2012 Brazilian Robotics Symposium and Latin American Robotics Symposium

A REAL TIME 6DOF VISUAL SLAM SYSTEM USING A MONOCULAR CAMERA

Andrés Díaz, Eduardo Caicedo
Escuela de Ing. Eléctica y Electrónica
Universidad del Valle
Santiago de Cali, Colombia
andres.a.diaz,eduardo.caicedo@correounivalle.edu.co

Lina Paz, Pedro Piniés
Instituto de Investigación en Ing. de Aragón
Universidad del Zaragoza
Zaragoza, España
linapaz,ppinies@unizar.es

***Abstract*—One of the most important properties that a robot must have in order to be considered autonomous is the ability to localize by itself in an unknown environment, using the information gathered by its sensors. The system uses a cheap web camera, carried by a mobile robot or by a person, while it builds a map and at the same time estimates its localization with respect to this map. To develop such system, problems such as camera calibration, corners detection, features parametrization, motion and measurement models, and the estimation of the joint state camera-features using the extended Kalman Filter, must be tackled.**

***Keywords*—SLAM; monocular camera; inverse depth; real time; 6DOF; EKF; radial distortion; active search;**

I. Introduction

Before carrying out tasks such as navigation, path planning, and objects and places recognition, a totally autonomous mobile robot must interpret the information obtained by its sensors and then estimate its position and the position of environmental features. The simultaneous localization and map building algorithms face both problems parallely [23], and they have been the focus of attention of the research community on mobile robotics during the last two decades.

This article presents the development of a visual SLAM system that works in real time and that navigates in structured environments, using only a monocular camera that makes simple trajectories with six degrees of freedom. The system is able to estimate the camera position and orientation, carried by a person or by a mobile platform, and to represent the trajectory that it makes, creating a three-dimentional map composed of the camera model and spatial points that represent object corners of the environment. Moreover, it can be adapted to different mobile platforms -terrestrial, aquatic, and aerial- because it is portable and has six degrees of freedom that reduce the motion restrictions. The system is of great importance when GPS information is not available and in applications where is not practical to carry heavy and bulky sensors in people and object tracking and mapping of environments in rescue operations.

Section II presents the most important SLAM researchers since it was first proposed at the end of the 1980s until the present. Section II-A defines the advantages of vision sensors, especially monocular cameras. Sections III and IV explain how the intrinsic parameters of the camera were calculated and how the radial distortion was corrected, respectively. Section V analyzes the corner detector based on the Harris technique and the occupancy criteria of the image regions. Section VI, VII, VIII and IX present the parametrization process with the inverse depth of the features, the constant velocity model, the position prediction of features in the image plane and the data association, respectively. Finally, we present the results and conclusions obtained in this work.

II. SIMULTANEOUS LOCALIZATION AND MAPPING

The probabilistic approach given to the simultaneous localization and mapping was initially introduced in 1986, in the *IEEE Robotics and Automation Conference*, held in San Francisco. In the following years were developed works that laid out the basis of SLAM (Smith and Cheeseman [23], Durrant-Whyte [10]). They proved the high degree of correlation between the different estimates of features positions and that this correlation grows with increasing observations. Smith, Self and Cheeseman [22] concluded that the estimated location of features are all correlated due to the common error in the estimation of robot position and that a vector composed of vehicle and features position is required to be updated after each observation. Leonard [15] and Manyika [11] proved techniques related to the Extended Kalman Filter, EKF. Implementations such as the ones developed by Castellanos [3] and Davison [7] proved the EKF in the building of small maps in SLAM systems with stereo vision, with the standard EKF and working in real time at 5 Hz. The system was able to build three-dimentional maps and to control a mobile robot. Jung and Lacroix [12] developed an autonomous system for mapping terrains using stereo vision as the only sensor and the standard EKF. Saez [21] presented a SLAM system with stereo vision for six degrees of freedom movements and indoor environments.

