Simultaneous Localization and Mapping for Tracked Wheel Robots Combining Monocular and Stereo Vision

Received 10th October 2012; accepted 22nd November 2012.

Filipe Jesus, Rodrigo Ventura

Abstract:
This paper addresses an online 6D SLAM method for a tracked wheel robot in an unknown and unstructured environment. While the robot pose is represented by its position and orientation over a 3D space, the environment is mapped with natural landmarks in the same space, autonomously collected using visual data from feature detectors. The observation model employs opportunistically features detected from either monocular and stereo vision. These features are represented using an inverse depth parametrization. The motion model uses odometry readings from motor encoders and orientation changes measured with an IMU. A dimensional-bounded EKF (DBEKF) is introduced here, that keeps the dimension of the state bounded. A new landmark classifier using a Temporal Difference Learning methodology is used to identify undesired landmarks from the state. By forcing an upper bound to the number of landmarks in the EKF state, the computational complexity is reduced to up to a constant while not compromising its integrity. All experimental work was done using real data from RAPOSA-NG, a tracked wheel robot developed for Search and Rescue missions.

Keywords: *simultaneous localisation and mapping, extended Kalman filter, feature detector, inverse depth parametrization, landmark evaluation, temporal difference learning*

1. Introduction

SLAM is one of the most promising fields in robotics, aiming at tracking the location of a robot and map its surroundings using external sensor data. EKF, when applied to SLAM, proves to work reasonably well with distinct, well-matched observations and a small state for estimation. However, insertion of new data over time without removal increases EKF complexity, hindering its scalability over time.

By memory-bounding the state, EKF complexity is assured to grow with time and, using proper classifiers, undesired features are automatically removed. A side effect from this removal procedure is that the map becomes visually sparse, but as long as it suffices the SLAM needs for stable predictions, one can use proper techniques to acquire visually more compelling maps.

This work was implemented on RAPOSA-NG, a tracked wheel robot for SaR missions (Figure 1). This robot has an adjustable frontal body, where both the IMU and camera are located. Since most SaR robots perform motion within irregular terrains, this paper focuses on estimating both its position and attitude in a 3D euclidean space. It uses an IMU to measure orientation changes and encodes odometry from both wheels to measure translation changes over time. Also, this paper introduces an elegant way to insert landmarks in the state from both monocular and stereo visualizations using the inverse depth parametrization. All landmarks are treated the same way, regardless of their origin. While monocular observations have no depth information, depth can be estimated through parallax changes over time. Stereo observations, on the other hand, provide depth, which not only allows for more accurate maps with fewer observations, but also solves the problem of map scale, common to most monocular SLAM techniques[1].

J.J. Leonard and H.F. Durrant-Whyte introduced *Simultaneous Localization and Mapping* (SLAM) terminology to the robotics field and the concept of *geometric beacons*: natural landmarks present in the environment that can be reliably observed, as well as described in terms of a concise geometric parametrization (referred in this paper simply as *landmarks*) [5]. Geometric beacons can be acquired with many different types of sensors, as long as the aforementioned qualities are maintained.

Davison *et al.* proposed a real-time algorithm which recovers the location of a monocular camera over time using SLAM with a random walk motion model [3]. However, feature initialization requires more than one observation, so that a proper triangulation for an initial depth estimate can be done. Also, it needs to acquire landmarks with known depth for scale initialization. Thus, Civera and Davison presented an *inverse depth parametrization* that represents landmarks uncertainty with more accuracy than the standard XYZ parametrization [2]. The increase of accuracy can be justified by the higher degree of linearity of the inverse depth parametrization over XYZ parametrization. However, this representation over-parametrizes each landmark (6 instead of the 3 components of XYZ), increasing the EKF complexity even further. They also defined a landmark classifier that removes 50% of all predicted landmarks that should be visible but are not detected by any feature detector. This approach leads to the landmark classifier introduced in this paper. The usage of a random walk model assumes a well behaved motion with smooth linear and angular velocities over time, a condition that often fails for tracked wheel robots in non-planar grounds (e.g., stair climbing).

Pinies *et al.* included the usage of an IMU to the vision SLAM with inverse depth parametrization [7]. In fact, having orientation changes measured with an IMU, the uncertainty of the camera location is reduced. However, it does not decrease the uncertainty when only linear motion is observed, which leads to the need of odometry inclusion presented in this paper. As for the map scale problem, in order to solve it, this paper extends the inverse depth parametrization usage for stereo vision as well.

