3rd International Conference on Automation, Control, Engineering and Computer Science (ACECS'16)

Proceedings of Engineering & Technology (PET)
pp. 468-474

Effective Localisation Method For a Mobile Robot and Optimization Procedure Using The Levenberg-Marquardt Algorithm

Noura Ayadi and Nabil Derbel
Electrical Engineering Department
National School of Engineers of Sfax
Sfax, Tunisia
Email: noura.ayadi85@gmail.com

Nicolas Morette, Cyril Novales
and Gérard Poisson
PRISME Laboratory, 63, av. Lattre de Tassigny, F-18020
Bourges cedex, France

***Abstract*—In recent years, autonomous navigation for mobile robots has been considered a highly active research field. It depends essentially on the perception, localization, cognition and motion control. Within this context, we are interested to apply a localization approach for a wheeled mobile robot in a known environment. The robot's model and map parameters are initially analyzed and defined. The localization algorithm is developed based on the robot's model and data obtained from exteroceptive sensors. A step of optimisation using the Levenberg Marquardt Algorithm is then followed. Thereafter, simulation results are described to show performances of the proposed algorithm.**

I. INTRODUCTION

Relatively to the artificial intelligence, cartography is a highly crucial step for localization and autonomous navigation of a mobile robot. Starting from an initial position, a robot relies on its exteroceptive sensors to build a map for the navigation environnement. The robot localizes itself and learns to react in function of existing constraints. This paper presents a combination of different tools in order to create a complete platform to develop robotic applications. We start essentially with modeling the navigation environment, then the localization method and finally the experimental validation on an existing mobile robot.

Once the map is fixed, the control task is presented. Localization and tracking the robot during its navigation is primordial in order to ensure an autonomous navigation. The localization purpose is to know the exact position of the robot to navigate without hitting obstacles and detect the goal location to reach [8]. In recent years, researchers have focused on the localization as it has been considered an important process to know the exact position of the robot during its navigation. They have been based on probabilistic methods that major ones are the Kalman Filter localization, the Markov localization and the particle filtering algorithm also known as the Monte Carlo algorithm.

On the one hand, the Extended Kalman Filter combines results from dead reckoning with extracted data from exteroceptive sensors [2]. It also assumes that the probability distribution of both the robot configuration and extracted data from sensors are continuous and Gaussian. Subsequently, only the mean value and variance of the Gaussian function are needed to be updated, therefore, the computational cost is very low [10].

On the other hand, the Markov method divides the configuration space into cells. For a mobile robot moving on a plane, the configuration space is $3D$ dimensional (x, y, θ) and each cell contains the probability of the robot to be in that cell. The probability distribution of the sensors model is also discrete and during action and perception, all the cells are updated [10] [12].

The Monte Carlo Localization (MCL) is derived from the Markov method. Comparing to these previous algorithms, the MCL is easier to implement and ensures higher accuracy. In practice, it shows empirical results by an order of magnitude of memory and computation requirements [4].

The rest of this work is organized as follows. In section II, we deal with the localization method that we propose to know the exact position of the robot while moving. We propose as well an optimization step needed to be sure that the obtained position is the optimal one. Simulation results are then analysed to give an idea about the performances of the localization algorithm in section III. Section IV is dedicated to validate the developed method by an experimental application on the mobile robot "Wifibot". After describing its platform and different sensors used in the process, we display localization measurements and compare between measured and calculated positions. Finally, conclusions are presented in section V.

II. PROPOSED METHOD FOR LOCALIZATION AND ROBOTIC MAPPING

In this paper, we propose an efficient approach in order to locate the robot during its navigation with accuracy. We assume that the robot's initial position is approximatively known. As to odometric feedback, the robot position can be easily but not precisely calculated. Some errors are caused by interaction of the robot with inevitable features of the environnement as wheel slippage, brutal or fast mouvements... Therefor, we also depend on other sensors to get more reliable data.

ISSN: 2356-5608

The presented method to locate the robot is inspired from the Kalman Filter method where we combine between dead reckoning and extracted data from robot's sensors. Considering the differential structure of the robot and the reliability of the Wifibot's sensors, we are able to obtain the robot's position with a high accuracy. Despite the fact that the robot navigates in a known environment, several constraints may appear especially when the robot must follow a certain trajectory or avoid different obstacles. That is why we need to optimize the localization algorithm. The Levenberg-Marquardt Method (LMM) is the most suitable technique to use [5], [7].

