REAL-TIME VISUAL SLAM USING PRE-EXISTING FLOOR LINES AS LANDMARKS AND A SINGLE CAMERA

Andre S. MACEDO*, Gutemberg S. SANTIAGO*, Adelardo A.D. MEDEIROS*

*Departement of Computer Engineering and Automation (DCA)
Federal University of Rio Grande do Norte (UFRN), Natal RN Brazil

Emails: [andremacedo, gutemberg, adelardo]@dca.ufrn.br

Abstract— This work proposes a SLAM (Simultaneous Localization and Mapping) 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.

Keywords— SLAM; Kalman filter; Hough transform

Resumo— Este trabalho propõe uma técnica para SLAM (*Simultaneous Localization and Mapping*) baseada no filtro de Kalman estendido (EKF) para navegar um robô em um ambiente *indoor* usando odometria e linhas pré-existentes no chão como marcos. As linhas são identificadas usando a transformada de Hough. A fase de predição do EKF é feita usando o modelo de odometria do robô. A fase de atualização usa diretamente os parâmetros das linhas detectados pela transformada de Hough sem cálculos adicionais intermediários. São apresentados experimentos com dados reais.

Palavras-chave— SLAM; Filtro de Kalman; Transformada de Hough

1 Introduction

In the problem of simultaneous localization and mapping (SLAM), a mobile robot explores the environment with its 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 (Goncalves et al., 2005; Mouragnon et al., 2006; Jensfelt et al., 2006). In the work of Folkesson et al. (2005), lines and points were extracted by image processing and used to solve the SLAM problem. Chen and Jagath (2006) 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) (Dissanayake et al., 2001; Castellanos et al., 2001). These SLAM algorithms usually require a sensor model that describes the robot's expected observation given its position. Davison (2003) uses a single camera, assuming a Gaussian noise sensor model whose covariance is determined by the image resolution. Zucchelli and Košecká (2001) 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 Zhang (2007) present a work whose principal focus is on how to model the sensor uncertainty and build a probabilistic 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 strategie 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. Using the 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.

2 Theoretical background

2.1 Extended Kalman filter

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

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

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;
- the **update phase** uses System 3 to correct the predicted state by verifying its compatibility with the actual sensor measurements.

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

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

where:

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

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

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

Table 1: Symbols in Equations (1), (2) and (3)

$\mathbf{s}_t$	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) returned 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 linearizes the process noise ε_t
$\mathbf{M}_t$	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

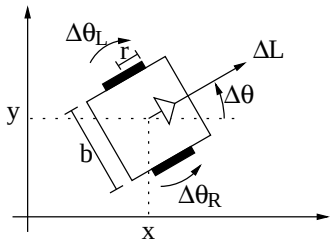

Figure 1: Variables of the kinematic model

2.2 EKF-SLAM

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 ${}^ic$ is the vector of coordinates of the i-th landmark and there are k landmarks, then the state vector is:

$$\mathbf{s}_t = \begin{bmatrix} x_t & y_t & \theta_t & {}^1c_t^T & \cdots & {}^kc_t^T \end{bmatrix}^T \quad (7)$$

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.

3 Modeling

3.1 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):

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

in which:

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

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 $\Delta\theta \to 0$, the model becomes the one in Equation 10, obtained from the limit of System 8.

$$\begin{cases} x_t = x_{t-1} + \Delta L \cos(\theta_{t-1}) \\ y_t = y_{t-1} + \Delta L \sin(\theta_{t-1}) \\ \theta_t = \theta_{t-1} \end{cases} \quad (10)$$

Adopting the approach advocated by Thrun et al. (2005), 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 11.

$$\begin{cases} \Delta\theta_R = \Delta\tilde{\theta}_R + \varepsilon_R \\ \Delta\theta_L = \Delta\tilde{\theta}_L + \varepsilon_L \end{cases} \tag{11}$$

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.

If the state vector $\mathbf{s}$ is given by Equation 7, the system model $p(\cdot)$ can be obtained from Systems 8 or 10 and from the fact that the landmarks coordinates ${}^ic$ are static:

$$p(\cdot) = \begin{cases} x_t = \cdots \\ y_t = \cdots \\ \theta_t = \cdots \\ {}^1c_t = {}^1c_{t-1} \\ \quad\vdots \\ {}^kc_t = {}^kc_{t-1} \end{cases} \quad n = 3 + k \cdot \text{order}({}^ic) \tag{12}$$

$$\mathbf{u}_t = \begin{bmatrix} \Delta\tilde{\theta}_R \; \Delta\tilde{\theta}_L \end{bmatrix}^T \quad l = 2 \tag{13}$$

$$\varepsilon_t = \begin{bmatrix} \varepsilon_R \; \varepsilon_L \end{bmatrix}^T \quad q = 2 \tag{14}$$

