

An Observability-Constrained Sliding Window Filter for SLAM

Guoquan P. Huang, Anastasios I. Mourikis, and Stergios I. Roumeliotis

Abstract—A sliding window filter (SWF) is an appealing smoothing algorithm for nonlinear estimation problems such as simultaneous localization and mapping (SLAM), since it is resource-adaptive by controlling the size of the sliding window, and can better address the nonlinearity of the problem by relinearizing available measurements. However, due to the marginalization employed to discard old states from the sliding window, the standard SWF has different parameter observability properties from the optimal batch maximum-a-posteriori (MAP) estimator. Specifically, the nullspace of the Fisher information matrix (or Hessian) has lower dimension than that of the batch MAP estimator. This implies that the standard SWF acquires spurious information, which can lead to inconsistency. To address this problem, we propose an observability-constrained (OC)-SWF where the linearization points are selected so as to ensure the correct dimension of the nullspace of the Hessian, as well as minimize the linearization errors. We present both Monte Carlo simulations and real-world experimental results which show that the OC-SWF’s performance is superior to the standard SWF, in terms of both accuracy and consistency.

I. INTRODUCTION AND RELATED WORK

Among the existing approaches for robot localization, the extended Kalman filter (EKF) is one of the most popular methods. This is primarily due to its ease of implementation and relatively low processing requirements. However, the EKF, as well as any linearization-based filtering approach, may suffer from the accumulation of linearization errors. This is because once linearization points are selected at a given time step for computing the filter Jacobians, they cannot be updated at later times, when more measurements become available for improving them. In contrast, a batch maximum a posteriori (MAP) estimator [1] can improve the estimation accuracy by computing consistent state estimates for all time steps based on all available measurements. Under a Gaussian prior and measurement noise assumption (which is common in practice), finding the MAP estimates requires solving a nonlinear least-squares problem (see Section II), whose counterpart in computer vision is known as bundle adjustment [2]. A variety of iterative algorithms have been employed for this problem. For example, the square-root smoothing and mapping (SAM) method [3] solves the SLAM problem efficiently by using variable reordering, a well-known technique for sparse linear systems. However, since the size of the state vector in the batch-MAP estimator

increases continuously over time, the processing and memory requirements become too high for real-time operation in large-scale problems (e.g., a robot exploring a large environment with millions of landmarks).

To overcome this limitation, a sliding window filter (SWF) [4] (also called a fixed-lag smoother (FLS) [5]–[7]) can be used to estimate the states over a sliding time window at a fixed computational cost. The SWF concurrently processes all the measurement constraints between states in the window, and better addresses the nonlinearity of the problem by iteratively relinearizing the process and measurement equations. This approach is resource-adaptive: depending on the available computational resources, it can scale from the iterated EKF solution if only a single time step is maintained, to the optimal batch-MAP solution if the sliding window spans the entire time horizon.

The key characteristic of the SWF is the marginalization of old states from the sliding window, a process that appropriately models the uncertainty of these states [2], [4], [6]. However, due to marginalization, *different estimates* of the *same states* are used as *linearization points* in computing the Hessian matrix during estimation (see Section III and [7]). This results in different parameter observability properties [8] as compared to the batch-MAP estimator. Specifically, the Hessian (Fisher information matrix) of the standard SWF has a nullspace of lower dimension than that of the batch-MAP estimator. This implies that the estimator erroneously *believes* it has information along more directions of the state space than those contained in the measurements. This leads to inconsistent estimates, i.e., estimates whose accuracy is worse than the one reported by the estimator. This inconsistency is a serious problem, since when an estimator is inconsistent, the accuracy of the produced estimates is unknown, which in turn makes the estimator unreliable [8].

