Large Scale SLAM Building Conditionally Independent Local Maps: Application to Monocular Vision

Pedro Piniés, *Student Member, IEEE,* Juan D. Tardós, *Member, IEEE*

***Abstract*—SLAM algorithms based on local maps have been demonstrated to be well suited for mapping large environments as they reduce the computational cost and improve the consistency of the final estimation. The main contribution of this paper is a novel submapping technique that does not require independence between maps. The technique is based on the intrinsic structure of the SLAM problem which allows the building of submaps that can share information, remaining *conditionally independent*. The resulting algorithm obtains local maps in constant time during the exploration of new terrain, and recovers the global map in linear time after simple loop closures, without introducing any approximations besides the inherent EKF linearizations. The memory requirements are also linear with the size of the map. As the algorithm works in covariance form, well-known data association techniques can be used in the usual manner.**

We present experimental results using a hand-held monocular camera, building a map along a closed loop trajectory of 140m in a public square, with people and other clutter. Our results show that the combination of conditional independence, that enables the system to share camera and feature states between submaps, and local coordinates, that reduce the effects of linearization errors, allow us to obtain precise maps of large areas with pure monocular SLAM in real time.

***Index Terms*—SLAM, Local Maps, Scalability, Loop closing, Visual SLAM.**

I. Introduction

THE simultaneous localization and mapping problem (SLAM) consists in processing the information obtained by a sensor installed on a mobile platform to obtain an estimate of its own pose while building a map of the environment. It has been the subject of continuous attention during the last two decades (for recent reviews, see [1]–[3]). The first consistent solution proposed, and still a popular one, is EKF-SLAM [4]–[6], that represents the vehicle pose and the location of a set of environment features in a joint state vector that is estimated using the Extended Kalman Filter. Under the assumption of white Gaussian noise, the EKF provides a suboptimal way to deal with the uncertainties associated with the motion and measurement processes, due to the inherent linearization errors. To clarify, in the rest of the paper we will refer to the EKF-SLAM solution as *suboptimal* as opposed to other *approximated* techniques that introduce additional approximations besides linearization.

This research has been funded in part by the European Union under project RAWSEEDS FP6-IST-045144 and the Dirección General de Investigación of Spain under project SLAM6DOF DPI2006-13578.

P. Piniés and J. D. Tardós are with Instituto de Investigación en Ingeniería de Aragón (I3A), Universidad de Zaragoza, María de Luna 1, E-50018, Zaragoza, Spain, e-mail: {ppinies,tardos}@unizar.es

Despite its relative success, the EKF-SLAM algorithm suffers from two main limitations:

1) It requires updating the full map covariance matrix after each measurement, giving a memory complexity of $O(n^2)$ and a time complexity of $O(n^2)$ per step, where n is the total number of features stored in the map [5].
2) The EKF linearization approximations produce optimistic values for the map covariance matrix and introduce errors in the estimation, which may result in inconsistency [7], [8].

Techniques based on building submaps confront both problems at the same time. The main motivation for using submaps is clear: if a large area is split into several submaps with the number of features bounded by a constant, the submaps can be built in constant time per step. To clarify terminology, in this paper we will use the generic term *submap* for a map of a small area inside a larger map. We will call *absolute submap* a submap expressed in global coordinates. We will use the term *local submap* or simply *local map* for a submap expressed with respect to a local coordinate frame. Although there is no formal proof, there is strong empirical evidence that using local submaps also improves the consistency of the EKF-SLAM [8]. The intuitive explanation is that in local maps uncertainty is small and the linearization errors introduced in the EKF remain small. Another advantage of these algorithms is that they allow direct implementation of data association methods since they work with covariance matrices.

The main contribution of this paper is a novel technique that allows the use of submap algorithms, avoiding the limitations imposed by the requirement of statistical independence between maps. The technique is based on the intrinsic structure of the SLAM problem which allows us to build submaps that can share information, remaining *conditionally independent*. During exploration of new terrain, it obtains local maps in constant time. After simple loop closures, it can recover the global map in linear time, without introducing any approximations besides the inherent EKF linearizations. As it works in covariance space, robust data association algorithms such as JCBB [9] can be directly used. The technique has been implemented using absolute submaps or local submaps. In the second case, the effects of linearization errors are minimized, and the maps obtained are actually more precise and consistent than the maps obtained by the techniques based on EKF or EIF that use global coordinates.

Section II discusses the related work. The basic technique for building conditionally independent submaps is introduced in section III and particularized in section IV for the case of Gaussian maps in covariance form. Section V presents the algorithms for exploration and loop closing. Section VI shows the application of the technique to the challenging case of pure monocular SLAM, closing a loop of 140m with a hand-held camera in a public square. Finally, in section VII we summarize the main characteristics of the algorithm presented and propose future work. A preliminary version of this work was presented in [10]. Apart from a more detailed presentation and discussion of the technique, this paper adds the loop closing technique and new experiments demonstrating the performance of the method.

II. Related Work

In the context of Gaussian filters, several techniques have been proposed to address the computational complexity problem. Postponement [11] and the Compressed EKF filter (CEKF) [12] reduce the computational cost by making updates in a local area around the robot, delaying the global map update until the vehicle moves to another area. The result obtained is suboptimal, but the global map update is still $O(n^2)$.

Techniques based on the Information Filter take advantage of the near to sparse structure of the information matrix (the inverse of the map covariance matrix) to reduce the computational burden. In the Sparse Extended Information Filter (SEIF) [13] the information matrix of the SLAM posterior is approximated by rounding to zero the small off-diagonal elements. This prevents the inter-landmark links from forming and therefore limits the density of the information matrix. The Thin Junction Tree Filter (TJTF) [14] and the Exactly Sparse Extended Information Filter (ESEIF) [15] maintain the sparsity by discarding some *weak* information such as the robot odometry. The Exactly Sparse Delayed state Filters (ESDF) [16] avoids the previous approximations by including the trajectory of the vehicle in the state vector which makes the information matrix exactly sparse. Nevertheless, some approximations are still performed when portions of the mean state vector are recovered from its canonical (information) form in order to evaluate the jacobians.

The main advantage of information filters is that both the measurement and motion steps can be performed by updating the information vector and matrix in constant time. However, to recover the estimated value of the map state, a sparse linear system has to be solved. This has been addressed using conjugate gradient [13], relaxation [17] or multi-level relaxation [18], which require quadratic or, at best, linear time to converge (see [19] for a discussion). A recent and very efficient technique that also works in information space is the Treemap algorithm [20], which requires $O(\log n)$ time per step to recover a part of the state and $O(n)$ to recover the whole map. However, it has only been tested using simulations, with known data association.

One important limitation of the techniques based on the information form is the difficulty of performing data association since the covariance matrix is not available. Most techniques resort to approximating the classical individual Mahalanobis gating, which is known to be problematic in difficult data association scenarios [9].

The problem of map consistency has motivated algorithms such as the UKF [21] which achieve better consistency properties, but do not take into account the computational complexity problem.

Finally, some techniques based on building submaps confront complexity and consistency issues at the same time. The first technique using absolute submaps was Decoupled Stochastic Mapping [22]. The main difficulty of the technique was that absolute submaps are not statistically independent and some approximations were needed to get rid of the dependencies, introducing inconsistency in the map. In local submaps, the base reference is usually chosen to be the first robot pose when the local map was started. This allows local maps to be initialized with zero uncertainty in the robot pose. Under the assumption of white noise and if no information is shared between maps, local maps are statically *independent* and thus, uncorrelated [23]. Local maps can be consistently combined using map joining [23], or the equivalent Constrained Local Submap Filter (CLSF) [24], to obtain the global map in an $O(n^2)$ operation. The more recent Divide and Conquer SLAM [25] is able to recover the global map in amortized $O(n)$ time, provided that the overlap between maps remains small.

However, it is important to note that for a set of local maps to be independent, no information can be shared between them. This has several consequences:

1) Features that are seen from two neighboring local maps have a different estimation in each map. If the information that both features are the same were used, the map independence would be destroyed. This information can only be used when recovering the global map with map joining, that has $O(n^2)$ cost. More efficient techniques such as CTS [26], Atlas [27] or Hierarchical SLAM [28] discard this information which results in weak links between maps, obtaining approximated solutions.
2) Loop consistency can be imposed at the global map level as Hierarchical SLAM does, but the corrections obtained cannot be propagated to the individual features inside the local maps because this would destroy map independence. Other techniques such as CTS and Atlas simply discard the loop constraints to remain efficient.
3) Sensors with partial observability such as monocular vision require the integration of measurements taken from several robot poses to obtain an accurate estimation of a feature. With independent maps, features that are observed at the end of one local map and at the beginning of the next map remain quite imprecise in both maps.
4) Vehicle states such as velocities or sensor biases that have been estimated in real time cannot be transferred between maps. For example, this precludes the use of inertial sensors. Also sensors that give absolute measurements such as GPS or compass cannot be used without destroying map independence.

These limitations are particularly important in the extreme case of pure monocular SLAM where the only sensory input is a single camera, with no odometry. Under these conditions, real-time EKF-SLAM has been successfully demonstrated in small areas [29]–[31]. The first system able to extend the approach to large outdoor areas is based on building independent local maps that are combined using the Hierarchical SLAM approach [32]. In that system the constraint of map independence forces to start each local map from scratch, without any information about the environment or camera velocities. This makes the system slightly unreliable as the most critical part, map initialization, is repeated once and again along the trajectory. Furthermore, as in monocular SLAM the scale is intrinsically unobservable, the different local maps obtained have quite different scale factors.

The method proposed in this paper avoids these problems by building *conditionally independent* submaps, that can share information about the environment and vehicle states: the submaps are conditionally independent given the common states. The idea of conditional independence has been previously used in Rao-Blackwellized particle filter (RBPF) SLAM in a different sense: the estimations of the elements in the map are conditionally independent given the robot trajectory [33]. Recent optimizations of this approach have produced very efficient and accurate techniques for indoor and outdoor SLAM with laser data [34]. The idea of conditional independence between local maps has been recently applied in [35], but the method makes the approximation that there are no common features between submaps; this approximation is not needed in our technique.