The $\mathbf{G}$ and $\mathbf{V}$ matrices are obtained by deriving the $p(\cdot)$ model[1] using Equations 4 and 5:

$$\mathbf{G} = \begin{pmatrix} 1 & 0 & g_{13} & 0 & \cdots & 0 \\ 0 & 1 & g_{23} & 0 & \cdots & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1 \end{pmatrix} \tag{15}$$

where:

$$g_{13} = \frac{\Delta\tilde{L}}{\Delta\tilde{\theta}}[\cos(\theta_{t-1} + \Delta\tilde{\theta}) - \cos(\theta_{t-1})]$$

$$g_{23} = \frac{\Delta\tilde{L}}{\Delta\tilde{\theta}}[\sin(\theta_{t-1} + \Delta\tilde{\theta}) - \sin(\theta_{t-1})]$$

$$\mathbf{V} = \begin{pmatrix} v_{11} & v_{12} \\ v_{21} & v_{22} \\ r_R/b & -r_L/b \\ 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \end{pmatrix} \tag{16}$$

[1]We only present the $\mathbf{G}$ and $\mathbf{V}$ matrices using the $p(\cdot)$ model based on Equations 8. Other matrices can be obtained by deriving Equations 10 to be used when $\Delta\theta \to 0$.

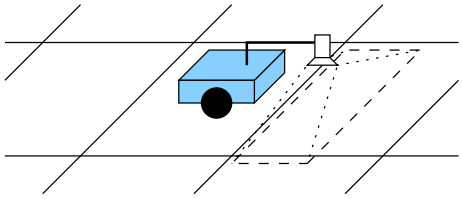

Figure 2: Robotic system.

where, considering $r_R = r_L = r$:

$$v_{11} = V_1\cos(\beta) + V_2[\sin(\beta) - \sin(\theta_{t-1})]$$
$$v_{12} = -V_1\cos(\beta) + V_3[\sin(\beta) - \sin(\theta_{t-1})]$$
$$v_{21} = V_1\sin(\beta) - V_2[\cos(\beta) - \cos(\theta_{t-1})]$$
$$v_{22} = -V_1\sin(\beta) - V_3[\cos(\beta) - \cos(\theta_{t-1})]$$

$$\beta = \theta_{t-1} + \frac{r(\Delta\tilde{\theta}_R - \Delta\tilde{\theta}_L)}{b} \quad V_1 = \frac{r(\Delta\tilde{\theta}_R + \Delta\tilde{\theta}_L)}{2(\Delta\tilde{\theta}_R - \Delta\tilde{\theta}_L)}$$

$$V_2 = \frac{-b\Delta\tilde{\theta}_L}{(\Delta\tilde{\theta}_R - \Delta\tilde{\theta}_L)^2} \qquad V_3 = \frac{b\Delta\tilde{\theta}_R}{(\Delta\tilde{\theta}_R - \Delta\tilde{\theta}_L)^2}$$

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 17.

$$\mathbf{M} = \begin{pmatrix} (M_R|\Delta\tilde{\theta}_R|)^2 & 0 \\ 0 & (M_L|\Delta\tilde{\theta}_L|)^2 \end{pmatrix} \tag{17}$$

3.2 Update phase: sensor model

The landmarks adopted in this work are lines formed by the grooves of the floor in the environment where the robot navigates. 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 4). The detected lines are described by the parameters ρ and α in Equation 18:

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

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

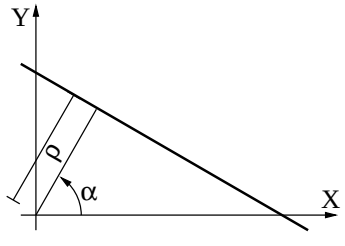

Figure 3: Line parameters ρ and α

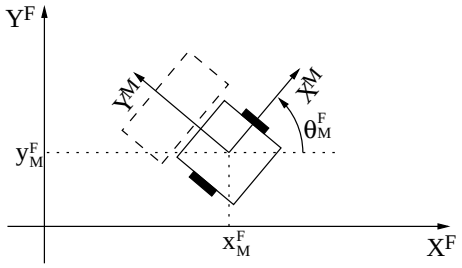

Figure 4: Mobile and fixed coordinate systems

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_M^F, y_M^F) in the fixed system. θ_M^F 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 $(x_F^M, y_F^M, \theta_F^M)$ and the robot's pose (x_t, y_t, θ_t), which is given by Equations 19.