Regarding nonlinear least square problems, the LMM tries to fit a parameterized function to a set of measured data points by minimizing the sum of the squares of the errors between measured values and calculated ones. This technique is in fact a combination between the gradient descent method and the Gauss-Newton method considered as standard minimization methods. Using the gradient descent method, the sum of squared errors is reduced by finding the minimum of the locally quadratic function. The LMM acts like a gradient descent method when parameters are far from their optimum values and acts like the Gauss-Newton method when parameters are close to their optimum values.

However, the algorithm effectiveness depends on local minima which can not be necessarily global ones. This fact, mislead to an incorrect information especially in presence of symmetric effects in the navigation environnement [9]. To avoid this problem, we rely on the approximation of the robot position provided by odometric sensors. Consequently, every time the robot moves, the algorithm initializes its actual position comparing to its previous one [6]. To this way and despite the deterministic nature of the LMM algorithm, it presents best convergence properties and ensures more precise and quicker solutions.

A. Formulation of the Levenberg-Marquardt algorithm

1) Optimisation proposition: We introduce the function $\hat{y}(t,p)$ defined for the variable t and a vector p of m points. In order to minimize the sum of the least squared error between the measured parameters of $y(t_i)$ and calculated ones of $\hat{y}(t,p)$, we define an error criterion known as Chi-square χ^2 with:

$$\chi^2(p) = \sum_{i=1}^{m}\left[\frac{y(t_i)-\hat{y}(t_i,p)}{w_i}\right]^2 \quad (1)$$

$$= (y-\hat{y}(p))^T W (y-\hat{y}(p)) \quad (2)$$

$$= y^T W y - 2y^T W\hat{y} + \hat{y}^T W\hat{y} \quad (3)$$

w_i is the measurement error of the function $y(t_i)$ and W is a diagonal matrix with:

$$W_{ii} = \frac{1}{{w_i}^2}$$

In every iteration, we aim finding the perturbation h of the parameter p minimising χ^2.

2) The gradient descent method: The gradient descent technique is highly known and used for optimisation problems especially for large scaled systems. We express the gradient of the χ^2 by:

$$\frac{\partial}{\partial p}\chi^2 = [y-\hat{y}(p)]^T W \frac{\partial}{\partial p}[y-\hat{y}(p)] \quad (4)$$

$$= -[y-\hat{y}(p)]^T W \frac{\partial \hat{y}(p)}{\partial p} \quad (5)$$

$$= -[y-\hat{y}(p)])^T W J \quad (6)$$

J is an $(n \times m)$ jacobian matrix representing the sensibility of the function $\hat{y}$ while the variable p is varying. As a result, the perturbation h responsable of heading parameters toward the highest descent is:

$$h_{gr} = \alpha J W (y-\hat{y})$$

α is a variable that gives an idea about the direction of the descent.

3) The Gauss-Newton method: This technique minimizes the sum of squares of a quadratic function that we consider very near to the optimal solution. It has proved its efficiency especially for medium scaled systems and converges faster than the gradient descent method. The Gauss-Newton method introduces a perturbation element h described by the following Taylor expansion:

$$\hat{y}(p+h) = \hat{y}(p) + \left[\frac{\partial \hat{y}}{\partial p}\right] h \quad (7)$$

$$= \hat{y} + Jh \quad (8)$$

Substituting the function $\hat{y}(p+h)$ in equation (3), we obtain:

$$\chi^2(p+h) = y^T W y + \hat{y}^T W \hat{y} - 2y^T W \hat{y} - 2(y-\hat{y})^T W J h + h^T J^T W J h \quad (9)$$

The result clearly proves that χ^2 is approximatively quadratic. The hessian matrix of the χ^2 verifying the minimisation criteria is defined by $J^T W J$. The perturbation h minimizing χ^2 is obtained by having $\frac{\partial \chi^2}{\partial h} = 0$. Thus:

$$\frac{\partial \chi^2(p+h)}{\partial h} \approx -2(y-\hat{y})^T W J + 2h^T J^T W J \quad (10)$$

The Gauss-Newton perturbation is defined by the next equation:

$$[J^T W J] h_{gn} = J^T W (y-\hat{y}) \quad (11)$$

4) The Levenberg Marquardt Method: When combining these two last methods, we obtain the Levenberg Marquardt technique which fits its parameters between both of the gradient descent and the Gauss-Newton optimisation methods. The equation becomes:

$$[J^T W J + \lambda I] h_{lm} = J^T W (y-\hat{y}) \quad (12)$$

In case of lowest values of λ, we apply the Gauss-Newton update. Otherwise, it is the gradient descent update.

