Amortized Constant Time State Estimation in SLAM using a Mixed Kalman-Information Filter

Viorela Ila Josep M. Porta Juan Andrade-Cetto
Institut de Robòtica i Informàtica Industrial, CSIC-UPC, Barcelona, Spain

***Abstract*— The computational bottleneck in all information-based algorithms for SLAM is the recovery of the state mean and covariance. The mean is needed to evaluate model Jacobians and the covariance is needed to generate data association hypotheses. Recovering the state mean and covariance requires the inversion of a matrix of the size of the state. Current state recovery methods use sparse linear algebra tools that have quadratic cost, either in memory or in time. In this paper, we present an approach to state estimation that is worst case linear both in execution time and in memory footprint at loop closure, and constant otherwise. The approach relies on a state representation that combines the Kalman and the information-based state representations. The strategy is valid for any SLAM system that maintains constraints between robot poses at different time slices. This includes both Pose SLAM, the variant of SLAM where only the robot trajectory is estimated, and hierarchical techniques in which submaps are registered with a network of relative geometric constraints.**

***Index Terms*— State recovery, Kalman filter, Information filter, Pose SLAM, Hierarchical SLAM.**

I. Introduction

Seminal solutions to the Simultaneous Localization and Mapping (SLAM) problem relied on the Extended Kalman Filter (EKF) to estimate the mean absolute position of landmarks and the robot pose and their associated covariance matrix. This has quadratic memory and computational cost, limiting its use only to small areas [16].

Instead of using the mean and the covariance, Gaussians can be represented using the information vector and the information matrix. In SLAM, the information matrix turns out to be approximately sparse, i.e., the matrix entries for distant landmarks are very small and the matrix can be sparsified with a minimal information loss, trading optimality for efficiency [17]. Efficiency without information loss is possible estimating the entire robot path along with the map, an approach typically referred to as full SLAM [3], [10], [13]. Exact sparsification is also possible if only a set of variables is maintained; either by keeping a small set of active landmarks kidnapping and relocating the robot [19], by decoupling the estimation problem maintaining the map only [20], or as it is done in Pose SLAM, by maintaining only the pose history [5], [11]. In Pose SLAM, landmarks are only used to obtain relative measurements linking pairs of poses. When working with sensors that are able to identify many landmarks per pose, Pose SLAM produces more compact maps than the other exactly sparse approaches.

Due to their small memory footprint, sparse representations enable SLAM solutions that scale nicely for very large maps. However, the information-based representation have some drawbacks from the estimation point of view. Off-line information-based SLAM approaches [3], [6], [14] obtain the maximum likelihood solution from the constraints encoded in the information matrix. The optimization iteratively approximates the mean solving a sequence of linear systems using the previously estimated mean as a linealization point for the constraints. This process assumes data association for granted, somehow limiting its applicability. On-line information-based approaches rely either on variants of the batch methods [10] or, more commonly, on filtering [5], [8] using the Extended Information Filter (EIF) as the estimation tool of choice. These on-line systems do not only have to recover the mean to evaluate the Jacobians, but also need to address the data association problem. Data association might be tackled without relying on the filtered pose priors [2]. The process, however, is prone to perceptual aliasing and it is often convenient to rely on filter estimates to limit the search space. In this case, false positives can be avoided performing prior-based data association tests that use cross covariances between match candidates. Again, those cross covariances are not directly available from the estimates of the information-based representations.

The EKF and the EIF applied to SLAM are radically different in nature. While in the former the estimate includes all the necessary data for linearization and data association, the latter is advantageous from the point of view of memory footprint. In this paper, we propose a combination of these two filters with the aim of getting the best of the two worlds: reduced memory complexity and easy access to the mean and the relevant blocks of the covariance matrix.

The work presented in this paper improves the formalization of the state estimation technique in [9], where we adopted an extended information filter approach. Here we definitively abandon this paradigm and propose a novel mixed Kalman-information filter. The strategy is valid for any SLAM approach that maintains constraints between robot poses at different time slices. This includes both Pose SLAM and submapping techniques in which submaps form a network of relative geometric constraints. For the sake of simplicity we center the discussion in the first case, but all the results are directly applicable to the second one as well.

The paper is structured as follows. In Section II, we formalize the Pose SLAM problem and describe its solution via EKF and EIF. In Section III we describe a combination of the two filters that allows state estimation in linear time and space complexities. Section IV describes a refinement of the presented approach that allows updates in constant time during open loop traverse. This is relevant in Pose SLAM approaches that carefully select the loops to close in order to avoid inconsistency as much as possible [8] or in hierarchical SLAM where submaps are scarcely re-visited [4]. With this enhancement the linear time complexity when closing a loop is amortized over long periods yielding an almost constant time state update. Section V presents results both with simulated data and with real datasets that validate the presented approach. Concluding remarks are given in Section VI.