In order to improve the consistency and accuracy of the SWF, in this paper we propose an observability-constrained (OC)-SWF as a general smoothing framework. In particular, we postulate that by ensuring the Hessian matrix has a nullspace of appropriate dimension, we can avoid the influx of spurious information in the unobservable directions of the parameter (state) space, thus improving the consistency of the estimates. Based on this, we develop an OC-SWF which extends the observability-based methodology for designing consistent EKFs [9]. The key idea behind our approach is to select the linearization points for computing the Jacobians, and hence the Hessian, so as to ensure that its nullspace dimension does not arbitrarily decrease.

It should be pointed out that a prior-linearization (PL)-SWF for motion estimation was proposed in [7]. In particular, the PL-SWF computes the Hessian using the prior,

This work was supported by the University of Minnesota (DTC), the UC Riverside Bourns College of Engineering, and the National Science Foundation (IIS-0643680, IIS-0811946, IIS-0835637, IIS-1117957).

G. P. Huang and S. I. Roumeliotis are with the Department of Computer Science and Engineering, University of Minnesota, Minneapolis, MN 55455, USA. Email: {ghuang|stergios}@cs.umn.edu

A. I. Mourikis is with the Department of Electrical Engineering, University of California, Riverside, CA 92521, USA. Email: mourikis@ee.ucr.edu

instead of the current, estimates, for the states connected via measurements to marginalized states. This ensures the same estimates for the same states are used, and the appropriate dimension of the Hessian's nullspace is preserved. However, if the prior estimates are inaccurate, the linearization errors will be large and may degrade the estimator's performance. In contrast, the proposed OC-SWF selects *optimal* linearization points for computing the Hessian, in the sense that they not only ensure the correct dimension for the nullspace of the Hessian, but also minimize the linearization errors. We stress that apart from the SLAM problem treated in this paper, the proposed OC-SWF is applicable to a large class of nonlinear estimation problems in robotics and computer vision, such as visual odometry [10] and vision-aided inertial navigation [11].

The remainder of the paper is structured as follows: After presenting the batch-MAP formulation of the SLAM problem in the next section, we describe the standard SWF and its parameter observability properties in Section III. The proposed OC-SWF is presented in Section IV, whose performance is compared against that of the standard SWF and the PL-SWF through Monte Carlo simulations and real-world experiments in Sections V and VI. Finally, Section VII outlines the main conclusions of this work and possible directions of future work.

II. SLAM BATCH-MAP FORMULATION

In this section, we describe the batch-MAP formulation of the SLAM problem, which forms the basis for the ensuing derivations of the SWF. In particular, we aim at estimating the entire robot trajectory up to the current time-step k , as well as the positions of all observed landmarks:¹

$$\mathbf{x}_{0:k} = [\mathbf{x}_{R_0}^T \quad \mathbf{x}_{R_1}^T \quad \cdots \quad \mathbf{x}_{R_k}^T \quad \mathbf{p}_{L_1}^T \quad \cdots \quad \mathbf{p}_{L_M}^T]^T \quad (1)$$

where $\mathbf{x}_{R_k} = [\mathbf{p}_{R_k}^T \quad \phi_{R_k}]^T$ denotes the robot pose (position and orientation) at time-step k , and \mathbf{p}_{L_i} is the position of the i -th landmark.

In what follows, we start by presenting the general motion and measurement models that will be used throughout the paper. Subsequently, we describe the batch-MAP estimator.

A. Motion Model

Consider a robot equipped with an odometry sensor moving on a plane. The odometry serves as the control input to propagate the robot pose, according to the following motion model:

$$\mathbf{p}_{R_k} = \mathbf{p}_{R_{k-1}} + \mathbf{C}(\phi_{R_{k-1}})^{R_{k-1}} \mathbf{p}_{R_k} \quad (2)$$

$$\phi_{R_k} = \phi_{R_{k-1}} + {}^{R_{k-1}}\phi_{R_k} \quad (3)$$

where $\mathbf{C}(\cdot)$ denotes the 2×2 rotation matrix, and $\mathbf{u}_{k-1} = {}^{R_{k-1}}\mathbf{x}_{R_k} = [{}^{R_{k-1}}\mathbf{p}_{R_k}^T \quad {}^{R_{k-1}}\phi_{R_k}]^T$ is the true odometry (control input), i.e., the robot's motion between time-steps