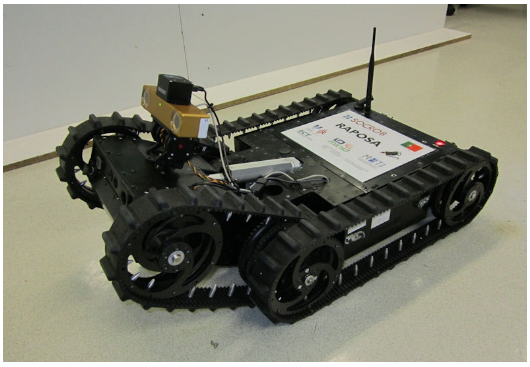


Fig. 1. RAPOSA-NG.

2. State and Model Definitions

We address the problem of simultaneously estimating the robot pose and landmark positions (SLAM) using a probabilistic approach based on the Extended Kalman Filter. The state of this filter encompasses both the robot pose and the landmark positions. The motion model is used in the predict step, while the observation model is used in the update step, as usual [5].

2.1. State Representation

The SLAM algorithm estimates the pose of the camera frame with respect to a world frame. The camera frame is considered to be located at the midpoint between the two stereo cameras.

The EKF state is defined as

$$s_t = \left(r_t^{\mathrm{T}}\, q_t^{\mathrm{T}}\, {y_1}^{\mathrm{T}}\, \cdots\, {y_n}^{\mathrm{T}}\right)^{\mathrm{T}}, \tag{1}$$

where vector r_t and unit quaternion q_t represent the camera position and attitude (*i.e.*, pose of the camera frame) in the world frame at time t. All y_i from $i \in \{0, \ldots, n\}$ correspond to 3D point landmarks represented using an inverse depth parametrization,

$$y_i = (X_i^o\, Y_i^o\, Z_i^o\, \theta_i\, \phi_i\, p_i)^{\mathrm{T}} \tag{2}$$

where $(X_i^o, Y_i^o, Z_i^o)^{\mathrm{T}}$ is an arbitrary point in XYZ, θ_i and ϕ_i are the azimuth and elevation of the semi-ray that crosses both this point and the landmark in the world frame and p_i is the inverse of the distance between $(X_i^o, Y_i^o, Z_i^o)^{\mathrm{T}}$ and the landmark. This parametrization is capable of representing any landmark in space. One can compute y_i^{XYZ}, the landmark position in XYZ, by

$$y_i^{XYZ} = o_i + \frac{1}{p_i} m_i \tag{3}$$

where

$$m_i = \begin{pmatrix} \cos\phi_i \, \sin\theta_i \\ -\sin\phi_i \\ \cos\phi_i \, \cos\theta_i \end{pmatrix}. \tag{4}$$

Usually, o_i corresponds to the focal point of the camera when the landmark was first observed. While this parametrization has more degrees of freedom than necessary (6 instead of 3), it has interesting properties regarding linearity over EKF [2].

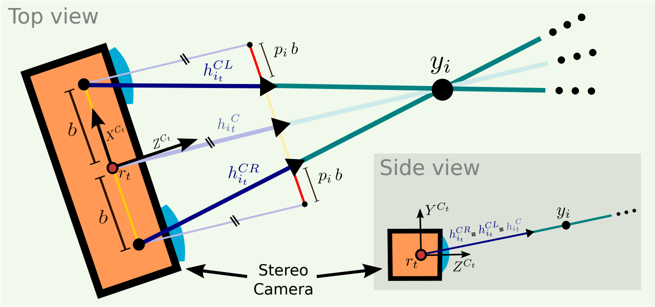


Fig. 2. Horizontal stereo camera representation.

2.2. Observation Model

The observation model describes how each feature is perceived by the sensors. Each feature can be perceived in either stereo by both cameras, or mono by only one of the cameras. In any case, the observation model provides the expected pixel position of each feature, for each one of the cameras. This computation is performed in two steps:

1) For each landmark y_i in state, compute a directional vector in the camera frame that points from the camera position to the landmark position as

$$h_{it}^{C} = R_{q_t^*}\left(p_i\left(o_i - r_t\right) + m_i\right) \tag{5}$$

where $R_{q_t^*}$ is the rotation matrix that rotates the world frame to the camera frame (from q_t conjugate, q_t^*).

