ROS-based human leader and robot follower using a Pioneer 3-DX robot

Implementación de ROS en sistema de control para un seguidor de personas usando robot Pioneer 3-DX

M.A Molina[1], D.R Avendaño[2], L.E Solaque[3], N.F Velasco[4], C.A Pulido[5]

[1]UMNG, Bogotá, Colombia, mmolina2127@hotmail.com; [2]UMNG, Bogotá, Colombia, daniellpu9@outlook.com; [3] UMNG, Bogotá, Colombia, leonardo.solaque@unimilitar.edu.co; [4]UMNG, Bogotá, Colombia, lnelson.velasco@unimilitar.edu.co; [5]UMNG, Bogotá, Colombia, camilopulidorojas@gmail.com

Recibido: ene 15, 2015. **Aceptado:** mar 8, 2015. **Versión final:** jun 8, 2016

RESUMEN

Robot Operating System (ROS) proporciona algoritmos y servicios para el control de diferentes tipos de robots. ROS es una herramienta útil en muchos problemas en el campo de los robots autónomos, especialmente en aplicaciones como la cartografía, localización y navegación autónoma en un ambiente interior. La navegación autónoma en un entorno dinámico contempla muchos factores ambientales que afectan el proceso de navegación. El enfoque de este trabajo es desarrollar un sistema modular para llegar a un seguidor humano utilizando un robot PIONEER 3-DX y un sensor Carmine Prime Sense, con el apoyo de hilos por cada paso de procesamiento a través de paquetes desarrollados en ROS. Estas etapas se dividen en: búsqueda y ubicación del usuario a seguir, estimación de la dinámica del robot, control de velocidades y por último, un módulo de seguridad de evasión de obstáculos.

Palabras clave: Carmine, Pioneer, robótica, ROS, seguidor de personas, sistemas de control.

ABSTRACT

Robot operating system (ROS) provides algorithms and services to control different kind of robots. ROS is a useful tool in many issues in the field of autonomous robots, especially in applications as mapping, localization and autonomous navigation in an indoor environment. Autonomous navigation in a dynamic environment is not only challenging, but also uncovers many indoor environmental factors which affect the navigation process. The presented work describes how a ROS-Based control system is used with a Pioneer 3-DX robot for a human leader and robot follower system. The approach for this work is develop a modular system to reach a human follower using PIONEER 3-DX and Carmine Prime sense Sensor supported by threads for each processing steps through ROS packages, these stages are divided in: coordinate human acquisition, robot´s pose estimation, angular and linear speed control and obstacles security module.

Keywords: Carmine, control system, human follower, Pioneer, robotics, ROS

1. INTRODUCTION

Human's beings have always looked for ways to perform several activities easily, avoiding to make a hand labor or painstaking work. Robotics offers supports for problems like transportation and physical rehabilitation which humans want to solve in a cooperative way [1], these solutions have been inspired by nature and human morphology, even production plants have adapted this approach using manipulators working together [2], where cooperative robotics comes in response, providing methods to develop team work. [1; 3].

Nowadays, cooperative robotics is used to perform dirty, dangerous, hard and repetitive tasks which can be of risk to humans. Actually, these activities are focused on industrial, research and development use. In some cases, cooperative robotics applications do not relate only to manipulator robots, but also to mobile robots. Then, their objectives and missions involve maintain robots formation for achieving a goal following a path [4; 5]. Some applications as transporting objects in restricted spaces, exploration and map construction of unstructured terrain, security and surveillance operations, search and rescue missions or entertainment robots, require a special care to keep them inside a minimum position error margin., [6].

In recent years, the interest in cooperative robotics and design of robots to assist people in their daily tasks has increased significantly. The main problem in mobile robotics is the navigation of agents working cooperatively with humans, solving problems such as moving loads, delivery of construction materials, delivery of medicines in hospitals, mine detection, among others.

The guide robots are helpful, however there is not a large number of investigations which have given the opportunity to robots working as guides or support robots in environments with a lot of people. One of the most important examples of robots interacting with people is Rhino robot. This is a robot who worked at the Deutsches Museum Bonn, where he guide hundreds of visitors through the museum.

The interaction between robots and people could be understood as a model by several dogs that lead a flock of sheep, guiding them toward a goal. Dogs and sheep have at least one form of communication explicit. However, currently there is no explicit signal for the guidance or for the path control, which is given by implicitly based on natural reactions between persons and robots. Moreover, the real human behaviors are very difficult to estimate, it should take about patterns of behavior as far as possible.