Several SLAM systems that use a monocular camera have proved to be viable in small environments; the most outstanding systems are the ones designed by Bailey [1], Kwok [13], Lemaire [14] and Davison [9]. Most of them are essentially EKF-SLAM systems and only change the initialization techniques and the kind of interest points extracted from the images (Harris corners, Shi and Tomasi corners, SIFT features,

978-0-7695-4906-4/12 $26.00 © 2012 IEEE
DOI 10.1109/SBR-LARS.2012.14

or any mixture of them). The works of Civera [4], Davison [9], Tully [24], Clemente [5] and Marzorati [17] show a tendency to use monocular cameras, inverse depth parametrization, and to perform in real time. The submapping techniques, such as the ones depevoped by Bosse [2], Leonard [16], Paz [19] and Piniés [20], allow the system to achive a performance in long trajectories.

A. *Visual SLAM*

Recently, the use of vision sensors has generated great interest in the research field of SLAM due to the large amount of texture information provided by these sensors of the objects found in a scene [8], [9]. Moreover, cameras are compact, accurate, and much cheaper than laser sensors. A monocular camera is a projective sensor that measures the orientation of features taken from an image. Given a sequence of images of the environment where the camera moves, it is posible to calculate the movements of the camera and the structure of the scene. To estimate the depth of the features, the moving camera must observe them repeatedly, and for each observation capture a ray passing through the feature and the camera optical center. The measured angle between the rays captured from different points of view is the feature parallax, which allows the system to estimate its depth. Using pure monocular vision, the map scale is not observable. However, by choosing appropiated values of the initial velocity and the covariance noise, the EKF-SLAM is able to obtain an approximate scale.

III. Camera Calibration

OpenCV was used to calculate the intrinsic parameters of the *Logitech Pro 9000* web camera. The calibration is based on the Zang and Brown methods that process the corners detected on a checkerboard, which are observed from different points of view. The data obtained are:

Focal Length	$[f_x \; f_y]$ = [265.34713 263.67229]
Principal point	$[c_x \; c_y]$ = [160.09206 108.20917]
Distortion coef.	$[k_1 \; k_2 \; p_1 \; p_2]$ = [-0.00035 -0.01892 -0.00211 0.00101]

The coefficients k_1 and k_2 are used in the model of radial distortion while p_1 and p_2 are used in the model of tangencial distortion. However, the last one is not significant so the correction is focused on the radial type.

IV. Correction of Radial Distortion

This distortion causes the movement of the points in radial way of the principal point of the image, such is shown in figure (1). The new position (x_{nd}, y_{nd}) of an ideal point with normalized coordinates (x_n, y_n) will be:

$$\begin{pmatrix} x_{nd} \\ y_{nd} \end{pmatrix} = (1 + k_1 r^2 + k_2 r^4) \begin{pmatrix} x_n \\ y_n \end{pmatrix} \qquad (1)$$

where x_n, and y_n are normalized coordinates and r is the radius, defined as:

$$x_n = \frac{x}{f} = \frac{u - c_x}{f} \qquad y_n = \frac{y}{f} = \frac{v - c_y}{f} \qquad (2)$$

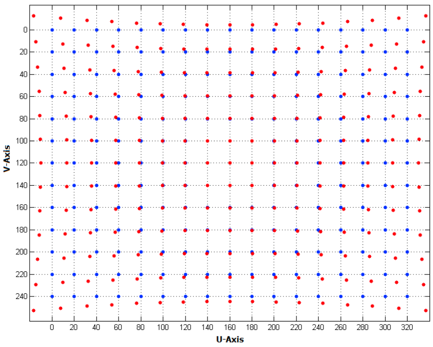


Fig. 1. Simulation of radial distortion for $k_1 = k_2 = 1.2$. In blue, ideal points. In red, points with radial distoriton

$$r^2 = x_n^2 + y_n^2 \qquad (3)$$

(u, v) are the coordinates of a point. The expression 1 allows us to obtain the normalized coordinates with distortion x_{nd} and y_{nd}, from the normalized coordinates without distortion. However, the reverse process is required. The expression that relates the radius r and the distorted radius r_d is:

$$r_d = r + k_1 r^3 + k_2 r^5 \qquad (4)$$

The correction of radial distortion consists of calculating the value r associated with r_d. Knowing r_d, the coordinates without distortion (u, v) can be calculated. This is done through the iterative method of Newton-Rapshon, which estimates a root of $f(r) = 0$ based on the information about the funtion $f(r)$ and its first derivative $df(r)/dt$, where $f(r)$ is:

$$f(r) = rd - r_{df} = r + k_1 r^3 + k_2 r^5 - r_{df} \qquad (5)$$

r_{df} is the value of r_d for which it is desired to know its equivalent r. The first approximation of the root of the equation $f(r)$, denoted r_0, is calculated by comparing r_{df} with the tabulated values of the radius with and without distortion r, r_d and assigning to r_0 the minimal difference:

$$min|r_d - rdf| \rightarrow r = r_0 \qquad (6)$$

where r_0 is the initial approximation of the root. The iterations are carried out using the expression:

$$r = r_0 + \frac{(r_{df} - r_{d0}))}{d(r_d)/dr \mid r_0} \qquad (7)$$

r_{d0} is the value of the funtion (4) evaluated at r_0. In the next iteration the estimated value of r is assigned to the initial approximation r_0. The iterations are made until the desired accuracy is obtained. With the value of r, the coordinates without radial distortion (u, v) are calculated:

$$u = \frac{u_d - c_x}{1 + k_1 r^2 + k_2 r^4} + cx \qquad (8)$$

$$v = \frac{v_d - c_y}{1 + k_1 r^2 + k_2 r^4} + cy \tag{9}$$

V. Corners Detector

The corner detection is used in image processing and in applications such as recogntion, object tracking, navigation, scene clasification and three-dimensional object modeling. It looks for unique features that can be parametrized and compared with points from other images. The Harris corner detector is one of the most used methods based on the image intensity. Harris implements the Plessey algorithm that uses the gradient of each pixel from the image. If the gradient in horizontal and vertical directions are both large, the pixel is considered as a corner.

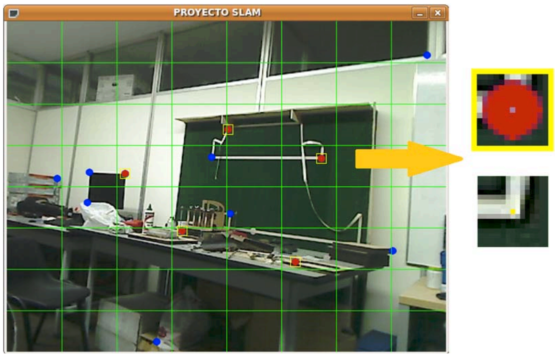


Fig. 2. Corners descriptors

The corners detected are stored together with a window of 15×15 pixels, where the central one is the corner detected with the Harris algorithm, such is shown in figure 2. It is desired that the features be of high quality and be distributed over the entire image to avoid feature agglomeration and incorrect associations. To this end, the image is split up on equal size regions and the detector is executed for each region independently. The minimun eigenvalue of the detected corners are evaluated: if it is higher than a threshold, the corner is of high quality and can be identified easily in subsequent iterations, otherwise it is discarded. Only features of high quality that are over empty regions and with a distance greater than 30 pixels of the other corners are used. Finally, when a region is empty, either because a feaure was deleted or because the feature passed from one region to another, a transition time must be taken to be occupied by a new corner.

VI. Parametrization with Inverse Depth

A significat limitation of the initial approaches of Davison [6], [9] and others was that the systems could only use features close to the camera and that had great parallax during the motion. This problem limited the robot navegation (or the camera navegation) to indoors. Montiel [18] proposed a technique to initialize features using the inverse distance between the feature and the camera where it was seen for first time. This technique allows the system to work with both close and distant features since the moment they were detected. The distant features are used to improve the motion estimation, acting initially as an orientation reference. These features are common in outdoor environments.