Our method presents some similitude with the Treemap algorithm [20] in the sense that flows of information are transferred between submaps to update previous map estimates. However, we use the covariance form instead of the information form, which allows us to apply effective data association algorithms such as JCBB. The second crucial difference is the use of local coordinates which improves precision, as shown in our experiments. Finally, our technique represents the information using sequential local maps instead of ordering features in a tree structure which has to be maintained and balanced, resulting in an algorithm which is easier to implement.

III. Building Conditionally Independent Submaps

A. Basic Probability Concepts

For the reader's convenience, this subsection summarizes the basic probability concepts that will be used in the rest of the paper. More detailed presentations can be found in [1], [36].

The *conditional probability* of a random variable $\mathbf{x}$ given the value of the random variable $\mathbf{y}$ is defined as:

$$p(\mathbf{x}|\mathbf{y}) = \frac{p(\mathbf{x},\mathbf{y})}{p(\mathbf{y})} \tag{1}$$

where $p(\mathbf{x},\mathbf{y})$ is the *joint distribution*, and $p(\mathbf{y})$ is the *marginal* distribution of $\mathbf{y}$. In a more general case:

$$p(\mathbf{x}|\mathbf{y},\mathbf{z}) = \frac{p(\mathbf{x},\mathbf{y}|\mathbf{z})}{p(\mathbf{y}|\mathbf{z})} \tag{2}$$

Two random variables are *independent* when:

$$p(\mathbf{x},\mathbf{y}) = p(\mathbf{x})p(\mathbf{y}) \tag{3}$$

which is equivalent to:

$$p(\mathbf{x}|\mathbf{y}) = p(\mathbf{x}) \tag{4}$$

Intuitively this means that knowledge of $\mathbf{y}$ does not provide any information about $\mathbf{x}$.

Two random variables $\mathbf{x}$ and $\mathbf{y}$ are *conditionally independent* given $\mathbf{z}$ when:

$$p(\mathbf{x},\mathbf{y}|\mathbf{z}) = p(\mathbf{x}|\mathbf{z})p(\mathbf{y}|\mathbf{z}) \tag{5}$$

which is equivalent to:

$$p(\mathbf{x}|\mathbf{y},\mathbf{z}) = p(\mathbf{x}|\mathbf{z}) \tag{6}$$

In this case, if $\mathbf{z}$ is known, $\mathbf{y}$ does not provide any additional information about $\mathbf{x}$.

In the case of two random variables that are jointly Gaussian, with mean and covariance given by:

$$p(\mathbf{x},\mathbf{y}) = \mathcal{N}\left(\left[\begin{array}{c}\hat{\mathbf{x}}\\ \hat{\mathbf{y}}\end{array}\right], \left[\begin{array}{cc}P_x & P_{xy}\\ P_{yx} & P_y\end{array}\right]\right) \tag{7}$$

the process of *marginalization* consists simply in choosing the appropriate rows and columns of the mean vector and covariance matrix:

$$p(\mathbf{y}) = \int p(\mathbf{x},\mathbf{y})\, d\mathbf{x} = \mathcal{N}\left(\hat{\mathbf{y}}, P_y\right) \tag{8}$$

and the process of *conditioning* is performed by [37]:

$$\begin{aligned} p(\mathbf{x}|\mathbf{y}) &= \frac{p(\mathbf{x},\mathbf{y})}{p(\mathbf{y})} = \mathcal{N}\left(\hat{\mathbf{x}}', P_x'\right) && (9)\\ \hat{\mathbf{x}}' &= \hat{\mathbf{x}} + P_{xy}P_{yy}^{-1}(\mathbf{y}-\hat{\mathbf{y}}) && (10)\\ P_x' &= P_x - P_{xy}P_{yy}^{-1}P_{yx} && (11) \end{aligned}$$

B. Conditionally Independent Absolute Submaps

Figure 1 shows on the top an example of a Bayesian Network that represents the probabilistic dependencies between stochastic variables involved in SLAM. Node $\mathbf{x}_i$ represents the state of the platform at the i^{th} time step, $\mathbf{u}_i$ models the motion applied to the system at $\mathbf{x}_i$, node $\mathbf{f}_j$ represents the j^{th} feature of the map and $\mathbf{z}_i$ compactly represents all feature observations taken from the i^{th} platform location. Without loss of generality, we will use this example to illustrate the development of the technique.

The graph describes a map in which the vehicle has moved along five different locations $\mathbf{x}_{1:5}$ and has observed five features $\mathbf{f}_{1:5}$ during the trajectory. As the inputs $\mathbf{u}_{1:5}$ and observations $\mathbf{z}_{1:4}$ are known, the pdf associated with the graph is given by:

$$p(\mathbf{x}_{1:5},\mathbf{f}_{1:5}|\mathbf{z}_{1:4},\mathbf{u}_{1:4}) \tag{12}$$

This pdf represents the joint distribution of the whole map and trajectory. Assume now that we want to estimate the same map by building two submaps as shown in the figure. In submap 1 the vehicle starts at $\mathbf{x}_1$ and finishes at $\mathbf{x}_3$ and has observed features $\mathbf{f}_{1:3}$ through measurements $\mathbf{z}_{1:2}$. Therefore

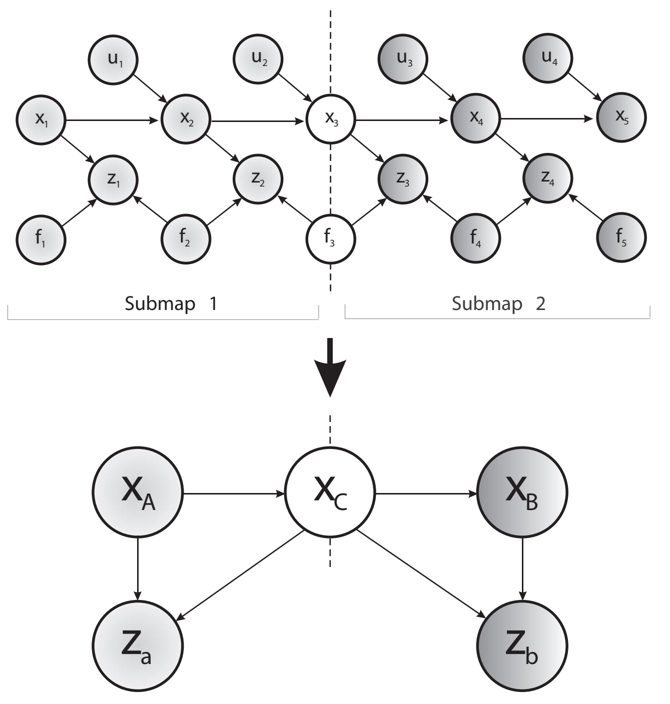

Fig. 1. Bayesian Network that describes the probabilistic dependencies between SLAM variables. It also reveals the intrinsic SLAM structure.

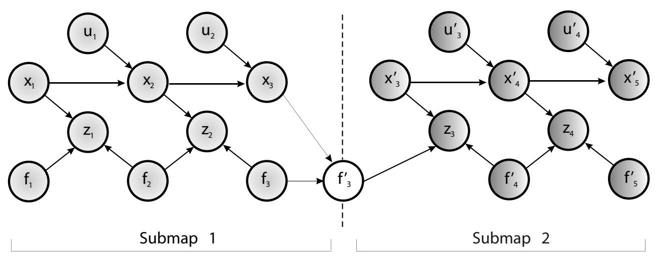

Fig. 2. Bayesian Network which illustrates the process to generate a local map with its own base reference

at the end of submap 1 the pdf that describes the map estimate is given by:

$$p(\mathbf{x}_{1:3}, \mathbf{f}_{1:3} | \mathbf{z}_{1:2}, \mathbf{u}_{1:2}) \tag{13}$$

Differences with current independent submap techniques begin now. Instead of starting submap 2 from scratch, we want to take advantage of the available estimation of features that are in the border between both submaps. In the example, feature $\mathbf{f}_3$, that is visible from both submaps, will be copied to the second map. In addition, if we want to build absolute submaps, we should also include in submap 2 the current vehicle estimate $\mathbf{x}_3$. So, the pdf that describes the initial state of submap 2 is just the result of marginalizing out the elements of the pdf (13) we are not interested in:

$$p(\mathbf{x}_3, \mathbf{f}_3 | \mathbf{z}_{1:2}, \mathbf{u}_{1:2}) = \int p(\mathbf{x}_{1:3}, \mathbf{f}_{1:3} | \mathbf{z}_{1:2}, \mathbf{u}_{1:2}) \, d\mathbf{x}_{1:2} \, d\mathbf{f}_{1:2} \tag{14}$$

Then, the vehicle continues traversing the second area, building submap 2. The vehicle has been in two new positions $\mathbf{x}_{4:5}$, has re-observed feature $\mathbf{f}_3$ and indirectly $\mathbf{x}_3$ through $\mathbf{z}_3$ and has observed two new features $\mathbf{f}_{4:5}$ through measurements $\mathbf{z}_{3:4}$. Therefore the final pdf of submap 2 is given by:

$$p(\mathbf{x}_{3:5}, \mathbf{f}_{3:5} | \mathbf{z}_{1:4}, \mathbf{u}_{1:4}) \tag{15}$$

As can be noticed in Eqs. (13,15), both local maps share in common a robot location $\mathbf{x}_3$, a feature $\mathbf{f}_3$ and some measurements $(\mathbf{z}_{1:2}, \mathbf{u}_{1:2})$, hence, they are not independent.

