Proceedings of the XVIII International Symposium on Dynamic Problems of Mechanics (DINAME 2019),
M.A. Savi, T.G. Ritto and W.M. Bessa (editors), ABCM, Buzios, RJ, Brazil, March 10th to 15th, 2019

# Project and Development of a Mecanum-wheeled Robot for Autonomous Navigation Tasks

**João Carlos Virgolino Soares[1], Gabriel Fischer Abati[1], Gustavo Henrique Duarte Lima[2], Carlos Luiz Machado de Souza Junior[3], Marco Antonio Meggiolaro[1]**

[1] Pontifícia Universidade Católica do Rio de Janeiro – Rua Marquês de São Vicente 255, Gávea, CEP 22451-900, Rio de Janeiro, RJ, Brasil, virgolinosoares@gmail.com, fischerabati@gmail.com, meggi@puc-rio.br

[2] Universidade Federal do Rio de Janeiro – Av. Pedro Calmon 550, Cidade Universitria, CEP 21941-901, Rio de Janeiro, RJ, Brasil, gughdl@gmail.com

[3] Instituto Militar de Engenharia – Praça Gen. Tibúrcio 80, Urca, CEP 22290-270, Rio de Janeiro, RJ, Brasil, carlos.m.s@icloud.com

*Abstract: This paper presents the project and development of a Mecanum-wheeled robot for autonomous navigation tasks. The robot is equipped with odometry and a laser sensor for range scans. A Monte Carlo Localization algorithm is used to estimate the pose of the robot in a global coordinate system and, simultaneously, a Grid Map is generated with the range scans to enable navigation. Once the map is constructed, the robot is able to perform autonomous navigation, given a desired goal. The Robot Operating System (ROS), an open-source framework for writing robot software, is used as middleware.*

***Keywords: Mecanum-wheeled Robot, Autonomous Navigation, Monte Carlo Localization, Simultaneous Localization and Mapping***

## INTRODUCTION

Research and development in autonomous mobile robotics are increasing each day considering its large potential applications such as autonomous vehicles, mobile industrial robots and home-service robots. In order to perform autonomous navigation, the robot needs a geometric representation of the environment, as a map. But usually a mobile robot has no prior knowledge of the environment and has to rely only on sensor information to create the map, such as odometry and range measurements. It also needs to estimate its own pose in this map. This problem is known as Simultaneous Localization and Mapping (SLAM).

Another important aspect of autonomous navigation is locomotion. Wheels are popular locomotion mechanisms for their efficiency and simplicity, and maneuverability is an important issue in many wheel types and configurations. A robot with four mecanum wheels driven by a separate motor has important capabilities, specially for industrial environments, as it is able to move in any direction whilst spinning around its vertical axis (Siegwart, Nourbakhsh and Scaramuzza, 2011). The mecanum wheel, shown in Fig. 1, was developed by Ilon (1975). It is composed of an active hub and a set of passive rollers attached around its circumference, usually positioned with a 45 degree angle (Siegwart, Nourbakhsh and Scaramuzza, 2011).

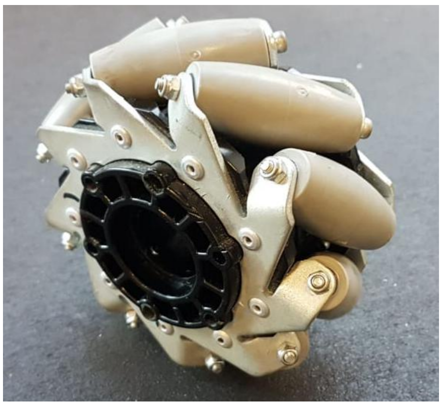

**Figure 1 – Mecanum Wheel**

Despite its advantages, mecanum wheels suffer from wheel slippage and vibration (Xie et al., 2015), which can considerably decrease the confiability in the odometry information and, consequently, compromise the SLAM solution. Futhermore, the range measurements from laser scanners are inherently uncertain. Therefore, this problem needs a probabilistic approach to handle these uncertainties. Introduced by Murphy (1999), the Rao-Blackwellized Particle Filter is a probabilistic approach for SLAM that uses the Monte Carlo Localization algorithm.