2) Using the Pinhole Camera Model, for each landmark i situated in front of the camera, project the landmark position, along that directional vector, to each one of the image planes of the cameras. This model assumes a single camera with no lenses, nor aperture radius. It does not model any type of image distortion or blur present in every camera. For this paper, information retrieved for observation analysis passed through a correction process using camera's proprietary software before being used by the EKF, returning an undistorted image with known intrinsic parameters, while maintaining a wide visual range. This software also rectifies each pair of stereo images, by projecting them to a common image plane [6]. If no software correction is available, distortion can be compensated with proper models using distortion parameters intrinsic to the camera, retrieved through calibration methods. An horizontal stereo camera is used in this paper to acquire image data from two different sources. Since all images are properly rectified, a given pair of features from both cameras only correspond to the same landmark if they both share the same horizontal axis. This rectification also results in a pair of images with the same size and intrinsic parameters. A set of coordinate frames, $(X^{CL_t}, Y^{CL_t}, Z^{CL_t})$ and $(X^{CR_t}, Y^{CR_t}, Z^{CR_t})$, are defined for the left and right camera, respectively. After the rectification process, both right and left camera frames share the same orientation as the camera frame and are displaced by b along the X^{C_t} axis. To simplify the formalism, a parameter k^{LR} is introduced, where

$$k^{LR} = \begin{cases} 0, & \text{from left (L) camera} \\ 1, & \text{from right (R) camera} \end{cases} \tag{6}$$

Both directional vectors from left and right cameras, ${h_{i}}_{t}^{CL}$ and ${h_{i}}_{t}^{CR}$, can easily computed from ${h_{i}}_{t}^{C}$,

$$ {h_{i}}_{t}^{CL/CR} = {h_{i}}_{t}^{C} - (-1)^{k^{LR}}\, h'_{i_t} \tag{7} $$

where

$$ h'_{i_t} = p_i\,\bar{b} \qquad \text{and} \qquad \bar{b} = \begin{pmatrix} b\;0\;0 \end{pmatrix}^{\mathrm{T}}. \tag{8} $$

With the pinhole model, one can either model an observation from both cameras,

$$ z_{i_t}^{\text{Stereo}} = \begin{pmatrix} z^{L}_{u_{i\,t}} \\ z^{R}_{u_{i\,t}} \\ z_{vit} \end{pmatrix} = \begin{pmatrix} z_{u_{i\,t}} \\ z_{u_{i\,t}} \\ z_{vit} \end{pmatrix} - \frac{f_C\, m_u\, p_i}{{h_{z}}_{i\ t}^{C}} \begin{pmatrix} -b \\ b \\ 0 \end{pmatrix}, \tag{9} $$

or from left or right camera only,

$$ z_{i_t}^{\text{Left}} = \begin{pmatrix} z^{L}_{u_{i\,t}} \\ z_{vit} \end{pmatrix} \qquad z_{i_t}^{\text{Right}} = \begin{pmatrix} z^{R}_{u_{i\,t}} \\ z_{vit} \end{pmatrix}. \tag{10} $$

2.3. Motion Model

The motion model employs the odometry readings to estimate linear movement along the body frame and IMU readings to estimate incremental rotations of the robot. The frame transformations among the defined frames propagate these movement measurements, as well as their uncertainties (as covariances), to the camera frame.

Three different frames are defined for the robot for each iteration t.

1) **Camera frame**, representing the camera pose at iteration t, as defined in subsection 2.1;
2) **Body frame**, representing the robot body pose at iteration t. In RAPOSA-NG, the transformation between this frame and the previous one depends solely on the angle of the frontal body of the robot;
3) **IMU frame,** representing the IMU pose at instant t. The angular velocity ω^{imu} can be modelled through the IMU gyroscopes. In RAPOSA-NG, this frame is attached to the frontal body.

The motion model employs the odometry readings to estimate linear movement along the body frame and IMU readings to estimate incremental rotations of the robot. In RAPOSA-NG the IMU is mounted on the frontal body, and thus the IMU frame shares the same attitude as the camera frame.

From odometry the robot obtains linear movement along the body frame, by averaging the velocity of both tracks. Differential movement is discarded, since tracked wheel robots provide unreliable angular movement measurements from odometry. From the IMU, the robot obtains attitude changes. These changes are modeled as an angular velocity ω^{imu}, defined by

$$ \omega_t^{\text{imu}} = \omega_t^{\text{gyro}} + \omega_t^{\text{bias}} + \omega_t^{\epsilon}, \tag{11} $$

where ω_t^{gyro} is the angular velocity retrieved from the IMU, ω_t^{bias} is the bias error normally associated with most IMUs (if the IMU uses optical or MEMS technology and is calibrated, it can be assumed no ω_t^{bias} for some period of time [10]) and ω_t^{ϵ} is a normally distributed error with zero-mean. From ω^{imu} one can obtain an incremental rotation q_t^{imu} using a zeroth-order integrator as described in [9].

The frame transformations among the frames defined in section 2.1 propagate these movement measurements, as well as their uncertainties (as covariances), to the camera frame.