For clarity and generality, several nodes in the Bayesian Network will be grouped together as shown in figure 1, bottom. The notation used is:

- $\mathbf{x}_A$: Features and robot positions that are only observed in the first submap. In the example this corresponds to $\mathbf{f}_{1:2}$ and $\mathbf{x}_{1:2}$.
- $\mathbf{x}_B$: Features and robot positions that are only observed in the second submap, i.e., $\mathbf{f}_{4:5}$ and $\mathbf{x}_{4:5}$.
- $\mathbf{x}_C$: Common features and robot position that are observed both in the first and second submaps, i.e., $\mathbf{f}_3$ and $\mathbf{x}_3$.
- $\mathbf{z}_a$: Inputs and observations in the first submap gathered from features in $\mathbf{x}_A$ and $\mathbf{x}_C$, i.e., $\mathbf{u}_{1:2}$ and $\mathbf{z}_{1:2}$.
- $\mathbf{z}_b$: Inputs and observations in the second submap gathered from features in $\mathbf{x}_B$ and $\mathbf{x}_C$, i.e., $\mathbf{u}_{3:4}$ and $\mathbf{z}_{3:4}$.

As can be seen in the figure, the only connection between the set of nodes ($\mathbf{x}_A$, $\mathbf{z}_a$) and ($\mathbf{x}_B$, $\mathbf{z}_b$) is through node $\mathbf{x}_C$, i.e. both subgraphs are *d-separated* given $\mathbf{x}_C$ [38]. This implies that nodes $\mathbf{x}_A$ and $\mathbf{z}_a$ are conditionally independent of nodes $\mathbf{x}_B$ and $\mathbf{z}_b$ given node $\mathbf{x}_C$. Intuitively this means that if $\mathbf{x}_C$ is known, submaps 1 and 2 do not carry any additional information about each other. In the following, we will call this the *Submap Conditional Independence (CI) Property*, that can be stated as:

$$\begin{aligned} p(\mathbf{x}_A | \mathbf{x}_B, \mathbf{x}_C, \mathbf{z}_a, \mathbf{z}_b) &= p(\mathbf{x}_A | \mathbf{x}_C, \mathbf{z}_a) \\ p(\mathbf{x}_B | \mathbf{x}_A, \mathbf{x}_C, \mathbf{z}_a, \mathbf{z}_b) &= p(\mathbf{x}_B | \mathbf{x}_C, \mathbf{z}_b) \end{aligned} \tag{16}$$

C. Conditionally Independent Local Maps

The method described above can be easily adapted to building sequences of conditionally independent local maps, each with its own local base reference. Let us return to the moment when the first map was finished in the example of figure 1. With absolute maps, the last vehicle position $\mathbf{x}_3$ and feature $\mathbf{f}_3$ were chosen to initialize submap 2 in order to represent both maps with respect to the same reference and take advantage of the available estimation of the vehicle and the feature. Instead, we want now to represent submap 2 with respect to a local reference given by the current vehicle position $\mathbf{x}_3$, and still use the information about feature $\mathbf{f}_3$ in submap 2. For doing so in a consistent way, a copy of feature $\mathbf{f}_3$ expressed in the new reference must be calculated and included in submap 1. In the following a prime will be used to denote entities relative to the new base reference:

$$\mathbf{f}'_3 = \ominus \mathbf{x}_3 \oplus \mathbf{f}_3 \tag{17}$$

After this process the pdf that describes submap 1 is:

$$p(\mathbf{x}_{1:3}, \mathbf{f}_{1:3}, \mathbf{f}'_3 | \mathbf{z}_{1:2}, \mathbf{u}_{1:2}) \tag{18}$$

The new local map will start with robot position $\mathbf{x}'_3$ being exactly zero. Obviously this variable is completely independent of submap 1. By marginalizing (18) we obtain the pdf that describes the initial state of submap 2:

$$p(\mathbf{f}'_3|\mathbf{z}_{1:2},\mathbf{u}_{1:2}) \tag{19}$$

Once the vehicle has traversed the second submap and has incorporated all observations gathered in it, the pdf associated with the final estimate of submap 2 is:

$$p(\mathbf{x}'_{4:5},\mathbf{f}'_{3:5}|\mathbf{z}_{1:4},\mathbf{u}_{1:4}) \tag{20}$$

Figure 2 shows the Bayesian Network that corresponds to the new algorithm. As it can be seen, the structure of the network is the same as in figure 1, bottom. The only difference is that the part shared by both maps $\mathbf{x}_C$ corresponds in this case to the local representation of feature $\mathbf{f}'_3$. As a consequence, the Submap CI Property (16) is valid for local submaps as well as for absolute submaps.

D. Recovering the Global Map

The process of building the two conditionally independent submaps can be summarized in three steps:

1) Build the first submap, obtaining:

$$p(\mathbf{x}_A,\mathbf{x}_C|\mathbf{z}_a) \tag{21}$$

2) In the case of local maps, add to the first map the common elements relative to the last robot pose. Start the second map with the result of marginalizing out the non-common elements:

$$p(\mathbf{x}_C|\mathbf{z}_a)=\int p(\mathbf{x}_A,\mathbf{x}_C|\mathbf{z}_a)\,d\mathbf{x}_A \tag{22}$$

3) Continue building the second submap adding new features to it, obtaining:

$$p(\mathbf{x}_B,\mathbf{x}_C|\mathbf{z}_a,\mathbf{z}_b) \tag{23}$$

Our objective now is to combine the maps in equations (21,23) to obtain the joint distribution that corresponds to the global map. For doing so, the global map can be factorized as follows:

$$\begin{aligned}&p(\mathbf{x}_A,\mathbf{x}_B,\mathbf{x}_C|\mathbf{z}_a,\mathbf{z}_b)=\\ &\quad= p(\mathbf{x}_A|\mathbf{x}_B,\mathbf{x}_C,\mathbf{z}_a,\mathbf{z}_b)p(\mathbf{x}_B,\mathbf{x}_C|\mathbf{z}_a,\mathbf{z}_b)\\ &\quad= p(\mathbf{x}_A|\mathbf{x}_C,\mathbf{z}_a)p(\mathbf{x}_B,\mathbf{x}_C|\mathbf{z}_a,\mathbf{z}_b)\end{aligned} \tag{24}$$

where the first equality comes from eq. (2) and the second from the Submap CI Property (16). The second term in the factorization is directly the second submap (23). The first term can be obtained from the first submap by conditioning:

$$p(\mathbf{x}_A|\mathbf{x}_C,\mathbf{z}_a)=\frac{p(\mathbf{x}_A,\mathbf{x}_C|\mathbf{z}_a)}{p(\mathbf{x}_C|\mathbf{z}_a)}$$

Therefore, all the information needed to recover the global map can be obtained from the information stored in each of the submaps. Notice that no assumptions have been made about the particular distribution of the probability densities. The previous factorizations only depend on general probabilistic theorems and on the intrinsic structure of SLAM.

IV. The Case of Gaussian Submaps

In this section we will focus on the case when the probability densities are Gaussians represented in covariance form. Suppose we have built two submaps:

$$p(\mathbf{x}_A,\mathbf{x}_C|\mathbf{z}_a)=\mathcal{N}\left(\begin{bmatrix}\hat{\mathbf{x}}_{A_a}\\ \hat{\mathbf{x}}_{C_a}\end{bmatrix},\begin{bmatrix}P_{A_a} & P_{AC_a}\\ P_{CA_a} & P_{C_a}\end{bmatrix}\right) \tag{25}$$

$$p(\mathbf{x}_C,\mathbf{x}_B|\mathbf{z}_a,\mathbf{z}_b)=\mathcal{N}\left(\begin{bmatrix}\hat{\mathbf{x}}_{C_{ab}}\\ \hat{\mathbf{x}}_{B_{ab}}\end{bmatrix},\begin{bmatrix}P_{C_{ab}} & P_{CB_{ab}}\\ P_{BC_{ab}} & P_{B_{ab}}\end{bmatrix}\right) \tag{26}$$

where upper case subindexes are for state vector components whereas lower case subindexes describe which observations $\mathbf{z}$ have been used to obtain the estimate. For example, in the first submap, common elements $\mathbf{x}_C$ have been estimated using only observations $\mathbf{z}_a$, hence, the mean and covariance estimates are denoted by $\hat{\mathbf{x}}_{C_a}$ and P_{C_a} respectively.

We are interested in recovering the global map, represented by:

$$\begin{aligned}&p(\mathbf{x}_A,\mathbf{x}_B,\mathbf{x}_C|\mathbf{z}_a,\mathbf{z}_b)=\\ &=\mathcal{N}\left(\begin{bmatrix}\hat{\mathbf{x}}_{A_{ab}}\\ \hat{\mathbf{x}}_{C_{ab}}\\ \hat{\mathbf{x}}_{B_{ab}}\end{bmatrix},\begin{bmatrix}P_{A_{ab}} & P_{AC_{ab}} & P_{AB_{ab}}\\ P_{CA_{ab}} & P_{C_{ab}} & P_{CB_{ab}}\\ P_{BA_{ab}} & P_{BC_{ab}} & P_{B_{ab}}\end{bmatrix}\right)\end{aligned} \tag{27}$$

Comparing equations (27) and (26) we observe that the second local map by itself coincides exactly with the last two blocks of the global map. Only the terms related to $\mathbf{x}_A$ in the global map will need to be computed. This is because the first submap has only been updated with the observations $\mathbf{z}_a$, but not with the more recent observations $\mathbf{z}_b$. In the next subsections we will show how to *back-propagate* $\mathbf{z}_b$ to update the first submap and how to compute the correlation between both submaps $P_{AB_{ab}}$.

