Development of a control platform for the mobile robot Roomba using ROS and a Kinect sensor

Elvis Ruiz, Raúl Acuña, Novel Certad, Angel Terrones, María Eugenia Cabrera
Simón Bolivar University (USB)
Mechatronics Research Group
Miranda, Venezuela
racuna@usb.ve

***Abstract*—A control scheme for the mobile robot Roomba was developed using ROS (robot operating system), which allowed the control of the robot using velocity vector references. A Kinect sensor was incorporated into the system and also a framework which provides communication with processes in other instances, monitoring and remote control. The system is flexible enough to be replicated in the future in other Roomba robots. With this framework it is possible to control the movement of the robot using velocity vector references generated from the spatial references extracted from de environment using the Kinect sensor**

Keywords-ROS; Kinect; Roomba; Velocity Vectors; SLAM; Control.

I. INTRODUCTION

Mobile robots are characterized by allowing access to different topologies of work areas, their versatility can be utilized in applications such as: mineral exploration, planetary exploration missions, search and rescue, hazardous waste cleanup, surveillance, land reconnaissance, among others.

Current research seeks the improvement of robotic systems by increasing the efficiency of scheduling multiple tasks and compatibility and integration of new equipment, sensors and applications.

In this regard, we wish to develop a control platform for the mobile robot Roomba, using velocity vector references generated from the information provided by a Kinect sensor.

A. Mobile Robots

One of the most important attributes of Wheeled Mobile Robots (WMR), is its efficiency on smooth and firm surfaces [1]. Motion control of WMR, can be defined as the management of the actuators of the robot to follow a desired path [2]. According to the way the robot performs the tasks of perception, localization, motion planning and control the navigation can or cannot be reactive [3]. In reactive navigation, the robots automatically react to the changes that occur in the environment in a dynamic way, this method show good performance in initially unknown environments, but is inefficient in complex ones [4]. Potential fields are reactive in nature, easy to program and efficient in terms of computation time [5]. The gradient vector fields and potential fields can be interpreted as speed reference fields or force, where to each point of the discretized space is assigned a unique reference vector. Despite their associated problems the potential fields are a good choice for unknown environments with obstacles [6].

B. Roomba Robot Overview

The robot "Roomba 4400", IRobot Create ® (ICR) is a mobile instrumented robotic platform that can be described as an open interface and hardware robot. It has disc-shaped form, with dimensions of 0.330m in diameter, maximum height 0.085m at rest and weighing no more than 2.5kg unloaded. It also has at its disposal a cargo bay of about 1150 cm3. The IRobot Create® is equipped with a wide variety of parts and sensors that can become useful in navigation tasks, such as crash sensors, encoders with a resolution of 16 bits, LED indicators, buttons and two DC motors. The maximum speed of the robot is 0.5m/s in both forward and reverse, rotations can be performed up to 0.2m circular radius. With a mass less than 2.5Kg it can accelerate at max speed in less than 2s [7]. The wheel arrangement is a differential type.

C. General description of the Kinect sensor

It is a sensor device that complements the Xbox 360 game console from Microsoft. The sensor is composed by:

• An RGB camera (Resolution 640x480 VGA, bit color to 32-RGB-@ 30fps).
• A depth sensor consisting of an infrared emitter (projector) and an infrared monochrome CMOS sensor (320x240 resolution QVGA, bit to 16-levels deep -65.536 @ 30fps).
• An angular field of view of 57° horizontally and 43° vertically.

The junction of the optical systems is used to generate a three-dimensional image. To find the distance of each pixel in the depth image, the infrared camera detects a constellation of points emitted by the infrared emitter and the disparity is calculated for each pixel. The detection of depth can be computed without relying on the ambient light [8]. Moreover, because the IR camera and the RGB camera are displaced in the X axis, it is important to set the reference system by which the measurements will be taken. For this, the values of translation and rotation matrices provided by ROS generic in Kinect OpenNI package are used.

D. *Robot Operating System (ROS)*

Developed in 2007 by the Artificial Intelligence Laboratory at Stanford. ROS focuses on providing a communication infrastructure and services for programs and processes based on a host operating system [9]. ROS provides libraries and tools to help software developers in creating applications of robots. Abstraction is provided with hardware, drivers for various devices, displays, message passing, package management, support for multiple programming languages and other features. One of its distinctive features is the fact that it is completely "open source" under the BSD license style.