This paper presents the project and development of a mecanum-wheeled robot able to map unknown environments and perform autonomous navigation, using odometry information and range measurements from a Lidar sensor. The low-level system is controled with an Arduino board that communicates to a Raspberry pi board, running Ubuntu and the Robot Operating System (ROS) (Quigley et al., 2009) framework, responsible for the high-level tasks. The ROS-based implementation of the Rao-Blackwellized Particle Filter, called gMapping, is used to solve the SLAM problem and a ROS-based navigation system combined with a custom base controller is used for autonomous navigation.

## METHODOLOGY

### Robot Model

Fig. 3 shows the model of the robot with four mecanum wheels. Fig. 2 shows the world and robot coordinate frames. Eqs. (1) describe the forward kinematics of the robot (Roehrig et al., 2014). Eq. (3) transforms the coordinates in the world frame $(X_w, Y_w)$ to the robot frame $(X_r, Y_r)$. The variables $\dot{\varphi}_n$ represent the velocity of each wheel, which is obtained from the odometry measurements of the encoders.

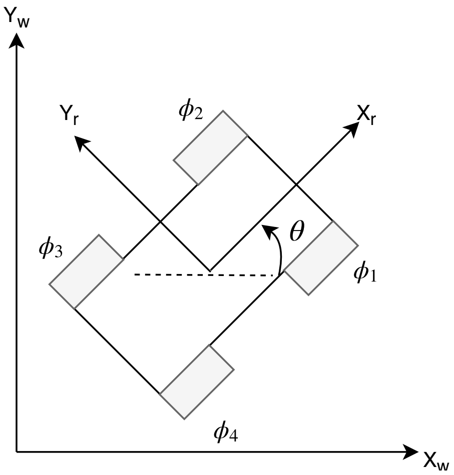


**Figure 2 – Coordinate frames**

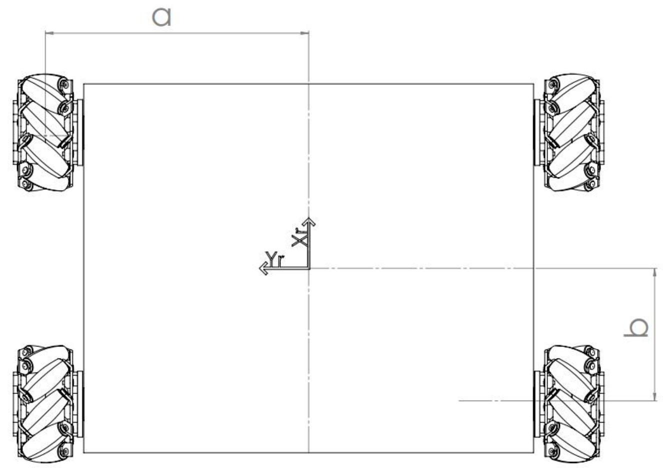


**Figure 3 – Robot model**

$$\begin{bmatrix} \dot{X}_r \\ \dot{Y}_r \\ \dot{\theta} \end{bmatrix} = J \begin{bmatrix} \dot{\varphi}_1 \\ \dot{\varphi}_2 \\ \dot{\varphi}_3 \\ \dot{\varphi}_4 \end{bmatrix} \tag{1}$$

where J is the Jacobian matrix, described by Eq. (2). The variable $r$ is the radius of the wheels.

$$J = \frac{r}{4} \begin{bmatrix} 1 & 1 & 1 & 1 \\ -1 & 1 & -1 & 1 \\ \frac{1}{a+b} & -\frac{1}{a+b} & -\frac{1}{a+b} & \frac{1}{a+b} \end{bmatrix} \tag{2}$$

$$\begin{bmatrix} \dot{X}_r \\ \dot{Y}_r \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{X}_w \\ \dot{Y}_w \\ \dot{\theta} \end{bmatrix} \tag{3}$$

### Robot Operating System

The Robot Operating System (ROS) is a framework for the development of robotics software, with a collection of libraries and tools specific for robotics applications. It is build in a graph-based structured, with several programs as the nodes, communicating with each other trough messages published on topics, as shown in Fig. 4, for example, where a Laser sensor driver sends the sensor information to a SLAM node. ROS is responsible for all high-level tasks in this work.