II. Pose SLAM Formulation

In the incremental form of Pose SLAM, the objective is to estimate the trajectory of the robot, $\mathbf{x}_n = \{x_0, \dots, x_n\}$, with x_i the robot pose at time i that can be defined in $SO(2)$ or in $SO(3)$ in any parametrization. Using a Bayesian recursion, the trajectory, $\mathbf{x}_n$, is updated given a set of observations, $\mathbf{z}_n$, of the relative displacement between the current robot pose and previous poses along the path

$$p(\mathbf{x}_n|\mathbf{z}_n, \mathbf{x}_{n-1}) \propto \; p(\mathbf{x}_n|\mathbf{x}_{n-1})\, p(\mathbf{z}_n|\mathbf{x}_n).$$

The observations can be split in two disjoint groups, a set of observations between the current robot pose and the immediate previous one, $\mathbf{u}_n$, and a set of observations linking the current pose with any other pose but the previous one, $\mathbf{y}_n$. With this, the probabilistic model becomes

$$\begin{aligned} p(\mathbf{x}_n|\mathbf{z}_n, \mathbf{x}_{n-1}) &\propto \; p(\mathbf{x}_n|\mathbf{x}_{n-1})\, p(\mathbf{u}_n, \mathbf{y}_n|\mathbf{x}_n) \\ &\propto \; p(\mathbf{x}_n|\mathbf{x}_{n-1})\, p(\mathbf{u}_n|\mathbf{x}_n)\, p(\mathbf{y}_n|\mathbf{x}_n) \\ &\propto \; p(\mathbf{x}_n|\mathbf{x}_{n-1}, \mathbf{u}_n)\, p(\mathbf{y}_n|\mathbf{x}_n). \end{aligned} \tag{1}$$

The estimation problem in Eq. (1) corresponds to the SLAM operations of augmenting the state, $p(\mathbf{x}_n|\mathbf{x}_{n-1}, \mathbf{u}_n)$, and updating the robot path using relative observations, $p(\mathbf{y}_n|\mathbf{x}_n)$.

Assuming Gaussian distributions, the probabilities in Eq. (1) can be parametrized either in terms of their mean and covariance, $\mathbf{x}_n \sim \mathcal{N}(\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n)$, or in terms of the information vector and matrix, $\mathbf{x}_n \sim \mathcal{N}^{-1}(\boldsymbol{\eta}_n, \boldsymbol{\Lambda}_n)$, with $\boldsymbol{\eta}_n = \boldsymbol{\Lambda}_n \boldsymbol{\mu}_n$, $\boldsymbol{\Lambda}_n = \boldsymbol{\Sigma}_n^{-1}$, and in which the estimation workhorses are the extended Kalman and information filters, respectively.

Note that simultaneous observations are independent and thus observations linking the same pair of poses can be fused before using them to update the filter. In particular, we can assume the set $\mathbf{u}_n$ to include a single element, u_n.

A. EKF State Estimation

The observation $u_n \sim \mathcal{N}(\mu_u, \boldsymbol{\Sigma}_u)$ is used to augment the state with a new pose. The state transition model is given by

$$\begin{aligned} x_n &= f(x_{n-1}, u_n) \\ &\approx f(\mu_{n-1}, \mu_u) + \mathbf{F}_n\, (\mathbf{x}_{n-1} - \boldsymbol{\mu}_{n-1}) + \mathbf{W}_n (u_n - \mu_u) \end{aligned}$$

with $\mathbf{F}_n$ and $\mathbf{W}_n$ the Jacobians of f with respect to x_{n-1} and u_n, evaluated at μ_{n-1} and μ_u, respectively. The EKF augments the state as

$$\boldsymbol{\mu}_n = \begin{bmatrix} \boldsymbol{\mu}_{1:n-1} \\ x_n \end{bmatrix}, \tag{2}$$

$$\boldsymbol{\Sigma}_n = \begin{bmatrix} \boldsymbol{\Sigma}_{1:n-2\;1:n-2} & \boldsymbol{\Sigma}_{1:n-2\;n-1}\, \mathbf{F}_n \\ \mathbf{F}_n^\top \boldsymbol{\Sigma}_{n-1\;1:n-2} & \mathbf{F}_n\, \boldsymbol{\Sigma}_{n-1\;n-1}\, \mathbf{F}_n^\top + \mathbf{Q} \end{bmatrix}, \tag{3}$$

with $\mathbf{Q} = \mathbf{W}_n \boldsymbol{\Sigma}_u \mathbf{W}_n^\top$ and where $\boldsymbol{\Sigma}_{n-1\;n-1}$ is used to denote the block of $\boldsymbol{\Sigma}_{n-1}$ corresponding to the $(n-1)$-th pose, and $\boldsymbol{\mu}_{1:n-1}$ and $\boldsymbol{\Sigma}_{1:n-1\;1:n-1}$ indicate the blocks ranging from the first to the $(n-1)$-th pose.

