Monocular-SLAM using Floor Lines

Andre M. SANTANA*, Adelardo A. D. MEDEIROS**
*Department of Informatics and Statistics (DIE)
Federal University of Piauí (UFPI - Teresina - PI - Brazil)
**Department of Computer Engineering and Automation (DCA)
Federal University of Rio Grande do Norte (UFRN - Natal - RN - Brazil)
Emails: andremacedo@ufpi.edu.br,adelardo@dca.ufrn.br

***Abstract*— This work proposes a SLAM technique based on Extended Kalman Filter (EKF) to navigate a robot in an indoor environment using odometry and pre-existing lines on the floor as landmarks. The lines are identified by using the Hough transform. The prediction phase of the EKF is implemented using the odometry model of the robot. The update phase directly uses the parameters of the lines detected by the Hough transform without additional intermediate calculations. Experiments with real data are presented.**

I. INTRODUCTION

In the problem of simultaneous localization and mapping (SLAM), a mobile robot autonomously explores the environment with its on board sensors, gains knowledge about it, interprets the scene, builds an appropriate map and localizes itself relative to this map. The representation of the maps can take various forms, such as occupancy grids and features maps. We are interested in the second representation.

Thanks to advances in computer vision and cheaper cameras, vision-based SLAM have become popular [8], [12], [10]. In the work of Folkesson et al. [6], lines and points were extracted by image processing and used to solve the SLAM problem. Chen and Jagath [3] proposed a SLAM method with two phases. Firstly, high level geometric information, such as lines and triangles constructed by observed feature points, is incorporated to EKF to enhance the robustness; secondly, a visual measurement approach, based on multiple view geometry, is employed to initialize new feature points.

A classical approach to SLAM is to use Extended Kalman Filter (EKF) [5], [2]. These SLAM algorithms usually require a sensor model that describes the robot's expected observation given its position. Davison [4] uses a single camera, assuming a Gaussian noise sensor model whose covariance is determined by the image resolution. Zucchelii [15] discusse how to propagate the uncertainty of the camera's intrinsic parameters into a covariance matrix that characterizes the noisy feature positions in the 3D space. Wu and Zang [14] present a work whose principal focus is on how to model the sensor uncertainty and build the probabilistic component of a camera model.

The main challenges in visual SLAM are: a) how to detect features in images; b) how to recognize that a detected feature is or is not the same as a previously detected one; c) how to decide if a new detected feature will or will not be adopted as a new mark; d) how to calculate the 3D position of the marks from 2D images; and e) how to estimate the uncertainty associated with the calculated values. In the general case, all these aspects have to be addressed. However, in particular situations, it is sometimes possible to develop a specific strategy to overcome these difficulties. That is the proposal of this work.

We present a SLAM technique suitable for flat indoor environments with lines on the floor. This is not a so restrictive assumption, since this is the case in many buildings such as universities, shopping malls, museums, hospitals, homes and airports, for example. Our approach is not a generic solution to the SLAM problem but an effective solution for indoor environments.

Using pre-existing lines as marks, the overall complexity of the SLAM problem is reduced, since: a) lines can be easily detected in images; b) lines on the floor are usually equally spaced well apart, so the possibility of confusion is reduced; c) as the number of lines in a image is not so big, every newly-detected line can define a new mark; d) a flat floor is a 2D surface and then there is a constant and easily calculable conversion matrix (a homography) between the image plane and the floor plane, without uncertainties concerning the 3D depth of points; and e) after processing, the number of pixels in the image belonging to the line is a good measure of confidence in the detected mark.

Lemaire & Lacroix [11] proposed the use of 3D lines as landmarks. They report the following advantages of using 3D lines: first, these primitives are very numerous in indoor environments; second, in contrast to sparse point maps, which are only useful for location purposes, a relevant segmentation map provides information on the structure of the environment. Also using lines, in this case, vertical, Fu et al. [7] carried out a study on the fusion of laser and camera information in an extended Kalman filter for SLAM. In this work, the lines are extracted from the image using Canny. Ahn et al. [1] shows a map with characteristics of 3D points and lines for indoor environments.