2.4. Feature Initialization

Over time, visual observations are made and new landmarks are attached to the state from observed visual features. Many criteria can be used to establish when new landmarks should be inserted and how many. For instance, one can add a new landmark every time a visual feature is observed that does not match any landmark in the state. However, doing so is computationally ineffective as it fills the state in a short time if no landmark removal procedure is performed.

Assuming the usage of the stereo camera, one can acquire monocular features either from the left or from the right camera. Also, some features acquired from both cameras correspond to the same landmark, resulting in a stereo feature. Depending on whether the new landmark in state results from a monocular feature or from a stereo feature, two different initializations are introduced:

1) **From a monocular observation:** If a new landmark y_{n+1} is to be appended to the state from a feature detected by only one of the cameras, first a directional vector for the respective camera frame is computed using the Pinhole Camera Model. The $(X_i^o, Y_i^o, Z_i^o)^{\mathrm{T}}$ is set to the respective camera center, and θ_i and ϕ_i are set to the azimuth and elevation of the semi-ray that crosses this point and the feature location in image plane.

$$ {h_{n}}^{CL/CR}_{\ \ t} = \begin{pmatrix} \frac{1}{f_C\, m_u}\left(c_u - z^{L/R}_{u_{n_t}}\right) \\ \frac{1}{f_C\, m_v}\left(c_v - z_{v_{n_t}}\right) \\ 1 \end{pmatrix} \tag{12} $$

Having R_{q_t} from q_t in state and knowing that both left and right camera frames share the same orientation from the robot state, the directional vector can be related to the world frame by

$$ h_{n_t} = R_{q_t}\, {h_{n}}^{CL/CR}_{\ \ t} \tag{13} $$

and

$$ \begin{pmatrix} o_n \\ \theta_n \\ \phi_n \\ p_n \end{pmatrix} = \begin{pmatrix} r_t + (-1)^{k^{LR}}\, R_{q_t}\, \bar{b} \\ \arctan(h_{xnt}, h_{znt}) \\ \arctan(-h_{y_{nt}}, |h_{xznt}|) \\ p_0 \end{pmatrix} \tag{14} $$

where

$$ |h_{xznt}| = \sqrt{(h_{xnt})^2 + (h_{xnt})^2} \tag{15} $$

It is impossible to gain depth information from just one observation, thus an initial arbitrary value p_0 serves

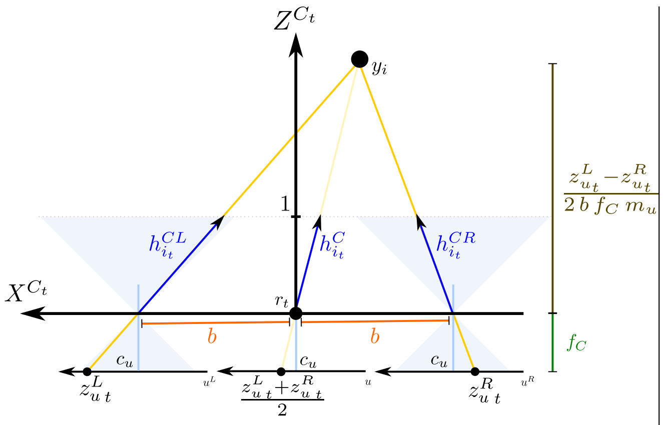


Fig. 3. Top view of camera frame for stereo vision of a landmark y_i.

as an initial estimation for the inverse depth given enough uncertainty. This parametrization is approximately linear along the corresponding semi-ray, allowing the EKF to sustain and correct large errors for the depth estimation.

2) **From a stereo observation:** Using epipolar geometry, one can compute the landmark parameters for the inverse depth parametrization, but including a measured depth, rather than a default value. The $(X_i^o, Y_i^o, Z_i^o)^{\mathrm{T}}$ are set to the camera frame, θ_i and ϕ_i are set to the azimuth and elevation of the semi-ray that crosses the camera frame and the landmark XYZ position extracted from stereo.

If the landmark comes from a stereo pair of features, using epipolar geometry it is possible to obtain the first a directional vector from the camera frame is computed using the Pinhole Camera Model. Since both left and right camera frames share the same distance from the camera frame but in opposite directions, the directional vector can be calculated as

$$h^C_{n_t} = \begin{pmatrix} \frac{1}{f_C\,m_u}\left(c_u - \frac{1}{2}(z^L_{u_{n_t}} + z^R_{u_{n_t}})\right) \\ \frac{1}{f_C\,m_v}\left(c_v - z_{v_{n_t}}\right) \\ 1 \end{pmatrix}. \qquad (16)$$