A. Back-Propagation

From the Submaps CI Property we know that:

$$p(\mathbf{x}_A|\mathbf{z}_a,\mathbf{z}_b,\mathbf{x}_C)=p(\mathbf{x}_A|\mathbf{z}_a,\mathbf{x}_C)=\mathcal{N}(\hat{\mathbf{x}}_{A|C},P_{A|C}) \tag{28}$$

The conditional distribution $p(\mathbf{x}_A|\mathbf{z}_a,\mathbf{z}_b,\mathbf{x}_C)$ can be obtained from the global map by marginalizing out $\mathbf{x}_B$ using eq. (8) and conditioning on $\mathbf{x}_C$ using eqs. (10, 11):

$$\hat{\mathbf{x}}_{A|C} = \hat{\mathbf{x}}_{A_{ab}}+P_{AC_{ab}}P_{C_{ab}}^{-1}(\mathbf{x}_C-\hat{\mathbf{x}}_{C_{ab}}) \tag{29}$$

$$P_{A|C} = P_{A_{ab}}-P_{AC_{ab}}P_{C_{ab}}^{-1}P_{CA_{ab}} \tag{30}$$

The conditional probability $p(\mathbf{x}_A|\mathbf{z}_a,\mathbf{x}_C)$ can also be obtained from the first map by conditioning on $\mathbf{x}_C$, which gives:

$$\hat{\mathbf{x}}_{A|C} = \hat{\mathbf{x}}_{A_a}+P_{AC_a}P_{C_a}^{-1}(\mathbf{x}_C-\hat{\mathbf{x}}_{C_a}) \tag{31}$$

$$P_{A|C} = P_{A_a}-P_{AC_a}P_{C_a}^{-1}P_{CA_a} \tag{32}$$

Equating (29,30) and (31,32) for all $\mathbf{x}_C$, and after some manipulations, we obtain the following back-propagation equations:

$$\begin{aligned} K &= P_{AC_a} P_{C_a}^{-1} \\ &= P_{AC_{ab}} P_{C_{ab}}^{-1} && (33) \\ P_{AC_{ab}} &= K P_{C_{ab}} && (34) \\ P_{A_{ab}} &= P_{A_a} + K(P_{CA_{ab}} - P_{CA_a}) && (35) \\ \hat{\mathbf{x}}_{A_{ab}} &= \hat{\mathbf{x}}_{A_a} + K(\hat{\mathbf{x}}_{C_{ab}} - \hat{\mathbf{x}}_{C_a}) && (36) \end{aligned}$$

Observe that, in order to propagate the influence of the new observations $\mathbf{z}_b$ to the first map, we only need the mean and covariance of the common elements from the second map: $\hat{\mathbf{x}}_{C_{ab}}$ and $P_{C_{ab}}$. An important property of the previous equations is that $\mathbf{x}_A$ can be updated with the information contained in $\mathbf{z}_b$ without having to compute the correlations between both maps $P_{AB_{ab}}$.

An interesting property of the back-propagation equations is that they can be applied at any moment. They work correctly even if we back-propagate twice the same information: the terms inside the parentheses in eqs (35,36) will be zero and the maps will remain unchanged. This allows us to schedule the back-propagation in moments with low CPU loads, or to delay it until a loop closure is detected.

B. Computing the Correlation between Submaps

If you want to obtain the covariance matrix of the whole map, the correlation term $P_{AB_{ab}}$ should also be computed. For doing so, we first obtain the expression of the covariance of $p(\mathbf{x}_A, \mathbf{x}_B | \mathbf{z}_a, \mathbf{z}_b, \mathbf{x}_C)$ by conditioning the global map on $\mathbf{x}_C$:

$$\begin{bmatrix} P_{A_{ab}} - P_{AC_{ab}} P_{C_{ab}}^{-1} P_{CA_{ab}} & P_{AB_{ab}} - P_{AC_{ab}} P_{C_{ab}}^{-1} P_{CB_{ab}} \\ P_{BA_{ab}} - P_{BC_{ab}} P_{C_{ab}}^{-1} P_{CA_{ab}} & P_{B_{ab}} - P_{BC_{ab}} P_{C_{ab}}^{-1} P_{CB_{ab}} \end{bmatrix} \quad (37)$$

Due to the submaps CI property we know that $\mathbf{x}_A$ and $\mathbf{x}_B$ are conditionally independent given $\mathbf{x}_C$ and therefore the correlation term in eq. (37) must be zero, which gives the following expression for the correlation term:

$$\begin{aligned} P_{AB_{ab}} &= P_{AC_{ab}} P_{C_{ab}}^{-1} P_{CB_{ab}} \\ &= K P_{CB_{ab}} && (38) \end{aligned}$$

However, computing all the correlation blocks in the global map is an $O(n^2)$ operation and, in fact, they are never required by our method, as will be explained next.

V. EKF-SLAM with Conditionally Independent Submaps

A. Exploration

Figure 3 shows a schematic view of the elements of the total covariance matrix that are actually calculated during the process of building up a sequence of conditionally independent submaps. Notice that the off-diagonal blocks of the matrix are not zero because the submaps are not *independent*. However, they are not required to obtain the global map. If the maximum size of the submaps is bounded by a constant, the process of building the CI submaps is $O(1)$ per step. In the absence of loop closures, the last submap, including the current robot pose, is already suboptimal. The suboptimal estimation of the previous submaps can be obtained with a complete back-propagation in $O(n)$. It is important to point out that the back-propagation operation is in fact delayed until a submap is revisited or a loop closing is detected reducing even more the computational complexity of the algorithm.

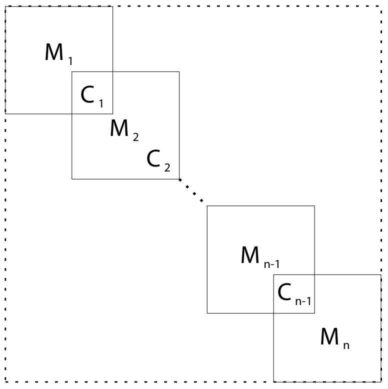

Fig. 3. Schematic representation of the elements of the total map covariance matrix that are indeed calculated with the method proposed. $\mathbf{M}_i$ represents Submap i elements whereas $\mathbf{C}_i$ are the common elements between Submap i and $i+1$.

Algorithm 1 : `EKF-SLAM with CI submaps`

```
$\mathbf{z}_0, \mathbf{R}_0 = get\_observations$
$\mathbf{m}_0 = init\_map(\mathbf{z}_0, \mathbf{R}_0)$
$j = 0$
for $k = 1$ to steps do
    $\mathbf{u}_{k-1}, \mathbf{Q}_{k-1} = get\_odometry$
    $\mathbf{m}_j = EKF\_prediction(\mathbf{m}_j, \mathbf{u}_{k-1}, \mathbf{Q}_{k-1})$
    $\mathbf{z}_k, \mathbf{R}_k = get\_observations$
    $\mathcal{DA}_k = data\_association(\mathbf{m}_j, \mathbf{z}_k, \mathbf{R}_k)$
    $\mathbf{m}_j = EKF\_update(\mathbf{m}_j, \mathbf{z}_k, \mathbf{R}_k, \mathcal{DA}_k)$
    $\mathbf{m}_j = add\_new\_features(\mathbf{m}_j, \mathbf{z}_k, \mathbf{R}_k, \mathcal{DA}_k)$
    if $size(\mathbf{m}_j) > max\_features$ then
        $\mathbf{m}_{j+1} = map\_transition(\mathbf{m}_j)$
        $j = j + 1$
    end if
end for
for $i = j$ downto 2 do
    $\mathbf{m}_{i-1} = back\_propagation(\mathbf{m}_i, \mathbf{m}_{i-1})$
end for
```

A simple implementation of our SLAM method is shown in algorithm 1. The implementation follows the structure of the standard EKF SLAM algorithm but introduces two new functions: *map_transition* and *back_propagation*. Function *back_propagation* is implemented directly using eqs. (33)-(36). Function *map_transition* creates a new submap when the number of features in the current map exceeds a given threshold. When using absolute submaps, the common features are directly copied to the new map $\mathbf{m}_{j+1}$ and the last robot pose in map $\mathbf{m}_j$ is replicated twice in the new submap. One of the copies will change as the robot moves through the new map carrying the current position, while the other will remain as a common element with map $\mathbf{m}_j$ to perform back-propagation.

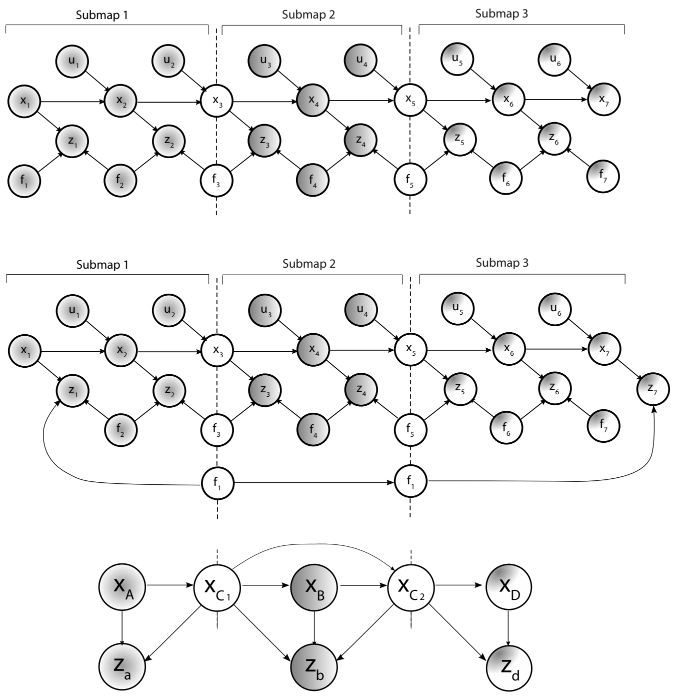

Fig. 4. Changes produced in the maps structure when our loop closing algorithm is applied: dependencies between submaps before closing the loop (top); final Bayesian Network after reobserving $\mathbf{f}_1$ and closing the loop (middle); some nodes are grouped together to show more clearly that the CI between submaps is maintained (bottom).