The idea of the present work is to develop a cooperative system using a mobile follower robot interacting with human leader. The robot must maintain a formation according to the user, while they are moving through space. As a follower robot, the mobile robot Pioneer 3-DX of MobileRobots Company is used, it is equipped with Carmine motion sensor. All the programming algorithms are developed based in ROS open source framework.

2. ROS

Robot Operating System is an open-source framework dedicated to robot applications; his purpose is to offer to developers some libraries, graphical and command-line tools under C++, Python and Java programming languages, [7]. ROS provides advantages respect communications infrastructure between processes using programs also known as nodes and topics sharing information on data structure called messages. Each running instance of ROS can publish messages on the defined topic likewise receive information subscribing to desire topic, this kind of decentralization improves system performance and allows executing packages on different stations.

3. ROSARIA

MobileRobots Inc provides C++ library called ARIA for Linux and Windows to interact with Pioneer Robots. Includes utilities to use network sockets, built-in robot features like velocity and position control, pose, battery level or sonar, as seen in [8]. Within ROS packages is ROSARIA, an interface developed to get and set information to robot using library named above, implemented via RosAria node, [9; 10].

4. COORDINATE HUMAN ACQUISITION

Alternatives to acquire human leader coordinates with respect to mobile robot have been named in [11]. The approach on the present work to estimate the target position to be followed is to use a device inexpensive with direct line-of-sight and recognized by developers to program skeleton tracking applications.

According to the description above, Carmine 1.08 PrimeSense 3D Sensor is used. It is thin, light, portable, fast (30 fps), and allows 3D features extraction in the environment with computational affordable cost and low consumption (Max 2.25 Watt via USB); also provides an open-source framework called OpenNI, [12]. This is the largest 3D sensing development framework. It provides an interface with devices to get visual data, RGB images or depth map even a user tracking inside field of view through nodes interaction as components to create understandable information for developer, from acquisition of sensor to full body pose presentation or identifying predefined gestures.

