Memorias del XVI Congreso Latinoamericano de Control Automático, CLCA 2014
Octubre 14-17, 2014. Cancún, Quintana Roo, México

Behavior-Based Navigation System for a Service Robot with a Mechatronic Head ⋆

Marco Negrete * **Jesús Savage** * **Jesús Cruz** * **Jaime Márquez** *

* *Universidad Nacional Autónoma de México*

Abstract: In this paper a navigation system for a service robot based on a scheme of cooperative behaviors, which control a mobile base and a mechatronic head, is proposed. The main idea is to achieve a more cognitive control system through a strong interaction between base and head behaviors, perception and task planning. Some behaviors depend directly on the perception information and some others depend both on the task planning and perception, i.e. position control is improved by incorporating artificial intelligence techniques. A Kalman Filter is used for robot localization, nevertheless, since the filter is just a part of the behavior-based scheme, it is turned on and off conveniently according to the environment representation. The use of the mechatronic head improves significantly the navigation performance. Experimental results are presented to show the effectiveness of the scheme.

Keywords: Mobile robots, autonomous vehicles, behavior-based robotics, navigation, localization.

1. INTRODUCTION

In last years, research on navigation systems for mobile robots has been focused in achieving more cognitive systems. For example Jae-Han et al. (2007) propose the use of a semantic map from which a path is planned and the localization of the robot is performed with an external sensor network, which is not desirable for service robots. Rawlinson and Jarvis (2008) developed a system that allows the robot to reach a goal through topological instructions without the necessity of a previous map, nevertheless, since no information is stored, there is no learning. In Rogers et al. (2011) it is proposed to extract high-level characteristics of the objects in the environment in order to relate them with their semantic meaning. Weixing et al. (2012) propose a method to navigate by recognizing arrows and traffic signs.

Arambula and Padilla (2011) improves the obstacle avoidance by incorporating a genetic algorithm for calculating the optimal potential field constants. Other related works, such as Palmeira et al. (2012) and González-Sarabia and Alvarado (2012) develop control laws for position or path tracking, nevertheless, they take odometry as the position measurement and this is not desirable for service robots, or they don't present experimental results.

These works proposed several methods to enhance a navigation system, nevertheless, none tested a complete system. In this paper, a navigation system for a service robot, Justina, is proposed. In section 2, the service robot Justina, in which all algorithms were tested, is described. Section 3 describes the navigation system: a highly interactive system composed by a path planner, a perception module, a set of behaviors controlling both the mobile base and the mechatronic head, and a localization subsystem containing a world representation and a Kalman filter for estimation. In section 4 some results are presented and discussed and finally, in section 5 conclusions and future work are given.

⋆ This work was partly supported by PAPIIT-DGAPA UNAM under Grant IN-107609 and by CONACYT

2. THE SERVICE ROBOT JUSTINA

Justina is a service robot built at the Biorobotics Lab of the Engineering School of the National University of Mexico. This robot and its predecessors have been participating in the Robocup@Home league since 2006 performing several task like cleaning up a room, serving drinks and several other tasks that the human beings ask for. It is based on the ViRbot architecture for the operation of mobile robots (Savage et al., 1998). To accomplish all these tasks Justina needs a navigation system capable to take her from one point to another in a safely manner and that means, with a safe obstacle avoidance, an efficient path planning and with a localization that allows her to know where she is in every moment.

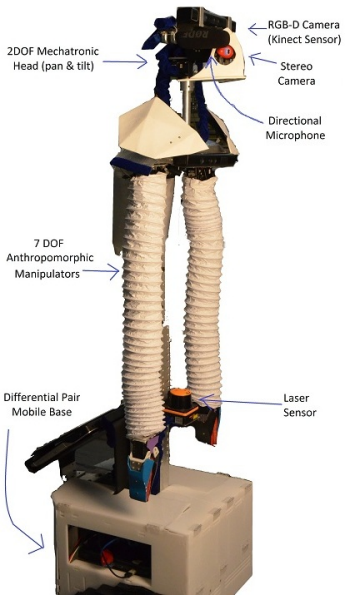

Fig. 1. The service robot Justina

For sensing the environment, Justina counts with several sensors: two laser range finders, a Kinect sensor, a stereo camera and a directional microphone. Also, Justina has