Each set of measures $\mathbf{y}_n = \{y_n^i, \dots, y_n^k\}$ constrains the relative position of the last pose to some other poses from the robot trajectory forming loops. The measurement model for each of these constraints is

$$\begin{aligned} y_n^i &= h(x_i, x_n) \\ &\approx h(\mu_i, \mu_n) + \mathbf{H}(\mathbf{x}_n - \boldsymbol{\mu}_n) + \mathbf{v}_n, \end{aligned}$$

where h gives $x_i - x_n$ in the reference frame of x_i, and $\mathbf{H}$ is

$$\mathbf{H} = [\mathbf{0} \,\dots \mathbf{0}\; \mathbf{H}_i\; \mathbf{0} \,\dots\, \mathbf{0}\; \mathbf{H}_n], \tag{4}$$

with $\mathbf{H}_i$ and $\mathbf{H}_n$ the Jacobians of h with respect to x_i and x_n, and $\mathbf{v}_n \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_y)$ the measurement white noise.

The information from observation y_n^i is merged into the filter applying the following increments

$$\Delta\boldsymbol{\mu} = \mathbf{K}\, (y_n^i - h(\mu_i, \mu_n)), \tag{5}$$

$$\Delta\boldsymbol{\Sigma} = -\mathbf{K}\, \mathbf{H}\, \boldsymbol{\Sigma}_n \tag{6}$$

to $\boldsymbol{\mu}_n$ and $\boldsymbol{\Sigma}_n$, respectively, where $\mathbf{K}$ is the Kalman gain, $\mathbf{K} = \boldsymbol{\Sigma}_n\, \mathbf{H}^\top\, \mathbf{S}^{-1}$, and with $\mathbf{S}$ the innovation matrix, $\mathbf{S} = \mathbf{H}\, \boldsymbol{\Sigma}_n\, \mathbf{H}^\top + \boldsymbol{\Sigma}_y$.

Measurements y_n^i result from the data association process. Instead of directly comparing the sensor readings for the current pose with all those along the trajectory, data association is generally tested on a limited region of the trajectory. To identify poses that are close enough to the current one so that the corresponding sensor readings are likely to match (i.e., to produce y_n^i observations), we can estimate the relative displacement, d, from the current robot pose, x_n, to any other previous pose in the trajectory, x_i, as a Gaussian with parameters

$$\mu_d = h(\mu_i, \mu_n), \tag{7}$$

$$\boldsymbol{\Sigma}_d = [\mathbf{H}_i\; \mathbf{H}_n] \begin{bmatrix} \boldsymbol{\Sigma}_{ii} & \boldsymbol{\Sigma}_{in} \\ \boldsymbol{\Sigma}_{in}^\top & \boldsymbol{\Sigma}_{nn} \end{bmatrix} [\mathbf{H}_i\; \mathbf{H}_n]^\top, \tag{8}$$

where $\boldsymbol{\Sigma}_{in}$ is the cross correlation between the i-th and the current poses. Only poses whose relative displacement, d, is likely to be inside sensor range need to be considered for sensor registration.

Whereas the EKF estimation maintains all the data necessary for linearization and for data association, its drawback is that storing and updating the whole covariance matrix entails quadratic cost both in memory and in execution time.

B. EIF State Estimation

In the EIF form of Pose SLAM the state is augmented as

$$\boldsymbol{\eta}_n = \begin{bmatrix} \boldsymbol{\eta}_{1:n-2} \\ \eta_{n-1} - \mathbf{F}_n{}^\top \mathbf{Q}^{-1}\, (f(\mu_{n-1}, \mu_u) - \mathbf{F}_n\, \mu_{n-1}) \\ \mathbf{Q}^{-1}\, (f(\mu_{n-1}, \mu_u) - \mathbf{F}_n\, \mu_{n-1}) \end{bmatrix},$$

$$\boldsymbol{\Lambda}_n = \begin{bmatrix} \boldsymbol{\Lambda}_{1:n-2\;1:n-2} & \boldsymbol{\Lambda}_{1:n-2\;n-1} & \mathbf{0} \\ \boldsymbol{\Lambda}_{n-1\;1:n-2} & \boldsymbol{\Lambda}_{n-1\;n-1} + \mathbf{F}_n^\top \mathbf{Q}^{-1} \mathbf{F}_n & -\mathbf{F}_n^\top \mathbf{Q}^{-1} \\ \mathbf{0} & -\mathbf{Q}^{-1} \mathbf{F}_n & \mathbf{Q}^{-1} \end{bmatrix}. \tag{9}$$

The information from observation y_n^i is fed to the filter by adding the following increments

$$\begin{aligned}\Delta\boldsymbol{\eta} &= \mathbf{H}^\top\, \boldsymbol{\Sigma}_y^{-1}((y_n^i - h(\mu_i,\mu_n) + \mathbf{H}\,\boldsymbol{\mu}_n),\\ \Delta\boldsymbol{\Lambda} &= \mathbf{H}^\top \boldsymbol{\Sigma}_y^{-1}\mathbf{H} \end{aligned} \tag{10}$$

to $\boldsymbol{\eta}_n$ and $\boldsymbol{\Lambda}_n$, respectively.

Equation (9) defines a block-tridiagonal matrix and Eq. (10) only adds off-diagonal elements to the positions corresponding to the two poses directly related by the observation y_n^i, preserving the sparsity of $\boldsymbol{\Lambda}$. Thus, the memory requirements for the information-based representation can be considered linear with the number of poses for practical purposes.

