indoor service robot.

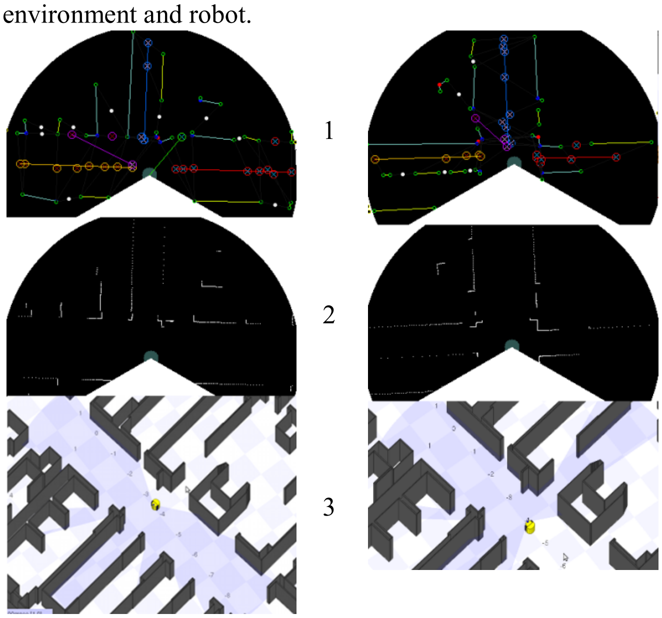
The software architecture that controls BENDER is fully distributed. It is based on the YARP middleware framework for communication [17]. High level software components exchange information with low level controllers in the vehicle and other high level processes by means of named YARP ports. The names of the ports used by the low level processes are the same for the real robot and the simulated one, so no changes in the high level software components are necessary to move from one environment to the other. This feature allows for easy and efficient system testing.

B. Semantic description of the environment and exploration tests.

This subsection shows how to use the semantic representation and the feasible exploration paths calculated in the previous sections to design a high level software component that allows to carry out a given task.

Specifically, the designed component uses the set of labelled polylines and the exploration paths to select a set of goal points that can be pursued using the well-known pure pursuit method [18]. The high level process selects the path that maximizes a function weighed by the number of points belonging to a cluster and the length of the corresponding fitting line. Normally, paths that accomplish these properties define corridors and wide exploration zones where many labelled features are found. When the robot finds a closed place (only one polyline is detected), the high level process changes its behaviour and sends commands to the robot in order to follow the perimeter of the polyline until new traversable segments are detected.

Fig. 11 shows the results obtained after applying the proposed method with the Player/Stage simulated environment and robot.



(a) (b)

Fig. 11. Experiments carried out in the simulated environment.

Fig. 11 (1a) and 11 (1b) show the set of labelled polylines and also the list of feasible paths. Fig. 11 (2a) and 11 (2b) show the raw data. Finally, the simulated environment and the position of the robot are shown in Fig. 11 (3a) and 11 (3b).

The experiments carried out with the real robot are presented in the video attached to this paper. BENDER navigates in the corridors of an office building. Initially, the robot does not have any knowledge about the environment. The proposed method and the high level process that defines the behaviour of the robot according to the set of labelled polylines are remotely executed in a laptop. The embedded computer within the vehicle sends the raw data scanned by the laser range finder and receives the commands using the YARP communication framework. The distributed application successfully works in real-time.

Fig. 12 shows the results obtained in a real environment which is geometrically and semantically described with high accuracy.

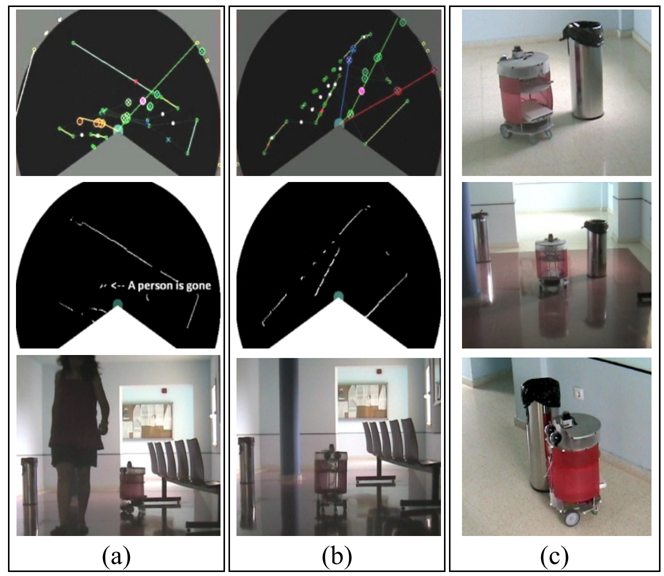


Fig. 12. Experiments carried out in a real environment using the robot BENDER.

Fig. 12 (a) shows several frames obtained from the first experiment in the video. They illustrate the point of view of the robot (above), the raw data (in the middle), and finally, a person walking in front of the robot. Fig. 12 (b) shows the second experiment in the video. Fig. 12 (c) shows three frames obtained from the third experiment; in this case, the robot detects and avoids dynamically inserted obstacles.

V. CONCLUSIONS AND FUTURE WORK

A wide range of experiments has been carried out in order to validate the proposed technique, both with the simulated environment and the real robot.

The method has demonstrated to be very robust even when velocity changes and the environment is cluttered and dynamic. The set of labelled polylines accurately fits the raw data both geometrically and semantically. Since this set is calculated by means of successive refinements, new semantic properties can be easily added if necessary. This feature is interesting if the robot should work in partially outdoor environments.