In the case of local submaps, map $\mathbf{m}_j$ is augmented with the common features expressed relative to the last robot pose in map $\mathbf{m}_j$. These features are then copied to map $\mathbf{m}_{j+1}$ which is started with robot pose equal to zero.

When using absolute submaps, our technique is similar to Postponement [11] and Compressed EKF [12] in the sense that most operations are performed in a local area and then the results are propagated to the rest of the map, obtaining the same solution as with the basic EKF-SLAM. However, we never need to compute the covariance matrix of the whole map, which reduces the computational to $O(1)$ for the local operations and to $O(n)$ for the complete back-propagation.

B. Loop Closing

Figure 4 shows on top the dependencies between three absolute submaps that have been built using our technique, before a loop closure. The pdfs that define the state of each submap are:

$$\begin{array}{lcl} Submap\ \ 1 & \rightarrow & p(\mathbf{x}_{1:3}, \mathbf{f}_{1:3}|\mathbf{z}_{1:2}) \\ Submap\ \ 2 & \rightarrow & p(\mathbf{x}_{3:5}, \mathbf{f}_{3:5}|\mathbf{z}_{1:4}) \\ Submap\ \ 3 & \rightarrow & p(\mathbf{x}_{5:7}, \mathbf{f}_{5:7}|\mathbf{z}_{1:6}) \end{array} \tag{39}$$

Observe that the most updated map is the current map, submap 3, that takes into account all the available observations.

Assume now that the robot is at position $\mathbf{x}_7$ and it closes a loop by observing feature $\mathbf{f}_1$ through measurement $\mathbf{z}_7$. The algorithm used to maintain the CI between submaps is as follows:

- The loop closing features, in this example $\mathbf{f}_1$, are copied to the common parts of all the intermediate submaps belonging to the loop, including the current submap. The correlation of the copied features with the elements of each submap is calculated with equation(38). The pdfs of the submaps are now given by:

$$\begin{array}{lcl} Submap\ \ 1 & \rightarrow & p(\mathbf{x}_{1:3}, \mathbf{f}_{1:3}|\mathbf{z}_{1:2}) \\ Submap\ \ 2 & \rightarrow & p(\mathbf{x}_{3:5}, \mathbf{f}_{3:5}, \mathbf{f}_1|\mathbf{z}_{1:4}) \\ Submap\ \ 3 & \rightarrow & p(\mathbf{x}_{5:7}, \mathbf{f}_{5:7}, \mathbf{f}_1|\mathbf{z}_{1:6}) \end{array} \tag{40}$$

- The current submap (submap 3) is updated with the loop closing observations ($\mathbf{z}_7$) using the standard EKF equations. The state of the Bayesian Network after performing the previous operations is shown in Figure 4 middle. In figure 4 bottom we have grouped some nodes together to clearly show that the CI property between submaps still holds. Submap 3 is now described by:

$$Submap\ \ 3 \ \rightarrow\ p(\mathbf{x}_{5:7}, \mathbf{f}_{5:7}, \mathbf{f}_1|\mathbf{z}_{1:7}) \tag{41}$$

- Thanks to the CI property, submaps 1 and 2 are updated using the *back-propagation* equations (33-36), obtaining:

$$\begin{array}{lcl} Submap\ \ 1 & \rightarrow & p(\mathbf{x}_{1:3}, \mathbf{f}_{1:3}|\mathbf{z}_{1:7}) \\ Submap\ \ 2 & \rightarrow & p(\mathbf{x}_{3:5}, \mathbf{f}_{3:5}, \mathbf{f}_1|\mathbf{z}_{1:7}) \end{array} \tag{42}$$

Notice that after applying this procedure all the submaps are suboptimal (up to EKF linearization errors) because they have been updated with all the available information. The price paid to maintain conditional independence is that all the submaps belonging to the loop contain a copy of the loop closing features.

In our algorithm the current submap is always kept suboptimal. With the loop closing procedure described above, when the information is propagated to a neighboring map, it becomes suboptimal. Repeating the process along the chain of submaps allows obtaining the global suboptimal map. Under the assumption that the size of the common part between submaps is bounded by a constant, the cost of each propagation is O(1), and the total cost of obtaining the global map after a loop closure is O(n). For this assumption to hold, the number of loops each local map belongs to must be bounded by a constant, regardless of the size of the environment. In extremely loopy environments, like a Manhattan-like world, this requirement can be easily violated. Maintaining efficiency in such situations is the subject of further investigations.

C. Revisiting a map

When a submap is revisited, the robot state that performs the transition to the revisited submap has to be included as a common element between both maps in order to preserve the CI property. Figure 5, top, shows an example in which the robot returns to the first submap when it is at $\mathbf{x}_6$ and reobserves feature $\mathbf{f}_2$. Including $\mathbf{x}_6$ as a common element of both maps preserves their CI, without introducing any approximation. A potential drawback of this approach is that the size of the common parts can grow without bound when revisiting the same environment indefinitely. However, if the number of

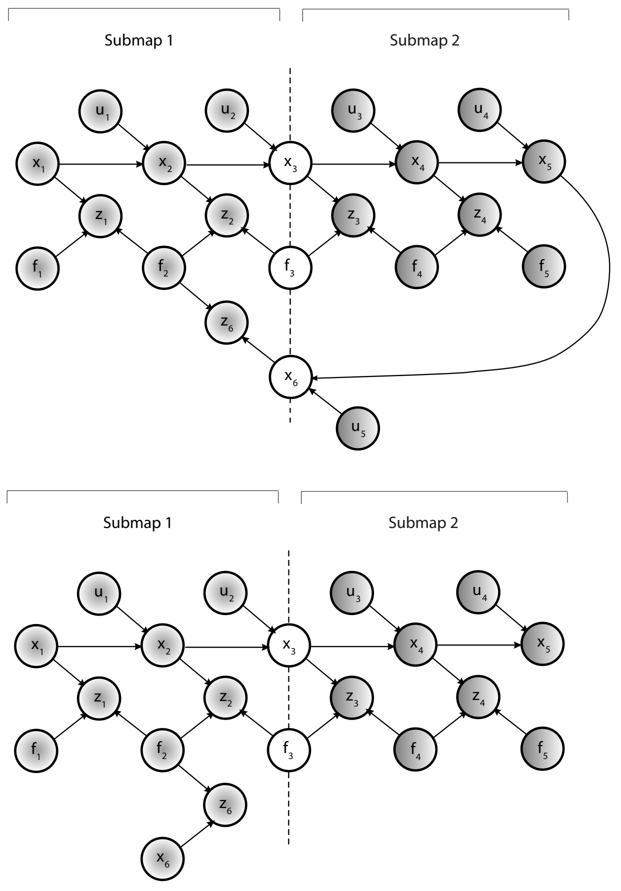

Fig. 5. When the first submap is revisited, the CI is preserved by including the robot position $\mathbf{x}_6$ in the common elements between both submaps (top). By marginalizing $\mathbf{x}_6$ from submap 2 and relocating the robot in submap 1, the CI property is also guaranteed but the odometry information is lost (bottom).

times the submaps are revisited is bounded by a constant, the global map can still be obtained in $O(n)$.

An alternative approximate solution that improves efficiency is to marginalize out the robot in the current map and relocate it in the revisited map, as shown in figure 5, bottom. A similar technique is used in ESEIF [15] to maintain the sparsity of the information matrix. In this case, the odometry link is disregarded and therefore we lose some information (in the figure, node $\mathbf{u}_5$ has disappeared). Nevertheless, the information loss is minimal because we can use the features common to both maps to relocate the robot with good precision. In our pure monocular SLAM application we do not even have odometry. Instead we simply have a prediction of the camera location using a constant velocity model, whose accuracy is negligible compared with the accuracy of the visual observations.

VI. Experimental Results

The algorithm proposed has been tested using real data obtained in an urban environment using a hand-held monocular camera. The features extracted from the images are Harris points. Data association is performed by predicting the feature locations in the image and searching for them with normalized correlation [30]. The set of matched features is further verified using the JCBB algorithm [9] which has demonstrated to add the needed robustness to build monocular maps in urban areas [32]. The method implemented to detect a loop closing is based on the map-to-map matching algorithm proposed in [32]. Basically, this method uses unary constraints, in this case the normalized correlation between features patches, and binary constraints, the relative distances between feature points in space, to find the maximal subset of geometrically compatible matchings. To speed up the search, a specialized version of the Geometric Constraints Branch and Bound (GCBB) algorithm [39] is implemented. In case of positive matches we obtain which subset of features in the current map corresponds to a subset of features in a previous map.

The state vector of each submap $\mathbf{M}_i$ contains the final camera location $\mathbf{x}_c^i$ and the 3D location of all features $(\mathbf{y}_1^i \dots \mathbf{y}_n^i)$, with respect to the map base reference (absolute or local). For the feature representation, we use the inverse-depth model proposed in [31]:

$$\mathbf{x}^T = (\mathbf{x}_c^T, \mathbf{y}_1^T, \mathbf{y}_2^T, \dots, \mathbf{y}_n^T) \tag{43}$$

$$\mathbf{x}_c^T = (\mathbf{r}^T, \mathbf{\Psi}^T, \mathbf{v}^T, \mathbf{w}^T) \tag{44}$$

$$\mathbf{y}_i = (x_i\, y_i\, z_i\, \theta_i\, \phi_i\, \rho_i)^T \tag{45}$$

This feature model represents the feature state as the camera optical center location $(x_i\, y_i\, z_i)$ when the feature point was first observed, and the azimuth and elevation $(\theta_i\, \phi_i)$ of the ray from the camera to the feature point. Finally, the depth d_i along this ray is represented by its inverse $\rho_i = 1/d_i$. The main advantage of the inverse-depth parametrization is that it allows consistent undelayed initialization of the 3D point features, regardless of their distance to the camera.