$$x_t = x_F^M \qquad y_t = y_F^M \qquad \theta_t = \theta_F^M + \pi/2 \tag{19}$$

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. So, the ${}^ic$ vector of coordinates of the i-th landmark that appears in Equations 7 and 12 is given by Equation 20:

$${}^ic = \begin{bmatrix} {}^i\rho^F \\ {}^i\alpha^F \end{bmatrix} \tag{20}$$

At each step the robot captures an image and identifies the parameters $(\tilde{\rho}, \tilde{\alpha})$[2] of the detected lines. These image parameters are then converted to the corresponding parameters $(\tilde{\rho}^M, \tilde{\alpha}^M)$ in the mobile coordinate system M attached to the robot, using the camera parameters (see Section 4). The vector of measurements $\mathbf{z}_t$ to be used in the update phase of the EKF algorithm (Equation 3) is defined by Equation 21[3]:

$$\mathbf{z}_t = \begin{bmatrix} \tilde{\rho}^M \\ \tilde{\alpha}^M \end{bmatrix} \tag{21}$$

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 (Sistem 22) and Equation 18 in both coordinate systems (Equations 23 and 24):

$$\begin{cases} x^F = \cos(\theta_F^M)x^M - \sin(\theta_F^M)y^M + x_M^F \\ y^F = \sin(\theta_F^M)x^M + \cos(\theta_F^M)y^M + y_M^F \end{cases} \tag{22}$$

$${}^i\rho^F = x^F\cos({}^i\alpha^F) + y^F\sin({}^i\alpha^F) \tag{23}$$

$$\rho^M = x^M\cos(\alpha^M) + y^M\sin(\alpha^M) \tag{24}$$

By replacing Equations 22 in Equation 23, doing the necessary equivalences with System 24 and replacing some variables using Equations 19, we obtain the Systems 25 and 26, 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 $\tilde{\alpha}^M$.

$$\begin{cases} \rho^M = {}^i\rho^F - x_t\cos({}^i\alpha^F) - y_t\sin({}^i\alpha^F) \\ \alpha^M = {}^i\alpha^F - \theta_t + \pi/2 \end{cases} \tag{25}$$

$$\begin{cases} \rho^M = -{}^i\rho^F + x_t\cos({}^i\alpha^F) + y_t\sin({}^i\alpha^F) \\ \alpha^M = {}^i\alpha^F - \theta_t - \pi/2 \end{cases} \tag{26}$$

The sensor model is incorporated into the EKF through the **H** matrix (Equation 6), given by Equation 27[4]:

$$\mathbf{H} = \begin{pmatrix} -\cos{}^i\alpha^F & -\sin{}^i\alpha^F & 0 & \cdots & 1 & 0 & \cdots \\ 0 & 0 & -1 & \cdots & 0 & 1 & \cdots \end{pmatrix} \tag{27}$$

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

3.3 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 25, if $\tilde{\alpha}^M \geq 0$, or in Equations 26, 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.

4 Image processing

4.1 Detection of lines

Due to the choice of floor lines as landmarks, the technique adopted to identify them was the Hough transform (Gonzalez and Woodes, 2000). The

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

[3] 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.

[4] We only present the **H** matrix corresponding to System 25.

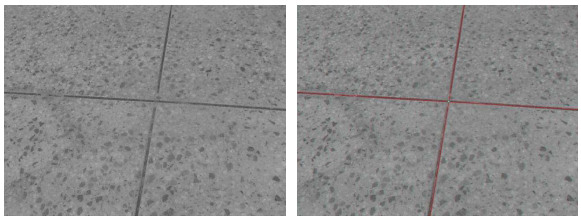

Figure 5: Detection of lines

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 18 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 shows a typical image of the floor and the lines detected by the Hough transform.

4.2 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 $\mathbf{A}$) between points in the floor plane (x, y) and points in the image plane (u, v):

$$s \cdot \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \mathbf{A} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{28}$$

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 6). After detecting the remarkable point in the image,

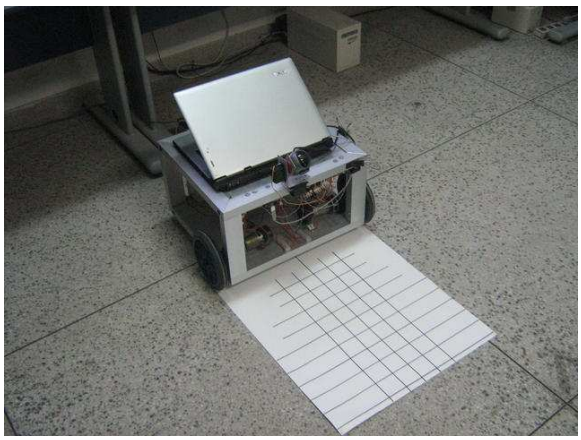

Figure 6: Calibration pattern

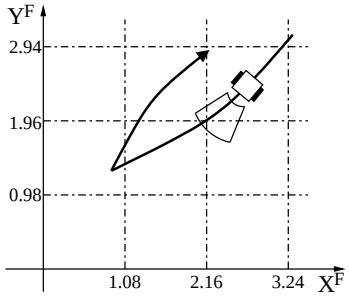

Figure 7: Executed trajectory

we have several correspondences between point coordinates in the mobile coordinate system M and in the image. Replacing these points in Equation 28, we obtain a linear system with which we can determine the 8 elements[5] of the homograpy matrix $\mathbf{A}$.