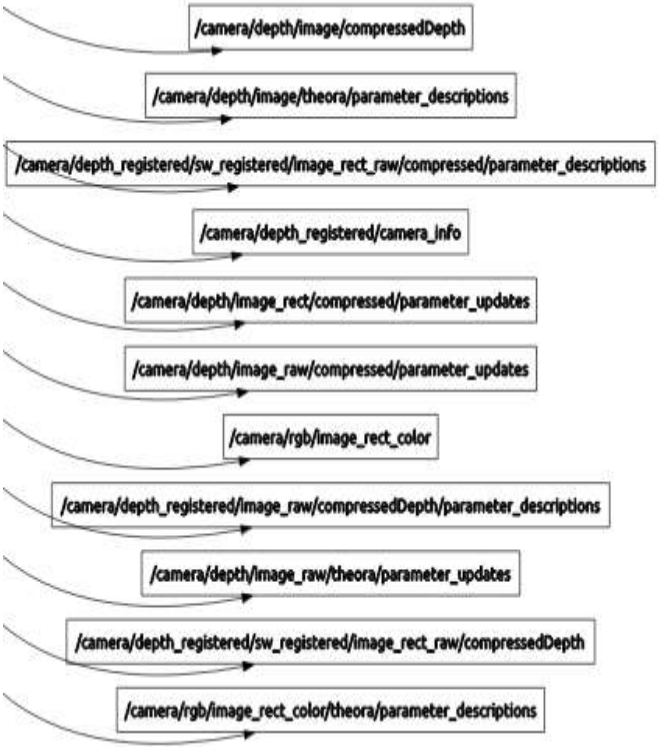

Figure 1. A segment of topics published when openni2_launch is executed

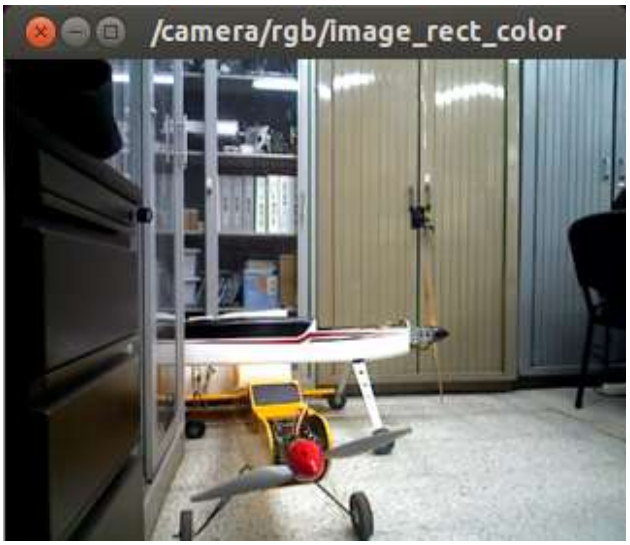

Figure 2. RGB Camera

ROS has the possibility to use this library with openni-camera and openni2-camera packages available on [9; 10]. To get data information from Carmine next command is used:

roslaunch openni2_launch openni2.launch

Then, the interface with device starts and publishes topics as shown in Figure 1 below:

This launch file provides RGB image, depth, camera parameters and so on; graphic above is displayed when *rqt_graph* command is executed.

To view the color image from RGB camera is necessary to run following command:

rosrun image_view image_view image:=/camera/rgb/image_rect_color

Figure 2 shows RGB visualizer launched with command above:

The overall objective is to locate a human in front of the robot for tracking as a reference position and distance to maintain, this becomes a problem of skeleton tracking on direct line of sight; get coordinate human and sent to nodes Control described below. To reach this purpose [13] is used.

NiTE2 is a middleware libraries for robust 3D computer vision, brings algorithms to hand and body tracking together with OpenNI. Carmine Sensor is placed on the PIONEER 3DX robot (See figure 3). For the present follower human application exists a restriction due to device field of view: 57.5, 45, 69 horizontal, vertical and diagonal degrees respectively. Operating rate is between 0.8 and 3.5 meters, [14].

Figure 3. Pioneer 3DX + Carmine 1.08 Sensor

The user tracker application is available for ROS [15], as a package which returns full body user´s joints transformations (See Figure 4) called openni2_tracker node.

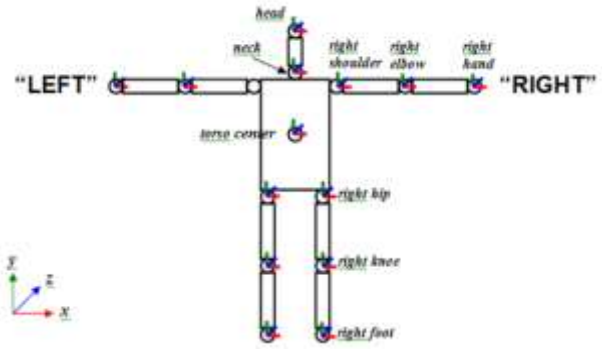

Figure 4. Joint definitions. Source: [16]

This source code is taken as reference to create a node to publish only torso´s coordinate through */torso_carmine/coordinates* topic with message type *geometry_msgs::Twist* which describes human leader position (X,Y,Z) relative to Pioneer 3DX (See figure 5).

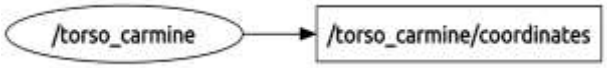

Figure 5. Torso_carmine node and topic published.

To run tracker, next command is used:

roslaunch torso_carmine tracker.launch

Torso_carmine starts with user identification (See Figure 6):

At this point, results necessary a calibration step to adjust the skeleton model to user according Skeleton Tracking chapter [16]. Figure 7, below, shows calibration pose and terminal message:

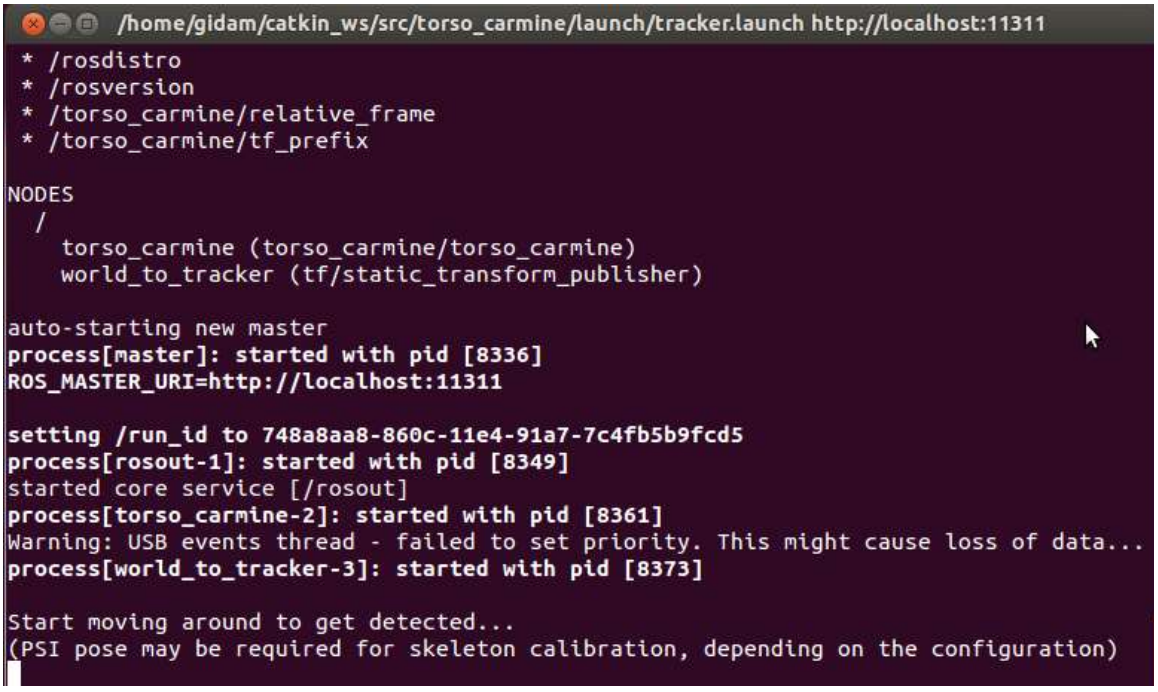

Figure 6. Terminal running Torso Carmine

Figure 7. Calibration Pose

Finally, after stay in front of robot few seconds, API returns positions of user´s torso, then publish and shows X, Y, Z coordinates on terminal (See Figure 8).

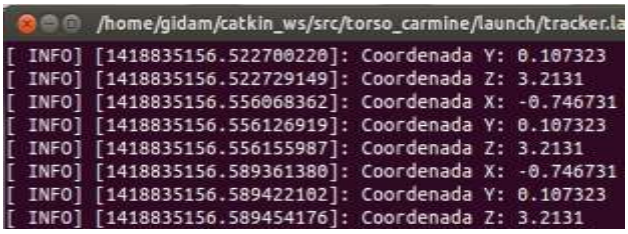

Figure 8. Tracking user´s torso

5. ROBOT´S POSE ESTIMATION

The Pioneer 3-DX robot works as a server in a client-server topology, this means, that the robot's microcontroller handles the low-level details of mobile robotics, as maintaining speed drive platform and direction, acquiring sensor readings, such as sonar or encoders. To complete the client-server architecture, the Pioneer 3-DX requires a connection to a PC, for this reason a HOST serial link is included in the system. The law controls are implemented in the computer and include obstacle avoidance, path planning, recognition features, location and navigation.

The client can be either an onboard piggy-back laptop, embedded PC or an off-board PC connected through radio modems or wireless serial Ethernet. In all cases, that client PC must connect to the internal HOST or User Control Panel SERIAL port in order to robot and software get to work. For the piggyback laptop or

embedded PC (See Figure 9), the serial connection is via a common "pass-through" serial cable.

Figure 9. Robot whit Piggyback laptop.

Rosaria provides a solution to the client-server connection; it has a port parameter which specifies the serial port to use. Also, it may contain a hostname and TCP port to connect remotely via TCP to connect using a Serial-Wifi bridge. The port parameter must be given in the command line when running the Rosaria rosrun node. The value is saved in the parameters of ROS and reused when Rosaria run.

To specify the port /dev/ttyUSB0 or COM1, as default for robot onboard computers the next line command is used:

rosrun rosaria RosAria _port:=/dev/ttyUSB0

To specify TCP connection to device with IP address 10.0.126.32 on port 8101 the next line command is used:

r*osrun rosaria RosAria _port:=10.0.126.32:8101*

RosAria provides several topics for publishing information received from the robot and devices by ARIA, and for receiving commands. The most important topics are Pose, Sonar, battery_voltage, and motors_state. They give all the necessary information to control and use the robot (See Figure 10). Also, RosAria just receives one important message, cmd_vel for desired velocities. The robot will achieve and maintain these set points.