The camera state $\mathbf{x}_c$ contains the position of the camera in cartesian coordinates $\mathbf{r}$, its attitude in Euler angles $\mathbf{\Psi}$, the linear velocity $\mathbf{v}$ and its angular velocity $\mathbf{w}$. The process model used for the camera motion is a constant velocity model with white Gaussian noise in the linear and angular accelerations. Using pure monocular vision, without any kind of odometry, the scale of the map is not observable. However, by choosing appropriate values for the initial velocities and the covariance of the process noise, the EKF-SLAM is able to obtain an approximate scale for the map.

The experiment has been carried out along a public square in our hometown. The trajectory was performed with the hand-held camera looking to the right and closing a loop following approximately the same path. The sequence contains 2700 images taken at 20Hz along a path of around 140m. During the map building process, approximately 500 salient features are extracted and tracked from the surrounding buildings and objects. Figure 6 shows one of the images obtained in the experiment with the corresponding features extracted, and the map that is being built. The process of building a sequence of conditionally independent local submaps can be seen in the accompanying video. On the images, the features are depicted on their predicted locations.

The characteristics of the experiment make it suitable to show the benefits of sharing information between maps. Since a feature can be seen from different local maps, the technique proposed turns out to be very useful, allowing us to reuse a

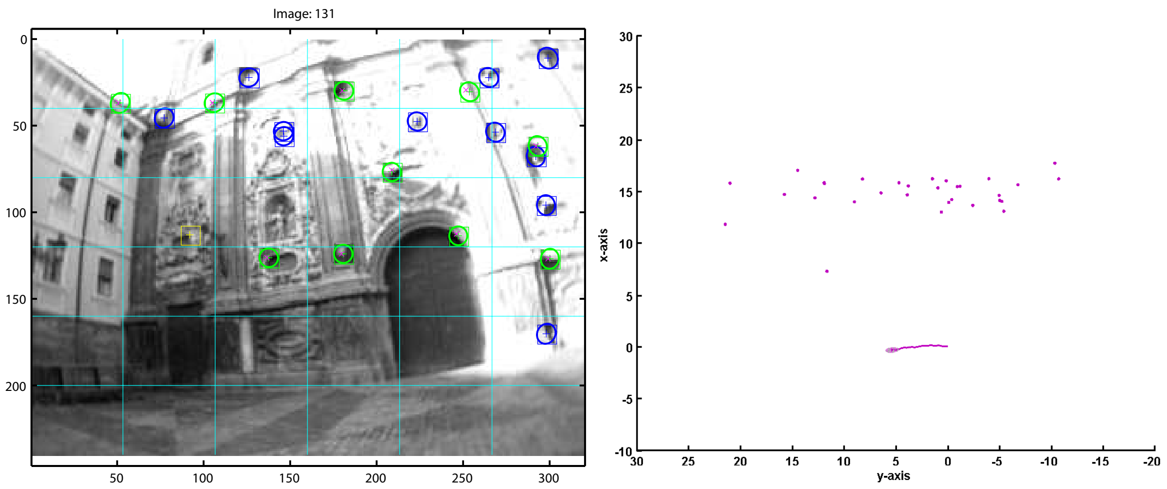

Fig. 6. A screenshot of the features extracted and the map built. The whole process of building a sequence of CI local maps can be seen in the accompanying video. The colors used in the video are: green for features predicted and matched, blue for predicted but not matched, and red for matchings rejected by JCBB.

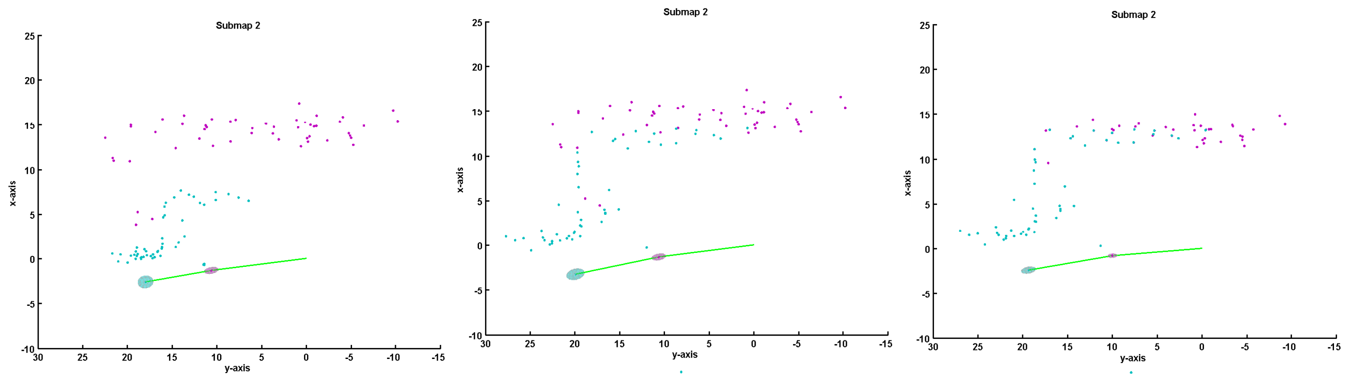

Fig. 7. Comparison of the results obtained building two local maps: classical independent local maps (left), conditionally independent local maps before back-propagation (middle) and after back-propagation (right).

feature without having to re-initialize it in each local map. In addition, linear and angular velocities of the camera, $\mathbf{v}$ and $\mathbf{w}$, can be consistently shared between consecutive submaps avoiding significant scale changes, a problem that needs to be addressed in techniques that build independent submaps [32]. Figure 7 shows an example of the advantages of our technique with respect to previous local mapping techniques. By sharing information with the first map, the second submap has the same scale and is more precise than in the case of independent maps. Actually, the second submap is suboptimal (up to the EKF linearization approximations). After back-propagation the estimation of the features in the first submap is improved to also become suboptimal.

Figure 8 compares the solutions obtained by a standard EKF algorithm and our method with absolute submaps for the first 1000 steps of the experiment. The number of absolute submaps created along this trajectory is 6. The EKF and the absolute submaps are superimposed in the figures to facilitate the comparison. Left figure shows our solution before performing the *back-propagation*, notice that both maps present several differences although the last absolute submap gives the same solution as the EKF since it is equally updated. On the right figure we can see that after updating the previous maps with the *back-propagation*, both solutions are exactly identical as was expected from the theoretical analysis.

The running times of both algorithms in a MATLAB implementation are shown in Figure 9. Notice that our algorithm runs in constant time since the size of the submaps is bounded whereas the standard EKF solution grows quadratically which makes it be more than 10 times slower than our method after the first 1000 steps. When the *back-propagation* is performed to update the previous maps, the extra time required turns out to be just 0.17 seconds which has little effect in the time of the last submap as can be seen in the figure.

For comparison purposes the whole dataset has been processed building absolute submaps and local submaps. In both cases the maximum number of features per map has been limited to 50 and the total number of local maps created is 15. Figure 10 presents the results obtained by both algorithms. The top plots show the maps obtained until the moment where the loop was detected by the map-to-map matching algorithm. The ellipsoids show the uncertainty in the camera position at the end of each local map, in absolute coordinates.

It can be noticed in the top left figure that the absolute submap technique gives an optimistic result. The uncertainty associated with the last camera position, around $x = -8$, $y =$

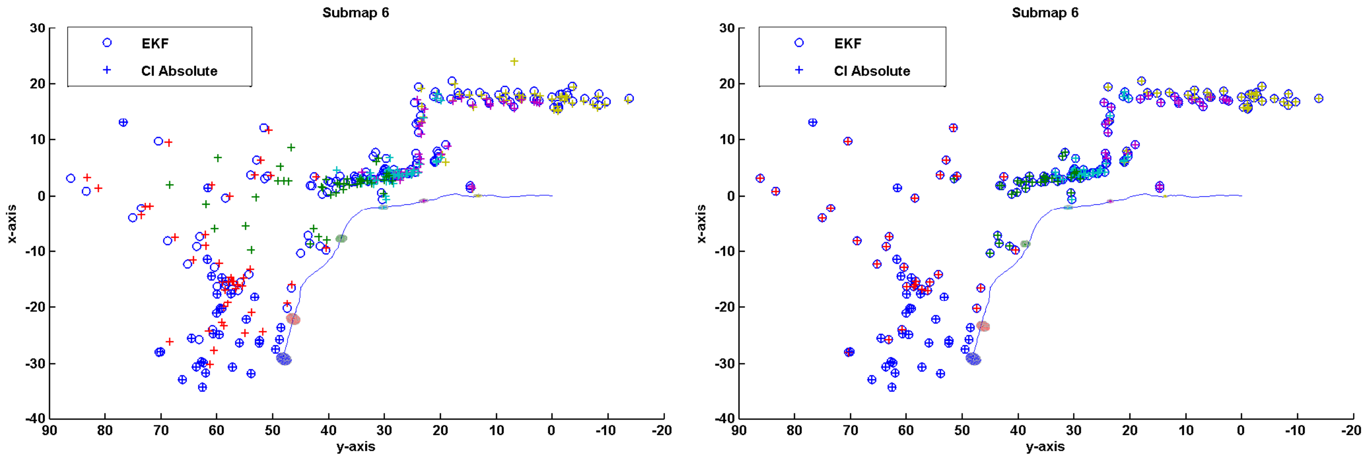

Fig. 8. Map solutions obtained during the first 1000 steps of the experiment using a standard EKF implementation and our method with absolute submaps. Both solutions are superimposed in the figures to stress the differences before performing the *back-propagation* (left) and to emphasize the identity of the solutions after *back-propagating* to all the previous maps (right).