Once calculated the homography, for each detected line we do the following:

1. using the values of $(\tilde{\rho}, \tilde{\alpha})$ in the image obtained by the Hough transform, calculate two point belonging to the image line;
2. convert the coordinates of these two points to the mobile coordinate systema M;
3. determine $(\tilde{\rho}^M, \tilde{\alpha}^M)$ of the line that passes through these two points.

5 Results

The experiments were conducted using the robot Karel[6], a mobile reconfigurable platform constructed in our laboratoy (see Figure 6). It has two wheels driven by DC motors with diferential drive. Each motor has an optical encoder and a dedicated board based on a PIC microcontroller that carries out a local velocity control. The boards communicate with a computer through a CAN bus, receiving the desired velocities of the wheels and sending data from the encoders.

The computer that controls the robot is an ordinary notebook with an USB-CAN bridge and a color webcam connected to its USB bus. The camera captures 160×120 images (as the one in Figure 5) and each image is processed in 300ms.

We present here an experiment in which the robot executes a forward mouvement, turnes over itself and retournes close to the original position. In Figure 7 we indicate the position of the lines, the approximated trajectory of the robot and the comparative size of the camera's field of view. The initial position of the robot is roughly $(3.3, 3.1, 110^o)$, but it is not used in the SLAM algorithm: the robot assumes that its initial position is $(0, 0, 0^o)$

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

[6] An homage to Karel Čapek, the Czech writer that first publicly used the word "robot" in 1921.

Table 2: Experimental results

Actual		Calculated		Corrected	
ρ	α	ρ	α	ρ	α
1.08	0	1.88	108	1.13	1.7
2.16	0	0.82	108	2.19	1.3
3.24	0	0.11	110	3.26	0
0.98	90	2.57	23	0.97	87
1.96	90	1.24	22	1.95	87
2.94	90	0.20	22	2.93	88

Table 2 presents the detected landmarks after approximately 250 iteractions. The first two columns contain the actual values. The result of the SLAM algorithm is in the second two columns: the values are expressed in the coordinate system assuming that the initial robot position is $(0, 0, 0^o)$. These lines are drawn in Figure 8. Finally, the last two columns show the SLAM result but corrected by assuming the initial robot position as $(3.3, 3.1, 110^o)$, for better comparison.

6 Conclusions and Perspectives

The main contribution in this paper is the modeling of the optical sensor made in such a way that it permits using the parameters obtained by the image processing algorithm directly into the equations of the EKF, without intermediate phases of pose or distance calculation. The values presented in Table 2 demonstrate that the proposed approach can obtain good results even with a relatively small quantity of samples.

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. Another required improvement is to deal with line segments with finite length, incorparating this characteristics to the step of matching lines.

Thanks

André Macedo was financially supported by CNPq during this work.

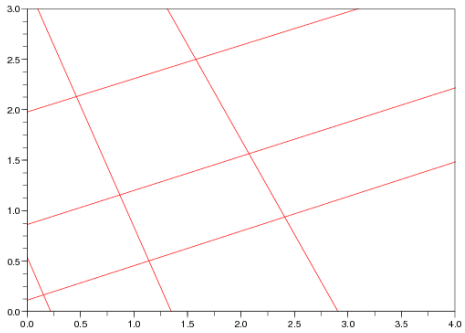

Figure 8: Lines calculated by EKF-SLAM

References

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

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

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

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

Folkesson, J., Jensfelt, P. and Christensen, H. I. (2005). Vision SLAM in the measurement subspace, *IEEE International Conference on Robotics and Automation*, pp. 30–35.

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

Gonzalez, R. C. and Woodes, R. E. (2000). *Processamento de Imagens Digitais*, Edgard Blucher. ISBN: 8521202644.

Jensfelt, P., Kragic, D., Folkesson, J. and Björkman, M. (2006). A framework for vision based bearing only 3D SLAM, *IEEE International Conference on Robotics and Automation*, pp. 1944– 1950.

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

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

Wu, J. and Zhang, H. (2007). Camera sensor model for visual SLAM, *IEEE Canadian Conference on Computer and Robot Vision*.

Zucchelli, M. and Košecká, J. (2001). Motion bias and structure distortion induced by calibration errors, *Technical report*, George Mason University. `http://citeseer.ist.psu.edu/kosecka01motion.html`.