In the same fashion as equation (13),

$$h_{n_t} = R_{q_t}\, h^C_{n_t} \qquad (17)$$

and

$$\begin{pmatrix} o_n \\ \theta_n \\ \phi_n \\ p_n \end{pmatrix} = \begin{pmatrix} r_t \\ \arctan(h_{x\,nt}, h_{z\,nt}) \\ \arctan(-h_{y_{nt}}, |h_{xz\,nt}|) \\ p_e \end{pmatrix}, \qquad (18)$$

where p_e can be computed using epipolar geometry [4],

$$p_e = \frac{z^L_{u_{n_t}} - z^R_{u_{n_t}}}{2\,b\,f_C\,m_u\,|h^C_{n_t}|}. \qquad (19)$$

2.5. Feature Detector and Descriptor

For landmark detection and matching, this work uses ORB (Oriented FAST and Rotated BRIEF), a rotation-only invariant feature detector and descriptor [8]. Although less reliable than SURF [1] and no scale invariant, it still behaves with great accuracy for small scale changes. While both methods have good performances, ORB is faster but less accurate than SURF regarding scale changes. However, if those changes are considered small, ORB accuracy suffices the SLAM needs. ORB computes FAST features with added orientation information from the intensity centroid. For the descriptors, it uses BRIEF descriptors, calculated from binary intensity tests and rotated using the orientation assigned.

3. Dimensional-Bounded EKF (DBEKF)

One of the major problems regarding the Extended Kalman Filter is the fact that its computational complexity increases over a quadratic order with the number of landmarks.

This paper introduces a *Dimensional-Bounded Extended Kalman Filter* (DBEKF) which equips EKF with criteria for landmarks insertion and removal. For that, a new Landmark Classifier has to be introduced first. Figure 4 shows the proposed DBEKF architecture.

3.1. Landmark Classifier

For the DBEKF, a landmark y_i is said to be *visible* in state s_t, $y_i \in \mathbf{V_{s_t}}$, if it is observable from state s_t. Also, y_i is *detected* at iteration t, $y_i \in \mathbf{D_t}$, if the feature detector points out a corresponding feature. In a perfect scenario without any physical occlusions, $\mathbf{V_{s_t}} = \mathbf{D_t}$, that is, if the landmark is *visible* it should be *detected*. However, feature detectors are prone to error: descriptors can fail to point out some correspondences and miss features from being detected. These inaccuracies are crucial to classify each landmark's usability in state. Since it is assumed that no landmarks have physical occlusions, a *visible* but not *detected* landmark can only represent a failed match. In these cases, failed matches promote the corresponding landmark to be removed from the state.

A *Temporal Difference Learning* approach is used to predict a measure of the *utility*, u^i_t, of each landmark at iteration t:

$$u^i_t = \begin{cases} G\,u^i_{t-1} + (1-G)\,\mathbf{1}^i_{\mathbf{D_t}} & \text{if} \quad y_i \in \mathbf{V_{s_t}} \\ u^i_{t-1} & \text{otherwise.} \end{cases} \qquad (20)$$

where G is a arbitrary weight set by the user and the indicator function ($G \in [0,1]$), $\mathbf{1}^i_{\mathbf{D_{s_t}}}$, is defined for detectability,

$$\mathbf{1}^i_{\mathbf{D_{s_t}}} = \begin{cases} 1 & \text{if} \quad y_i \in \mathbf{D_{s_t}} \\ 0 & \text{else} \end{cases}. \qquad (21)$$

The lower the utility of a landmark, more likely it is to be removed from the state. The initial value for utility is $u^i_0 = 1$. Due to (20) and (21), $u^i_t \in [0,1]$.

3.2. Landmark Removal

Assuming M_l as the maximum number of landmarks imposed by the user to the DBEKF, the Landmark removal procedure is composed of three criteria:

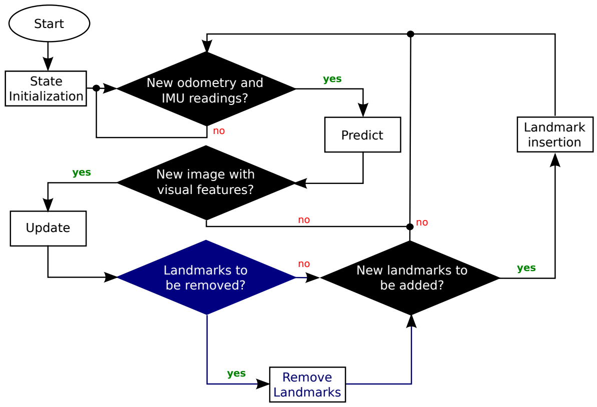


Fig. 4. DBEKF flowchart.