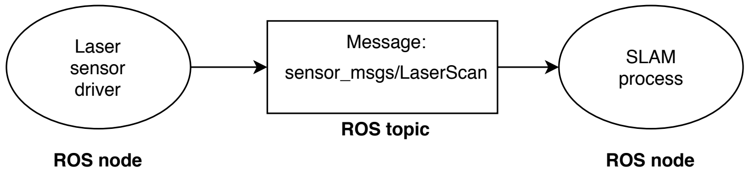


**Figure 4 – Example of a ROS system**

## Simultaneous Localization and Mapping

In the SLAM problem, the robot does not have knowledge of it own poses $x_t$ nor have access to a map m of the environment, having instead, only the sensor measurements $z_{1:t} = z_1, ..., z_t$ and odometry information $u_{1:t} = u_1, ..., u_t$. The problem consists on estimating a posterior belief $p(x_{1:t}, m|z_{1:t}, u_{1:t_1})$, in other words, estimating a belief of the pose of the robot and the map at each time step given its observations and odometry data (Grisetti, Stachniss and Burgard, 2007). This posterior belief is approximated by particles $x_t^i$ and each particle carry one individual map of the environment. The posterior estimation can be separated in two factors: the map estimation and the pose estimation, as stated in Eq. (4).

$$p(x_{1:t}, m|z_{1:t}, u_{1:t_1}) = p(m|x_{1:t}, z_{1:t})p(x_{1:t}|z_{1:t}, u_{1:t_1}) \tag{4}$$

The Rao-Blackwellized particle filter can be explained in four steps: Sampling, Importance Weighting, Resampling and Map Estimating. In Sampling step, the next generation of particles is sampled from the odometry motion model. The Importance Sampling method is, then, applied to assign an importance weight to each particle, incorporating the sensor measurement into the particle set. In the resampling step, particles with lower importance weight are discarded. After resampling, each particle is associated with a map estimation $p(m_i|x_{1:t}^i, z_{1:t})$. The ROS package *gmapping* is used in this work to perform the SLAM task, which is an implementation of the Rao-Blackwellized Particle Filter for SLAM.

## Navigation

Once the robot has a map of the environment it is necessary to localize the robot in the built map, in orther to perform a navigation task given a desired goal. The AMCL ROS package is used for this purpose. It consists in a Monte Carlo localization approach that uses particle filters to track the pose of the robot. Particle filters have advantages of being capable of dealing with non-Gaussian noise, represent complex multimodal beliefs and can be applied to global localization problems, such as the kidnapped robot problem (when the robot is suddenly carried to another location), instead of Kalman Filters that can only be applied to local localization problems.

The navigation system uses odometry, laser range information, the map and the estimated pose of the robot. A global planner generates the best trajectory to the goal and a local planner is used to avoid obstacles. The base controller receives the intructions from the local planner and translates into low level commands for the motors. In Fig. 5 is shown the proposed architeture of the system.