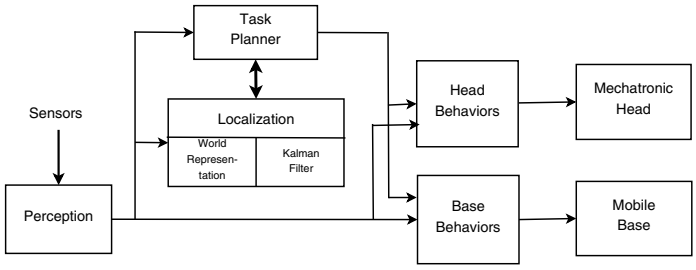

Fig. 2. The Navigation System

encoders in each motor (mobile base and its two arms). The navigation system uses the Kinect sensor, lasers and the mobile base encoders. Figure 1 shows the robot Justina and the position of its sensors and actuators.

2.1 Hardware

Mobile base. Justina has a differential base that allows motor control and encoder reading via RS232. This mobile base has a maximum speed of 2.0 [m/s] in each motor and the encoders give 8000 ticks per turn of the wheel.

Mechatronic head. The mechatronic head design is based on the corresponding movements of the human head: pan and tilt. It carries three sensors: the stereo camera, directional microphone and Kinect sensor. The pan and tilt movements allow to point these sensors in the best direction to obtain more useful readings of the environment. As will be described in section 3, the mechatronic head and the Kinect sensor it carries are used to improve the navigation system.

3. NAVIGATION SYSTEM

The proposed navigation system is composed of several subsystems performing different tasks in different levels of abstraction: a task planner, a set of behaviors controlling the mobile base and another set controlling the mechatronic head, a localization subsystem, which contains the world representation and a Kalman filter for estimating robot position, and a perception module, responsible for processing the raw sensor data. Figure 2 shows a block diagram of the different subsystems and its connections. Following subsections explain each subsystem and the way they interact between them.

3.1 Perception

Human perception is defined as the set of processes by which we organize, recognize and make sense of the stimulus we receive from the environment Sternberg and Sternberg (2012). Inspired on this definition we can say that perception of robot Justina is the set of signal processing algorithms used to extract, from raw sensor data, useful information to deduce aspects of the environment. Information generated by Justina's perception module includes object and face detection and recognition, landmark extraction, speech recognition, detection of obstacles and their positions, skeleton detection and propioception, that includes odometry and position estimations of the head and arms. Processing of data coming from the encoders is also part of the perception process, or more specifically, of

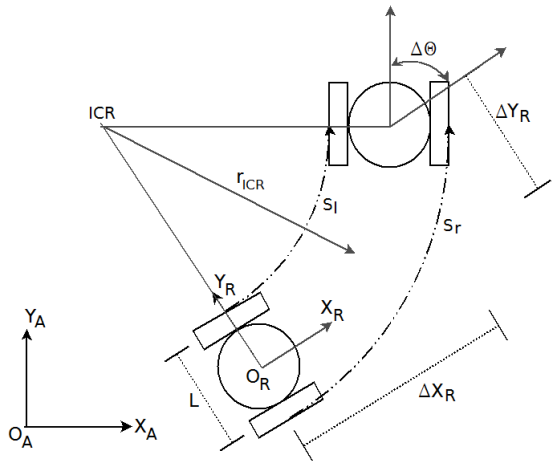

Fig. 3. Variables used for odometry

the proprioception process. The navigation system uses the subprocess of landmark extraction and obstacle detection.

3.1.0.1. Landmark Extraction Landmarks used for localization are line segments in the 3D space. They are extracted from laser readings and point clouds generated by the Kinect sensor, using an algorithm based on the work of Yan et al. (2012). It consist of two main steps: clustering the points according to its closeness and calculating the line equation by Principal Component Analysis (PCA).