Our approach differs from the last three studies since it uses 2D lines of the environment as landmarks. These lines are extracted from the images through Hough transform and mapped to the robot plane by means of homography. The ρ and α parameters represent the characteristics of each line and are used in Kalman equations without an intermediary position or distance calculation stage.

978-0-7695-4231-7/10 $26.00 © 2010 IEEE
DOI 10.1109/LARS.2010.15

II. MODELING

A. Extended Kalman Filter - EKF

In this work, the Extended Kalman Filter (EKF) deals with a system that is modeled by System 1, whose variables are described in Table II-A. The signals ε_t and δ_t are supposed to be Gaussian white noises with zero mean.

$$\begin{cases} \mathbf{s}_t = p(\mathbf{s}_{t-1}, \mathbf{u}_{t-1}, \varepsilon_{t-1}) \\ \mathbf{z}_t = h(\mathbf{s}_t) + \delta_t \end{cases} \tag{1}$$

At each sampling time, the EKF calculates the best estimate of the state vector in two phases: the **prediction phase** uses System 2 to predict the current state based on the previous state and on the applied input signals and the **update phase** uses System 3 to correct the predicted state by verifying its compatibility with the actual sensor measurements.

$$\begin{cases} \bar{\mu}_t = p(\mu_{t-1}, \mathbf{u}_{t-1}, 0) \\ \bar{\Sigma}_t = \mathbf{G}_t \Sigma_{t-1} \mathbf{G}_t^T + \mathbf{V}_t \mathbf{M}_t \mathbf{V}_t^T \end{cases} \tag{2}$$

$$\begin{cases} \mathbf{K}_t = \bar{\Sigma}_t \mathbf{H}_t^T (\mathbf{H}_t \bar{\Sigma}_t \mathbf{H}_t^T + \mathbf{Q}_t)^{-1} \\ \mu_t = \bar{\mu}_t + \mathbf{K}_t (\mathbf{z}_t - h(\bar{\mu}_t)) \\ \Sigma_t = (\mathbf{I} - \mathbf{K}_t \mathbf{H}_t) \bar{\Sigma}_t \end{cases} \tag{3}$$

where:

$$\mathbf{G}_t = \left. \frac{\partial p(\mathbf{s}, \mathbf{u}, \varepsilon)}{\partial \mathbf{s}} \right|_{\mathbf{s}=\mu_{t-1}, \mathbf{u}=\mathbf{u}_{t-1}, \varepsilon=0} \tag{4}$$

$$\mathbf{V}_t = \left. \frac{\partial p(\mathbf{s}, \mathbf{u}, \varepsilon)}{\partial \varepsilon} \right|_{\mathbf{s}=\mu_{t-1}, \mathbf{u}=\mathbf{u}_{t-1}, \varepsilon=0} \tag{5}$$

$$\mathbf{H}_t = \left. \frac{\partial h(\mathbf{s})}{\partial \mathbf{s}} \right|_{\mathbf{s}=\mu_{t-1}} \tag{6}$$

In SLAM, besides estimating the robot pose, we also estimate the coordinates of all landmarks encountered along the way. This makes necessary to include the landmark coordinates into the state vector. If ${}^i c$ is the vector of coordinates of the i-th landmark and there are k landmarks, then the state vector is:

$$\mathbf{s}_t = \begin{bmatrix} x_t \; y_t \; \theta_t \; {}^1c_t^T \; \cdots \; {}^k c_t^T \end{bmatrix}^T \tag{7}$$

When the number of marks (k) is *a priori* known, the dimension of the state vector is static; otherwise, it grows up when a new mark is found.