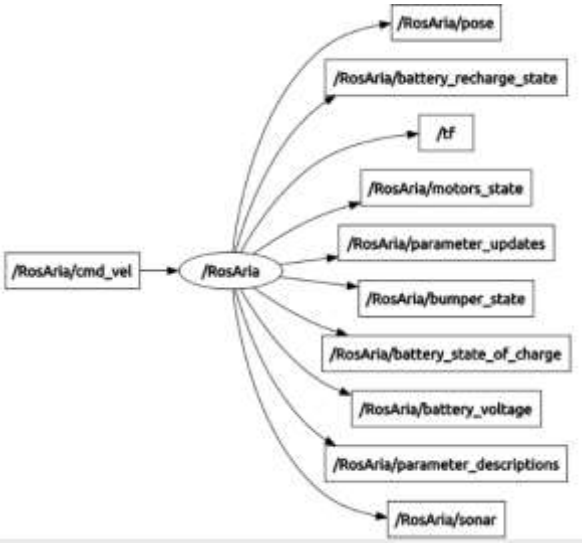

Figure 10. RosAria node and topics.

Finally, each topic has important information of the robot, for example, in this case RosAria publishes the current pose estimation received from the robot in /RosAria/pose. Also, each topic is published in a specific type of message, for example Pose has its data as a nav_msgs/Odometry message. This means that, the pose estimation is calculated by the robot firmware as ARIA based on wheel odometry, and if preset, the optional internal gyroscope.

The easiest way to read the Pose information is using rostopic command:

Rostopic echo /RosAria/pose

After that command, RosAria going to return information as the position, orientation, and velocities (angular and linear) of the robot (see Figure 11).

6. ANGULAR AND LINEAR SPEED CONTROLS.

In this application, a differential robot is used and it is considered like a solid mechanism, rigid and without flexible parts. The non-holonomic restrictions are important because the robot cannot move to the sides without an external perturbation force. On this kind of applications, the exact global position and robot orientation are the important variables to control; also to have a specific movement of the robot must to have a direct mathematic relation between the robot's inputs and the space-state variables *(X, Y, ϕ).* The control system must ensure that the robot moves with a specific linear velocity and rotates at determined angular velocity.

Rosaria brings an internal controller for the Pioneer robot. It controls the linear and angular velocity. The desired velocity of the robot is set via the /RosAria/cmd_vel topic. This desired velocity state is set in ARIA, which sends commands every robot cycle to

effect this state in the robot's embedded motion controller, which performs motor control to achieve then maintain the robot velocity automatically using previously configured acceleration parameters.

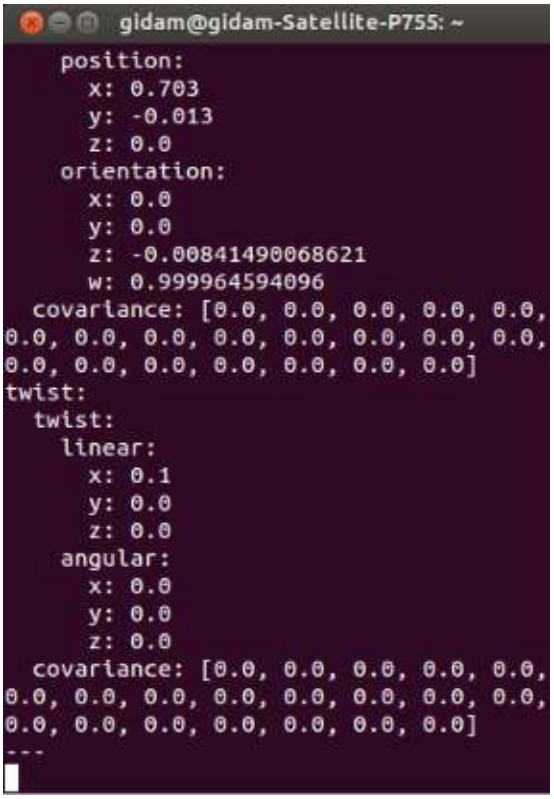

Figure 11. Pose information.

To achieve the principal objective of this paper is important to control the exact position and orientation of the robot in reference to the leader user. Also, is important to make the robot see all the time the user, and it must to be to a determinate distance from him. As conclusion of this idea, the problem is divided into two different controllers, the first one is an orientation controller and the second one would be a position controller. This is a stabilization problem that can be solved by output feedback. Each controller was developed by using Routh-Hurwitz criterion, which guarantees the stability of the close loop system.