The feature parametrization with *inverse depth* involves these steps:

- The distorted coordinates (u_d, v_d) are obtined from the outstanding features of each image with the Harris detector.
- The coordinates without distortion (u, v) are obtained through the process described in section V.
- The back projection model is applied to the coordinates (u, v), obtaining normalized coordinates x_n and y_n:

$$\begin{pmatrix} x_n \\ y_n \\ 1 \end{pmatrix} = \begin{pmatrix} 1/f & 0 & -C_x/f \\ 0 & 1/f & -C_y/f \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \tag{10}$$

- The normalized coordinates give information about the ray h_c that passes through the optical center of the camera and the point in the world whose image coordinates are (u_d, v_d). The ray can be defined by two angles, the azimuth and the elevation angles:

$$\theta = tan^{-1} x_n \qquad \phi = tan^{-1} y_n \tag{11}$$

- The rays are in the camera frame. They are referenced to the world frame through transformations.
- The camera state is defined with six parameters:

$$y_i = \left[\begin{array}{cccc} x_{wc} & \theta_i & \phi_i & \rho_i \end{array}\right]^T \tag{12}$$

The vector $x_{wc} = [x_{wc}\ y_{wc}\ z_{wc}]^T$ corresponds to the camera position, in cartesian coordinates, from where the feaures were seen for first time, θ_i is the azimuth angle, ϕ_i is the elevation angle and $\rho_i = 1/d_i$ is the inverse distance between the camera position and the feature.

A. Addition of Features to the State Vector

The state vector stores the information of the camera and outstanding features:

$$x(k) = \left[\begin{array}{c} x_c(k) \\ Y(k) \end{array}\right] \tag{13}$$

The dynamics of the camera is modeled using a constant linear and angular velocity model that defines the state of the camera at the time k:

$$x_c(k) = \left[\begin{array}{cccc} r^w(k) & \psi^w(k) & v^w(k) & \omega^c(k) \end{array}\right]^T \tag{14}$$

where r^w corresponds to the three cartesian coordinates of the camera position, ψ^w is the camera orientation in *Roll, Pitch, Yaw* angles $[\psi_z,\ \psi_y,\ \psi_z]^T$, v^w is the linear velocity of the camera and ω^c is the angular velocity with respect to the camera frame. The vector $Y(k)$ contains the features information:

$$Y(k) = \left[\begin{array}{ccc} y_1(k) & \dots & y_n(k) \end{array} \right]^T \tag{15}$$

where each feature y_i was defined in equation (12).

VII. Motion Model

The camera is connected to a laptop and is carried by a mobile robot or by a person, recovering in real time the trajectory and building a map with well distributed features. The camera moves freely in three dimensions in an unknown structured environment. A constant linear and angular velocity model is used. The joint state camera-features, in the time k, is defined in the expression (13). The following transition function is used to pass from the state x_k to the state x_{k+1}:

$$x(k+1) = f(x(k), w(k)) \tag{16}$$

where f only changes the components of the camera state x_c because the features are assumed to be static. The vector $w(k)$ represents a zero-mean Gaussian noise with covariance Q that affects the linear and angular velocities of the camera to detect small changes in the model:

$$w(k) = \left[\begin{array}{c} \delta v^m(k) \\ \delta \omega^c(k) \end{array} \right] \tag{17}$$

The camera state x_c evolves according to the following equation:

$$\left[\begin{array}{c} r^w(k+1) \\ \psi^w(k+1) \\ v^w(k+1) \\ \omega^c(k+1) \end{array} \right] = \left[\begin{array}{c} r^w(k) + v^w(k+1)\Delta t \\ \psi^w(k) + E_c^w \omega^c(k+1)\Delta t \\ v^w(k) + \delta v^w(k) \\ \omega^c(k) + \delta \omega^c(k) \end{array} \right] \tag{18}$$

where E_c^w is a matrix that transforms angular velocities with respect to the camera frame to equivalent angular velocities in the world frame:

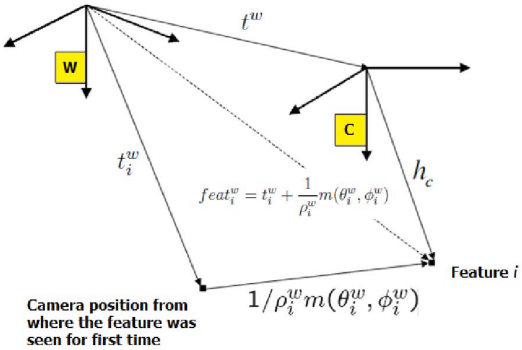
$$E_c^w = \left[\begin{array}{ccc} 1 & sin\psi_x \; tan\psi_y & cos\psi_x \; tan\psi_y \\ 0 & cos\psi_x & -sin\psi_x \\ 0 & sin\psi_x \; sec\psi_y & cos\psi_x \; sec\psi_y \end{array} \right] \tag{19}$$

where $[\psi_x \;\; \psi_y \;\; \psi_z]^T$ are the components of $\psi^w(k)$, correspondig to the *Roll-Pitch-Yaw* angles, respectively.

VIII. Prediction of the features position in the image plane

This process consists of predicting the features position in the next image, without making a new observation. Figure 3 provides a graphical representation of the vectors of the camera and features position.

The vector t_i^w represents the camera position from where a feature i is observed for first time. The vector t^w represents the current camera position, estimated with the motion model described in section VII. The vector m is a unitary vector with the same direction of ray h^c, stated when the feature was seen for first time. This vector is calculated using the azimuth and elevation angles of equation (11):



t^w
W
C
t_i^w
$feat_i^w = t_i^w + \frac{1}{\rho_i^w} m(\theta_i^w, \phi_i^w)$
h_c
Feature i
Camera position from where the feature was seen for first time
$1/\rho_i^w m(\theta_i^w, \phi_i^w)$

Fig. 3. Current and initial camera position used in the measurement prediction.

$$m(\theta_i^w, \phi_i^w) = \left[\begin{array}{c} cos(\theta_i^w)cos(\phi_i^w) \\ sin(\theta_i^w)cos(\phi_i^w) \\ sin(\phi_i^w) \end{array} \right] \tag{20}$$

The vector that describes the feature position with respect to the camera position from where the feature was seen for first time, t_i^w, is:

$$d_i^w = \frac{1}{\rho_i^w} m(\theta_i^w, \phi_i^w) \tag{21}$$

The sum of the vectors t_i^w and d_i^w is a vector referenced to the world frame that defines the feature position:

$$feat_i^w = t_i^w + \frac{1}{\rho_i^w} m(\theta_i^w, \phi_i^w) \tag{22}$$

The vector that goes from the current camera position t^w to the feature position, parametrized with the inverse depth $y_i = [r_i^w, \theta_i^w, \phi_i^w, \rho_i^w]^T$, is:

$$h^w = feat_i^w - t^w = (t_i^w + \frac{1}{\rho_i^w} m(\theta_i^w, \phi_i^w)) - t^w \tag{23}$$

where h^w is the vector expressed in the world frame. This ray is normalized using the inverse depth as is shown below:

$$h_\rho^w = \rho(t_i^w - t^w) + m(\theta_i^w, \phi_i^w) \tag{24}$$

The vector h^w has to be transformed to the camera frame, obtaining h^c. The equation used to predict the azimuth and elevation angles of a feature is based on the components of the vector h^c, $[h_x^c, h_y^c, h_z^c]$:

$$x_n = \frac{h_y^c}{h_x^c} \qquad y_n = \frac{h_z^c}{\sqrt{(h_x^c)^2 + (h_y^c)^2}} \tag{25}$$

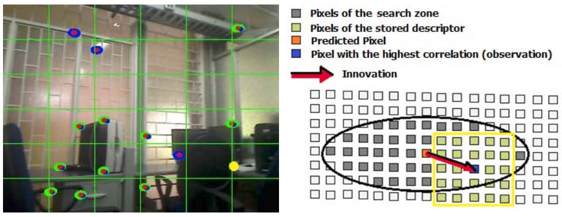
The coordinates (u, v) are calculated from the normalized coordinates x_n and y_n:

$$u = x_n f + c_x \qquad v = y_n f + c_y \tag{26}$$

where f is the focal length and (c_x, c_y) is the principal point.