Operational bases in which ROS is implemented are: management nodes, posts, topics, client-server relationships and hierarchical organization of packages [9].

E. *Point Cloud Library*

It is framed as a project for 3D point cloud processing, and is under the BSD license [10]. Among the diverse content of the library three of them were selected for data processing of Kinect information: Point Type, Voxel-Grid and Pass-through Filter.

II. RELATED WORKS

The robots Turtlebot and Bilibot are differential wheeled platforms that integrate with Kinect sensor and ROS software. The Bilibot is a free software project that uses the ICR sensor as a mobile platform, the Kinect as a sensor and ROS as software platform; also it has a robotic arm. With this the robot can track people and manipulate objects. The Bilibot was developed by Garratt Gallagher at MIT (Massachusetts Institute of Technology) Fig. 9 [11]. The Turtlebot, is a similar project developed by Willow Garage, among its attributes it has the following features: keyboard or joystick teleoperation, tracking people through recognition using the Kinect, navigation and SLAM (Simultaneous Localization and Mapping) [12].

Moreover, ROS systems developed with direct applications in the ICR are mainly at Brown University with the package "Ros Brown Package" [13] and the University of Colorado with the package "prairiedog-ros-pkg [12].

The Mechatronics Group at the Simon Bolivar University has been developing angle control of a mobile robot moving at constant speed [14], which served as a bridge to develop control module for the speed control system velocity fields [15]. Additionally, a control based on speed reference vectors was developed, the magnitude and angle act as inputs for a kinematic model of the vehicle. All this was employed for coordinated navigation based on differential velocity fields in a simulation environment [16].

III. EXPERIMENTAL DESIGN

ROS was used as the basis of the Roomba-ROS-Kinect Integrated System, Fig. 1. First, a control platform for the Roomba robot was developed using velocity vector references taking in consideration the characteristics and limitations of the mobile robot. Secondly, the Kinect data acquisition was implemented, which had access to the information of the sensor locally (on the robot) or remotely in a different computer. There were two groups or types of data: the first is focused for general purposes and to check the sensor data service, and the second was added for navigation in a corridor of particular conditions. From the point cloud obtained from the Kinect a set of lines of approximation are created which serves as references for the control of the robot.

The following are the main modules in which the system is divided, Fig 2:

- Control of the ICR.
- Kinect Integration: Data Service for general navigation purposes.
- Exporting Information.

Moreover, in a classic control scheme the reference parameters, the controller and the feedback are defined as:

- Reference: Velocity vectors, in magnitude and angle. This is the main input of ICR control module.
- Controller: Control is used on two levels; the higher order is responsible for navigation using velocity vectors tracking and the lower order which controls the speed of the robot according to the velocity reference.
- The feedback: It consists in the generation of the velocity vectors. These can be created from data services of the Kinect, from odometry calculations of the ICR and from the instant reading of the encoders located in the wheels of the ICR.

Two main work environments are defined; both share the similarity of being rectangular areas. The first one is an unobstructed area 2.5 m wide by 3.5 m long, the second is a nonuniform walled corridor with a separation between them of about 2.8 m, see Fig 3.