B. Prediction phase: process model

Consider a robot with diferential drive in which $\Delta\theta_R$ and $\Delta\theta_L$ are the right and left angular displacement of the respective wheels, according to Figure 1. Assuming that the speeds can be considered constant during one sampling period, we can determine the kinematic geometric model of the robot's movement (System 8):

$$\begin{cases} x_t = x_{t-1} + \dfrac{\Delta L}{\Delta\theta}[\sin(\theta_{t-1} + \Delta\theta) - \sin(\theta_{t-1})] \\ y_t = y_{t-1} - \dfrac{\Delta L}{\Delta\theta}[\cos(\theta_{t-1} + \Delta\theta) - \cos(\theta_{t-1})] \\ \theta_t = \theta_{t-1} + \Delta\theta \end{cases} \tag{8}$$

in which:

$$\begin{cases} \Delta L = (\Delta\theta_R r_R + \Delta\theta_L r_L)/2 \\ \Delta\theta = (\Delta\theta_R r_R - \Delta\theta_L r_L)/b \end{cases} \tag{9}$$

where ΔL and $\Delta\theta$ are the linear and angular displacement of the robot; b represents the distance between wheels and r_R and r_L are the radii of the right and the left wheels, respectively. When $\Delta\theta \to 0$, another system, obtained from the limit of System 8, must be used.

Adopting the approach advocated by Thrun et al. [13], we consider odometric information as input signals to be incorporated to the robot's model, rather than as sensorial measurements. The differences between the actual angular displacements of the wheels ($\Delta\theta_R$ and $\Delta\theta_L$) and those ones measured by the encoders ($\Delta\tilde{\theta}_R$ and $\Delta\tilde{\theta}_L$) are modeled by a zero mean Gaussian white noise accordingly to System 10.

$$\begin{cases} \Delta\theta_R = \Delta\tilde{\theta}_R + \varepsilon_R \\ \Delta\theta_L = \Delta\tilde{\theta}_L + \varepsilon_L \end{cases} \tag{10}$$

The measured $\Delta\tilde{L}$ and $\Delta\tilde{\theta}$ are defined by replacing ($\Delta\theta_R$ and $\Delta\theta_L$) by ($\Delta\tilde{\theta}_R$ and $\Delta\tilde{\theta}_L$) in Equations 9. Using Equations 10, 9 and 8 to calculate the state model $p(.)$ (System 2) we can calculate, by deriving the model, the matrices $\mathbf{G}$ and $\mathbf{V}$ used during the prediction phase (Equations 4 and 5).

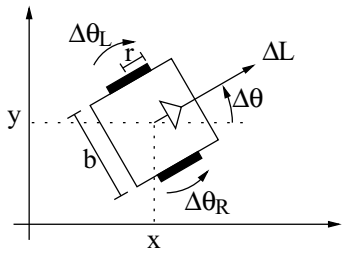

Fig. 1. Variables of the kinematic model.

$\mathbf{s}_t$	state vector (order n) at instant t
$p(\cdot)$	non-linear model of the system
$\mathbf{u}_{t-1}$	input signals (order l), instant $t-1$
ε_{t-1}	process noise (order q), instant $t-1$
$\mathbf{z}_t$	vector of measurements (order m) retourned by the sensors
$h(\cdot)$	non-linear model of the sensors
δ_t	measurement noise
$\bar{\mu}_t$, μ_t	mean (order n) of the state vector $\mathbf{s}_t$, before and after the update phase
$\bar{\Sigma}_t$,Σ_t	covariance ($n \times n$) of the state vector $\mathbf{s}_t$
$\mathbf{G}_t$	Jacobian matrix ($n \times n$) that linearizes the system model $p(\cdot)$
$\mathbf{V}_t$	Jacobian matrix ($n \times q$) that maps the process noise ε_t to a n-dimensional linearized additive noise
$\mathbf{M}_t$	covariance ($q \times q$) of the process noise ε_t
$\mathbf{K}_t$	gain of the Kalman filter ($n \times m$)
$\mathbf{H}_t$	Jacobian matrix ($m \times n$) that linearizes the model of the sensors $h(\cdot)$
$\mathbf{Q}_t$	covariance matrix ($m \times m$) of the measurement noise δ_t

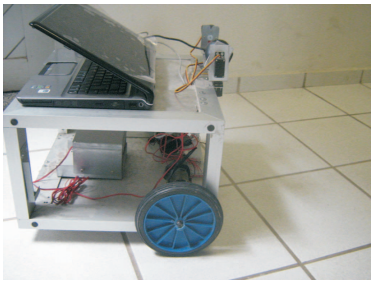

Fig. 2. Robotic system.