Notice, however, that the Jacobians above have to be evaluated at the state mean which is not directly available in the information representation. Moreover, the displacement measure in Eqs. (7) and (8) used for data association requires marginalising out some blocks of the covariance matrix (its block diagonal and the last column) which are also not available in the information representation. On the one hand, the mean can be recovered solving the following linear system

$$\boldsymbol{\Lambda}_n\, \boldsymbol{\mu}_n = \boldsymbol{\eta}_n,$$

that using sparse Cholesky factorization [1] can be solved in linear time for realistic problems. On the other hand, the covariance can be recovered solving

$$\boldsymbol{\Lambda}_n\, \boldsymbol{\Sigma}_n = \mathbf{I},$$

with $\mathbf{I}$ the identity matrix. Sparse Cholesky factorization also allows to solve this system efficiently but with quadratic memory cost to store $\boldsymbol{\Sigma}_n$, which is not sparse. This quadratic memory cost can be alleviated by solving n independent systems, one for each block column of the covariance matrix, $\mathbf{T}_i$, $i \in \{1,\ldots,n\}$

$$\boldsymbol{\Lambda}_n\, \mathbf{T}_i = \mathbf{I}_i \tag{11}$$

where $\mathbf{I}_i$ is the sparse block column matrix with an identity block only at the position corresponding to pose i. In this way space complexity is linear, but time complexity is still quadratic.

III. Pose SLAM with a Mixed Kalman-Information Representation

To obtain a state recovery strategy that scales linearly both in execution time and in memory usage we propose a mixed Kalman-information representation. We store the state mean, $\boldsymbol{\mu}_n$, the block-diagonal and the block-last column of the covariance matrix, $\mathbf{D}_n$ and $\mathbf{T}_n$ respectively, and the information matrix $\boldsymbol{\Lambda}_n$. The mean is used to evaluate Jacobians, $\mathbf{D}_n$ and $\mathbf{T}_n$ are used for data association, and $\boldsymbol{\Lambda}_n$ stores in a very compact way the full set of correlations between all poses that are necessary to propagate the effects of each loop closure all over the trajectory. Neither the rest of entries in $\boldsymbol{\Sigma}_n$ nor the information vector $\boldsymbol{\eta}_n$ are maintained. The largest stored element is $\boldsymbol{\Lambda}_n$ that, as mentioned before, scales linearly with the number of states. Therefore, the whole representation scales linearly.

During state augmentation, $\boldsymbol{\mu}_n$ is enlarged with Eq. (2), the new blocks of $\mathbf{D}_n$ and $\mathbf{T}_n$ are computed using the relevant parts of Eq. (3) and $\boldsymbol{\Lambda}_n$ is extended as in Eq. (9). Augmenting the mean block-vector, $\boldsymbol{\mu}_n$, and the block-vector of diagonal covariance entries, $\mathbf{D}_n$, can be done in constant time since only a new block is added. However, state augmentation produces a new block column $\mathbf{T}_n$ with $n-1$ elements. Therefore updating $\mathbf{T}_n$ has linear computational cost. Data association can be carried out at the same time $\mathbf{T}_n$ is updated, with linear complexity as well.

When establishing a link between the last pose and any i-th pose from the trajectory one can realize that, due to the sparse form of the Jacobian $\mathbf{H}$ in Eq. (4), Eqs. (5) and (6) do not use the full covariance matrix but only $\mathbf{D}_n$, $\mathbf{T}_n$ and the i-th block columns of the covariance matrix, $\mathbf{T}_i$. Notice that $\mathbf{T}_i$ is the only element missing from our representation. However, it can be obtained in linear time by solving the system in Eq. (11).

Furthermore, applying Cholesky decomposition to the inverse of the Kalman innovation $\mathbf{S}^{-1} = \mathbf{V}^\top\mathbf{V}$ we define the block column matrix

$$\mathbf{B} = \boldsymbol{\Sigma}\mathbf{H}^\top\mathbf{V}^\top,$$

that considering Eq. (4) becomes

$$\mathbf{B} = [\mathbf{T}_i\;\mathbf{T}_n]\begin{bmatrix}\mathbf{H}_i^\top\\ \mathbf{H}_n^\top\end{bmatrix}\mathbf{V}^\top.$$

With this, the mean can be updated as in Eq. (5) with $\mathbf{K} = \mathbf{B}\,\mathbf{V}$ and the block diagonal entries of the new covariance matrix can be updated adding the following increment to $\mathbf{D}_n$

$$\Delta\mathbf{D} = -\begin{bmatrix}\mathbf{B}_1\,\mathbf{B}_1^\top\\ \vdots\\ \mathbf{B}_n\,\mathbf{B}_n^\top\end{bmatrix},$$

where $\mathbf{B}_i$ is the i-th block row of $\mathbf{B}$. Moreover, $\mathbf{T}_n$ is updated with

$$\Delta\mathbf{T} = -\mathbf{B}\,\mathbf{B}_n^\top.$$

Finally, the information matrix is updated as in Eq. (10).

This process is applied for all loops closed at the same time slice. In practice and due to sensor limitations, a bounded number of loops per step are closed and, therefore, the whole state update process scales linearly in time and memory.