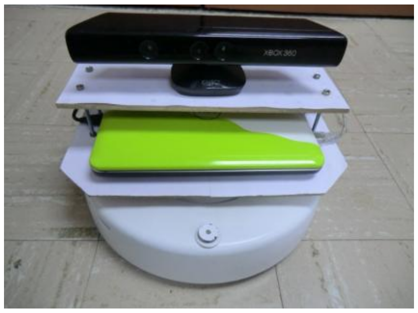


Figure 1. Roomba-ROS-Kinect Integrated System

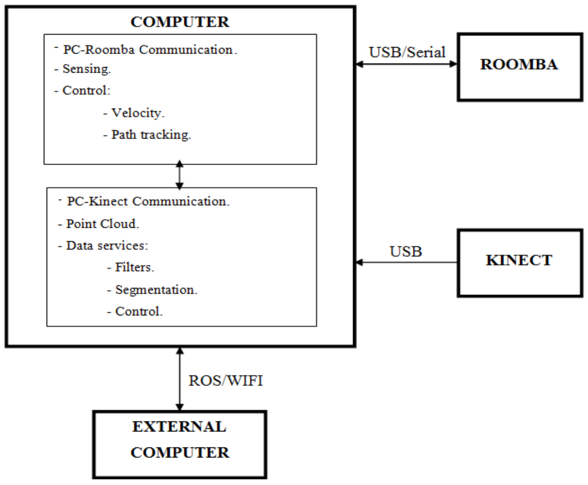


Figure 2. Global System Diagram

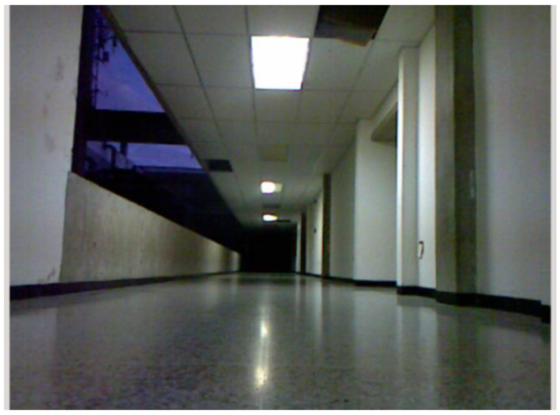


Figure 3. Second Work Environment

A. *Control*

The control loops are defined using as manipulated variables the module (V) and the angle (φ) of a velocity vector. Both references are followed with control loops tuned empirically by the method of Ziegler-Nichols, Fig 4 [17].

After achieving independent monitoring of the references, we present the problem of getting the right module while the angle reference is still not achieved, creating movement in the wrong direction. To avoid this problem, the module speed reference (Vr) is scaled with the cosine of the error angle (φe). Thus the moving platform advances with the projection of the velocity vector on its current direction. This allows the platform to go backwards when the reference is totally opposite to the direction of the robot [16]. The error value of the module of velocity (Ve) is expressed as follows:

$$V_e = V_r\cos(\varphi_e) - V_m \qquad (1)$$

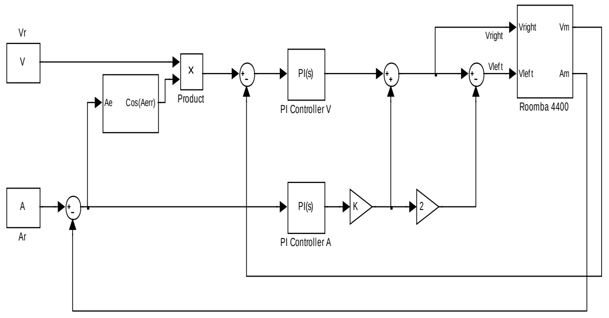


Figure 4. Second Work Environment

B. *Path following*

Using the control by reference vectors it is possible to follow a predefined trajectory. First, an approximation vector is defined (V_{aprox}), from the mobile robot to the nearest point (P_{prox}) to the predefined path. Then in Pprox a tangent vector is defined, the direction of this vector is from that point to the next consecutive point in the path forming the tangent vector (V_{tsn}) [18]. Thus, by adding V_{tsn} and V_{aprox} a reference vector (V_r) is obtained. Equation 2 can be decomposed into module and angle, to be then subjected to control. Note that the value of the modulus is directly proportional to the distance, so it is scaled and limited to the permitted speeds accepted by the mobile. For distance values greater tan 2m the module is escalated with a factor of 0.11, and for minor distances the factor is 0.08. This arbitrary values depends of the maximum acceleration and velocity of the robot and the processing time of the sensors, thus it's empirically defined.

$$V_r = V_{aprox} + V_{tan} \qquad (2)$$

C. *ROS configuration*

The *DiamondBack-Desktop-Full* of ROS was used in this work, which operates without problems in the operating system Ubuntu 10.10 (Maverick Meerkat). It makes use of the repository "Prairiedog-ros-pkg" at the University of Colorado and specifically the Irobot_Create_Rustic package. Below is a description of the nodes used in the ICR:

- Move the robot in a closed loop with a fixed speed.
- Move the robot through a computer keyboard.
- Send speed instructions using the client-server system.
- Control, given a vector of speed reference.
- Calculate reference vectors from a planned path and unobstructed access to the robot's position.