Consequently,we define the convergence step of Levenberg Marquardt h_{lm} as follows:

$$h_{lm} = (H_i + \lambda I)^{-1} G_i$$

with:

- H is the hessian matrix of the minimization criteria and given by $J^T W J$.
- G_i is the calculated gradient defined by $J^T W(y - \hat{y})$.
- λ is a setting parameter which increases when the cost function diverges.

B. Presentation of the proposed localization approach

The developed technique of localization followed by the defined optimisation step can be summarized in the next organisation chart.

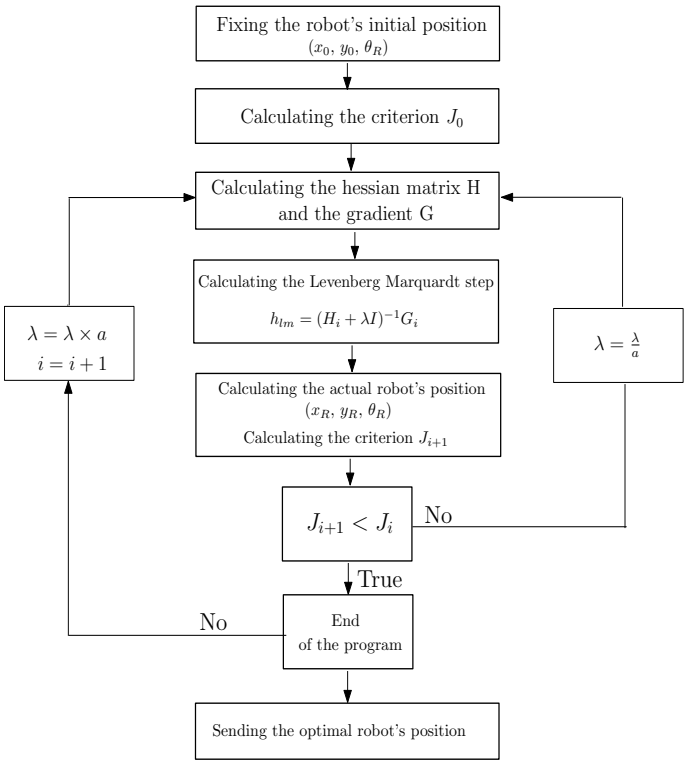

Fig. 1. Description of localization steps

C. Application of the localization method

We have developed a tracking algorithm depending on the robot's position defined by (x_R, y_R) and its orientation θ_R. We assume that the robot' initial position is its actual one. While the optimal real position is thereafter given by the LM algorithm. The localization technique is based on the following steps:

- Acquisition of sensors measurements.
- Computation of the distance between the obstacle and the robot based on its actual position.
- Defining the convenient cost function Z for the system.
- Minimizing Z using the optimization method.

The mapping step requires a known environment where the robot navigates in presence of obstacles that it should avoid. Map's dimensions are given as $(430 \times 460)\ cm^2$ and an obstacle of $(45 \times 30)\ cm^2$ placed in the map at the position $(138 \times 286)\ cm^2$.

1) Acquisition of odometric data: In the localization process, we divide the environment into n areas. We note those measurements $(m_{t1}, ..., m_{tn})$.

2) Distance calculation: Based on the robot's position, we are able to determine n distance values which represent in fact theoretical measures noted $(m_1, ..., m_n)$. We create a loop that increments the variable d until an obstacle occurs. As a result, the obstacle position is defined by this form:

$$\begin{aligned} x &= x_R + d\cos(\theta_R + \theta_C) \\ y &= y_R + d\sin(\theta_R + \theta_C) \end{aligned}$$

where:

- $(x,\ y)$ is the calculated obstacle's position depending on the actual position of the robot $(x_R,\ y_R)$.
- θ_R is the orientation angle given by the robot.
- θ_C is a vector containing n values.