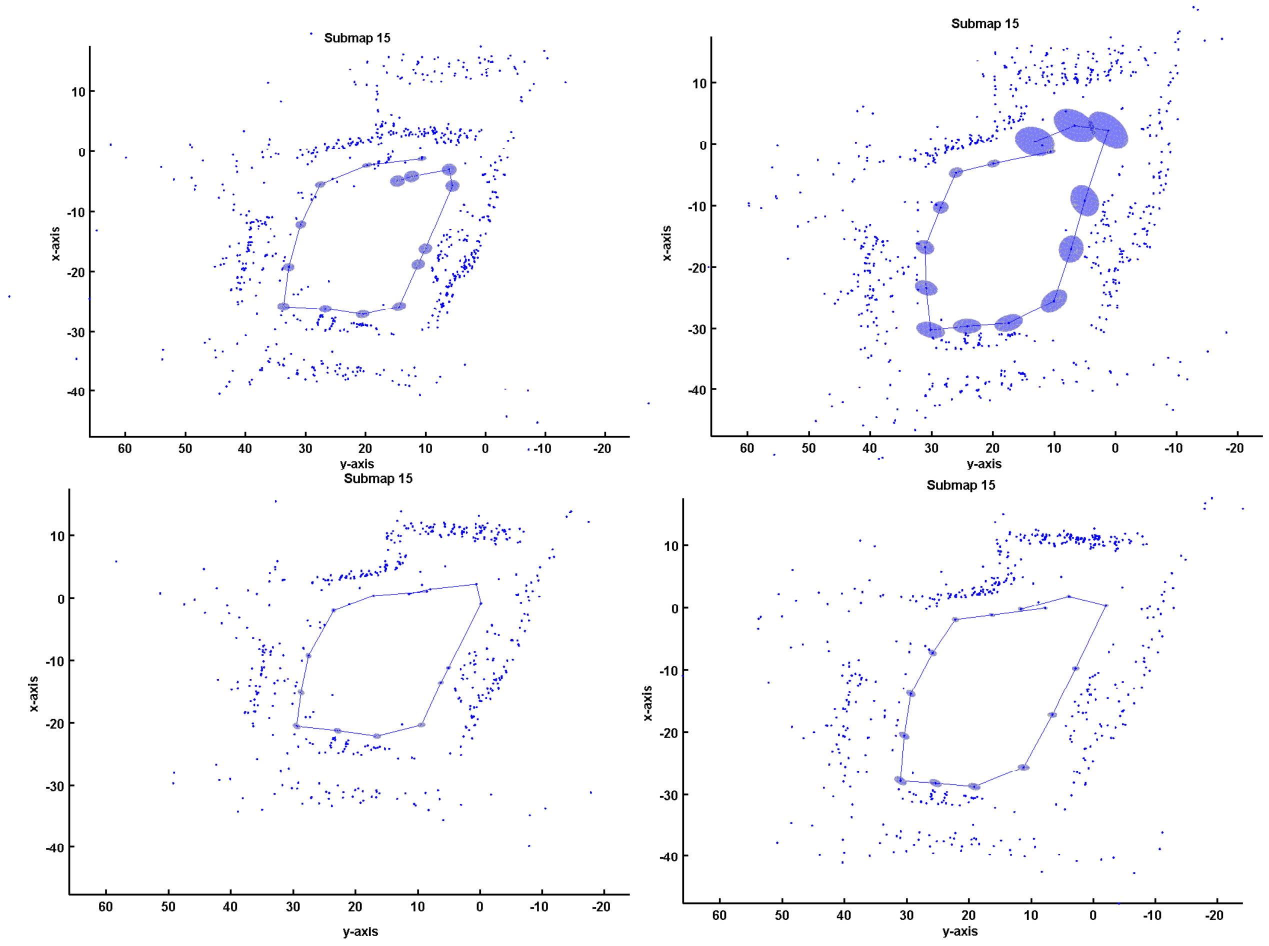

Fig. 10. Final maps obtained with pure monocular SLAM building conditionally independent submaps in absolute coordinates (left) and in local coordinates (right). The plots on the top show the maps obtained without back-propagation or loop closing. The ellipsoids correspond to the absolute uncertainty in the camera position at the end of each submap. The plots on the bottom show the final maps obtained obtained after loop closing and back-propagation.

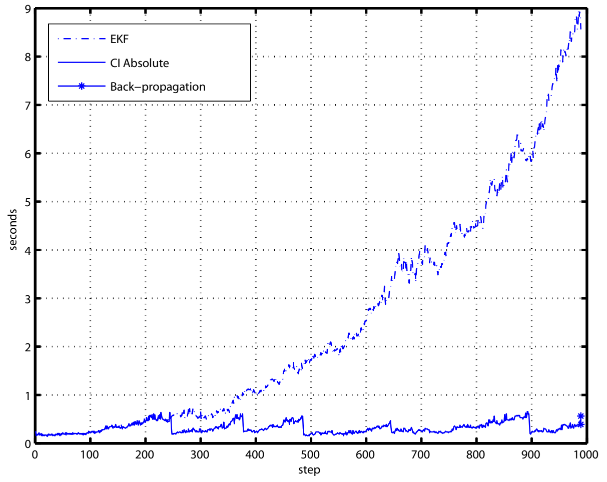

Fig. 9. Comparison of the computational times required by the standard EKF and our CI method using Absolute submaps after processing the first 1000 images of the experiment in a MATLAB implementation. At the end of the CI Absolute time we show as well the minimal time spent on performing a *back-propagation* to update all previous submaps

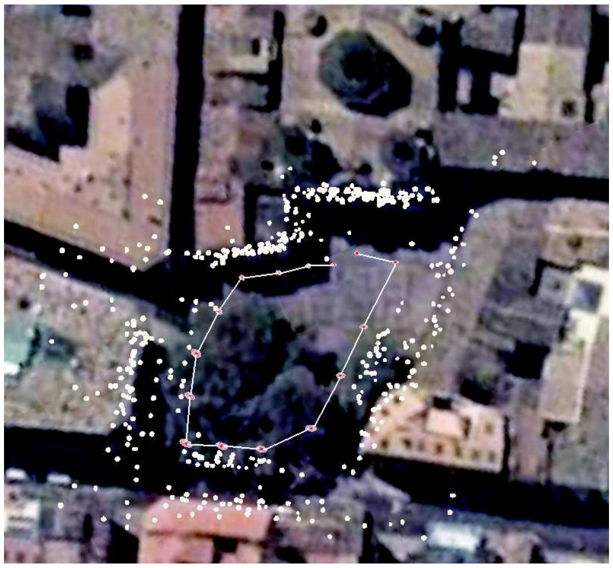

Fig. 11. Final map and trajectory estimates superimposed to an aerial image of the real environment.

17, cannot explain the big gap that appears between the first estimate of the top wall and the new estimate obtained on the second pass. Nevertheless the map matching algorithm used allows us to realize that both walls are indeed the same and the loop can be closed as it is shown in the bottom left figure. However, as the estimate was inconsistent, the final map has to be slightly deformed to accomplish the loop constraint.

In contrast, the local submap technique achieves better consistency and as a consequence a better estimate of the map and the trajectory. This is noticeable in the larger size of ellipsoids before the loop closure constraint is applied, that include the path performed during the first pass. After imposing the loop closure, the map obtained is quite precise. Figure 11 shows the final map superimposed on a satellite image of the environment obtained from Google Earth. The scale and the absolute position and orientation of the map, that are not observable with pure monocular SLAM, were adjusted by hand to draw the figure. Notice how the feature points mapped follow the shape formed by the surrounding buildings.

Using the implementation described in [32], both algorithms are able to build the sequence of submaps up to 60 features in real-time at 20Hz, including all image processing. For this map size, the running time for a standard EKF-SLAM implementation would increase quadratically up to about 2s per step. In our current MATLAB implementation, the whole process of loop detection, loop closing and back-propagation takes 1.15, 0.3 and 1.8 seconds respectively. We expect that an optimized C++ implementation will take a fraction of a second. To maintain real time performance at video frequency, loop closing can be implemented on a separate lower-priority thread.

VII. Conclusion

In this paper we have proposed a new technique that allows the use of submap algorithms, avoiding the constraint imposed by the requirement of probabilistic independence between them. Using this method, salient features of the environment or vehicle state components, such as velocity or global attitude, can be shared between local maps in a consistent manner. Our experiments show that this is extremely valuable to reduce the errors committed during the first steps of map initialization, specially for monocular vision.

Under the assumption that the size of the common parts between submaps is bounded by a constant, the back-propagation algorithm allows us to make updates from local map to local map in constant time. In addition, a loop-closing algorithm that takes advantage of the structure of the conditionally independent maps has been proposed. By means of this algorithm, the loop closure can be performed with a computational cost that is linear in the number of local maps instead of quadratic in the total number of features. So, the global cost of our method is $O(1)$ during exploration and $O(n)$ during loop closing. Memory requirements are also $O(n)$, because the whole covariance matrix is never computed.

Unlike many other techniques, this performance gain is not obtained by sacrificing precision. Our technique does not use sparsification or other approximations, apart from the intrinsic EKF linearizations. Using absolute submaps, the result obtained is the same as with the classical EKF-SLAM algorithm. Using local submaps, the inconsistencies introduced by the linearization errors are reduced, and the results obtained are much better, for a small fraction of the cost.

We believe that this work opens the way for developing new efficient submapping algorithms. We plan to extend the technique to larger environments, where hierarchical map decomposition and non-linear optimization techniques may be useful. The method presented here relies on the common part between submaps being small to achieve efficiency. Environments with more complicated topologies may require the development of new algorithms and maybe approximations. Regarding applications, we have demonstrated real-time monocular SLAM in moderately large urban environments. For reliable loop detection in larger areas appearance based methods [40] or image-to-map matching techniques [41] will be investigated. We are also interested in large-scale SLAM with systems that include inertial or other sensors, where the proposed technique will allow us to consistently share global information or sensor biases across submaps. A work in this line is [42] where the CI technique proposed here allows the consistent sharing of vehicle states and compass measurements between the local maps.

ACKNOWLEDGMENT

The authors would like to thank José Neira, Lina M. Paz, José M. M. Montiel and Javier Civera for the fruitful discussions and their help with the experimental setup.