Regarding the Kinect sensor, the ROS packages OpenNI, Perception Addons and Turtlebot Pcl. The first is the basis for establishing communication between the Kinect and computer, the second to perform operations with the library of PCL in ROS and the third is taken as a platform to operate with another type of message, the LaserScan. With this ROS package framework is possible to develop and use new nodes or modify existing ones to generate a data service with the Kinect. The features of this framework are:

- Access the RGB camera, depth images and point cloud as a standard message type PointCloud2 in ROS.
- Filter the point cloud with Voxel-Grid (obtaining a less dense point cloud) or a Pass-through Filter and access an image plane.
- Perform a stratification of the point cloud, i.e., create a message with only one row of the matrix of 640x480. And to allow row selection to be carried out by a client-server pair.
- Perform a format conversion of the PointCloud2 message to LaserScan. This is characterized by a representation of the point cloud in a single plane (like a laser) and each point is represented by its radius and angle (r, φ).
- Make data segmentation of the LaserScan standard format.
- Approximation lines calculation from the point cloud of a surface.
- Velocity vector generation using the approximation lines.

Additionally, the graphical tool rviz is used, which allows visualization of the point cloud Kinect and other structures ROS messages, such as the LaserScan format.

D. Materials and equipment

- PC: DELL Inspiron mini Intel Atom CPU N270 @ 1.60 Ghz.
- OS: Ubuntu NetBook Edition 10.10 (Maverick). Kernel Linux 2.6.35-28-generic. GNOME 2.32.0.
- Software framework: ROS DiamondBack desktop full.
- Mobile Robot: Robot Roomba 4400 Irobot Create®.
- Kinect Sensor.

IV. PREPARE YOUR PAPER BEFORE STYLING

A. Roomba Robot Control

To carry out these tests the first workspace was selected, given that in this environment we can easily study the behavior of the velocity to a step input.

The PI control tests show a settling time to 95% of the reference signal, in about five seconds and a steady state error that tends to zero, for any input value, Fig. 5.

We thus have a system capable of providing control signals from module and angle reference (V, φ). With this system the path following ROS node was verified, for a planned trajectory free of obstacles.

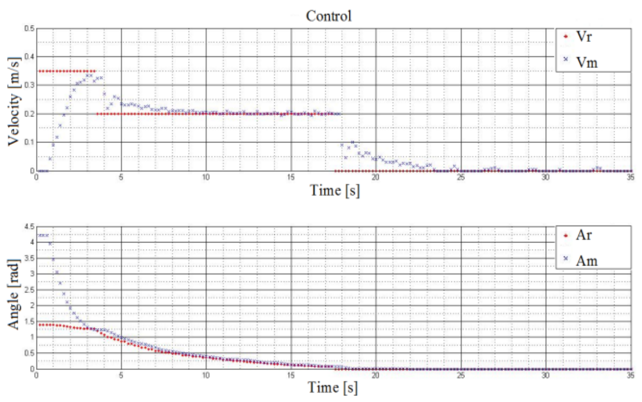
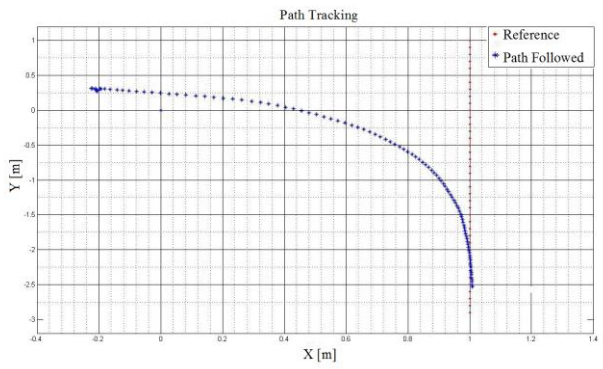
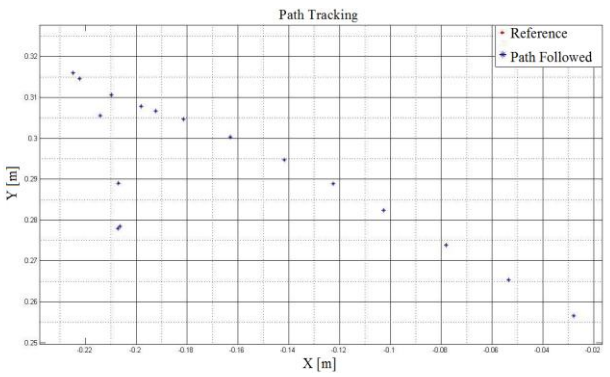


Figure 5. PI Velocity Control to different inputs



(a) Final route followed by the robot



(b) Detail of the beginning of the movement and the effect of the cosine of the angle error to scale the velocity magnitude.