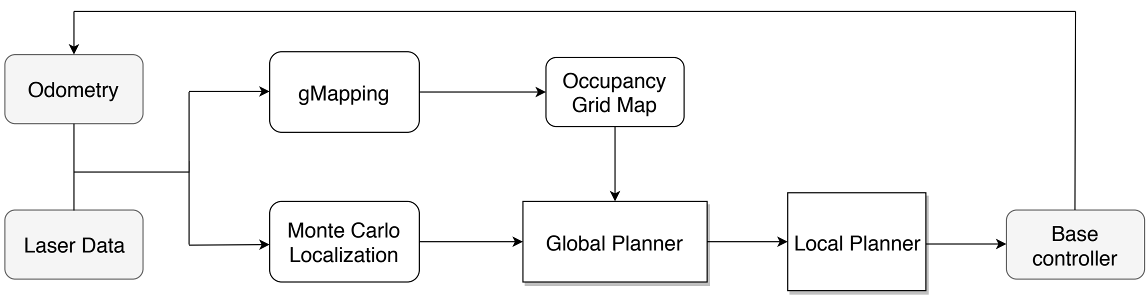


**Figure 5 – System pipeline**

## Base Controller

The base controller acts in two different situations. First, it receives commands to move the robot around and create a map of the environment. After the map is finished, the base controller receives high level commands from the planner. For the mapping step, the commands are sent from a cellphone with a bluetooth interface to the robot with an Arduino microcontroller through a bluetooth module, as shown in Fig. 6. The module translates the received commands and send them to the arduino through a serial communication. Once the commands are obtained by the arduino, it sets the desired speed and compares the command with the associated direction of translation or rotation. Finally, the arduino sends a Pulse Width Modulated (PWM) signal to establish the motor speed and two digital signals to define the motor direction of rotation. The arduino board and the raspberry pi board communicate through a USB-serial connection.

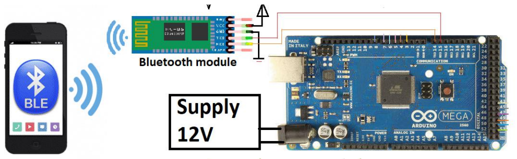


**Figure 6 – Communication system**

Fig. 7 shows a schematic diagram of the control system of the robot, with the locomotion system, the arduino board and motor controls. The system is powered by a 12V battery.

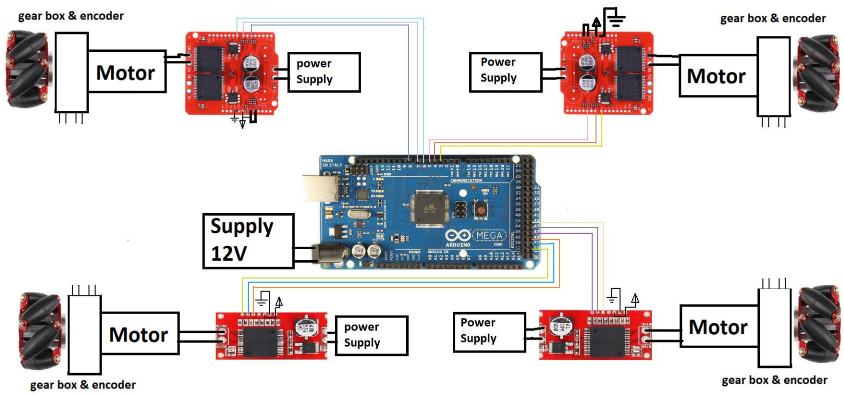


**Figure 7 – Control System**

Fig. 8 shows the achievable movements by the robot, given the individual wheel movements.

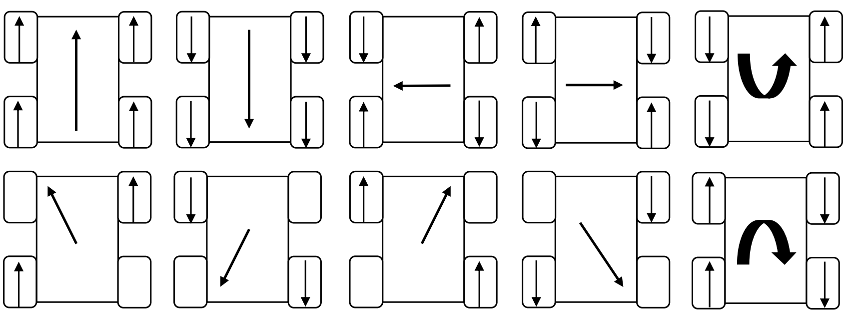

**Figure 8 – Robot Movements**

### Experimental Platform

Fig. 9 shows the robot specially built for this work, composed of 4 DC motors with enconders, 4 mecanum wheels, a Hokuyo URG-04LX-UG01 Laser Range finder, an Arduino board, and a Raspberry Pi 3 board with Ubuntu and ROS. The Laser Range Finder has a detection area of 240 degrees and a maximum measurement distance of 4 meters, which is sufficient for indoor applications.