The controller node receives a position topic from the carmine node, it topic gives the information about the exact user position (*X, Y, Z*) (See Figure 12). The orientation PID controller uses the *X* signal to guide the robot in front the user, with "0" reference (the user is in front of the robot) (See Figure 13), the position of the user in the *X* axis feedback loop and the angular velocity control signal. On the other hand, the position PID controller uses the *Z* signal to maintain the robot a specific distance. In this case, the desired distance is the system reference, the linear velocity is the control signal and the *Z* signal is the feedback loop.

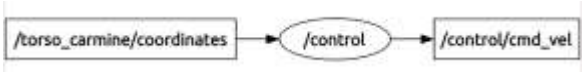

Figure 12. *Control node.*

Figure 13. Variable to control.

Finally, the control node, it prints on screen actual robot position (X, Y) and orientation. The controller has an error equal to zero and proves that it works correctly for the application. (See Figure 14)

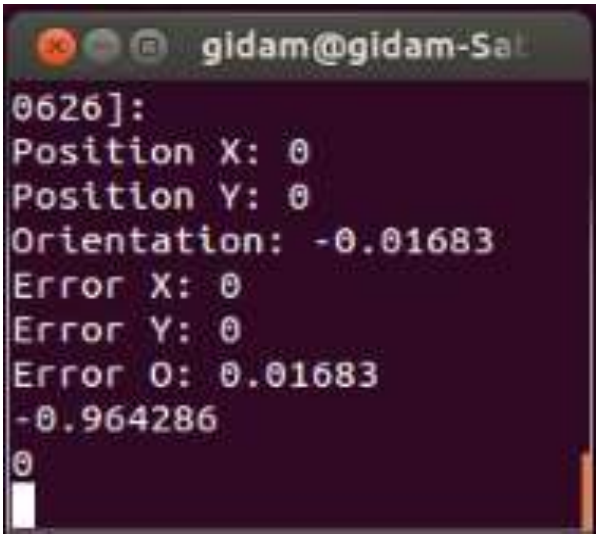

Figure 14. Control node error.

7. OBSTACLES SECURITY MODULE

In this paper, obstacles avoid module is developed looking to achieve that the robot could moves inside somewhere without collide something or somebody. Since, the objective of this application is introduce robotics inside space where the robots would interact with humans, is important to protect the human health, as well as objects that are in the environment where the robot is moving.

To develop the avoid obstacles control, the pioneer 3-

DX has a sonar array with eight transducers that provide object detection and range information for collision avoidance, features recognition, localization, and navigation. The sonar positions in all Pioneer 3 sonar arrays are fixed one on each side and six facing outward at 20-degree intervals. Together, sonar arrays provide 180 degrees of nearly seamless sensing for the platform.

To read the sonar array signal, the RosAria node has a singular topic that sends each 10 ms this information. */RosAria/sonar* message publishes sonar readings, is important to say, that the sonar arrays is turned on while there are subscribers on the topic. The RosAria node provides a two dimensional point cloud, and its coordinate frame matches the robot position. Each sonar gives the X and Y position on the plane of the object in front it.

The logic implemented is easy, the security node receives the velocities that the robot has; according of those velocities it calculates the minimum distance that the robot must have in order to move. If the actual distance is smaller than the minimum distance necessary, the security node control changes the lineal velocity to zero. After that, it calculates which side has more space (left or right) and it change the angular velocity making that the robot turn to that side. When the actual distance is bigger than the minimum distance, the security control returns the same velocity that the follower control sends.

8. RESULTS

The results obtained from tests over the platform with the Carmine and the piggy-back laptop, the Pioneer 3-DX robot is able to follow a person (See Figure 15. The tests were performed inside an indoor environment, to prove the obstacles avoidance control system.

The system performance results acceptable. The robot followed a user with a desired gap. If the user gets closer to the Pionner 3-DX, it goes back, maintaining the gap. This gap is the control system reference, measured in meters (m). These tests prove the control system performance. Also, the robot was able to avoid obstacles, allowing the robot to go behind a person without crashing with objects or walls.