Figure 6. Path following for a line parallel to the Y axis and initial position of the robot with an offset.

In the scenario in which the robot has to follow a parallel line to the Y axis starting from and offset position and the original orientation of the robot is opposite to the line, the robot behaves as expected, see Fig 6. In the lower graph Fig. 6a it is possible to appreciate the effect of using the cosine of the angle error to scale the magnitude of the velocity, which has an effect when the movement of the robot starts (because its heading is opposite to the line) [16]. Using this scaling, the robot first moves backwards to correct its heading and then it proceeds forward.

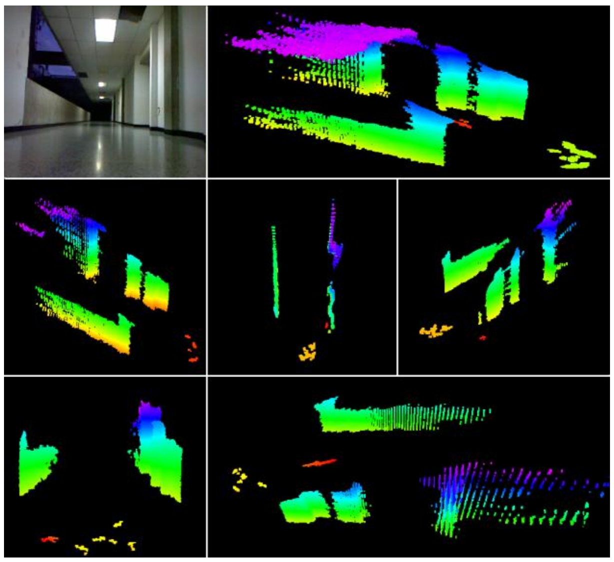


Figure 7. Fig. 7. Visualization of the Kinect sensor data using Rviz. View of the hall in different perspectives (frontal, plane and lateral).

B. *Kinect sensor*

A data service for acquisition and processing of Kinect information aimed at laying the groundwork for future work was developed. Additionally, it has a service designed to navigate in a corridor of known characteristics. In this sense we have the following results:

General Purpose Data Service: The tests were performed in the second work environment. The data obtained from the data service can be viewed through: graphic visualization tools of PCL, Rviz ROS environment as in Fig. 7, or by other programs that are compatible with the format in which information is stored, for example Matlab®.

Another way to visualize a plane is the transformation of the point cloud to a Laser type structure, the data handled from the Kinect can be interpreted as if it were a laser scanner. From a depth plane a line equidistant from the walls defining the passage was calculated, the result was a virtual line in the middle of the hall at equal distance from the walls, see Fig. 8.

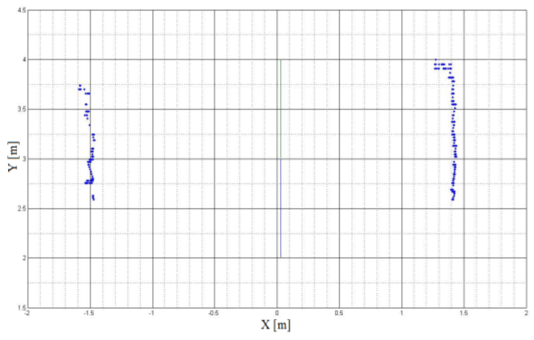


Figure 8. Creation of a virtual line in the middle of the hall.

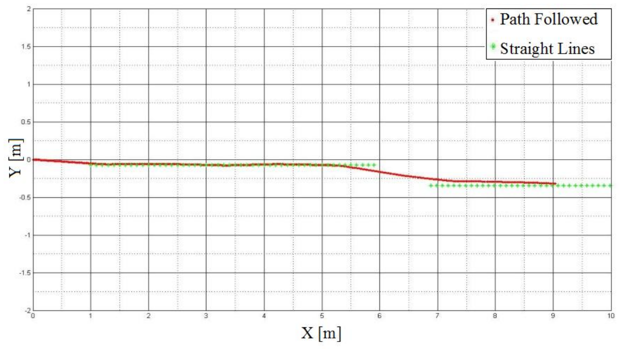


Figure 9. Hall trajectories generation and the path followed by the Roomba Robot.