$k-1$ and k , expressed with respect to the robot's frame at time-step $k-1$, $\{R_{k-1}\}$. The corresponding odometry measurement, $\mathbf{u}_{m_{k-1}}$, is assumed to be corrupted by zero-mean, white Gaussian noise, $\mathbf{w}_{k-1} = \mathbf{u}_{k-1} - \mathbf{u}_{m_{k-1}}$, with covariance \mathbf{Q}_{k-1} . This motion model is described by the following generic nonlinear function:

$$\mathbf{g}(\mathbf{x}_{0:k}, \mathbf{u}_{k-1}) = \mathbf{x}_{R_k} - \mathbf{f}(\mathbf{x}_{R_{k-1}}, \mathbf{u}_{m_{k-1}} + \mathbf{w}_{k-1}) = \mathbf{0} \quad (4)$$

To employ a batch-MAP estimator, it is necessary to linearize (4) and compute the Jacobians with respect to the state vector (1) and the noise, respectively, i.e.,

$$\Phi_{k-1} \triangleq \left. \frac{\partial \mathbf{g}}{\partial \mathbf{x}_{0:k}} \right|_{\{\mathbf{x}_{0:k}^*, \mathbf{0}\}} \quad (5)$$

$$= [\mathbf{0}_{3 \times 3} \quad \cdots \quad \Phi_{R_{k-1}} \quad \mathbf{I}_3 \quad \mathbf{0}_{3 \times 2} \quad \cdots \quad \mathbf{0}_{3 \times 2}]$$

$$\mathbf{G}_{k-1} \triangleq \left. \frac{\partial \mathbf{g}}{\partial \mathbf{w}_{k-1}} \right|_{\{\mathbf{x}_{0:k}^*, \mathbf{0}\}} = \begin{bmatrix} \mathbf{C}(\phi_{R_{k-1}}^*) & \mathbf{0}_{2 \times 1} \\ \mathbf{0}_{1 \times 2} & 1 \end{bmatrix} \quad (6)$$

with

$$\Phi_{R_{k-1}} = - \begin{bmatrix} \mathbf{I}_2 & \mathbf{J}(\mathbf{p}_{R_k}^* - \mathbf{p}_{R_{k-1}}^*) \\ \mathbf{0}_{1 \times 2} & 1 \end{bmatrix} \quad (7)$$

where $\mathbf{x}_{0:k}^*$ denotes the linearization point for the state (1), while a zero vector is used as the linearization point for the noise, and $\mathbf{J} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$. Clearly, the values of the Jacobian matrices depend on the choice of linearization points, which is the key fact our approach relies on. Note also that the form of the motion model presented above is general, and holds for any robot kinematic model (e.g., unicycle, bicycle, or Ackerman model) [9].

B. Measurement Model

During SLAM, the robot-to-landmark measurements are a function of the relative position of the observed landmark with respect to the robot:

$$\mathbf{z}_{ij} = \mathbf{h}_{ij}(\mathbf{x}_{0:k}) + \mathbf{v}_{ij} = \mathbf{h}({}^{R_j}\mathbf{p}_{L_i}) + \mathbf{v}_{ij} \quad (8)$$

where ${}^{R_j}\mathbf{p}_{L_i} = \mathbf{C}^T(\phi_{R_j})(\mathbf{p}_{L_i} - \mathbf{p}_{R_j})$ is the position of the i -th landmark with respect to the robot at time-step j , and \mathbf{v}_{ij} is zero-mean Gaussian measurement noise with covariance \mathbf{R}_{ij} . In this work, we allow $\mathbf{h}(\cdot)$ to be *any* measurement function (e.g., a direct measurement of relative position, a pair of range and bearing measurements, bearing-only measurements, etc.). In general, the measurement function is nonlinear, and its Jacobian matrix is given by:

$$\mathbf{H}_{ij} \triangleq \left. \frac{\partial \mathbf{h}_{ij}}{\partial \mathbf{x}_{0:k}} \right|_{\{\mathbf{x}_{0:k}^*, \mathbf{0}\}} \quad (9)$$

$$= [\mathbf{0} \quad \cdots \quad \mathbf{H}_{R_{ij}} \quad \mathbf{0} \quad \cdots \quad \mathbf{H}_{L_{ij}} \quad \mathbf{0} \quad \cdots \quad \mathbf{0}]$$

with

$$\mathbf{H}_{R_{ij}} = (\nabla \mathbf{h}_{ij}) \mathbf{C}^T(\phi_{R_j}^*) [-\mathbf{I}_2 \quad -\mathbf{J}(\mathbf{p}_{L_i}^* - \mathbf{p}_{R_j}^*)] \quad (10)$$

$$\mathbf{H}_{L_{ij}} = (\nabla \mathbf{h}_{ij}) \mathbf{C}^T(\phi_{R_j}^*) \quad (11)$$

where $\mathbf{H}_{R_{ij}}$ and $\mathbf{H}_{L_{ij}}$ are the Jacobians with respect to the robot pose at time-step j and the i -th landmark position, respectively, and $\nabla \mathbf{h}_{ij}$ denotes the Jacobian of \mathbf{h}_{ij} with respect to the robot-relative landmark position, ${}^{R_j}\mathbf{p}_{L_i}$, evaluated at the linearization point, $\mathbf{x}_{0:k}^*$.

¹Throughout this paper, the subscript $\ell|j$ refers to the estimate of a quantity at time-step ℓ , after all measurements up to time-step j have been processed. \hat{x} is used to denote the estimate of a random variable x , while $\tilde{x} = x - \hat{x}$ is the error in this estimate. $\mathbf{0}_{m \times n}$ and $\mathbf{1}_{m \times n}$ denote $m \times n$ matrices of zeros and ones, respectively, and \mathbf{I}_n is the $n \times n$ identity matrix.

C. Batch-MAP Estimator

The batch-MAP estimator utilizes all the available information to estimate the state vector (1). The information used includes: (i) the prior information about the initial state, described by a Gaussian pdf with mean $\hat{\mathbf{x}}_{0|0}$ and covariance $\mathbf{P}_{0|0}$, (ii) the motion information (4), and (iii) the sensor measurements (8). In particular, the batch-MAP estimator seeks to determine the estimate $\hat{\mathbf{x}}_{0:k|k}$ that maximizes the posterior pdf:

$$p(\mathbf{x}_{0:k}|\mathcal{Z}_{0:k}) \propto p(\mathbf{x}_{R_0}) \prod_{\kappa=1}^k p(\mathbf{x}_{R_\kappa}|\mathbf{x}_{R_{\kappa-1}}) \prod_{\mathbf{z}_{ij} \in \mathcal{Z}_{0:k}} p(\mathbf{z}_{ij}|\mathbf{x}_{R_j}, \mathbf{p}_{L_i}) \quad (12)$$

where $\mathcal{Z}_{0:k}$ denotes all the available measurements in the time interval $[0, k]$. For Gaussian state and measurement noise (see (4), and (8), respectively), maximizing (12) is equivalent to minimizing the following cost function [12]:

$$\begin{aligned} c(\mathbf{x}_{0:k}) &= \frac{1}{2} \|\mathbf{x}_{R_0} - \hat{\mathbf{x}}_{0|0}\|_{\mathbf{P}_{0|0}}^2 \\ &+ \sum_{\kappa=1}^k \frac{1}{2} \|\mathbf{x}_{R_\kappa} - \mathbf{f}(\mathbf{x}_{R_{\kappa-1}}, \mathbf{u}_{m_{\kappa-1}})\|_{\mathbf{Q}'_{\kappa-1}}^2 \\ &+ \sum_{\mathbf{z}_{ij} \in \mathcal{Z}_{0:k}} \frac{1}{2} \|\mathbf{z}_{ij} - \mathbf{h}_{ij}(\mathbf{x}_{0:k})\|_{\mathbf{R}_{ij}}^2 \end{aligned} \quad (13)$$

where $\|\mathbf{a}\|_{\mathbf{M}}^2 = \mathbf{a}^T \mathbf{M}^{-1} \mathbf{a}$ and $\mathbf{Q}'_k = \mathbf{G}_k \mathbf{Q}_k \mathbf{G}_k^T$ (see (4)).