The new approach avoids quadratic memory requirements of an EKF by maintaining the state in information form while, at the same time, it allows direct access to the mean and the covariance entries needed for data association. In this way, the proposed filtering scheme gets the best of the two worlds, Kalman state availability and information filter sparsity.

IV. Open Loop State Recovery in Constant Time

The mixed Kalman-information approach presented above can be applied regardless of the number of asserted loop closures, giving linear time execution per time slice. However, in many cases loops are scarcely closed. For instance in Pose SLAM, in order to avoid filter inconsistency as much as possible, it is desired to close only highly informative loops [8]. The same happens in hierarchical SLAM where many loops are formed inside local maps but few are formed at the inter-map level. In these cases, the robot operates most of the time in exploration mode, when the most expensive step is that of updating $\mathbf{T}_n$. However, in this situation this cost can be reduced by factoring $\mathbf{T}_n$ as

$$\mathbf{T}_n = \mathbf{\Phi}_n \, \mathbf{G}_n .$$

This factorization can be updated in constant time as follows.

Suppose a loop closure occurred at time l and that at that time the procedure presented in the previous section is used to compute $\mathbf{T}_l$. At this point we define $\mathbf{\Phi}_l = \mathbf{T}_l$ and $\mathbf{G}_l = \mathbf{I}$, with $\mathbf{I}$ the identity matrix. After the loop is closed, when the robot moves to a new pose x_n, $n > l$, we compute $\mathbf{\Phi}_n$ and $\mathbf{G}_n$ as

$$\mathbf{\Phi}_n = \begin{bmatrix} \mathbf{\Phi}_{1:n-1} \\ \mathbf{\Sigma}_{n-1\,n-1}\,\mathbf{G}_{n-1}^{-1} \end{bmatrix},$$
$$\mathbf{G}_n = \mathbf{G}_{n-1}\,\mathbf{F}_n,$$

where $\mathbf{\Sigma}_{n-1\,n-1}$ is the last block of $\mathbf{D}_{n-1}$, and $\mathbf{F}_n$ is the Jacobian of the state transition function f at time n. With this factorization we do not need to store $\mathbf{T}_n$ since by book-keeping $\mathbf{\Phi}_n$ and $\mathbf{G}_n$, any block of $\mathbf{T}_n$ can be computed in constant time when required in the data association process.

The constant time open loop update prompts the necessity to perform data association in times better than linear. This can be done, for instance, in logarithmic time per iteration using a KD-tree [18] or even in constant time using grid techniques when covariances are bounded [15].

V. Experiments and Results

This section describes experiments to validate the presented Kalman-information filtering approach applied to Pose SLAM, first using synthetic data and then using a real dataset obtained form a public repository.

In the first experiment, we simulate a robot moving about 0.8 m per step looping around two concentric ellipses, the first with semi-axes 10 m and 6 m and the second with semi-axes 20 m and 6 m. In the simulation, the motion of the robot is measured with an odometric sensor whose error is 5% of the displacement in x and y, and 0.0175 rad in orientation. A second sensor is able to establish a link between any two poses closer than ± 3 m in x and y, and ± 0.26 rad in orientation, respectively. This sensor has a noise covariance of $\mathbf{\Sigma}_y = \text{diag}(0.2, 0.2, 0.009)^2$. The simulation is implemented in Matlab running under Linux on a Intel Core 2 at 2.4 GHz.

Fig. 1 shows the result obtained when incorporating all possible loop closure links. We compare the loop closure state update proposed in this paper with the two alternative methods described in Section II-B. Fig. 2 shows the execution time and the memory footprint for the three approaches. The blue dotted-lines depict the time and memory requirements when recovering the whole covariance matrix $\mathbf{\Sigma}$. The red dashed-lines show the time and memory requirements for the strategy which recovers each block column of the covariance matrix solving a sequence of linear systems, one at a time. The results corresponding to the method introduced in Section III are

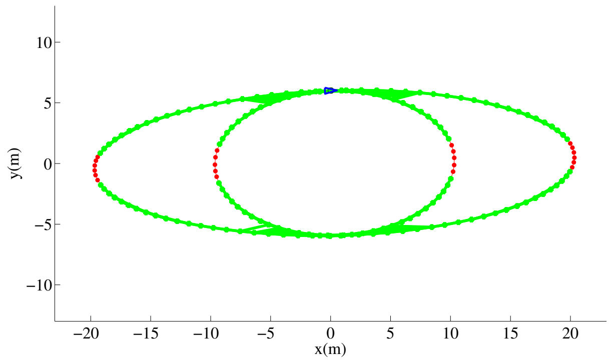
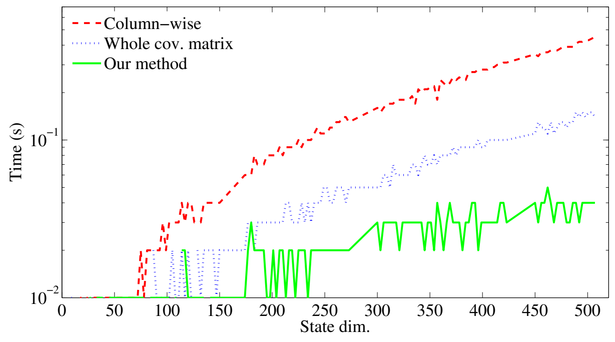
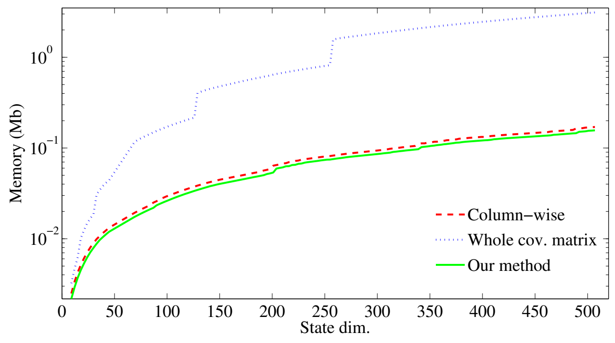