REFERENCES

[1] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. The MIT Press, September 2005.

[2] H. Durrant-Whyte and T. Bailey, “Simultaneous localization and mapping: part I,” *IEEE Robotics & Automation Magazine*, vol. 13, no. 2, pp. 99–110, 2006.

[3] T. Bailey and H. Durrant-Whyte, “Simultaneous localization and mapping (SLAM): Part II,” *IEEE Robotics & Automation Magazine*, vol. 13, no. 3, pp. 108–117, 2006.

[4] R. Smith, M. Self, and P. Cheeseman, “A stochastic map for uncertain spatial relationships,” in *Robotics Research, The Fourth Int. Symposium*, O. Faugeras and G. Giralt, Eds. The MIT Press, 1988, pp. 467–474.

[5] J. A. Castellanos, J. M. M. Montiel, J. Neira, and J. D. Tardós, “The SPmap: A probabilistic framework for simultaneous localization and map building,” *IEEE Trans. Robotics and Automation*, vol. 15, no. 5, pp. 948–953, 1999.

[6] M. W. M. G. Dissanayake, P. Newman, S. Clark, H. F. Durrant-Whyte, and M. Csorba, “A solution to the simultaneous localization and map building (SLAM) problem,” *IEEE Trans. Robotics and Automation*, vol. 17, no. 3, pp. 229–241, 2001.

[7] S. J. Julier and J. K. Uhlmann, “A Counter Example to the Theory of Simultaneous Localization and Map Building,” in *Proc. IEEE Int. Conf. Robotics and Automation*, Seoul, Korea, 2001, pp. 4238–4243.

[8] J. A. Castellanos, R. Martínez-Cantín, J. Neira, and J. D. Tardós, “Robocentric map joining: Improving the consistency of EKF-SLAM,” *Robotics and Autonomous Systems*, vol. 55, no. 1, pp. 21–29, January 2007.

[9] J. Neira and J. D. Tardós, “Data association in stochastic mapping using the joint compatibility test,” *IEEE Trans. Robotics and Automation*, vol. 17, no. 6, pp. 890–897, 2001.

[10] P. Piniés and J. D. Tardós, “Scalable SLAM building conditionally independent local maps,” in *Proc. IEEE/RJS Int. Conference on Intelligent Robots and Systems*, San Diego, CA, USA, October 2007, pp. 3466–3471.

[11] J. Knight, A. Davison, and I. Reid, “Towards constant time SLAM using postponement,” in *Proc. IEEE/RJS Int. Conference on Intelligent Robots and Systems*, Maui, Hawaii, 2001, pp. 406–412.

[12] J. E. Guivant and E. M. Nebot, “Optimization of the simultaneous localization and map-building algorithm for real-time implementation,” *IEEE Trans. Robotics and Automation*, vol. 17, no. 3, pp. 242–257, 2001.

[13] S. Thrun, Y. Liu, D. Koller, A. Y. Ng, Z. Ghahramani, and H. Durrant-Whyte, “Simultaneous Localization and Mapping with Sparse Extended Information Filters,” *Int. J. Robotics Research*, vol. 23, no. 7-8, pp. 693–716, 2004.

[14] M. A. Paskin, “Thin Junction Tree Filters for Simultaneous Localization and Mapping,” in *Proc. Int. Joint Conf. Artificial Intelligence*, San Francisco, CA., 2003, pp. 1157–1164.

[15] M. R. Walter, R. M. Eustice, and J. J. Leonard, “Exactly sparse extended information filters for feature-based SLAM,” *Int. J. Robotics Research*, vol. 26, no. 4, pp. 335–359, 2007.

[16] R. M. Eustice, H. Singh, and J. J. Leonard, “Exactly sparse delayed-state filters for view-based SLAM,” *IEEE Trans. Robotics*, vol. 22, no. 6, pp. 1100–1114, Dec 2006.

[17] T. Duckett, S. Marsland, and J. Shapiro, “Fast, on-line learning of globally consistent maps,” *Autonomous Robots*, vol. 12, no. 3, pp. 287–300, 2002.

[18] U. Frese, P. Larsson, and T. Duckett, “A multigrid algorithm for simultaneous localization and mapping,” *IEEE Trans. Robotics*, vol. 21, no. 2, pp. 1–12, 2004.

[19] U. Frese, “A discussion of simultaneous localization and mapping,” *Autonomous Robots*, vol. 20, no. 1, pp. 25–42, January 2006.

[20] ——, “Treemap: An O(log n) algorithm for indoor simultaneous localization and mapping,” *Autonomous Robots*, vol. 21, no. 2, pp. 103–122, September 2006.

[21] S. Julier and J. Uhlmann, “A new extension of the Kalman Filter to nonlinear systems,” in *International Symposium on Aerospace/Defense Sensing, Simulate and Controls*, Orlando, FL, 1997, pp. 182–193.

[22] J. J. Leonard and H. J. S. Feder, “A computationally efficient method for large-scale concurrent mapping and localization,” in *Robotics Research: The Ninth International Symposium*, D. Koditschek and J. Hollerbach, Eds. Snowbird, Utah: Springer Verlag, 2000, pp. 169–176.

[23] J. D. Tardós, J. Neira, P. M. Newman, and J. J. Leonard, “Robust mapping and localization in indoor environments using sonar data,” *Int. J. Robotics Research*, vol. 21, no. 4, pp. 311–330, 2002.

[24] S. B. Williams, G. Dissanayake, and H. Durrant-Whyte, “An efficient approach to the simultaneous localisation and mapping problem,” in *Proc. IEEE Int. Conf. Robotics and Automation*, vol. 1, Washington DC, 2002, pp. 406–411.

[25] L. M. Paz, J. D. Tardós, and J. Neira, “Divide and conquer: EKF SLAM in O(n),” *IEEE Trans. Robotics*, vol. 24, no. 5, October 2008.

[26] J. Leonard and P. Newman, “Consistent, convergent and constant-time SLAM,” in *Proc. Int. Joint Conf. Artificial Intelligence*, Acapulco, Mexico, August 2003.

[27] M. Bosse, P. M. Newman, J. J. Leonard, and S. Teller, “SLAM in large-scale cyclic environments using the atlas framework,” *Int. J. Robotics Research*, vol. 23, no. 12, pp. 1113–1139, December 2004.

[28] C. Estrada, J. Neira, and J. D. Tardós, “Hierarchical SLAM: real-time accurate mapping of large environments,” *IEEE Trans. Robotics*, vol. 21, no. 4, pp. 588–596, August 2005.

[29] A. J. Davison, “Real-time simultaneous localisation and mapping with a single camera,” in *Proc. Int. Conf. Computer Vision*, vol. 2, Nice, Oct 2003, pp. 1403–1410.

[30] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, “MonoSLAM: Real-time single camera SLAM,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 1052–1067, June 2007.

[31] J. Civera, A. J. Davison, and J. M. M. Montiel, “Inverse depth parametrization for monocular SLAM,” *IEEE Trans. Robotics*, vol. 24, no. 5, October 2008.

[32] L. Clemente, A. J. Davison, I. D. Reid, J. Neira, and J. D. Tardós, “Mapping large loops with a single hand-held camera,” in *Proc. Robotics: Science and Systems*, Atlanta, GA, USA, June 2007.

[33] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, “FastSLAM: A factored solution to the simultaneous localization and mapping problem,” in *Proceedings of the AAAI National Conference on Artificial Intelligence*. Edmonton, Canada: AAAI, 2002.

[34] G. Grisetti, C. Stachniss, and W. Burgard, “Improved Techniques for Grid Mapping With Rao-Blackwellized Particle Filters,” *IEEE Trans. Robotics*, vol. 23, no. 1, pp. 34–46, February 2007.

[35] J. L. Blanco, J. A. Fernandez-Madrigal, and J. González, “Toward a unified bayesian approach to hibrid metric-topological SLAM,” *IEEE Trans. Robotics*, vol. 24, no. 2, pp. 259–270, April 2008.

[36] A. Papoulis and S. Pillai, *Probability, Random Variables and Stochastic Processes*. McGraw-Hill, 2002.

[37] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation*. New York: John Willey and Sons, 2001.

[38] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[39] J. Neira, J. D. Tardós, and J. A. Castellanos, “Linear time vehicle relocation in SLAM,” in *Proc. IEEE Int. Conf. Robotics and Automation*, Taipei, Taiwan, September 2003, pp. 427–433.

[40] M. Cummins and P. Newman, “Probabilistic appearance based navigation and loop closing,” in *Proc. IEEE Int. Conf. Robotics and Automation*, Rome, April 2007, pp. 2042–2048.

[41] B. Williams, P. Smith, and I. Reid, “Automatic relocalisation for a single-camera simultaneous localisation and mapping system,” in *Proc. IEEE Int. Conf. Robotics and Automation*, Roma, Italy, April 2007, pp. 2784–2790.

[42] D. Rivas, P. Ridao, J. D. Tardós, and J. Neira, "Underwater SLAM in man-made structured environments," *Journal of Field Robotics*, vol. 25, no. 8, pp. 1–24, August 2008.

Pedro Piniés was born in Bilbao, Spain, in 1979. He earned the M.S. degree in telecommunication engineering from the University of Zaragoza, Spain, in 2004. Currently he is a Ph.D. student in the Robotics, Perception and Real Time Group of the University of Zaragoza. His current research interests include SLAM, mobile robotics, computer vision and probabilistic inference.

Juan D. Tardós was born in Huesca, Spain, in 1961. He earned the M.S. and Ph.D. degrees in electrical engineering from the University of Zaragoza, Spain, in 1985 and 1991, respectively. He is professor with the Departamento de Informática e Ingeniería de Sistemas, University of Zaragoza, where he is in charge of courses in robotics, computer vision, and artificial intelligence. His current research interests include SLAM, perception and mobile robotics.