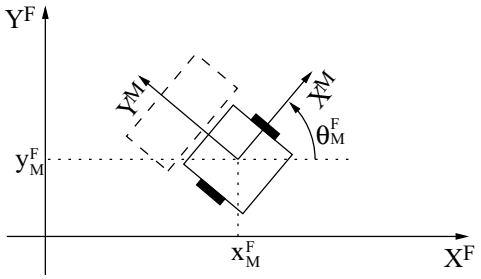

Fig. 4. Mobile and fixed coordinate systems.

It is well known that odometry introduces accumulative error. Therefore, the standard deviation of the noises ε_R and ε_L is assumed to be proportional to the module of the measured angular displacement. These considerations lead to a definition of the matrix $\mathbf{M}$ given by Equation 11.

$$\mathbf{M} = \begin{pmatrix} (M_R|\Delta\tilde{\theta}_R|)^2 & 0 \\ 0 & (M_L|\Delta\tilde{\theta}_L|)^2 \end{pmatrix} \tag{11}$$

C. Update phase: sensor model

The landmarks adopted in this work are lines on the floor. The system is based on a robot with differential drive and a fixed camera, as in Figure 2. The landmarks are detected by processing images using the Hough transform (see Section III). The detected lines are described by the parameters ρ and α in Equation 12.

Figure 3 shows the geometric representation of these parameters: ρ is the module and α is the angle of the vector from the origin of the system of coordinates to the line and perpendicular to it.

$$\rho = x\cos(\alpha) + y\sin(\alpha) \tag{12}$$

We define a fixed coordinate system (F) and a mobile one (M), attached to the robot, both illustrated in Figure 4. The origin of the mobile system has coordinates (x^F_M, y^F_M) in the fixed system. θ^F_M represents the rotation of the mobile system with respect to the fixed one. One should note that there is a straight relation among these variables $(x^M_F, y^M_F, \theta^M_F)$ and the robot's pose (x_t, y_t, θ_t), which is given by Equations 13.

$$x_t = x^M_F \qquad y_t = y^M_F \qquad \theta_t = \theta^M_F + \pi/2 \tag{13}$$

Each line on the floor is described by two static parameters (ρ^F, α^F). The map to be produced by the SLAM process is composed of a set of these pairs of parameters.

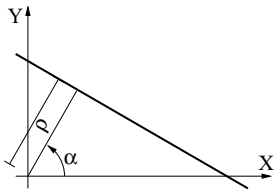

Fig. 3. Line parameters ρ and α.

So, the ${}^ic$ vector of coordinates of the i-th landmark that appear in Equations 7 is given by Equation 14:

$${}^ic = \begin{bmatrix} {}^i\rho^F \\ {}^i\alpha^F \end{bmatrix} \tag{14}$$

At each step the robot captures an image and identifies the parameters $(\tilde{\rho}, \tilde{\alpha})$[1] of the detected lines. These image parameters are then converted to the corresponding parameters $(\tilde{\rho}^M, \tilde{\alpha}^M)$ in the coordinate system M attached to the robot, using the camera parameters (see Section III). The vector of measurements $\mathbf{z}_t$ to be used in the update phase of the EKF algorithm (Equation 3) is composed of these parameters expressed in the mobile coordinate system, as defined in Equation 15[2]:

$$\mathbf{z}_t = \begin{bmatrix} \tilde{\rho}^M \\ \tilde{\alpha}^M \end{bmatrix} \tag{15}$$

To use information directly obtained by image processing $(\tilde{\rho}^M, \tilde{\alpha}^M)$ in the update phase of the EKF-SLAM, one must deduct the sensor model $h(\cdot)$, that is, the expected value of these parameters in function of the state variables.