The distance d is fixed initially at 0 and incremented in each iteration. In presence of an obstacle, we deduce the measure that we note:

$$m = d$$

3) Cost function: We define the cost function as:

$$Z = \sum_{i=1}^{n} (m_{ti} - m_i)^2$$

The main objective is to use the minimal cost function Z in order to locate the robot. We consider the gradient method in this part. To refine the calculation, we integrate in the algorithm a comparison between calculated position coordinates and odometry data that the robot's sensors provide. As a result, coordinates (x_R,y_R) corresponding to the lowest cost function is the most probable real position of the robot.

III. Evaluation of the localization method

A. Programming of the navigation map

In order to obtain simulation results of the localization algorithm, we program the navigation environment of the mobile robot. In $C++$, the map is transformed into an array containing values of $\{1\}$ to indicate the presence of an obstacle and $\{0\}$ to show an empty space. The created array is formed of pixels with 1 pixel $= 4cm^2$. Figure 2 describes the designed environment.

B. Simulation results

We choose to test the developed algorithm's performance using the simulation calculator MATLAB. The robot is dedicated to reach a target fixed by the algorithm and starting from a random position. The robot's orientation is already known.

1) First case : The target point is (50,50) with an initial orientation $-90°$. The initial position is defined along a 50 cm surrounding the target. Figures 3, 4, 5 and 6 present respectively the trajectory made by the robot to reach its destination, the convergence of the two parameters x and y to their fixed values and finally the progression of the criteria cost function Z that we aim minimizing.

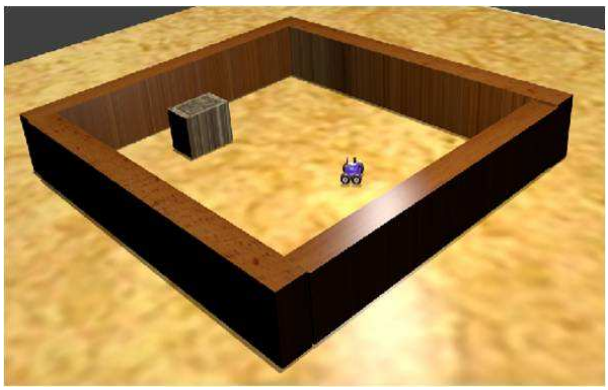

Fig. 2. Designed environment

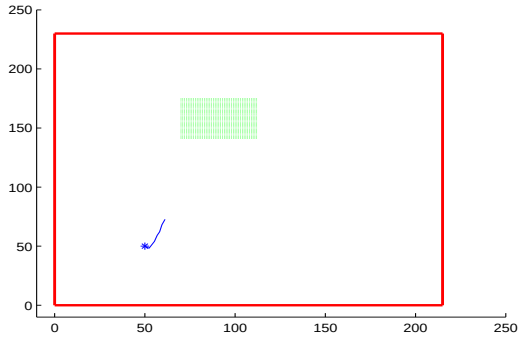

Fig. 3. The robot path for reaching the target

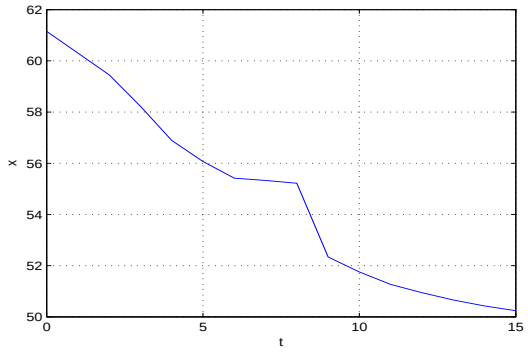

Fig. 4. Convergence of the x position toward its desired value

2) Second case : The target point is (180,200) with an initial orientation $45°$. The initial position is defined along a 100 cm surrounding the target. Figures 7, 8, 9 and 10 present respectively the trajectory made by the robot to reach its destination, the convergence of coordinates x and y to their fixed values and finally the progression of the criteria cost function Z that we aim minimizing.

All presented curves and robot's trajectories show clearly that the robot reaches its defined target successfully. In spite of the robot's random initial position, we see that every time, it finds its way to go to its desired destination.