Fig. 1. Simulated trajectory when closing all possible loops. The trajectory is shown in red and the links forming loops are shown in green.



(a) Execution time.



(b) Memory footprint.

Fig. 2. Execution time and memory footprint for different state recovery strategies when closing a loop in the simulated experiment.

shown in green. In all cases, linear systems are solved using supernodal sparse Cholesky factorization [1]. Note from the plot that the time needed to recover the whole $\mathbf{\Sigma}$ is smaller than that of solving separate systems per each block column of $\mathbf{\Sigma}$ due to the extra cost of defining the different linear systems to be solved in this second case. However, the memory requirements to solve the whole $\mathbf{\Sigma}$ increase much faster than when solving the systems column-wise. The method that recovers the whole $\mathbf{\Sigma}$ is too memory demanding to be applied to large mapping problems. In contrast, the execution time and memory usage of our strategy outperforms the two other methods in both aspects, time and memory usage.

When carefully selecting the loops to be closed using for instance, information-based criteria [8], the robot operates most of the time in open loop. Thus, we can take advantage

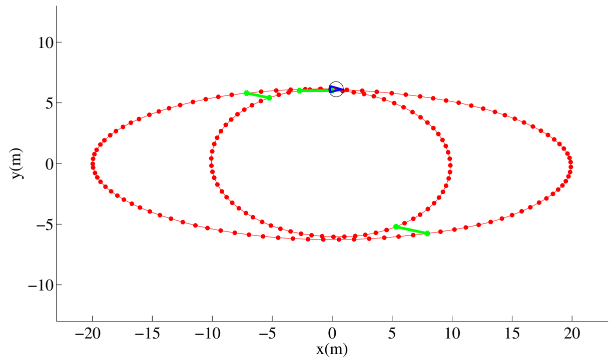


Fig. 3. Simulated trajectory when carefully selecting the loops to close using information-based criteria. The trajectory is shown in red and the loop closure links in green.

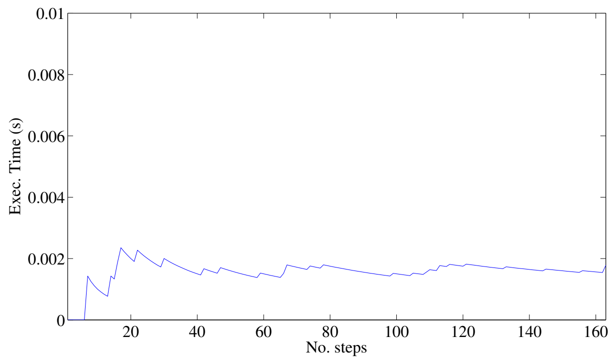


Fig. 4. Amortized execution time when controlling the number of loop closures in the simulated experiment.

of the factorization proposed in Section IV. Fig. 3 shows the result of a simulation of the same experiment as that in Fig. 1 when using this strategy. Fig. 4 shows the amortized cost, c_i for each step, i, computed as

$$c_i = \frac{1}{i}\sum_{k=1}^{i} t_k, \tag{12}$$

where t_k only includes the time for filter related operations (including the time to compute $\boldsymbol{\mu}$, $\mathbf{D}$, $\mathbf{\Phi}$, and $\mathbf{\Lambda}$) at time slice k, disregarding the cost of sensor registration. As expected, the plot indicates that the amortized time is almost constant for the entire experiment.

To test the performance of the proposed approach in larger problems, we used the Intel dataset from [7]. The dataset includes 26915 odometry readings and 13631 laser scans. The laser scans are used to generate sensor-based odometry and to assert loop closures aligning them using an ICP scan matching algorithm [12]. The robot odometry and the laser scan match are modelled with noise covariances $\mathbf{\Sigma}_u = \mathrm{diag}(0.05, 0.05, 0.03)^2$ and $\mathbf{\Sigma}_y = \mathrm{diag}(0.05, 0.05, 0.009)^2$, respectively. Finally, the covariance of the initial pose is set to $\mathbf{\Sigma}_{00} = \mathrm{diag}(0.1, 0.1, 0.09)^2$. Due to its large size, this dataset is typically pre-processed and reduced to about 1000 poses with about 3500 loop closure links [10]. By carefully selecting the most informative loops [8] we only establish about 100 links. Fig. 5 shows the final estimated trajectory in this case.