From the picture we can see that the separation of the walls is about 2.8 m, this reflects that the measurements made by the Kinect sensor and therefore implemented by nodes are correct with reality. Now, the association is established between the ICR nodes and Kinect nodes, some are responsible for generating a line with the Kinect sensor data and the others transform it into a speed command for the mobile.

The test conducted was that from the depth plane a straight line, similar to the one in Fig. 8 is created. This line is the path reference for the Roomba Robot, using the control nodes the robot follows this path, once it reaches the end of the path another line is generated from a new set of data from the Kinect sensor, as seen in Fig 9.

Regarding the control signals, it can be seen in Fig. 10 that the angle that is sent as reference is between zero and the equivalent in radians to a complete turn of 2π radians, indicating an attempt to stay in the middle of the hallway.

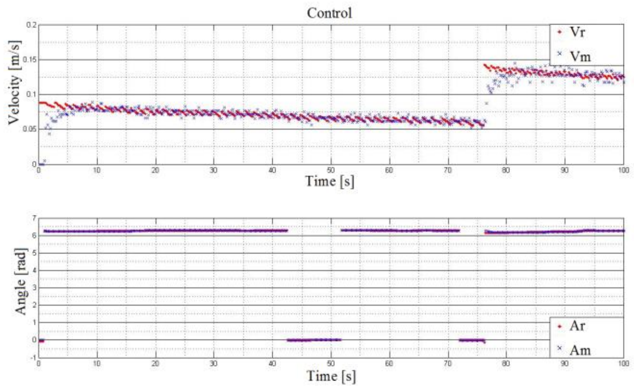


Figure 10. Velocity control in the hall.

Finally, as an extra validation of the applicability of the developed system, the acquired data using the Kinect sensor in the second workspace, were exported to Matlab® and an algorithm for detection of corners and edges was applied, this algorithm is the product of a related research in SLAM [19]. In Fig. 11 are shown the points obtained from the acquisition. The points marked with blue diamonds are detected corners and the red squares correspond to edges. The points marked in black are laser measurements, and the points marked in red are points that correspond to ruptures or discontinuities in the data (since the distance between them is very large), or refers to data which are outside the measurement range of the Kinect. The black solid line represents an approximation of the actual shape of the hall, made with hand measurements taken at the site. This shows that the Kinect data service based in ROS can be an input similar to a laser rangefinder for feature based EKF-SLAM applications.

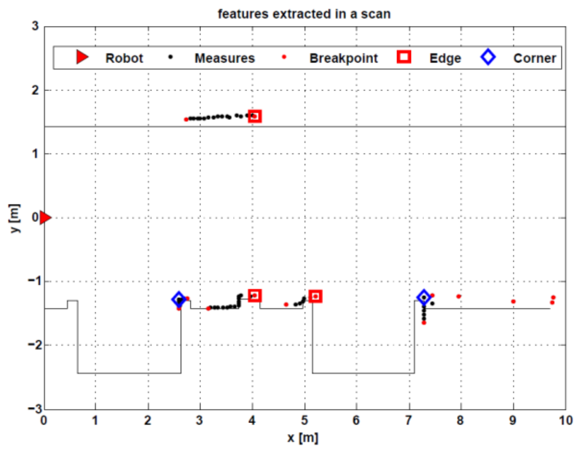


Figure 11. Features extract in a scan.

V. CONCLUSIONS

In this work the Kinect-Roomba-Ros framework was presented. It is clear from the results that this framework is an excellent low cost development platform for the test of autonomous navigation algorithms. Using this scheme it is possible to control a differential wheels robot like the Roomba using the Kinect sensor as input and ROS as control platform using autonomous navigation strategies such as the Velocity Vector Fields. Also, it is possible to use this low cost platform for SLAM using the Kinect as input for a reference extraction system.

REFERENCES

[1] P. Muir,C. Neuman. “Kinematic modeling of wheeled mobile robots”, Robotics Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, Tech. Rep. CMU-RI-TR-86-12, 1986.