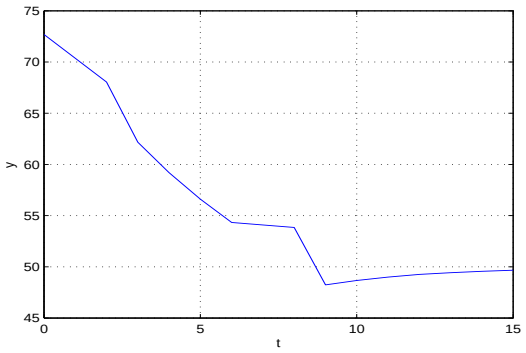

Fig. 5. Convergence of the y position toward its desired value

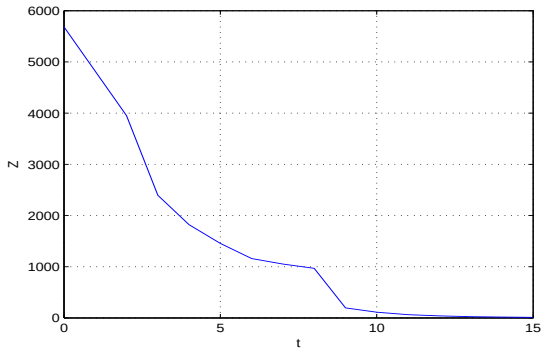

Fig. 6. Convergence of the criterion Z toward the minimal value

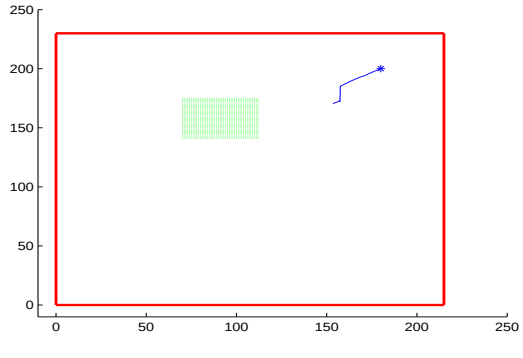

Fig. 7. The robot path for reaching the target

IV. Experimental validation of the localization method

We choose to apply the developed localization method along with optimization technique on a real mobile robot knows as Wifibot.

A. Description of the Wifibot

Wifibot presents a multi-purpose robot. It is a Wifi enabled, low cost and running Linux platform. Wifibot is useful for several applications in the fields of education, research, supervision and entertainment. It is a differential robot with 4