IX. Data Association

The position in the image plane (u_i, v_i) where the features $feat_i$, for $i = 1, 2, 3, ..., n$, will be observed again, is predicted together with the innovation covariance matrix S_i. This matrix defines an elliptical zone of uncertainty where there is high probability to reobserve the feature. In this zone a correlation algorithm is executed, comparing the distribution of the digital levels of the pixels. The position that shows the strongest similarity will be taken as the equivalent point to the central pixel of a corner descriptor and will be the *observed position* of the feature from the new camera position.


Pixels of the search zone
Pixels of the stored descriptor
Predicted Pixel
Pixel with the highest correlation (observation)
Innovation

(a) Prediction of the feature position in the image plane (red points). The ellipses represent the prediction uncertainty: in green, with successful correlation; in blue, with failed correlation.

(b) Elliptical search zone and feature descriptor. The red arrow represents the innovation and allows the system to update the EKF filter.

Fig. 4. Predictions, search zones and innovations

Figure 4.a shows the predicted position in the image plane of the features and their uncertainty. Figure 4.b shows the difference (innovation, red arrow) between the predicted position of the feature (orange pixel) and the observed position (blue pixel), and it allows the system to make the correction step of the Extended Kalman Filter.

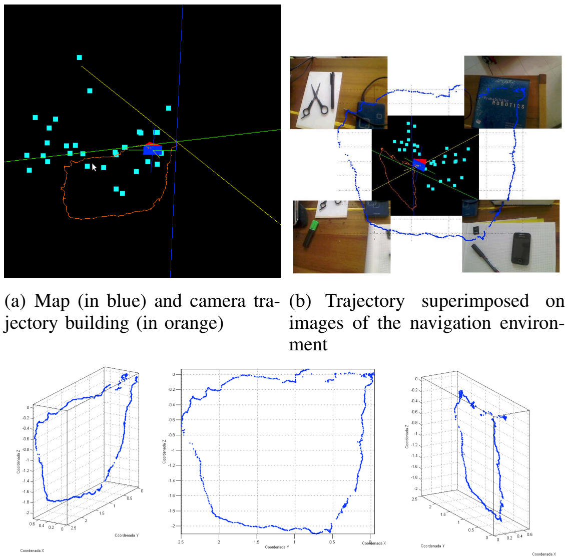
A joint compatibility test based on the Mahalanobis distance is carried out to deal with spurious associations between observations and predicted features that come from dynamic objects into the mapped environment.

X. Results

The visual SLAM system that was carried out builds in real time a three-dimensional representation of the environmental features, the camera and its trajectory. Figure 5.a shows this representation, using OpenGL. The system processes 18 frames per second, taked with a hand held camera. The features correspond to corners of objects on a desktop. Figure 5.b shows the trajectory scaled manually and overlaying the images of the environment. Figure 5.c shows the trajectory drawn in Matlab from three different points of view.

Figure 6.a shows how the inverse depth estimates evolve over time. The estimates converge to a given value after about 50 iterations. At steady state, the estimates don't vary significantly, which means that the map is consistent. As time passes, the parallax angles increase, yielding better estimates of the inverse depth, which is evidenced by a reduction in standard deviation (fig. 6.b).

Figure 7.a shows the total number of features in the map (in blue), the number of features that are predicted inside the image plane (in green), and the number of features that have been predicted inside the image plane and have been observed again (in red). The association process is made with all of the features predicted inside the image plane. The covariance matrix is updated with the information of the features reobserved. Figure 7.b shows the low computational cost of the association process (in green) obtained using an active search. This figure also shows the computational cost of the measurement prediction process (in blue), which increases with the number of features in the map. The occupancy criteria also contributes to reduce the total cost by decreasing the number of features and therefore the number of search zones. Moreover, this criteria avoids that features appear very close