1) **Utility Threshold:** when u_t^i reaches a value below $T \in [0, 1]$ at iteration t, it is discarded from the state;
2) **Negative Inverse Depth:** all features with negative inverse depth (*e.g.*, due to a feature mismatch) are automatically discarded from the state;
3) **Emergency Removal**: if the amount of matched landmarks is below a threshold m_l, the oldest landmarks are removed from the state as such to leave room for the new ones.

4. Results

All experimental results presented in this paper are from two different datasets made with RAPOSA-NG. Each dataset is the result of a ROS[2] log file recording during operation. It contains odometry readings from left track, right track and inclination arm position at 10 Hz each, angular velocity readings from IMU at 30 Hz, rectified images from both cameras at 15 Hz, and all features retrieved from image readings using feature detector ORB at 15 Hz. Unless otherwise stated, all tests performed with DBEKF have an upper bound of $M_l = 60$ landmarks in state, an utility weight factor of $G = 0.8$, an utility threshold of $T = 0.01$ and a minimal number of matched landmarks per observation of $m_l = 10$. Results may slightly vary for the same dataset in different runs, since the log file is played in real time (as well as the SLAM algorithm).

The datasets are denoted here A and B, both illustrated in figure 5. In dataset A, RAPOSA-NG performs a near-squared trip of 3×3 meters in a soccer field full of newspaper pages, wooden planks and other sort of objects, simulating debris. Dataset B comprises RAPOSA-NG climbing up and down a set of stairs. The stairs set has 0.62 meters of height.

By upper limiting the number of landmarks in state by a value $M_{\text{landmarks}}$, EKF computational complexity becomes upper bounded as well. If there are enough observations to grant $M_{\text{landmarks}}$ in state for estimation at every iteration, the computational load should be constant for all time. This situation happens to all experiments presented on this thesis.

Figure 6 shows the time duration and number of removed features per DBEKF iteration with both monocular and stereo observations from datasets "A" and "B", respectively. As expected, the computational complexity is near constant for all time during both experiments, presenting some peaks and fluctuations due to new landmark initialization, the number of feature observations per update and other processing tasks unrelated to this software. Regarding dataset "A", one can notice some time intervals where a larger number of features are removed. These intervals happen when the robot finishes rotating 90 degrees

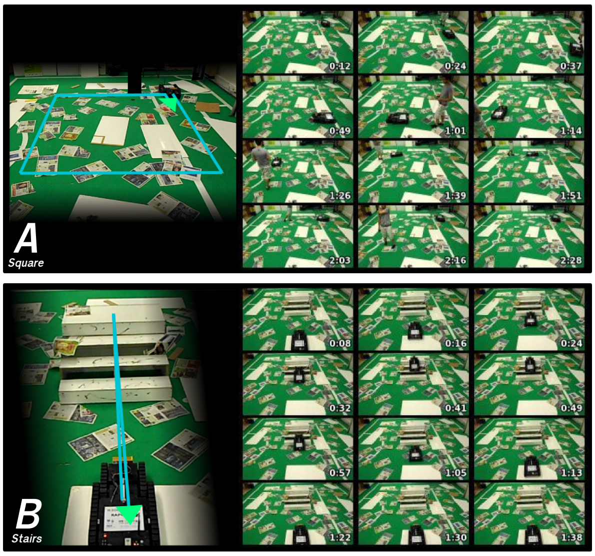
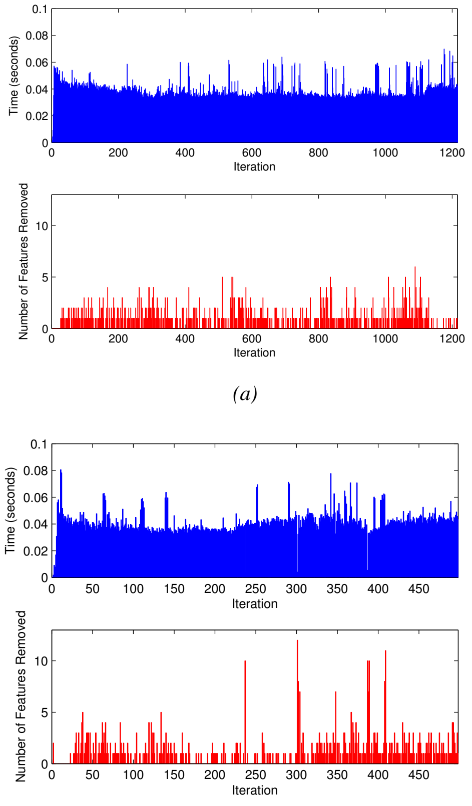


Fig. 5. Datasets A (top) and B (bottom). The indicated line represents the trajectory travelled by RAPOSA-NG during the experiment, in the direction pointed by the arrowhead.