$c(\mathbf{x}_{0:k})$ is a nonlinear function, and a standard approach to determine its minimum is to employ Gauss-Newton iterative minimization [2]. Specifically, at the ℓ -th iteration of this method, a correction, $\delta \mathbf{x}_{0:k}^{(\ell)}$, to the current estimate, $\hat{\mathbf{x}}_{0:k|k}^{(\ell)}$, is computed by minimizing the second-order Taylor-series approximation of the cost function which is given by:

$$c(\hat{\mathbf{x}}_{0:k|k}^{(\ell)} + \delta \mathbf{x}_{0:k}^{(\ell)}) \simeq c(\hat{\mathbf{x}}_{0:k|k}^{(\ell)}) + \mathbf{b}_b^{(\ell)T} \delta \mathbf{x}_{0:k}^{(\ell)} + \frac{1}{2} \delta \mathbf{x}_{0:k}^{(\ell)T} \mathbf{A}_b^{(\ell)} \delta \mathbf{x}_{0:k}^{(\ell)} \quad (14)$$

where

$$\mathbf{b}_b^{(\ell)} \triangleq \nabla_{\mathbf{x}_{0:k}} c(\cdot) \Big|_{\{\mathbf{x}_{0:k}^* = \hat{\mathbf{x}}_{0:k|k}^{(\ell)}\}} \quad (15)$$

$$\mathbf{A}_b^{(\ell)} \triangleq \nabla_{\mathbf{x}_{0:k}}^2 c(\cdot) \Big|_{\{\mathbf{x}_{0:k}^* = \hat{\mathbf{x}}_{0:k|k}^{(\ell)}\}} \quad (16)$$

are the gradient and Hessian of $c(\cdot)$ with respect to $\mathbf{x}_{0:k}$, evaluated at the current state estimate $\hat{\mathbf{x}}_{0:k|k}^{(\ell)}$.

Specifically, at the ℓ -th iteration, $\mathbf{b}_b^{(\ell)}$ is (see (5) and (9)):

$$\begin{aligned} \mathbf{b}_b^{(\ell)} &= \mathbf{\Pi}^T \mathbf{P}_{0|0}^{-1} \left(\hat{\mathbf{x}}_{R_{0|k}}^{(\ell)} - \hat{\mathbf{x}}_{0|0} \right) \\ &- \sum_{\mathbf{z}_{ij} \in \mathcal{Z}_{0:k}} \mathbf{H}_{ij}^{(\ell)T} \mathbf{R}_{ij}^{-1} \left(\mathbf{z}_{ij} - \mathbf{h}_{ij}(\hat{\mathbf{x}}_{0:k|k}^{(\ell)}) \right) \\ &+ \sum_{\kappa=1}^k \mathbf{\Phi}_{\kappa-1}^{(\ell)T} \mathbf{Q}_{\kappa-1}'^{-1} \left(\hat{\mathbf{x}}_{R_\kappa}^{(\ell)} - \mathbf{f}(\hat{\mathbf{x}}_{R_{\kappa-1}|k}^{(\ell)}, \mathbf{u}_{m_{\kappa-1}}) \right) \end{aligned} \quad (17)$$

where $\mathbf{\Pi} = [\mathbf{I}_n \ \mathbf{0} \ \cdots \ \mathbf{0}]$, and $n = \dim(\mathbf{x}_{R_0})$. On the other hand, the Hessian matrix, $\mathbf{A}_b^{(\ell)}$, is approximated in the