We use the relation between coordinates in the M and F systems (System 16) and Equation 12 in both coordinate systems (Equations 17 and 18):

$$\begin{cases} x^F = \cos(\theta^M_F)x^M - \sin(\theta^M_F)y^M + x^F_M \\ y^F = \sin(\theta^M_F)x^M + \cos(\theta^M_F)y^M + y^F_M \end{cases} \tag{16}$$

$${}^i\rho^F = x^F\cos({}^i\alpha^F) + y^F\sin({}^i\alpha^F) \tag{17}$$

$$\rho^M = x^M\cos(\alpha^M) + y^M\sin(\alpha^M) \tag{18}$$

By replacing Equations 16 in Equation 17, doing the necessary equivalences with System 18 and replacing some variables using Equations 13, we obtain the Systems 19 and 20, which represent two possible sensor models $h(\cdot)$ to be used in the filter. To decide about which model to use, we calculate both values of α^M and use the model which generate the value closer to the measured value of α^M.

[1] We use a ˜ over a variable to indicate measured values, rather than calculated ones.

[2] To simplify the notation, we are assuming that there is exactly one line per image. In fact, we can have none, one or more than one line per image. At each step, we execute the update phase of the EKF as many times as the number of detected lines in the image, even none.

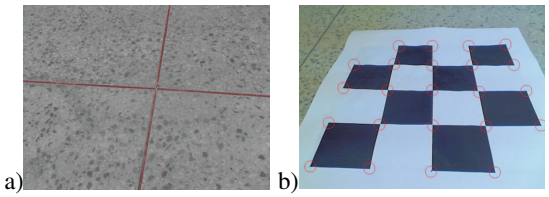

Fig. 5. Image Processing: a) detection of lines, b) calibration pattern.

$$\begin{cases} \rho^M = {}^i\rho^F - x_t\cos({}^i\alpha^F) - y_t\sin({}^i\alpha^F) \\ \alpha^M = {}^i\alpha^F - \theta_t + \pi/2 \end{cases} \quad (19)$$

$$\begin{cases} \rho^M = -{}^i\rho^F + x_t\cos({}^i\alpha^F) + y_t\sin({}^i\alpha^F) \\ \alpha^M = {}^i\alpha^F - \theta_t - \pi/2 \end{cases} \quad (20)$$

The sensor model is incorporated into the EKF through the **H** matrix (Equation 6), given by Equation 21:

$$\mathbf{H} = \begin{pmatrix} -\cos{}^i\alpha^F & -\sin{}^i\alpha^F & 0 & \cdots & 1 & 0 & \cdots \\ 0 & 0 & -1 & \cdots & 0 & 1 & \cdots \end{pmatrix} \quad (21)$$

The final columns of the **H** matrix are almost all null, except for the columns corresponding to the landmark in the vector state that matches the detected line on the image.

D. Matching

A crucial aspect of the SLAM algorithm is to establish a match between the detected line on the image and one of the landmarks represented in the state vector. To choose the correct landmark, we first calculate the predicted values of (ρ^F, α^F) using the measured values of $(\tilde{\rho}^M, \tilde{\alpha}^M)$ and the model in Equations 19, if $\tilde{\alpha}^M \geq 0$, or in Equations 20, if $\tilde{\alpha}^M < 0$. Then, these predicted values are compared to each one of the values $({}^i\rho^F, {}^i\alpha^F)$ in the state vector. If the difference between the predicted values and the best $({}^i\rho^F, {}^i\alpha^F)$ is small enough, a match was found. If not, we consider that a new mark was detected and the size of the state vector is increased.

III. Image Processing

A. Detection of lines

Due to the choice of floor lines as landmarks, the technique adopted to identify them was the Hough transform [9]. The purpose of this technique is to find imperfect instances of objects within a certain class of shapes by a voting procedure. This voting procedure is carried out in a parameter space, from which object candidates are obtained as local maxima in a accumulator grid that is constructed by the algorithm for computing the Hough transform.

In our case, the shapes are lines described by Equation 12 and the parameter space has coordinates (ρ, α). The images are captured in grayscale and converted to black and white using a threshold level, determined off-line. Figure 5.a shows a typical image of the floor and the lines detected by the Hough transform.

B. From images to the world

We assume that the floor is flat and that the camera is fixed. So, there is a constant relation (a homography **A**) between points in the floor plane (x, y) and points in the image plane (u, v):

$$s \cdot \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \mathbf{A} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \quad (22)$$

The scale factor s is determined for each point in such a way that the value of the third element of the vector is always 1.