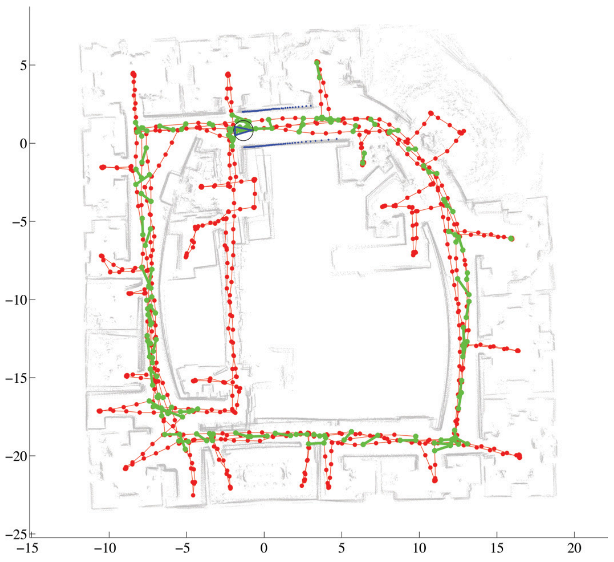


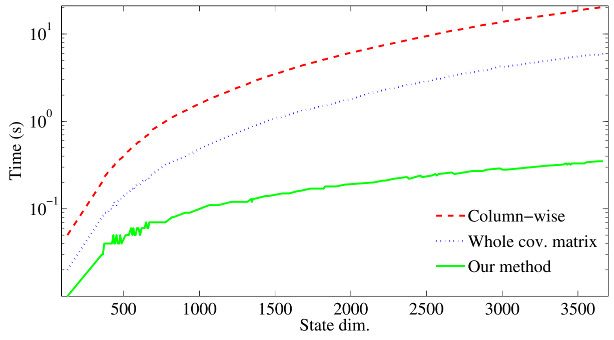
Fig. 5. Filtered trajectory using encoder and laser odometry of the Intel dataset. The blue arrow indicates the final pose of the robot and the black ellipse the associated covariance at a 95% confidence level.

Fig. 6 shows the execution time and memory footprint at each step using the different state recovery strategies for loop closure discussed in this paper: recovering the whole $\mathbf{\Sigma}$, recovering it column-wise, and the method proposed in this paper. The result confirms that for larger SLAM problems, our method clearly outperforms the two other methods both in memory usage and in execution time. The result also confirms that the hypothesis behind our approach hold even for large problems where the robot re-traverses many times the same places.

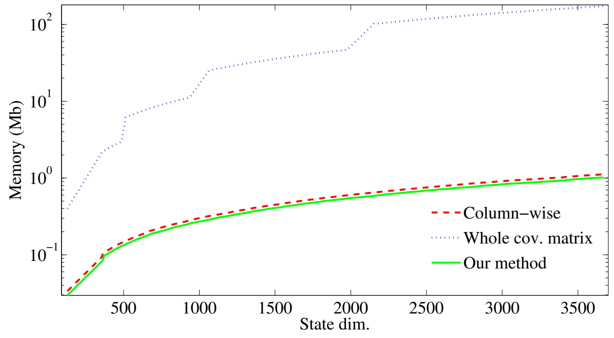
Fig. 7 shows the amortized time for the whole execution on the Intel experiment when using a restrictive policy for loop closure. The amortized cost is almost constant which makes the total cost of the SLAM system linear with the number of iterations, taking into account state estimation but without considering sensor registration.

VI. Conclusions

The problem of estimating a set of reference frames with relative constraints between them is a fundamental problem in SLAM. It appears, for instance, in Pose SLAM where reference frames are attached to each one of the poses along the robot trajectory or in hierarchical SLAM where reference frames are attached to each submap. When assuming Gaussian distributions, the Kalman and the information filters are the two alternative filtering schemes that have been applied to this problem. In the Kalman filter the mean and the covariance are directly available for linearization and data association, but at the cost of quadratic memory and time complexity. The information filter offers linear memory cost, but to linearize the state transition and observation models and to perform prior-based data association, the mean and the covariance need to be recovered from the information vector and the information matrix. This can only be achieved at the cost of quadratic memory (when recovering the whole covariance



(a) Execution time.



(b) Memory footprint.

Fig. 6. Execution time and memory footprint for different state recovery strategies when closing a loop in the Intel experiment.

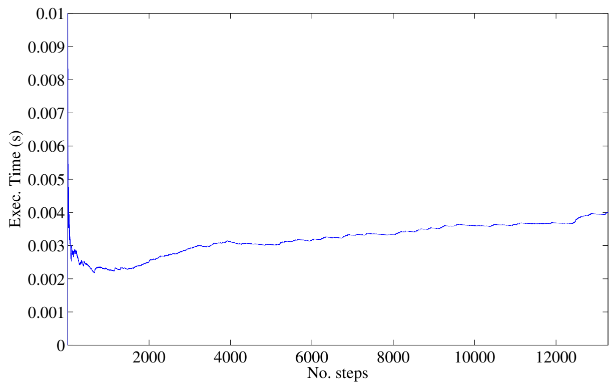


Fig. 7. Amortized execution time when controlling the number of loop closures in the Intel experiment.

matrix) or quadratic execution time (when recovering the marginal covariances for each pose one at a time).