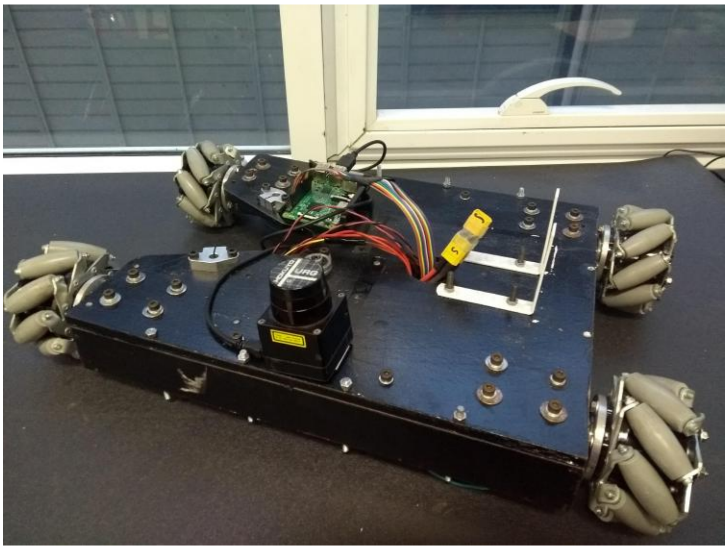

**Figure 9 – Assembled Robot**

## EXPERIMENTS AND RESULTS

Experiments are performed in order to evaluate if the robot is able to build a map and localize itself in this map, even with its high wheel slippage. All the experiments are conducted in the Robotics Laboratory from the Pontifical Catholic University of Rio de Janeiro. First, the robot is driven performing SLAM. Figs. 10 and 11 show the map being created by the robot with its trajectory in the map. Fig. 12 shows the map completely built. The mapping task is successful, as the map does not show misalignments and inconsistencies.

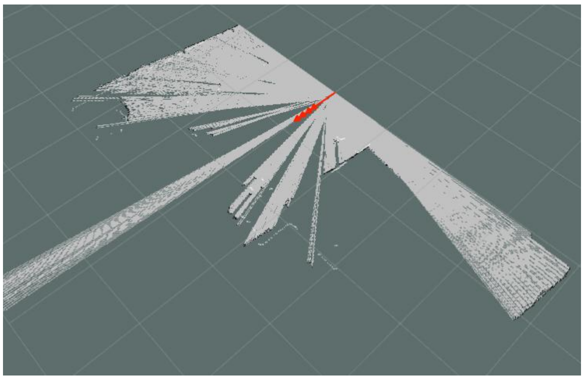

**Figure 10 – Initial Map**

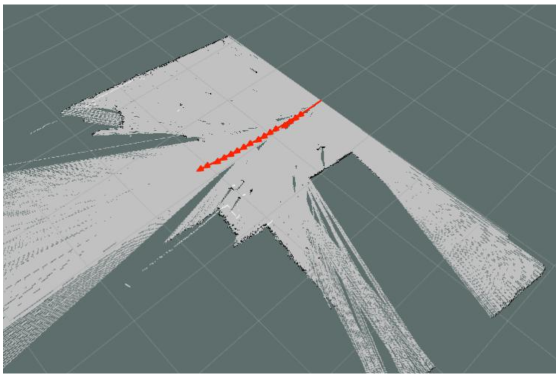

**Figure 11 – Robot Mapping**

After the map is constructed, the robot is able to localize itself in this map through the Monte Carlo localization system and to navigate. Fig. 15 shows the particles of initial pose estimation of the robot in the created map by the Monte Carlo localization algorithm. The particles are highly diffuse, showing that there is a high uncertainty about the initial pose of the robot. As the robot continues to move, it obtains more information about the environment through the laser sensor, matching those informations with the map. This is used to refine the initial pose estimation and the particles become more agglomerated, as shown in Figs. 14 through 16, which means that the system is more certain about the pose of the robot.