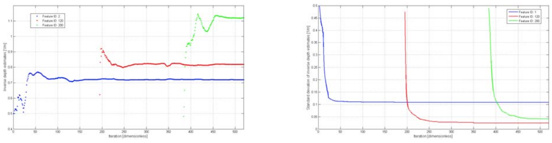


(a) Map (in blue) and camera trajectory building (in orange)

(b) Trajectory superimposed on images of the navigation environment

(c) Trajectory seen from different points of view

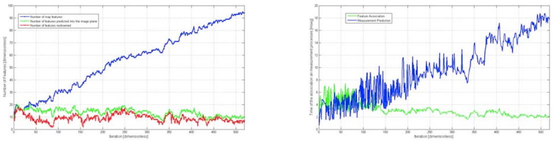
Fig. 5. SLAM performed in indoor environments



(a) Convergence of the inverse depth estimates

(b) Standard deviation of the inverse depth estimates

Fig. 6. Inverse depth estimates



(a) Number of features.

(b) Computational cost of the association process (correlation).

Fig. 7. Computational cost of the data association.

to one another, which would produce false associations. The occupancy criteria takes a constant time of 0.3mseg.

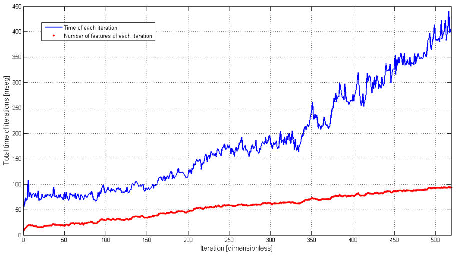


Fig. 8. Computational times (in blue) and number of features (in red) of each iteration

Figure 8 shows the computational times of each iteration of the SLAM algorithm and its quadratical dependence on the number of features. The algorithm that requieres the most time is the EKF update due to the computational effort of matrix operations. This is the reason why the number of features is limited to 40, achieving to process at least 10 images per second (at the critical point).

XI. Conclusions

We have presented a 6DOF real time visual SLAM system based on the work described in [9], using a camera as the only sensor. The core of the system relies on the well known incremental Extended Kalman Filter such that the positions of camera and a feature-based map can be estimated in real time.

The radial distortion causes significant displacements of the pixels from their ideal positions. Feature initialization and measurement prediction steps will accumulate inaccuracies, which will affect the feature association, and finally, the filter will diverge. During the measurement prediction step, a direct model is adopted to emulate the distortion effects, and during the feature initialization step, a reverse model is used to correct the distortion of the observed images.

The feature detection is made by regions of interest and an occupancy algorithm is executed for avoiding features agglomeration, achieving high quality corners that are well distributed. Features are mainly aimed to enable an estimate of the camera position and orientation. The Harris detector has a constant consumption of 12ms.

An active search of the feature descriptor is executed over an elliptical zone, defined by the innovation covariance matrix. By reducing the search to a small area, the computational cost of the association algorithm is significantly reduced.

The increase in the computational cost of the SLAM algorithm is caused by the EKF update. To achieve real time performance and without affecting the motion estimates and measurement predictions, the number of features is limited to 40. A future work could be the implementation of submapping techniques to achieve navigation and maps building of large environments.

References

[1] T. Bailey. Constrained initialisation for bearing-only SLAM. In *Proc. IEEE Int. Conf. on Robotics and Automation, (ICRA'03)*, volume 2, 2003.

[2] M. Bosse, P. M. Newman, J. J. Leonard, M. Soika, W. Feiten, and S. Teller. An atlas framework for scalable mapping. In *Proc. IEEE Int. Conf. Robotics and Automation*, pages 1899–1906, Taipei, Taiwan, 2003.

[3] J. A. Castellanos. *Mobile Robot Localization and Map Building: A Multisensor Fusion Approach.* PhD thesis, Dpto. de Informática e Ingeniería de Sistemas, University of Zaragoza, Spain, May 1998.

[4] J. Civera, A. J. Davison, and J. M. M. Montiel. Inverse depth parametrization for monocular SLAM. *IEEE Transactions on Robotics*, 24(5):932–945, October 2008.