In this paper we proposed a mixed Kalman-information approach in which we store and maintain the state mean, the relevant block entries of the covariance matrix (its block diagonal and its block last column) and the information matrix. The mean and the covariance entries are used to linearize the system when necessary and to perform data association. The information matrix stores in a very compact way the whole set of correlations between the poses. The result is an estimation mechanism that scales linearly both in memory and in execution time. Moreover, both in Pose SLAM and in hierarchical mapping, it is typical to operate most of the time in open loop while exploring new areas or when defining new submaps, as well as to establish only few constraints between the current robot pose (or current submap) and previous poses (or submaps). We have shown that this particular property can be exploited to derive a system whose amortized cost per step is constant instead of linear. The presented results using both simulated experiments and standard SLAM data sets validate the approach.

Acknowledgment

This work has been partially supported by the Spanish Ministry of Science and Innovation under a Juan de la Cierva Postdoctoral Fellowship to V. Ila and the projects DPI-2007-60858, DPI-2008-06022, MIPRCV Consolider-Ingenio 2010, and the EU URUS project IST-FP6-STREP-045062.

References

[1] Y. Chen, T. A. Davis, W. W. Hager, and S. Rajamanickam. Algorithm 887: CHOLMOD, supernodal sparse Cholesky factorization and update/downdate. *ACM T. Math. Soft.*, 35(3), 2008.

[2] M. Cummins and P. Newman. FAB-MAP: Probabilistic localization and mapping in the space of appearance. *Int. J. Robot. Res.*, 27(6):647–665, 2008.

[3] F. Dellaert and M. Kaess. Square root SAM: Simultaneous localization and mapping via square root information smoothing. *Int. J. Robot. Res.*, 25(12):1181–1204, 2006.

[4] C. Estrada, J. Neira, and J.D. Tardós. Hierarchical SLAM: Real-time accurate mapping of large environments. *IEEE Trans. Robot.*, 21(4):588–596, Aug. 2005.

[5] R. M. Eustice, H. Singh, and J. J. Leonard. Exactly sparse delayed-state filters for view-based SLAM. *IEEE Trans. Robot.*, 22(6):1100–1114, Dec. 2006.

[6] G. Grisetti, C. Stachniss, S. Grzonka, and W. Burgard. A tree parameterization for efficiently computing maximum likelihood maps using gradient descent. In *Robotics: Science and Systems III*, Atlanta, Jun. 2007.

[7] A. Howard and N. Roy. The robotics data set repository (Radish). `http://radish.sourceforge.net`, 2003.

[8] V. Ila, J. Andrade-Cetto, R. Valencia, and A. Sanfeliu. Vision-based loop closing for delayed state robot mapping. In *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, pages 3892–3897, San Diego, Nov. 2007.

[9] V. Ila, J. M. Porta, and J. Andrade-Cetto. Information-based compact Pose SLAM. *IEEE Trans. Robot.*, 2009. To appear.

[10] M. Kaess, A. Ranganathan, and F. Dellaert. iSAM: Incremental smoothing and mapping. *IEEE Trans. Robot.*, 24(6):1365–1378, 2008.

[11] K. Konolige and M. Agrawal. FrameSLAM: from bundle adjustment to realtime visual mapping. *IEEE Trans. Robot.*, 24(5):1066–1077, 2008.

[12] F. Lu and E. Milios. Globally consistent range scan alignment for environment mapping. *Auton. Robot.*, 4(4):333–349, 1997.

[13] M. Montemerlo and S. Thrun. *FastSLAM: A Scalable Method for the Simultaneous Localization and Mapping Problem in Robotics*, volume 27 of *Springer Tracts in Advanced Robotics*. Springer, 2007.

[14] E. Olson, J. Leonard, and S. Teller. Fast iterative optimization of pose graphs with poor initial estimates. In *Proc. IEEE Int. Conf. Robot. Automat.*, pages 2262–2269, Orlando, May 2006.

[15] L. M. Paz, J. D. Tardós, and J. Neira. Divide and conquer: EKF SLAM in $O(n)$. *IEEE Trans. Robot.*, 24(5):1107–1120, 2008.

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

[17] 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. Robot. Res.*, 23(7-8):693–716, Jul. 2004.

[18] J. Uhlmann. Introduction to the algorithmics of data association in multiple-target tracking. In M. E. Liggins, D. E. Hall, and J. Llinas, editors, *Handbook of Multisensor Data Fusion*. CRC Press, Boca Raton, 2001.

[19] M. R. Walter, R. M. Eustice, and J. J. Leonard. Exactly sparse extended information filters for feature-based SLAM. *Int. J. Robot. Res.*, 26(4):335–359, 2007.

[20] Z. Wang, S. Huang, and G. Dissanayake. D-SLAM: A decoupled solution to simultaneous localization and mapping. *Int. J. Robot. Res.*, 26(2):187–204, 2007.