(a)

(b)

Fig. 6. Time duration (top) and number of removed features (bottom) per DBEKF iteration for datasets "A" (a) and "B" (b) with both monocular and stereo observations.

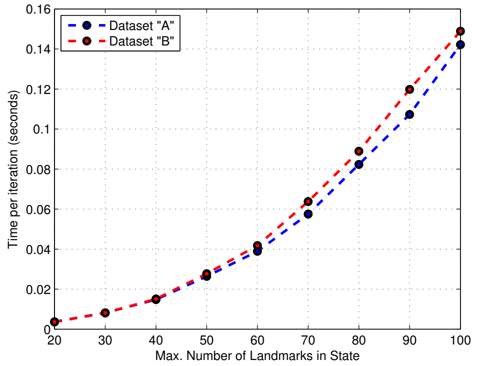


Fig. 7. Average time per iteration for different upper bounds in DBEKF for datasets "A" and "B".

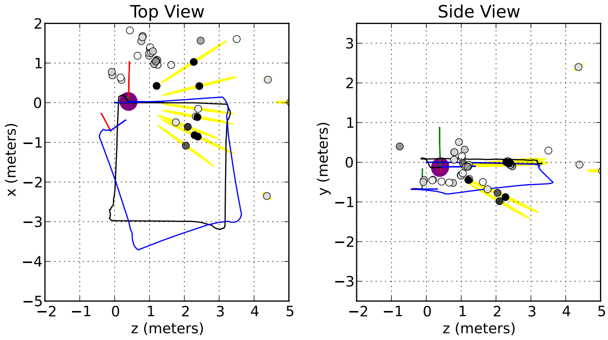


Fig. 8. SLAM results using DBEKF with dataset A. The trajectories shown are: in blue/grey is the camera trajectory with only odometry and IMU, while the black one used SLAM estimation. Covariance for the final position is also shown, as well as the final covariances of the landmarks in state.

and faces a new plane of observations, requiring space for new landmarks in state. It then discards older landmarks in order to acquire new ones. Dataset "B" presents some peaks regarding the number of feature observations as well, where the robot experienced drastic observation changes due to the rough movement of RAPOSA when finishing climbing up or starting to climb down the stairs.

From the average time reading in both experiments, it is clear that the SLAM algorithm fully performs in real time. However, it does not take into account the time needed for feature acquisition using feature detectors such as SURF or ORB.

Figure 7 shows the average time per iteration for different upper bounds in DBEKF with both datasets. As expected, despite the dataset, the time per iteration rises near a cubic order. It is important that, although the computational power becomes constant, a reasonable upper bound is chosen to avoid large time intervals for the EKF that can otherwise reveal linearity problems.

Figures 8 and 9 present the trajectories estimated by DBEKF for datasets A and B, in comparison with using odometry alone. Note that on both cases the algorithm was capable of correcting the errors induced by odometry, being able to return to the initial position of each dataset (note that no closure detection algorithm was employed).

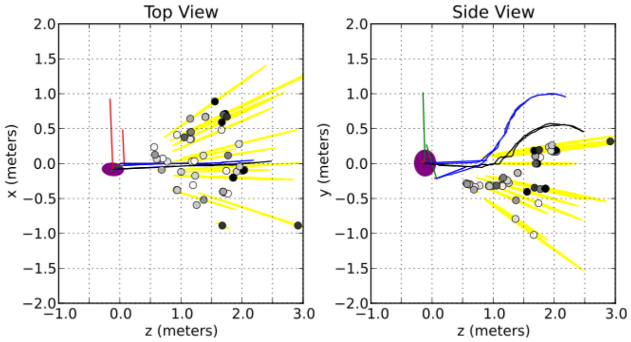
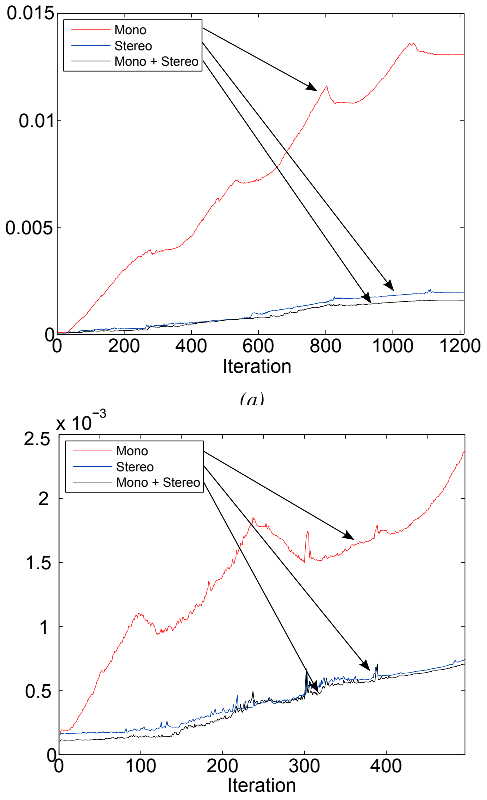


Fig. 9. SLAM results using DBEKF with dataset B. Refer to figure 8 for graphical notation.