The compact semantic representation of the environment (providing abstract relations between features that can be used by higher level processes to generate an intelligent behaviour of the robot), and the generation of traversable segments whose middle points are processed to calculate safe exploration paths, together with the low computational complexity allows to achieve real-time performance using standard hardware.

The proposed method provides advantages with respect to the standard line extractions techniques, which just build a geometric representation of the environment, or other classic topological representations based on Voronoi/Delaunay tessellation, for instance, whose computational complexity is higher.

At present, a method to build topological maps is under development. It will obtain a cell decomposition of the free space based on labelled triangles, using the set of labelled polylines and traversable segments. This new method is intended to facilitate localization and loop closure tasks in cluttered scenarios.

Future work will also address an improved semantic representation of the environment using not only a polygonal approximation of the objects, but also more complex descriptions of them, including visual features.

On the other hand, interaction with humans is also an emergent issue and the linguistic representation of features makes easier its translation to a human-compatible description by higher level software components, so exploration tasks will be improved allowing humans to send topological commands expressed in natural language.

REFERENCES

[1] S. Thrun, "When robot meet people", IEEE Intelligent Systems and their Applications, vol. 13, issue 3, August 2002, pp. 27 - 29.

[2] M. Tresanchez, T. Palleja, M. Teixido, J. Palacín, "Estimating a room size using encoders and collision detectors: application to a cleaning mobile robot", IEEE International Symposium on Intelligent Signal Processing, 2007. WISP 2007, vol. 51, no. 3, Alcalá de Henares, Spain, October 2007, pp. 1 - 6.

[3] M. Schofield, "Neither master nor slave..." A practical case study in the development and employment of cleaning robots", Proceedings of the 7th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA '99, vol. 2, Barcelona, Spain, October 1999, pp. 1427 - 1434.

[4] Joon Seop Oh, Yoon Ho Choi, Jin Bae Park, Y.F. Zheng, "Complete coverage navigation of cleaning robots using triangular-cell-based map", IEEE Transactions on Industrial Electronics, vol. 51, no. 3, June 2004, pp. 718 - 726.

[5] J. Palacín, J. A. Salse, I. Valgañón, X. Clua, "Building a Mobile Robot for a Floor-Cleaning Operation in Domestic Environments", IEEE Transactions on instrumentation and measurement, vol. 53, no. 5, October 2004, pp. 1418 – 1424.

[6] Bum-Jae You, Myung Hwangbo, Sung-On Lee, Sang-Rok Oh, Young Do Kwon, San Lim, "Development of a home service robot 'ISSAC'", IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003), vol. 3, Las Vegas, Nevada, USA, October 2003, pp. 2630 - 2635.

[7] R.D. Schraft, C. Schaeffer, T. May, "Care-0-botTM: The Concept of a System for Assisting Elderly or Disabled Persons in Home Environments", Proceedings of the 24th Annual Conference of the IEEE Industrial Electronics Society, 1998. IECON '98, vol. 4, September 1998, pp. 2476 - 2481.

[8] G. Nejat, M. Ficocelli, "Can I be of Assistance? The Intelligence behind an Assistive Robot", Proceedings of the IEEE International Conference on Robotics and Automation, ICRA 2008, Pasadena, CA, USA, May 2008, pp. 3564 - 3569.

[9] T. H. J. Collett, B. A. MacDonald, and B. P. Gerkey, “Player 2.0: Toward a Practical Robot Programming Framework”, Proceedings of the Australasian Conference on Robotics and Automation (ACRA 2005), Sydney, Australia, December 2005.

[10] J. González, A. Stentz, A. Ollero, "An Iconic Position Estimator for a 2D Laser Range Finder", Proceedings of the 1992 IEEE International Conference on Robotics and Automation, vol. 3, Nice, May 1992, pp. 2646 - 2651.

[11] V. Nguyen, A. Martinelli, N. Tomatis, R. Siegwart, "A Comparison of Line Extraction Algorithms using 2D Laser Rangefinder for Indoor Mobile Robotics", IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2005), August 2005, pp. 1929 - 1934.

[12] L. Charbonnier, O. Strauss, "A suitable polygonal approximation for laser rangefinder data", Proceedings of the Intelligent Vehicles '95 Symposium, Detroit, USA, September 1995, pp. 118 - 123.

[13] B. Kuipers, Y. T. Byun, "A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations", Journal of Robotics and Autonomous Systems, vol. 8, 1991, pp. 47-63.

[14] D. Rawlinson, R. Jarvis, "Ways to Tell Robots Where to Go. Directing Autonomous Robots Using Topological Instructions", IEEE Robotics & Automation Magazine, vol. 15, no. 2, June 2008, pp. 27 - 36.

[15] N. Pavón, J. Ferruz, “B.EN.DE.R. 2.0: Basic ENvironment for DEveloping Robotic software: Application to educational purposes”, Proceedings of the 5th IEEE International Conference on Mechatronics, Málaga, April 14-17, 2009.

[16] N. Pavón, J. Ferruz, “BENDER 3.0, una Plataforma Robótica Remota para Aplicaciones Docentes: Aplicación a Programación Concurrente”, Revista Iberoamericana de Automática e Informática Industrial (RIAI). CEA-IFAC, Vol. 7, Núm. 1, Enero 2010, pp. 54-63.

[17] G. Metta, P. Fitzpatrick, L. Natale, “YARP: Yet Another Robotic Platform”, International Journal of Advanced Robotic Systems (ARS), vol. 3, no. 1, 2006, pp. 43–48.

[18] A. Ollero, G. Heredia, “Stability analysis of mobile robot path tracking”, Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS’95, vol. 3, Pittsburgh, PA , USA, August, 1995, pp. 461 – 466.