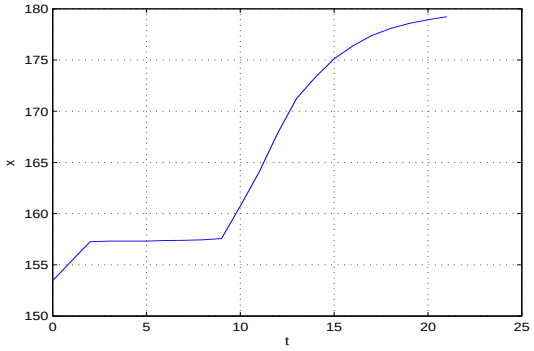

Fig. 8. Convergence of the x position toward its desired value

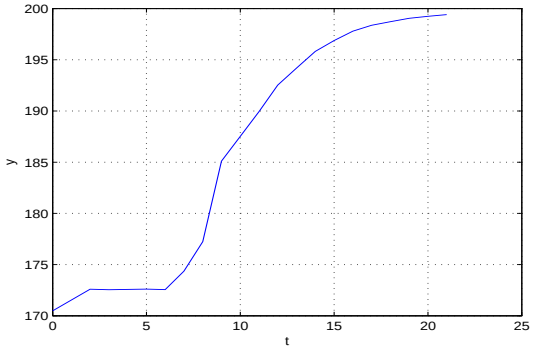

Fig. 9. Convergence of the y position toward its desired value

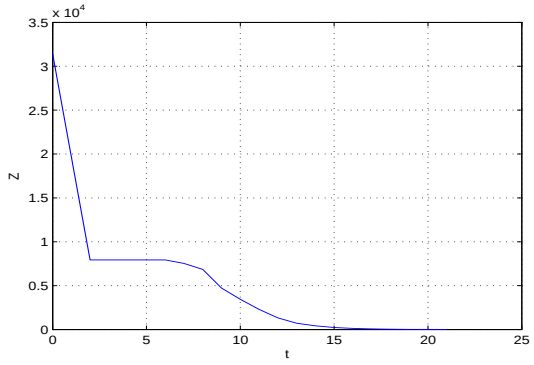

Fig. 10. Convergence of the criterion Z toward the minimal value

driving wheels. It is characterized by an open and modular architecture controlled by a serial communication link and by Wifi. It integrates a computing board unit running on Linux, Ubuntu. This platform stands by its simplicity and efficiency. The Wifibot is composed by an anodized aluminum frame, an USB motorized camera, 4 infrared sensors, central inertia $V_n - 100$ and a laser sheet Hokuyo $UTM - 30LX$. The robot platform is controlled using a $RS232$ port. The Wifi board ensures a wireless connection of the system with the configured access point which is provided freely. The structure of the Wifibot platform presents an advantage to the user to develop and manipulate different applications in an easy way. Figure 11 describes the Wifibot robot.

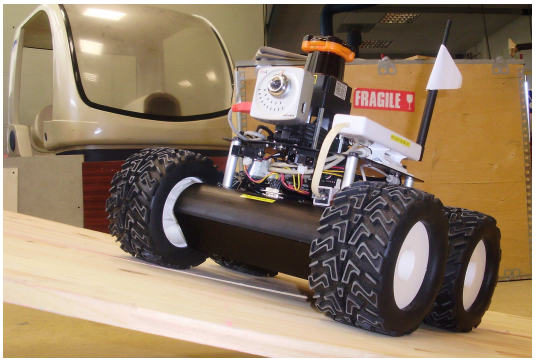

Fig. 11. The Wifibot robot

The Inertial Measurement Unit, odometer and laser telemeter are sensors involved in the process.Realized tests have proved that the laser telemeter is the most reliable sensor to obtain an exact position of the robot. It will be considered as the main sensor for the proposed method. In order to avoid failure in the program, orientation angle is chosen to be extracted from the robot. Starting from a fixed orientation, it is due to the inertial unit that we are able to obtain the relative robot's orientation while moving.

B. Calculated distances from the Wifibot

As the Laser sensor supplies 1081 measures sweeping from $-135°$ to $+135°$, we restrained the test to a limited number of samples n. Considering a step of $27°$, we obtain $n = 11$ measurements. Figure 12 illustrates calculated distances in the map in presence of an obstacle and the robot. The robot is situated in the center of the map at the position defined by:

$$\begin{aligned} x_0 &= 215cm \\ y_0 &= 230cm \end{aligned}$$

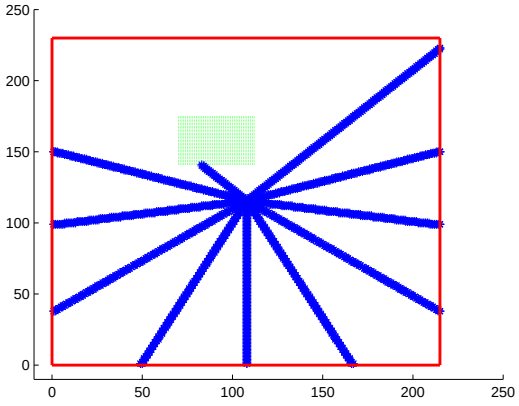

Fig. 12. Calculated distances between the robot and obstacles

C. Experimental results

The localization method has been transformed in $C++$ langage and implemented then in RTMaps Studio which represent the appropriate middleware to control the Wifibot. Two projects have been created, the first was implemented on the robot itself and the second was within reach of the user to extract different mesures. Figure 13 and 14 represent respectively measured positions from laser telemeter and calculated position resulted from the algorithm.