Gauss-Newton method by (see (5) and (9)):

$$\begin{aligned} \mathbf{A}_b^{(\ell)} &= \mathbf{\Pi}^T \mathbf{P}_{0|0}^{-1} \mathbf{\Pi} + \\ &\sum_{\mathbf{z}_{ij} \in \mathcal{Z}_{0:k}} \mathbf{H}_{ij}^{(\ell)T} \mathbf{R}_{ij}^{-1} \mathbf{H}_{ij}^{(\ell)} + \sum_{\kappa=1}^k \mathbf{\Phi}_{\kappa-1}^{(\ell)T} \mathbf{Q}_{\kappa-1}'^{-1} \mathbf{\Phi}_{\kappa-1}^{(\ell)} \end{aligned} \quad (18)$$

which is a good approximation for small-residual problems [2]. Due to the sparse structure of the matrices $\mathbf{H}_{ij}^{(\ell)}$ and $\mathbf{\Phi}_{\kappa}^{(\ell)}$ (see (5) and (9)), the matrix $\mathbf{A}_b^{(\ell)}$ is also sparse, which can be exploited to speed-up the solution of the linear system in (19) [2]. The value $\delta \mathbf{x}_{0:k}^{(\ell)}$ that minimizes (14) is found by solving the following linear system:

$$\mathbf{A}_b^{(\ell)} \delta \mathbf{x}_{0:k}^{(\ell)} = -\mathbf{b}_b^{(\ell)} \quad (19)$$

Once $\delta \mathbf{x}_{0:k}^{(\ell)}$ is found, the new state estimate is computed as:

$$\hat{\mathbf{x}}_{0:k|k}^{(\ell+1)} = \hat{\mathbf{x}}_{0:k|k}^{(\ell)} + \delta \mathbf{x}_{0:k}^{(\ell)} \quad (20)$$

Given an initial estimate $\hat{\mathbf{x}}_{0:k|k}^{(0)}$ that resides within the attraction basin of the global optimum, this iterative algorithm will compute the global minimum (i.e., MAP estimate) for the entire state given all measurements up to time-step k .

III. SLIDING WINDOW FILTER (SWF)-BASED SLAM AND OBSERVABILITY PROPERTIES

It is clear from the preceding section that, as the robot continuously moves and observes new landmarks, the size of the state vector of the batch-MAP estimator, $\mathbf{x}_{0:k}$, increases. Consequently, the computational cost of obtaining a state estimate continuously increases, and at some point it will inevitably become too high for real-time operation. In order to adapt to the available computational resources, marginalization [4], [6], [7] can be used to discard old, matured states. This results in a constant-cost SWF which maintains a constant-size window of states [4], [13]. In this section, we describe the effects of the marginalization used by the standard SWF on the system's observability properties. This analysis forms the basis for our proposed algorithm (see Section IV). For more details on the derivation of the marginalization equations, the interested reader is referred to [7].

We consider the scenario where marginalization of old states is carried out at time-step k_o , when all the measurements during the time interval $[0, k_o]$ are available. Subsequently the robot keeps moving and collects new measurements in the time interval $[k_o + 1, k]$, and estimation takes place again at time-step k . The old states that are marginalized out at time-step k_o are denoted by:

$$\mathbf{x}_M \triangleq \left[\mathbf{x}_{R_{0:k_m}}^T \quad \mathbf{p}_{L_{M_1}}^T \quad \cdots \quad \mathbf{p}_{L_{M_m}}^T \right]^T$$

Note that it is not necessary to sequentially marginalize out the old robot poses, $\mathbf{x}_{R_{0:k_m}}$; instead, we can selectively discard the most matured (i.e., accurately estimated) ones. The remaining states that stay active in the sliding window after marginalization are denoted by:

$$\mathbf{x}_R \triangleq \left[\mathbf{x}_{R_{k_m+1:k_o}}^T \quad \mathbf{p}_{L_{R_1}}^T \quad \cdots \quad \mathbf{p}_{L_{R_r}}^T \right]^T$$