3.1.0.2. Obstacle Detection An important issue for a safe navigation is the detection of obstacles with which the robot could crash. Perception module extracts obstacles from the laser readings using K-means algorithm. Detected obstacles are represented by the perception module as parallelepipeds with a XYZ position and its corresponding dimensions width, depth and height. Also, detected obstacles are use by the task planner to replan a path if there is risk of collision.

3.1.0.3. Odometry Given a differential pair mobile base of diameter L whose state is defined by three values $[x\, y\, \theta]^T$, if initial conditions $[x_0\, y_0\, \theta_0]^T$ are known, then the position can be calculated in an incremental form from the distances traveled by each wheel. Consider the scheme showed in figure 3 and the following variables and frames:

- S_l and S_r: Distances traveled by wheels left and right respectively.
- L: Robot diameter.
- r_{ICR}: Distance from the robot center to the instant center of rotation.
- O_A: Absolute frame.
- O_R. Robot frame. It moves with the robot and can be obtained by rotating O_A an angle of θ

Increments are calculated as

$$\Delta\theta = \frac{S_r - S_l}{L}$$
$$\Delta X_R = r_{ICR} \sin \Delta\theta$$
$$\Delta Y_R = r_{ICR} (1 - \cos \Delta\theta)$$

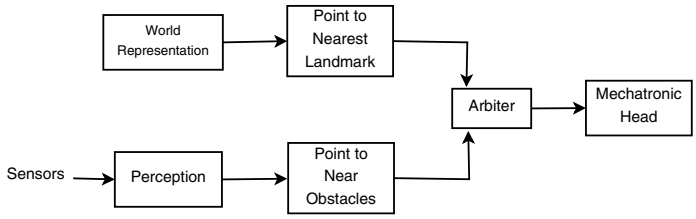

Fig. 4. Behaviors controlling the mechatronic head

where

$$r_{ICR} = \frac{S_l + S_r}{2\Delta\theta}$$

These increments are w.r.t. O_R and need to be transformed into increments w.r.t. O_A. Thus, robot's absolute position can be calculated as

$$X_{A_{k+1}} = X_{A_k} + \Delta X_R \cos\theta_k - \Delta Y_R \sin\theta_k \quad (1)$$

$$Y_{A_{k+1}} = Y_{A_k} + \Delta X_R \sin\theta_k + \Delta Y_R \cos\theta_k \quad (2)$$

$$\theta_{k+1} = \theta_k + \Delta\theta \quad (3)$$

3.2 Task Planner

Task planner calculates the optimal path to reach a goal point and this is achieved using the Dijkstra algorithm. This module interacts in a very strong way with the localization, perception and behavior modules. Paths are calculated according with the topological network contained in the world representation and nodes of this network are used in the Dijkstra algorithm. Nevertheless, if the perception module detects an obstacle with which the robot could crash, the stop-if-risk-of-collision behavior will stop the robot and a new path will be calculated. Also, paths are calculated considering the objects stored in the geometric map of the world representation and the obstacles detected by the perception module. Thus, path planning is made in interaction with the perception and world representation module.

3.3 Behaviors

Head Behaviors Mechatronic head is controlled by two behaviors mediated by an arbiter as shown in figure 4. The point-to-nearest-landmark behavior sets the desired pan θ_{Lm} and tilt ϕ_{Lm} according to:

$$e_p = [x_{Lm} - x_R \; y_{lm} - y_R \; z_{Lm} - z_H]^T = [e_x \, e_y \, e_z]^T \quad (4)$$

$$\theta_{Lm} = Atan2(e_y, e_x) \quad (5)$$

$$\phi_{Lm} = Atan2(e_z, \sqrt{e_x^2 + e_y^2}) \quad (6)$$

where $[x_R \; y_R \; \theta_R]^T$ is the current robot position and orientation, $[x_{Lm} \; y_{Lm} \; z_{Lm}]$ is the position of the nearest landmark to the robot and z_H is the height of the head position, which in Justina is 1.73 [m]. Nearest landmark is taken from the set of landmarks stored in the world representation. The point-to-near-obstacle behavior sets the desired pan θ_{no} and tilt ϕ_{no} according to:

$$e_p = [x_{no} - x_R \; y_{no} - y_R \; z_{no} - z_H]^T = [e_x \, e_y \, e_z]^T \quad (7)$$

$$\theta_{no} = Atan2(e_y, e_x) \quad (8)$$

$$\phi_{no} = Atan2(e_z, \sqrt{e_x^2 + e_y^2}) \quad (9)$$

where $[x_{no} \; y_{no} \; z_{no}]$ is the position of the nearest object to the robot from the set of obstacles detected by the perception module and robot position is the same as previous. This behavior allow an improvement in the potential field calculation.