The homography can be calculated off-line by using a pattern containing 4 or more remarkable points with known coordinates (see Figure 5.b). After detecting the remarkable point in the image, we have several correspondences between point coordinates in the floor plane and in the image. Replacing these points in Equation 22, we obtain a linear system with which we can determine[3] the elements of the homograpy matrix **A**. Once calculated the homography, for each detected line we do the following: a) using the values of $(\tilde{\rho}, \tilde{\alpha})$ obtained by the Hough transform, calculate two point belonging to the line, b) convert the coordinates of these two points to the floor plane and c)determine $(\tilde{\rho}^M, \tilde{\alpha}^M)$ of the line that passes through these two points.

C. Sensor noise

As it is showed in Figure 2, the camera position is such that the image plane is not parallel to the floor plane. The resulting effect caused by the camera slope can also be seen in Figure 5.b. From experimentation, one observed that existing information at the top of the image suffered higher noise if compared to the bottom area, what made us consider that noise variance must be proportional to the distance (ρ) of the straight line on the image. Besides, one noticed that quality of horizontal lines which appeared on the image was better than that of vertical ones, what allowed us to understand that noise variance was also related to the straight line angle (α) on the image.

If those aspects above are taken into consideration, then the sensor noise variance adopted in this work is in accordance with the equation 23. In this equation, the term $[\exp^{(\rho/c)} - 1]$ represents the distance proportionality, and the term $[\sin(\alpha)]$, the angle influence. The behavior of the function described through the equation 23 is showed in Figure 6.a. The values of the constants a , b and c were calculated through experimentation. Their values are: $a = 0.004$, $b = 0.3$ and $c = 45$. Figure 6.b shows noise variance function used in this work.

$$\sigma(\rho, \alpha) = a + b \cdot \sin(\alpha) \cdot (\exp^{(\rho/c)} - 1) \quad (23)$$

[3] **A** can be determined up to a scale factor. To obtain a unique answer, we impose $a_{33} = 1$.

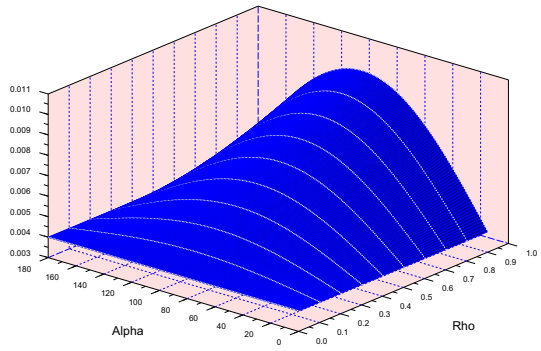

Fig. 6. Variance function: behavior of the noise variance function.

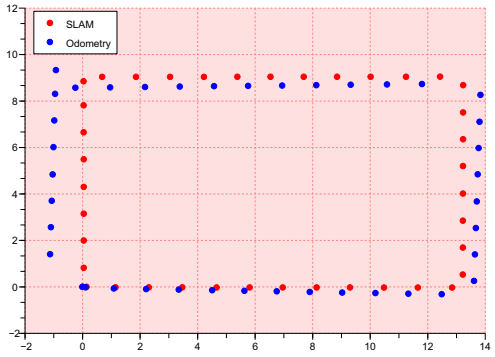

Fig. 8. Trajectory.

IV. Results

The experiments were carried out using a robot whose two wheels are driven by DC motors with differential steering. Each motor has an optical encoder and a dedicated card based on a PIC microcontroller that controls local velocity.

The computer that controls the robot is a notebook with a color webcam. The camera captures 640 x 480 images (Figure 5.a) and each image is processed at 200 ms. Figure 5.b shows the pattern which was used at the beginning of the experiment (robot view) to calculate the homography.

The camera was positioned so it was possible to have a view angle of the order of twice the robot size. The equation 24 shows the calculated homography matrix $\mathbf{A}$.

$$\mathbf{A} = \begin{pmatrix} 0.1417 & 0.0009 & -49.2065 \\ 0.0073 & -0.0761 & 98.6323 \\ 0.0001 & 0.0029 & 1 \end{pmatrix} \qquad (24)$$

The experiment was carried out in an 13.5×9.0 meters environment[4] with horizontal and vertical lines approximately each 25cm (Figure 7). The robot executed an almost-rectangular closed loop trajectory inside this building, during which 1832 images were processed.