[2] A. Luca, G. Oriolo y C. Samson. "Feedback control of a nonholonomic car-like robot". *Robot motion planning and control*, 1998, pp 171-253, doi: 10.1007/BFb0036073

[3] J. Putney. "Reactive navigation of an autonomous ground vehicle using dynamic expanding zones". MSc. Thesis, Virginia Tech. 2006.

[4] L. Wang, L. Yong y M. Ang. "Hybrid of Global Path Planning and Local Navigation implemented on a Mobile Robot in Indoor Environment". *Proceedings of the 2002 IEEE International Symposium on Intelligent Control*. Vancouver, Canada. October 2002, pp 821-826, doi: 10.1109/ISIC.2002.1157868

[5] W. Medina-Meléndez. "Generación dinámica de campos de velocidad usando visión artificial". MSc. Thesis, Simón Bolívar University, Caracas,Venezuela. 2007.

[6] R. Acuña. “Generación de campos de velocidad para entornos dinámicos empleando predicción de movimiento". MSc. Thesis. Simon Bolivar University, Caracas, Venezuela, 2013.

[7] S. Guy, D. Manocha, J. Snape, J. Van den Berg. “Smooth Coordination and Navigation for Multiple Differential-Drive Robots”. *12th International Symposium On Experimental Robotics*. University of North Carolina at Chapel Hill, USA, 2012.

[8] L. Aritz. " Sistema de localización y seguimiento de personas en interiores mediante camara PTZ basado en las tecnologías Kinect y Ubisense". MSc. Thesis. Basque Country University, Spain, 2011.

[9] E. Berger , K. Conley, J. Faust, T. Foote, B. Gerkey, J. Leibs, A. Ng, M. Quigley, R. Wheeler. "ROS: an open-source Robot Operating System". *ICRA Workshop on Open Source Software*, USA, 2009.

[10] Point Clouds. Official Web Site. http://www.pointclouds.org, 2010.

[11] The Bilibot Project. Official Web Site. http://www.bilibot.com, 2011.

[12] Robot Operating System. Official Web Site. http://www.ros.org, 2010.

[13] Crick C, Jay G, Jenkins O, Osentoski S. "Brown ROS Package: Reproducibility for Shared Experimentation and Learning from Demonstration", Brown University Providence, USA, 2010.

[14] W. Medina, L. Fermín, J. Cappelletto, C. Murrugarra, G. Fernandez, J. Grieco. "Vision-Based Dynamic Velocity Field Generation for Mobile Robots". Lecture Notes In Control And Information Sciences. Vol 360, pp. 69-79. United Kingdom, 2007.

[15] C. D´Arpino, W. Medina-Meléndez, L. Fermín, J. Guzmán, G. Fernández y J. Grieco. "Dynamic Velocity Field Angle Generation for Obstacles Avoidance in Mobile robots Using Hydrodynamics". *11th Ibero-American Conference on Artificial Intelligence*. Lisbon, Portugal. October 2008, pp 372-381, doi: 10.1007/978-3-540-88309-8_38

[16] P. Estévez, J. Cappelletto, F. Álvarez, R. Acuña y G. Fernández. "Coordinated navigation Using Dynamically Varying Velocity Fields". *9th International Conference on Artificial Intelligence and Soft Computing*. Zakopane, Poland. June 2008.

[17] K. Ogata. "Modern Control Engineering". 5th Edition. Prentice-Hall Pearson, 2010. ISBN: 978-0136156734

[18] W. Medina-Meléndez, L. Fermín, G. Fernéndez y J. Grieco. "Optical Flow Based Velocity Field Control". Modalidad: *9th International Conference on Artificial Intelligence and Soft Computing*. Zakopane, Poland. June 2008, pp 771-781, doi: 10.1007/978-3-540-69731-2_74

[19] N. Certad, R. Acuña, A. Terrones, D. Ralev, J. Cappelletto, J. Grieco. "Study and Improvements in Landmarks Extraction in 2D Range Images Based on an Adaptive Curvature Estimation". *Andean Region International Conference (ANDESCON) 2012 VI*. Cuenca, Ecuador. November 2012, pp 95-98, doi: 10.1109/Andescon.2012.31