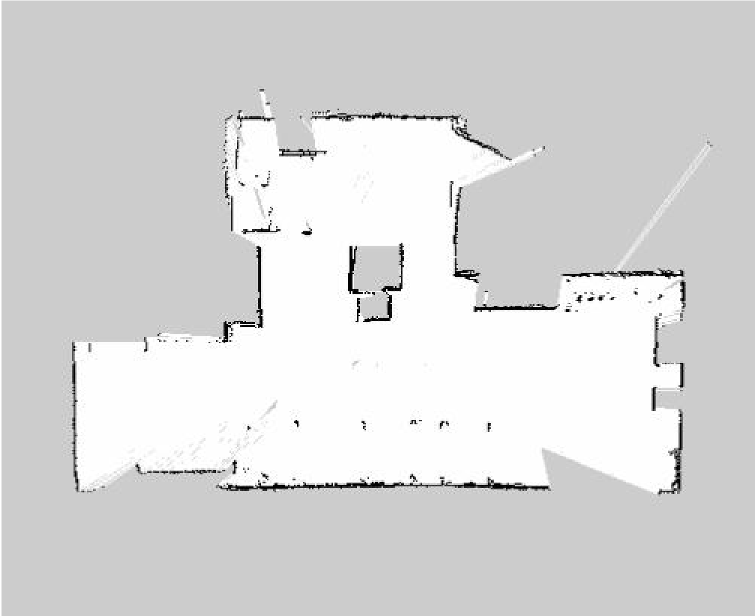

**Figure 12 – Final Map**

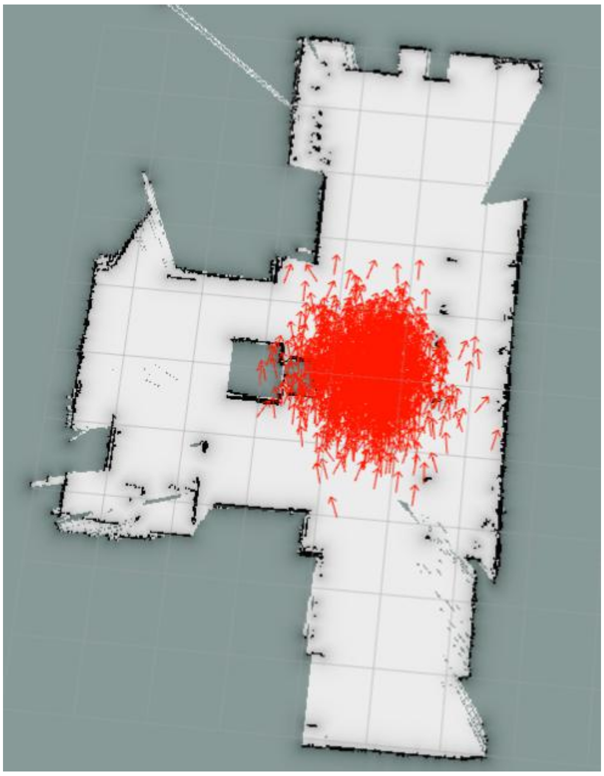

**Figure 13 – Initial Localization**

**Figure 14 – Improved Localization**

**Figure 15 – Improved Localization**

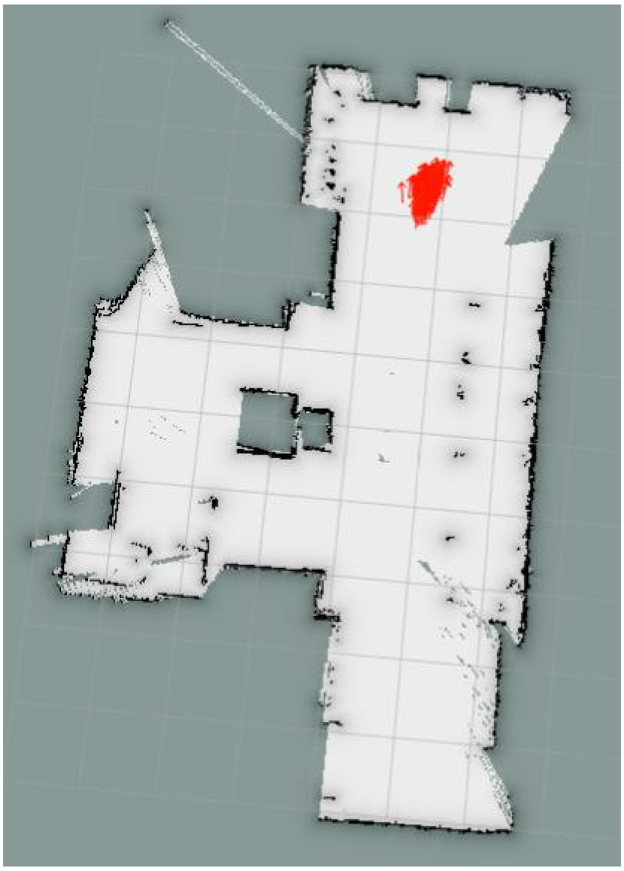

**Figure 16 – Improved Localization**

Fig. 17 shows the costmap, a map created and used by the planner that carries information about where the robot should navigate and should avoid.