Both behaviors are running concurrently. The arbiter assign priority to each behavior according to a time-sharing pattern. In this implementation, time is assigned 50-50 with intervals of three seconds.

Base Behaviors

3.3.2.1. Go-To-Goal-Point Behavior Mobile base is controlled by three behaviors. The first one is the go-to-goal-point behavior which directs the robot to a desired goal point. Based on the kinematic model

$$\dot{x} = \frac{v_l + v_r}{2}\cos\theta \quad (10)$$

$$\dot{y} = \frac{v_l + v_r}{2}\sin\theta \quad (11)$$

$$\dot{\theta} = \frac{v_r - v_l}{L} \quad (12)$$

where v_l and v_r are the linear speeds of left and right wheels respectively, taken as input signals, and the goal point $P_g = [X_g \; Y_g]^T$, speeds are calculated as

$$v_l = v_{max} e^{-\frac{e_a^2}{\alpha}} + \frac{D}{2}\omega_{max}\left(\frac{2}{1 + e^{-\frac{e_a}{\beta}}} - 1\right) \quad (13)$$

$$v_r = v_{max} e^{-\frac{e_a^2}{\alpha}} - \frac{D}{2}\omega_{max}\left(\frac{2}{1 + e^{-\frac{e_a}{\beta}}} - 1\right) \quad (14)$$

with $e_p = P_g - P = [X_g - X \quad Y_g - Y] = [e_x \; e_y]$ and $e_a = \text{Atan2}\,(e_y \; e_x) - \theta$ where $P = [X \, Y \, \theta]^T$ is the current robot position and orientation. v_{max}, ω_{max}, α and β are design constants greater than zero.

3.3.2.2. Avoid-Obstacles Behavior The second behavior uses potential fields (PF) to avoid obstacles. The equations used to calculate attractive and rejection forces are those proposed by Arambula and Padilla (2011):

$$F_{atr}(q) = -\xi\,(q - q_{atr})\,\frac{1}{|q - q_{atr}|} \quad (15)$$

$$F_{rep}(q) = \begin{cases} \eta\sqrt{\dfrac{1}{d} - \dfrac{1}{d_0}}\left(\dfrac{q - q_{obs}}{|q - q_{obs}|^3}\right) & \text{if } d \le d_0 \\ 0 & \text{if } d > d_0 \end{cases} \quad (16)$$

with the resultant force $F_{res} = F_{atr} + F_{rep}$, where q is the robot position, q_{atr}, the goal point and q_{obs} is the position of the object generating the rejection force. $\xi > 0$, $\eta > 0$ and $d_0 > 0$ are all design constants named, respectively, attraction constant, rejection constant and distance of influence. With the resultant force F_{res} a new goal point $P_g = P + F_{res}$ is set and the robot will try to reach this new goal point with the control law given by (13)-(14).

PFs are calculated from the laser reading and from the point cloud generated by the Kinect sensor. Kinect data are processed in parallel using an NVIDA Quadro GPU and the CUDA toolkit 5.0. The point-to-near-obstacle

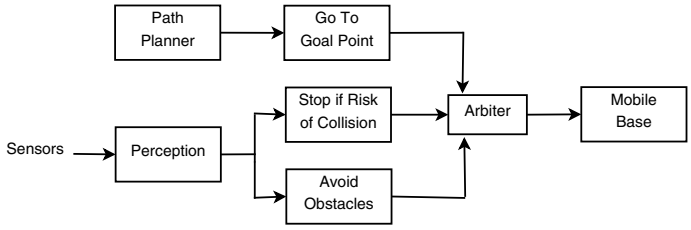

Fig. 5. Behaviors controlling the mobile base

behavior of the head helps to obtain a better view of the near obstacles. Figure 5 shows a block diagram of the behaviors controlling the mobile base.