(a)

(b)

Fig. 10. Evolution of the pose covariance from SLAM using DBEKF for dataset A (a) and B (b) along time. The different traces correspond to the use of mono features only, of stereo features only, and of all features.

In Figure 10 a comparison between several configurations is made, in terms of estimated uncertainty of the robot pose. This uncertainty is measured in terms of the trace of the position estimation covariance matrix (the corresponding 3×3 submatrix of the state covariance). The configurations are: use of monocular features only, use of stereo features only, and use of all features. This result shows that the combination of mono and stereo features outperform any other configuration.

5. Conclusions

The usage of both cameras as a stereo vision decreases the uncertainty from all landmarks and allows a better initialization for the SLAM algorithm, but the lack of stereo features may offer some problems to the SLAM problem if no other type of observations are used. In this work, both stereo and monocular features are used as observations, and as such one can use monocular information without worrying with the map scale problem (referred in Section 1) as long as stereo observations are available. From the presented results, it is clear that using both monocular and stereo observations in the way introduced here increases the overall quality of SLAM over monocular only or stereo only observations.

Although the usage of the *Extended Kalman Filter* (EKF) has been extensively used to solve the SLAM problem, its computational complexity grows unbounded with the number of landmarks. This paper showed that, with DBEKF, one can achieve good estimations with constant complexity when removing landmarks from state according to an utility evaluation criterion.

Notes

[1] Assuming no *a priori* initialization of map scale.

[2] http://www.ros.org

AUTHORS
Filipe Jesus* – Institute for Systems and Robotics, Instituto Superior Técnico, Av. Rovisco Pais, 1, Lisbon, Portugal, e-mail: filipejesus@ist.utl.pt.
Rodrigo Ventura – Institute for Systems and Robotics, Instituto Superior Técnico, Av. Rovisco Pais, 1, Lisbon, Portugal, e-mail: rodrigo.ventura@isr.ist.utl.pt.

*Corresponding author

Acknowledgements

This work was supported by the FCT projects [PEst-OE/EEI/LA0009/2011] and [PTDC/EIA-CCO/113257/2009].

References

[1] H. Bay, A. Ess, T. Tuytelaars, L. Van Gool, “Speeded-up robust features (SURF)”, *Comput. Vis. Image Underst.*, 110(3), 2008, pp. 346–359.

[2] J. Civera, A.J. Davison, and J. Montiel, “Inverse depth parametrization for monocular SLAM”, *IEEE Transactions on Robotics*, 24(5), 2008, pp. 932–945.

[3] A.J. Davison, I.D. Reid, N.D. Molton, O. Stasse, “MonoSLAM: Real-Time Single Camera SLAM”, *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(6), 2007, pp. 1052–1067.

[4] R.I. Hartley, A. Zisserman, “Multiple View Geometry in Computer Vision”, Cambridge University Press, 2nd edition, 2004.

[5] J.J. Leonard, H.F. Durrant-Whyte, “Simultaneous map building and localization for an autonomous mobile robot”. In: *IEEE/RSJ International Workshop on Intelligent Robots and Systems*, 1991, pp. 1442–1447.

[6] D. Oram, “Rectification for any epipolar geometry”. In: *BMVC’01* 2001, pp. 653–662.

[7] P. Pinies, T. Lupton, S. Sukkarieh, J.D. Tardos, “Inertial aiding of inverse depth SLAM using a monocular camera”. In: *IEEE International Conference on Robotics and Automation*, 2007, pp. 2797–2802.

[8] E. Rublee, V. Rabaud, K. Konolige, G. Bradski, “ORB: An efficient alternative to SIFT or SURF”. In: *International Conference on Computer Vision*, Barcelona, 2011.

[9] N. Trawny, S.I. Roumeliotis, “Indirect kalman filter for 3D attitude estimation”, Technical Report, University of Minnesota, Dept. of Comp. Sci. & Eng., 2005.

[10] O.J. Woodman, “An introduction to inertial navigation”, Technical Report, UCAM-CL-TR-696, University of Cambridge, Computer Laboratory, 2007.