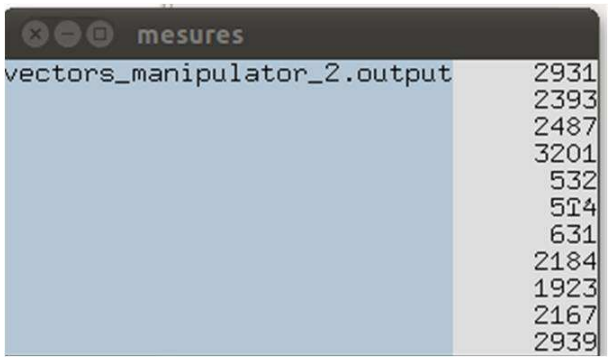

Fig. 13. Measured positions in mm

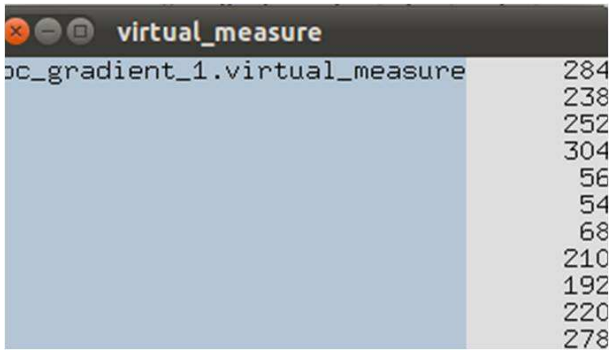

Fig. 14. Calculated position in cm

Presented results are approximatively equal which reveal the effectiveness of followed localization and optimization techniques. Many improvements are recommended to strengthen more the algorithm.

V. Conclusion

As it is primordial to know the robot's position and orientation all along its navigation, we have proposed a method to track the robot and to determine coordinates of its control point besides its orientation θ. This method is based on the robot's model and the extracted data from its exteroceptive sensors. We have proceeded then with an optimization using the Levenberg Marquardt algorithm to ensure the convergence of the cost function to its minimal value. Simulation and direct experimental tests on the robot show the robustness of the proposed method and its efficiency to locate the robot in the map. The robot is able to avoid the obstacle and map fronts with success. The next phase is to handle the navigator which tasks are now easier thanks to the localization's robustness.

References

[1] C. Novales, G. Mourioux, G. Poisson, "Une Architecture Modulaire de Commande de Robots: de l'Autonomie la tl-opration", Journal Europen des Systmes Automatiss(JESA), vol5, pp.1-20, 2008.

[2] C. F. Olson, "Probabilistic Self-Localization for Mobile Robots", IEEE Transactions on Robotics and Automation, Vol. 16, No. 1, 2000.

[3] D.W. Marquardt, "An algorithm for least-squares estimation of nonlinear parameters", J. Soc. Ind. Appl. Math, vol.11, no.2, pp.431441,1963.

[4] D. Fox, W. Burgard, F. Dellaert, and S. Thrun, "Monte Carlo Localization: Efficient Position Estimation forMobile Robots", American Association for Artificial Intelligence, pp.343-349, 1999.

[5] H. P. Gavin, "The Levenberg-Marquardt method for nonlinear least squares curve-fitting problems", Department of Civil and Environmental Engineering Duke University, 2013.

[6] M. K. Transtrum and J. P. Sethna, "Improvements to the Levenberg-Marquardt algorithm for nonlinear least-squares minimization", Journal of Computational Physics (Data Analysis, Statistics and Probability), pp. 1-32, 2012.

[7] M. O. Efe and O. Kaynak, "A novel optimization procedure for training of fuzzy inference systems by combining variable structure systems technique and Levenberg-Marquardt algorithm", Fuzzy Sets and Systems, Vol. 122, No. 1, pp. 153-165, 2001.

[8] M. Pfingsthorn and A. Birk, "Simultaneous Localization and Mapping (SLAM) with Multimodal Probability Distributions", The International Journal of Robotics Research, pp.129, October 2012.

[9] N.Morette ,C. Novales ,L. Josserand , and P. Vieyres, "Direct Model Navigation issue shifted in the continuous domain by a predictive control approach for mobile robots", International Conference on Robotics and Automation, pp.2566 - 2573, 2011.

[10] R. Siegwart, D. Scaramuzza, "Autonomous Mobile Robots". vol.5-Localization and Mapping, pp.292-338.

[11] U. A. Sheikh, M. Jamil and Y. Ayaz, "A comparison of various robotic control architectures for autonomous navigation of mobile robots", International Conference on Robotics and Emerging Allied Technologies in Engineering (iCREATE) Islamabad, Pakistan, pp.239-243, April 22-24, 2014.

[12] W. Burgard, D. Fox and S. Thrun, "Active Mobile Robot Localization", International Joint Conference on Artificial Intelligence (IJCAI), pp.1346-1352, 1997.