3.3.2.3. Stop-if-Risk-of-Collision Behavior Third behavior stops the robot when a risk of collision is detected. Condition of risk is determined as follows: Let P_{no} be the position of the nearest obstacle detected by the perception module, P_R the current robot position and θ_R the current robot orientation. If

$$d_c = |e_p| < K_{dr} \qquad and$$
$$\theta_c = Atan2(e_{py}, e_{px}) - \theta_R < K_{\theta r}$$

with $e_p = P_{no} - P_R$, then there is a risk of collision. With the correct values of K_{dr} and $K_{\theta r}$ it is also possible to avoid local minima because, when a risk of collision is detected, a new path is calculated by the task planner.

3.4 Localization

World Representation Environment is represented symbolically through a topological network, a geometric map and a map of landmarks. Topological network contains a set of nodes, each of which has a name and a list of nodes with which is linked. The list of linked nodes is updated every time a new goal point is set in order to consider information about obstacles detected by perception module. This network is used for global path planning through Dijsktra algorithm. Path is taken as a global goal and each node is taken as a local goal which serves as goal point in the go-to-goal-point behavior.

Geometric map contains a simple representation of the objects in the environment as rectangles with an associated position and width-height values. Further, every object has some ontological information like the object type (table, chair, desk) and a property indicating whether the object is *good for localization* or not. Determining whether an object is *good for localization* or not is made statistically. *Good for localization* indicates that lines can be easily extracted. For example, a table or desk are objects that have well defined lines (borders) while an office chair does not. In the localization process, this information is used to determine the best angle toward which the head should be pointed in order to get the biggest confidence of landmarks. This process is described in subsection 3.4.2.

Regions are defined similarly to map objects: by an XY coordinates and width-depth-height values. For example, a region with a desk and a table in the surroundings could be good to perform the localization algorithm, while another one with only chairs and more complex furniture

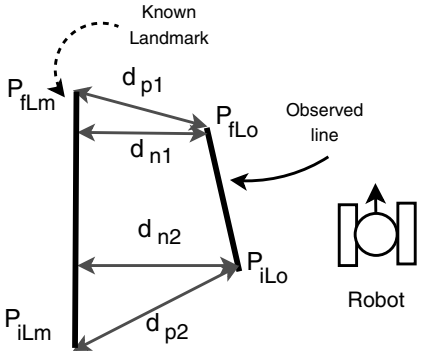

Fig. 6. Matching an observed and a known line

could be harder to extract reliable landmarks. These regions marked also as "good for localization" are used in the Kalman filter implementation described in subsection 3.4.2.

In a training stage, a map of landmarks is built. In this stage, the most reliable landmarks observed by the perception module are stored. Confidence of a landmark is determined by the number of times it is observed. Landmark map stores the coordinates of the end point of every landmark.

Kalman Filter

3.4.2.1. Position Measurement The Kalman filter is an estimator of the robot state. To perform estimations its necessary to "measure" some states of the robot. In this work, the three states $[X\,Y\,\theta]$ are considered measurable since their values can be obtained from the observed landmarks.

Consider figure 6. Lets call observed lines those given by the perception module and known lines, those represented in the landmark map. To triangulate the robot position each observed line is compared with every known line. To determine the map line that matches an observed line, lets define a pseudo-distance between an observed and a known line D_{LoLm} as the sum of the four distances d_{p1}, d_{p2}, d_{n1} and d_{n2} (see figure 6). Considering that a line segment is defined by its initial and final points and the known line has an equation of the form $A_{Lm}x + B_{Lm}y + C_{Lm} = 0$, the pseudo-distance D_{LoLm} is calculated as