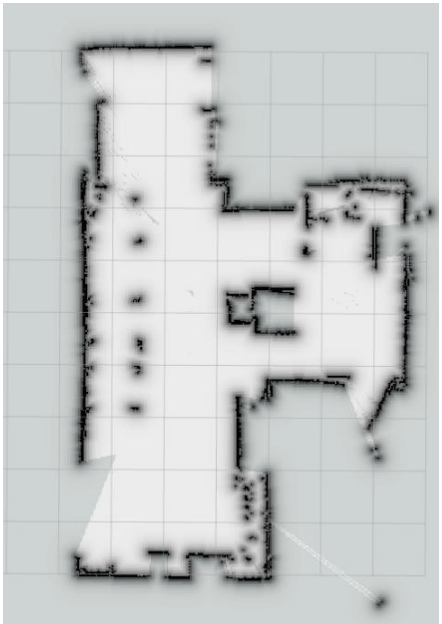

**Figure 17 – Cost Map**

## CONCLUSION

This paper presented the project and development of a mecanum-wheeled robot able to perform autonomous navigation tasks. The robot was built with 4 DC motors with attached wheel encoders and an arduino board. A low-level base controller was developed together with a bluetooth communication system to control the robot during the mapping process. A probabilistic approach was used to assure the robustness of the system against the uncertainties of motion and sensor measurements. The system was able to create consistent maps, being capable of localizing itself in this map and able to navigate through the environment. The mecanum-wheel configuration allowed the robot to reach any space in the room, which assures a fast and robust map building. This robot is highly versatile and can be used in several scenarios, such as an office, civil construction, surveillance and for industrial tasks. Future works include adding an IMU sensor into the system to improve the pose estimation. Furthermore, a future goal is to perform autonomous mapping with exploration.

## ACKNOWLEDGMENTS

The authors of this work would like to thank CNPq for the financial support.

## REFERENCES

Grisetti, G., Stachniss, C., and Burgard, W., 2007, "Improved techniques for grid mapping with rao-blackwellized particle filters". IEEE transactions on Robotics, Vol. 23, No. 1, pp. 34–46.

Ilon, B. E., 1975, "Wheels for a course stable self-propelling vehicle movable in any desired direction on the ground or some other base", US Patent 3876255.

Roehrig, C., Heller, A., Hess, D., and Kuenemund, F., 2014, "Global localization and position tracking of automatic guided vehicles using passive RFID technology", Proceedings of 41st International Symposium on Robotics, pp. 1–8.

Murphy, K., 1999, "Bayesian map learning in dynamic environments", Proceedings of the Conference on Neural Information Processing Systems (NIPS), Denver, CO, USA, pp. 1015–1021.

Quigley, M., Gerkey, B., Conley, K., Faust, J., Foote, T., Leibs, J., Berger, E., Wheeler, R., and Ng, A., 2009, "ROS: an open-source Robot Operating System", Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Workshop on Open Source Robotics.

Siegwart, R., Nourbakhsh, I., and Scaramuzza, D., 2011, "Introduction to autonomous mobile robots", MIT press.

Thrun, S., Burgard, W., and Fox, D., 2005, "Probabilistic Robotics", MIT press.

Xie, L., Scheifele, C., Xu, W., and Stol, K. A., 2015, "Heavy-duty omni-directional Mecanum-wheeled robot for autonomous navigation: System development and simulation realization", Proceedings of IEEE International Conference on Mechatronics (ICM), pp. 256–261.

## RESPONSIBILITY NOTICE

The authors are the only responsible for the printed material included in this paper.