Upon marginalization, all the states in \mathbf{x}_M , as well as all the measurements that involve these states (denoted by \mathcal{Z}_M) are discarded. In their place, we maintain a Gaussian pdf, that describes the information that the discarded measurements convey about the active states, \mathbf{x}_R . The information matrix of this Gaussian is given by:

$$\mathbf{A}_p(k_o) = \mathbf{A}_{RR}(k_o) - \mathbf{A}_{RM}(k_o)\mathbf{A}_{MM}^{-1}(k_o)\mathbf{A}_{MR}(k_o) \quad (21)$$

where the matrices appearing in the above equation are defined as partitions of the following matrix:

$$\begin{aligned} \mathbf{A}_m(k_o) &= \mathbf{\Pi}^T \mathbf{P}_{0|0}^{-1} \mathbf{\Pi} + \sum_{\kappa=0}^{k_m-1} \mathbf{\Phi}_\kappa^T(k_o) \mathbf{Q}_\kappa'^{-1} \mathbf{\Phi}_\kappa(k_o) \\ &+ \sum_{\mathbf{z}_{ij} \in \mathcal{Z}_M} \mathbf{H}_{ij}^T(k_o) \mathbf{R}_{ij}^{-1} \mathbf{H}_{ij}(k_o) \quad (22) \\ &= \begin{bmatrix} \mathbf{A}_{MM}(k_o) & \mathbf{A}_{MR}(k_o) \\ \mathbf{A}_{RM}(k_o) & \mathbf{A}_{RR}(k_o) \end{bmatrix} \quad (23) \end{aligned}$$

Close inspection reveals that $\mathbf{A}_m(k_o)$ is the matrix describing the information contained in all the discarded measurements (odometry, robot-to-landmark, and prior). Thus, $\mathbf{A}_p(k_o)$, which is the Schur complement of $\mathbf{A}_{MM}(k_o)$ in $\mathbf{A}_m(k_o)$, describes the information that the discarded measurements give us about \mathbf{x}_R . We also note that, in the above, the time index (k_o) has been added to denote the fact that all the Jacobians are computed using the estimate $\hat{\mathbf{x}}_{0:k_o|k_o}$ as the linearization point.

After marginalization the robot continues moving in its environment, and new states are added to the state vector during $[k_o + 1, k]$. These are denoted by:

$$\mathbf{x}_N \triangleq \begin{bmatrix} \mathbf{x}_{R_{k_o+1:k}}^T & \mathbf{p}_{L_{N_1}}^T & \cdots & \mathbf{p}_{L_{N_n}}^T \end{bmatrix}^T$$

Now, at time step k , the ‘‘active states’’ are \mathbf{x}_R and \mathbf{x}_N . In order to compute estimates for the active states, the SWF employs the ‘‘active’’ measurements, $\mathcal{Z}_A = \mathcal{Z}_{0:k} \setminus \mathcal{Z}_M$, along with the motion model and the information from the marginalized states (expressed by $\mathbf{A}_p(k_o)$ (21)) [7].

As described above, the key idea in the SWF is that the information of all the marginalized measurements is represented using a single Gaussian. While this entails an approximation, it also enables the SWF to maintain constant computational complexity, that depends only on the number of currently active states, and not on the past history of marginalized states.

A. Parameter Observability Properties

We now examine the parameter observability properties [8] of the standard SWF-based SLAM, which, for the time being, is considered as a parameter (instead of state) estimation problem. The study of parameter observability examines whether the information provided by the available measurements is sufficient for estimating the parameters without ambiguity. When parameter observability holds, the Fisher information matrix (i.e., the Hessian matrix) is invertible. Since the Fisher information matrix describes the information available in the measurements, by studying its nullspace we can gain insight about the directions in the parameter (state)

space along which the estimator acquires information. In what follows, we will compare the parameter observability properties of the standard SWF with those of the batch-MAP estimator, to draw conclusions about the estimator’s consistency.