$$D_{LoLm} = d_{p1} + d_{p2} + d_{n1} + d_{n2} \tag{17}$$

with

$$P_{fLo} = [x_{fLo}\ y_{fLo}]^T \qquad d_{p1} = |P_{fLm} - P_{fLo}|$$
$$P_{iLo} = [x_{iLo}\ y_{iLo}]^T \qquad d_{p2} = |P_{iLm} - P_{iLo}|$$
$$P_{fLm} = [x_{fLm}\ y_{fLm}]^T \qquad d_{n1} = \frac{|A_{Lm}x_{fLo} + B_{Lm}y_{fLo} + C_{Lm}|}{\sqrt{A_{Lm}^2 + B_{Lm}^2}}$$
$$P_{iLm} = [x_{iLm}\ y_{iLm}]^T \qquad d_{n2} = \frac{|A_{Lm}x_{iLo} + B_{Lm}y_{iLo} + C_{Lm}|}{\sqrt{A_{Lm}^2 + B_{Lm}^2}}$$

An observed line L_o is said to be matched with a known line L_m if the distance D_{LoLm} between L_o and L_m is less than the distance between L_o and any other map line.

It its assumed that all objects in the real environment (tables, walls, desks) are in an quasi-orthogonal position,

thus, it is expected that every observed line has an angle near to 0° or 90°. An observed line L_o, with equation $A_{Lo}x + B_{Lo}y + C_{Lo} = 0$, is considered vertical if $|B_{Lo}/A_{Lo}| < 0.6$ and horizontal if $|A_{Lo}/B_{Lo}| < 0.6$. If the observed line is not vertical nor horizontal, then it is considered as a wrong observation and is not taken into account. For each pair $L_m - L_o$ (map line, observed line) a position error $e_{pm} = [e_{xpm}\ e_{ypm}\ e_{\theta pm}]^T$ is calculated. If L_o is vertical, the following equations are used:

$$e_{xpm} = (x_{fLm} - x_{fLo} + x_{iLm} - x_{iLo})/2 \quad (18)$$
$$e_{ypm} = 0 \quad (19)$$
$$e_{\theta pm} = \pi/2 - \text{atan2}(A_{Lo}, -B_{Lo}) \quad (20)$$

And, if L_o is horizontal:

$$e_{xpm} = 0 \quad (21)$$
$$e_{ypm} = (y_{fLm} - y_{fLo} + y_{iLm} - y_{iLo})/2 \quad (22)$$
$$e_{\theta pm} = -\text{atan2}(A_{Lo}, -B_{Lo}) \quad (23)$$

Finally, an average error e_{av} is obtained from all known-observed line pairs and $P_{Rm} = P_R + e_{av} = Z$ is taken as the measured position.

3.4.2.2. Extended Kalman Filter From (10)-(12) a discrete model can be obtained using an approximation of the derivatives.

$$x_{k+1} = x_k + \Delta t \frac{v_l + v_r}{2} \cos\theta_k + \nu_1 \quad (24)$$
$$y_{k+1} = y_k + \Delta t \frac{v_l + v_r}{2} \sin\theta_k + \nu_2 \quad (25)$$
$$\theta_{k+1} = \theta_k + \Delta t \frac{v_r - v_i l}{L} + \nu_3 \quad (26)$$

where $X = [x\ y\ \theta]$ is the state vector, L the robot diameter, Δt the sampling step, v_l and v_r the left and right wheel speeds respectively and $\nu = [\nu_1\ \nu_2\ \nu_3]^T$ is gaussian noise without temporal correlation, zero mean and covariance matrix Q.

While Justina is moving, it is trying to localize itself, but it does not search landmarks all the time. It only performs the position estimation when it is in a "good-for-localization" (GFL) region (see section 3.4.1). This helps to reduce the uncertainty in the landmark extraction. Furthermore, the mechatronic head is always pointing to the nearest object marked as "good for localization" (GFL) thanks to the point-to-nearest-landmark behavior. Then, sometimes the measured position Z (see equation (3.4.2.1)) is the calculated from the observed lines and sometimes is the calculated from odometry. That is, the measured state Z is given by

$$Z = \begin{cases} X_{lines} & \text{if } \hat{X} \text{ is in a GFL region} \\ X_{odometry} & otherwise \end{cases} \quad (27)$$