[5] Laura Clemente, Andrew J. Davison, Ian D. Reid, Jose Neira, and J. D. Tardos. Mapping large loops with a single hand-held camera. In *Proc. Robotics: Science and Systems*, Atlanta, GA, USA, June 2007.

[6] A. J. Davison. Real-time simultaneous localisation and mapping with a single camera. In *Proc. Int. Conf. Computer Vision*, Nice, Oct 2003.

[7] A.J. Davison. *Mobile Robot Navigation using Active Vision.* PhD thesis, University of Oxford, 1998.

[8] Andrew J. Davison and David W. Murray. Simultaneous localization and map-building using active vision. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 24(7):865–880, 2002.

[9] Andrew J. Davison, Ian D. Reid, Nicholas D. Molton, and Oliver Stasse. Monoslam: Real-time single camera slam. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 29(6):1052–1067, June 2007.

[10] H. F. Durrant-Whyte. Uncertain geometry in robotics. *IEEE, Trans. Robotics and Automation*, pages 23–31, 1988.

[11] J.Manyika. *An Information-Theoretic Approach to Data Fusion and Sensor Management.* Ph.D., University of Oxford, 1993.

[12] I.K. Jung and S. Lacroix. High resolution terrain mapping using low altitude aerial stereo imagery. In *Proc. of the 9th Int. Conf. on Computer Vision*, pages 946–951, 2003.

[13] NM Kwok and G. Dissanayake. An efficient multiple hypothesis filter for bearing-only SLAM. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, (IROS'04).*, volume 1, 2004.

[14] T. Lemaire, S. Lacroix, and J. Sola. A practical 3D bearing-only SLAM algorithm. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS'05)*, pages 2449–2454, 2005.

[15] J.J. Leonard and H.F. Durrant-Whyte. *Directed Sonar Sensing for Mobile Robot Navigation.* Kluwer Academic Publishers, London, 1992.

[16] J.J. Leonard and P.M. Newman. Consistent, convergent and constant-time SLAM. In *Int. Joint Conf. Artificial Intelligence*, Acapulco, Mexico, August 2003.

[17] Daniele Marzorati, Matteo Matteucci, Davide Migliore, and Dominico Sorrenti. On the use of inverse scaling in monocular slam. In *IEEE International Conference on Robotics and Automation*, 2009.

[18] J. M. M. Montiel, Javier Civera, and Andrew J. Davison. Unified inverse depth parametrization for monocular SLAM. In *Proc. Robotics: Science and Systems*, Philadelphia, USA, August 2006.

[19] Lina M. Paz, Juan D. Tardós, and José Neira. Divide and Conquer: EKF SLAM in $O(n)$. *Acepted in Transactions on Robotics (in Print)*, 24(5), October 2008.

[20] Pedro Piniés and Juan D. Tardós. Large Scale SLAM Building Conditionally Independent Local Maps: Application to Monocular Vision. *Accepted in Transactions on Robotics (in print).*, 24(5), October 2008.

[21] J.M. Saez, F. Escolano, and A. Penalver. First Steps towards Stereo-based 6DOF SLAM for the Visually Impaired. In *Proc. IEEE Computer Society Conf. on Computer Vision and Pattern Recognition (CVPR'05)-Workshops-Volume 03*. IEEE Computer Society Washington, DC, USA, 2005.

[22] R. Smith, M. Self, and P. Cheeseman. Estimating uncertain spatial relationships in robotics. In J.F. Lemmer and L.N. Kanal, editors, *Uncertainty in Artificial Intelligence 2*, pages 435–461. Elsevier Science Pub., 1988.

[23] R. C. Smith and P. Cheeseman. On the representation and estimation of spatial uncertainty. *Int. J. Robotics Research*, 5(4):56–68, 1986.

[24] Stephen Tully, Hyungpil Moon, George Kantor, and Howie Choset. Iterated filters for bearing-only slam. In *IEEE International Conference on Robotics and Automation*, 2008.