[4]The building can be seen in online map systems at coordinates 5° 50' 29.50"S, 35° 11' 49.48"W

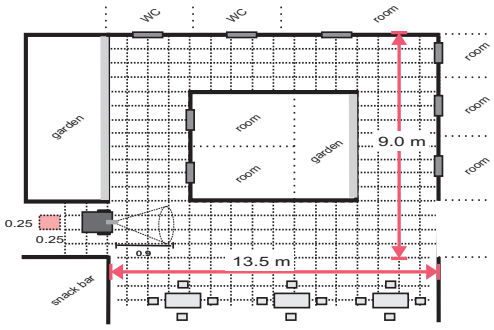

Fig. 7. Department of Physics - UFRN.

Figure 8 shows the calculated trajectory, with a zoom in the region where the loop is closed (Figure 9). The blue dots correspond to the trajectory calculated by odometry only and the red dots, to the trajectory calculated by SLAM. In this experiment, each line was observed in 15 consecutive images, in average. In 98% of the images the system detected lines: tree lines in 61%, four lines in 26% and five lines in 11% of the images. The remaining 2% were errors of the processing image algorithm, due to illumination changes. The matching of detected lines was correct in 95% of the cases.

The distance from the initial position to the final one, calculated assuming an initial pose $(0,0,0^o)$, was 0.89m, using odometry only, and 0.02m, using SLAM; the actual final distance, measured *in loco*, was very close to the one calculated by SLAM.

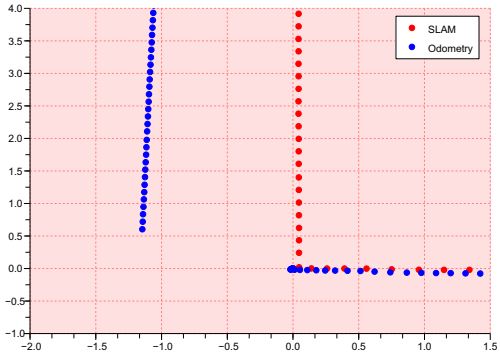

Fig. 9. Trajectory: zoom in the region where the loop is closed.

Figure 10 shows the behavior of the y components of the robot pose when closing the loop and Figure 11 shows the variance (σ^2) in ρ of the first re-observed line when the loop is closed. This line was first observed in step 23, and first re-observed in step 1591.

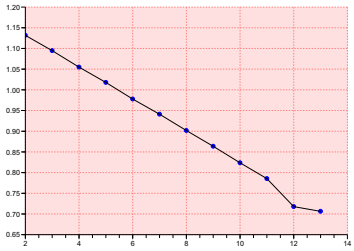

Fig. 10. Robot Pose: coordinate y.

It is possible to verify the variance update just when the lines were observed and re-observed. Before the first re-observation (closed loop) the value of the variance in ρ was 4.6×10^{-5} ($\sigma = 0,68$cm) and after re-observations the value was decreased to 9.2×10^{-6} ($\sigma = 0,30$cm). Finally, the variances of the robot pose are shown in Figure 12.

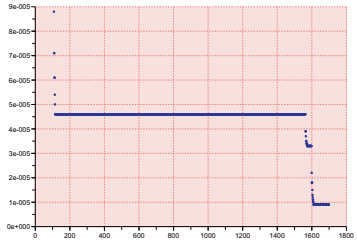

Fig. 11. Behavior of the variance in *rho*.

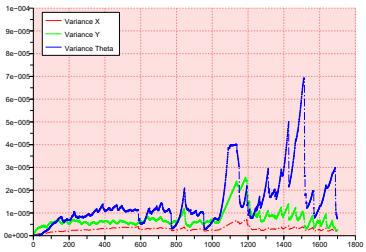

Fig. 12. Variance (x, y, θ).