The observation model for the Kalman filter, considering measurement noise, is given by $Z_k = X_k + \omega_k$ where ω_k is gaussian noise without temporal correlation, zero mean and covariance matrix R. The robot pose estimation by the Extended Kalman Filter consist of the following steps:

Fig. 7. Biorobotics Lab, where the navigation system was tested

Prediction: Based on the kinematic model and the observation model, the next state and the output are predicted considering that noise is equal to zero both in the transition state model and the observation model:

$$\hat{X}_{(k+1|k)} = \hat{F}(X_{(k|k)}, u_{(k)})$$
$$\hat{Z}_{(k+1|k)} = \hat{X}_{(k+1|k)}$$
$$P_{(k+1|k)} = J_{(k)} P_{(k|k)} J_{(k)}^T + Q$$

where P is the covariance matrix of the estimation error and J is the Jacobian of the function F w.r.t. X.

Update: Based on the observation error and the estimation error covariance, the next estimated state is calculated according to:

$$S_{(k+1)} = H_{(k+1)} P_{(k|k+1)} H_{(k+1)}^T + R$$
$$K_{(k+1)} = P_{(k+1|k)} H_{(k+1)}^T S_{(k+1)}^{-1}$$
$$\hat{X}_{(k+1|k+1)} = \hat{X}_{(k+1|k)} + K_{(k+1)} \left(Z_{(k+1)} - \hat{Z}_{(k+1|k)} \right)$$
$$P_{(k+1|k+1)} = \left(I - P_{(k+1)} H_{(k+1)} \right) P_{(k+1|k)}$$

where H is the Jacobian of the observation model, which in this case is the identity since it is considered that the whole robot state is measured. Matrix K is known as the Kalman gain.

In the prediction step, position estimation based on the kinematic model is not calculated by solving the difference equations (24)-(26), but taking as predicted position the value given by the odometry. Actually, odometry is per se an estimation based on the kinematic model.

In the update state, measurements $Z_{(k+1)}$ are those determined by landmarks or odometry, depending on whether the robot is in a good-for-localization region or not.

4. EXPERIMENTAL RESULTS

The proposed navigation system was tested in the Laboratory of Biorobotics at the National University of Mexico. It is an indoor environment with several types of furniture like tables, desks, chairs, shelves and other kind of objects like computers, instruments, printers, etc. Figure 7 shows a panoramic view of the lab.

To make a friendly system, a graphic interface was designed. All the system was implemented in C# language, except the line extraction from Kinect sensor, which was implemented in C++. Figure 8 shows the GUI. In the left

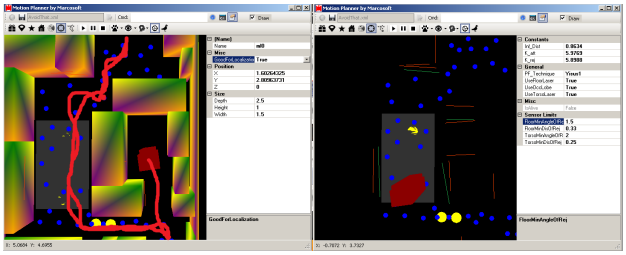

Fig. 8. The navigation system user interface

image, objects of the environment are drawn in green and yellow. Blue dots are the nodes of the topological network used for global path planning. Robot is drawn in red. In the right image, thin orange lines are the known landmarks and green lines are those extracted by perception module.

Constants used for the control law in the go-to-goal-point behavior are:

$$\omega_{max} = 1.5 \quad v_{max} = 0.6$$

$$\alpha = 1.26626 \quad \beta = 0.3659$$

The attraction and rejection constants, and distance of influence used for the avoid-obstacle behavior were, respectively:

$$\xi = 5.8988 \qquad \eta = 5.9769 \qquad d_0 = 0.8634$$

For the line extraction algorithm, constants K_1 and K_2 were, respectively, 0.05 and 0.06. Finally, constant matrices used in the Kalman Filter were

$$Q = \begin{bmatrix} 0.001 & 0 & 0 \\ 0 & 0.001 & 0 \\ 0 & 0 & 0.001 \end{bmatrix}$$

$$R = \begin{bmatrix} 0.02 & 0 & 0 \\ 0 & 0.02 & 0 \\ 0 & 0 & 0.04 \end{bmatrix}$$

with a time step $\Delta t = 0.1$ [s].