After that test, person are introduced in the indoor environment. The reason for perform the test with people around the robot was to prove the Carmine tracking algorithm. This algorithm does not track just one person. For example, if somebody, crosses in front of the initial user, inside the Carmine's line of view, that person is tracked. Then, in many cases, the robot suddenly change the user and follow the latter one, or it lost and did not follow anybody, since the Carmine detected nobody.

9. CONCLUSIONS

ROS is a system based on algorithms that allowing controlling different kind of robots. Those algorithms, called nodes, give essential information about them, such as velocity, position, sensors, battery, and so on. Besides, the fast and easy communication between nodes minimizes the processing time, giving real-time capabilities to the system. The information about the robot's sensors is reliable and accurate, then, the platform's controller has an appropriate performance.

Also, Carmine's tracker algorithms and its ROS packages ease to get the skeleton distance data (x, y, z). Therefore, like the previous paragraph, saves costs and time for the Project development.

This kind of projects could be implemented in environments where the interaction between humans and robots will be huge, for example, in a mall, where the customer needs information about stores' location. The robot could guide the customer until it achieves the goal (the desired store). The robot will sense the customer position to not get away from him. This kind of situations could be consider as Artificial Intelligence, because the robot should has enough capabilities and performance to process all the sensors' signals, execute all the control algorithms with minimum error and a considerable speed. In addition, the robot should choose the best way to get to the desired location (faster way, shortest way), by itself.

10. ACKNOWLEDGEMENTS

This work is supported by the project ING-1537 namely "Estudio de técnicas de control cooperativo descentralizado en formación de robots móviles no holonómicos –Fase II", funded by the research vice rectory of the Nueva Granada Military University in Bogotá, Colombia

11. BIBLIOGRAPHY

[1] M. Uchiyama, "Multirobots and cooperative systems," in *Control Problems in Robotics and Automation*, by B. Siciliano and K. P. Valavanis, Eds.London, U.K.: Springer-Verlag, 1998.

[2] Tinós, R.; Terra, M.H. ; Ishihara, J.Y. "Motion And Force Control Of Cooperative Robotic Manipulators With Passive Joints". *IEEE Transactions on Control Systems Technology*, (Volume:14 , Issue: 4). July 2006.

[3] KWOW, et al. "PSOBased Cooperative Control of Multiple Mobile Robots in Parameter-Tuned Formations". *Proceed. 3rd Ann. IEEE Conf. on Automation Sc. and Eng. Scottsdale*, US, pp. 332 – 337, 2007.

[4] Y.Q. CHEN y Z. Wang,"FormationControl: A Review And A New Consideration". In IEEE/RSJ International Conference on Intelligent Robots and Systems, 3181-3186 pp. 2005.

[5] Diaz del Rio, F. ; Jiménez, G. ; Sevillano, J.L.; Vicente, S. ; Civit Balcells, A. "A Generalization Of Path Following For Mobile Robots". *IEEE International Conference on Robotics and Automation, 1999*. Proceedings. 1999 (Volume:1). 1999.

[6] F. Bravo, et al, "Switching control and modeling of mobile robots formation". Fac. of Eng., Dept. of Electronics, PUJ,. Bogotá, Colombia.

[7] J.M. O´aKane, *A Gentle Introduction to ROS. Columbia: Department of Computer Science and Engineering*, 2014

[8] MobileRobots. (2014, oct 12). Mobile Robots. [Online] Available: http://www .mobilerobots.com/Software/ARIA.aspx

[9] ROS. (2014, 12 11). ROS. [Online] Available: http://wiki.ros.org/ROSARIA

[10] ROS. (2014, 12 16). ROS - openni_camera. Available: https://goo.gl/q0SDjh

[11] J. Borenstain et al, Human Leader and Robot Follower Team: Correcting Leader´s Position from Follower´s Heading. Orlando, 2010.

[12] PrimeSense, Willow Garage, ASUS, & Open Perception. (2014, 12 16). OpenNI. [Online]. Available: http://openni.ru/index .html

[13] PrimeSense. (2014, 12 16). OpenNI - NITE 2. [Online]. Available: http://www.openni .ru/files/nite/index.html

[14] PrimeSense. (2014, 12 16). i3du. [Online]. Available: https://goo.gl/uNwI6b

[15] Guerin, K. (2014, 12 16). github. [Online]. Available:https://github.com/futureneer/openn i2-tracker/blob/master/src/tracker.cpp

[16] PrimeSense. (2010). Prime Sensor NITE 1.3 Algorithms notes.