V. Conclusions and Perspectives

The main contribution of this work is an optical sensor model that enables the use of parameters obtained from the image processing algorithm directly in the Kalman filter equations, without intermediate phases to calculate position or distance, and the representation of the environment using floor lines to reduce the number of landmarks used in the SLAM process. Another important characteristic of our approach is using a monocular vision system and no other sensors (lasers, sonars, etc.) besides the odometry.

The proposed approach has no pretension of being general, as it requires a flat floor with lines. However, in the cases where it can be used, when compared to other approaches to visual SLAM, it is much more efficient both in computational cost, because of the reduced number of features, and in precision, because of the low rate of mismatches and the accuracy when determining the 3D position of landmarks. Even in closed-loop environments, as the one presented in the results, the system correctly recognized previously-detected lines with no need of special procedures to deal with this problem.

As future works, we intend: to improve the real-time properties of the image processing algorithm, by adopting some of the less time-consuming variants of the Hough transform; to deal with line segments with finite length, incorporating this characteristics to the step of matching lines and using the position of terminal points of the segments as another feature to be included into the map; and, finally, to test our approach with other statistical filters.

References

[1] Ahn, S., Chung, W. and Oh, S., "Construction of hybrid visual map for indoor SLAM", *IEEE/RSJ International Conference on Intelligent Robots andSystems*, 2007.

[2] Castellanos, J., Neira, J. and Tardos, J., "Multisensor fusion for simultaneous localization and map building", *IEEE Transactions on Robotics and Automation* **17**(6), 2001.

[3] Chen, Z. and Jagath, S., "A visual SLAM solution based on high level geometry knowledge and Kalman filtering", *IEEE Canadian Conference on Electrical and Computer Engineering*, 2006.

[4] Davison, A. J., "Real-time simultaneous localisation and mapping with a single camera", *IEEE International Conference on Computer Vision*, Vol. 2, Nice, France, pp. 1403–1410. 2003.

[5] Dissanayake, M. G., Newman, P., Clark, S. and Durrant-Whyte, H. F., "A solution to the simultaneous localization and map building (SLAM) problem", *IEEE Transactions on Robotics and Automation* **17**(3), 2001.

[6] Folkesson, J., Jensfelt, P. and Christensen, H., "Vision SLAM in the measurement subspace", *IEEE International Conference on Robotics and Automation*, pp. 30–35, 2005.

[7] Fu, S., Liu, C., Gao, L. and Gai, Y., "SLAM for mobile robots using laser range finder and monocular vision", *IEEE Int. Conference on Robotics and Automation*, 2007.

[8] Goncalves, L., di Bernardo, E., Benson, D., Svedman, M., Ostrovski and J., Karlsson, N., "A visual front-end for simultaneous localization and mapping", *IEEE International Conference on Robotics and Automation*, pp. 44–49, 2005.

[9] Gonzalez, R. C. and Woodes, R. E., *Processamento de Imagens Digitais*, Edgard Blucher. ISBN: 8521202644, 2000.

[10] Jensfelt, P., Kragic, D., Folkesson, J. and Bjorkman, M., "A framework for vision based bearing only 3D SLAM", *IEEE International Conference on Robotics and Automation*, pp. 1944–1950, 2006.

[11] Lemaire, T. and Lacroix, S., "Monocular SLAM as a graph of coalesced observations", *IEEE Int. Conference on Robotics and Automation*, pp.2791–2796, 2007.

[12] Mouragnon, E., Lhuillier, M., Dhome, M., Dekeyser, F. and Sayd, P., "3D reconstruction of complex structures with bundle adjustment: an incremental approach", *IEEE International Conference on Robotics and Automation*, pp. 3055–3061, 2006.

[13] Thrun, S., Burgard, W. and Fox, D., *Probabilistic Robotics*, 1 edn, MIT Press, USA. ISBN 978-0262201629, 2005.

[14] Wu, J. and Zhang, H., "Camera sensor model for visual SLAM", *IEEE Canadian Conference on Computer and Robot Vision*, 2007.

[15] Zucchelli, M. and Kosecka, J., "Motion bias and structure distortion induced by calibration errors", *Technical report*, George Mason University, 2001.