The navigation system showed a good performance since the robot could travel along the Biorobotics Lab without crashing any object and without getting lost. To test the proposed system, the robot was commanded to navigate from the lab's door to the end four times, navigating 40 m approx. Left side of figure 8 shows in red the path followed by the robot. During tests, people working in the lab were the unexpected obstacles and the robot avoided them successfully. Moreover, this navigation system was tested in several Robocup@Home tests in the Robocup Mexican Open, Mexico, and Robocup 2014, Brazil, showing a good performance in tests of first and second stage.

5. CONCLUSIONS

A navigation system based in a scheme of cooperation was proposed. In this work, two paradigms of robotics are used: hierarchical and hybrid. The proposed system has a more intelligent behavior which is achieved with the interaction of several levels of abstraction and different hardware modules. The system was successfully tested along with several Robocup test. Several previous works showed a good performance of obstacle avoidance, localization, mapping or path planning, nevertheless, many of them only showed on of these abilities separately, while this work showed a reliable complete navigation system tested experimentally. A position control law was tested and a Extended Kalman Filter was implemented successfully. While in previous works odometry is used as the position measurement, in this proposal a localization system is integrated, based not only on a common observer but integrating control and artificial intelligence techniques. It should be noted that, for all algorithms, experimental results are presented, implementing modern computational techniques as parallel processing, used for calculating the potential field forces. Future work is, on one side, the integration of pose estimation based on high-level characteristics of the environment, to determine, semantically, where the robot is located and, in the other side, integration of dynamic models not only of the robot base but also of the arms and head to achieve a more precise movements.

REFERENCES

Arambula, F. and Padilla, M. (2011). Autonomus robot navigation using adaptive potential fields. *Mathematical and Computer Modelling*, 40, 1141–1156.

González-Sarabia, A. and Alvarado, M. (2012). Navegación de un robot con ruedas en exteriores evadiendo obstáculos. In *Proceedings of the 15th Latin American Congress on Automatic Control CLCA*.

Jae-Han, P., Seung-Ho, B., and Moon-Hong, B. (2007). An intelligent navigation method for service robots in the smart environment. In *Control, Automation and Systems, 2007. ICCAS '07. International Conference on*, 494 –499. doi:10.1109/ICCAS.2007.4406960.

Palmeira, A., Magalhaes, R., Contente, C., and Ferreira, A. (2012). Comparison between classical pic and fuzzy pi+pd controller applied to a mobile robot. In *Proceedings of the 15th Latin American Congress on Automatic Control CLCA*.

Rawlinson, D. and Jarvis, R. (2008). Ways to tell robots where to go - directing autonomous robots using topological instructions. *Robotics Automation Magazine, IEEE*, 15(2), 27 –36. doi:10.1109/MRA.2008.921538.

Rogers, J., Trevor, A., Nieto-Granda, C., and Christensen, H.I. (2011). Simultaneous localization and mapping with learned object recognition and semantic data association. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, 1264 –1270. doi:10.1109/IROS.2011.6095152.

Savage, J., Billinghurst, M., and Holden, A. (1998). The virbot: a virtual reality robot driven with multimodal commands. *Expert Systems with Applications*, 15(3), 413–419.

Sternberg, R. and Sternberg, K. (2012). *Cognitive psychology*. Cengage Learning.

Weixing, M., Wei, P., and Lidong, X. (2012). Semantic-understand-based landmark navigation method of robots. In *Computer Science and Automation Engineering (CSAE), 2012 IEEE International Conference on*, volume 1, 760 –764. doi:10.1109/CSAE.2012.6272702.

Yan, R., Wu, J., Wang, W., Lim, S., Lee, J., and Han, C. (2012). Natural corners extraction algorithm in 2d unknown indoor environment with laser sensor. In *Control, Automation and Systems (ICCAS), 2012 12th International Conference on*, 983–987. IEEE.