We first notice that the nullspace of the Hessian matrix of the batch-MAP estimator (18) at time-step k , is given by:²

$$\text{null}(\mathbf{A}_b(k)) = \text{span}_{\text{col.}} \begin{bmatrix} \mathbf{I}_2 & \mathbf{J}\hat{\mathbf{p}}_{R_{0|k}} \\ \mathbf{0}_{1 \times 2} & 1 \\ \vdots & \vdots \\ \mathbf{I}_2 & \mathbf{J}\hat{\mathbf{p}}_{R_{k|k}} \\ \mathbf{0}_{1 \times 2} & 1 \\ \mathbf{I}_2 & \mathbf{J}\hat{\mathbf{p}}_{L_{1|k}} \\ \vdots & \vdots \\ \mathbf{I}_2 & \mathbf{J}\hat{\mathbf{p}}_{L_{M|k}} \end{bmatrix} \quad (24)$$

which is of dimension three. This agrees with the fact that in SLAM, three degrees of freedom corresponding to the global translation and rotation are unobservable [9]. However, as shown below, this is not the case for the standard SWF.

In the standard SWF, the matrix that describes the information for the *entire* history of states, $\mathbf{x}_{0:k} = [\mathbf{x}_M^T \quad \mathbf{x}_R^T \quad \mathbf{x}_N^T]^T$, is given by [7]:

$$\begin{aligned} \mathbf{A}(k) &= \underbrace{\sum_{\kappa=0}^{k_m-1} \mathbf{\Phi}_\kappa^T(k_o) \mathbf{Q}_\kappa'^{-1} \mathbf{\Phi}_\kappa(k_o)}_{\mathbf{A}_1(k_o)} + \sum_{\mathbf{z}_{ij} \in \mathcal{Z}_M} \mathbf{H}_{ij}^T(k_o) \mathbf{R}_{ij}^{-1} \mathbf{H}_{ij}(k_o) \\ &+ \underbrace{\sum_{\kappa=k_m}^{k-1} \mathbf{\Phi}_\kappa^T(k) \mathbf{Q}_\kappa'^{-1} \mathbf{\Phi}_\kappa(k)}_{\mathbf{A}_2(k)} + \sum_{\mathbf{z}_{ij} \in \mathcal{Z}_A} \mathbf{H}_{ij}^T(k) \mathbf{R}_{ij}^{-1} \mathbf{H}_{ij}(k) \quad (25) \end{aligned}$$

where the matrix $\mathbf{A}_1(k_o)$ contains all the information pertaining to the marginalized states, and $\mathbf{A}_2(k)$ the information pertaining to the active states at time-step k . Again, we note that the time indices (k_o) and (k) indicate the state estimates ($\hat{\mathbf{x}}_{0:k_o|k_o}$ and $\hat{\mathbf{x}}_{0:k|k}$, respectively) used as linearization points in computing each of the above terms. The Hessian $\mathbf{A}(k)$ in (25) has the following interesting structure:

$$\begin{aligned} \mathbf{A}(k) &= \underbrace{\begin{bmatrix} \mathbf{A}_{MM}(k_o) & \mathbf{A}_{MR}(k_o) & \mathbf{0} \\ \mathbf{A}_{RM}(k_o) & \mathbf{A}_{RR}(k_o) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}}_{\mathbf{A}_1(k_o)} + \underbrace{\begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_{RR}(k) & \mathbf{A}_{RN}(k) \\ \mathbf{0} & \mathbf{A}_{NR}(k) & \mathbf{A}_{NN}(k) \end{bmatrix}}_{\mathbf{A}_2(k)} \\ &= \begin{bmatrix} \mathbf{A}_{MM}(k_o) & \mathbf{A}_{MR}(k_o) & \mathbf{0} \\ \mathbf{A}_{RM}(k_o) & \mathbf{A}_{RR}(k_o) + \mathbf{A}_{RR}(k) & \mathbf{A}_{RN}(k) \\ \mathbf{0} & \mathbf{A}_{NR}(k) & \mathbf{A}_{NN}(k) \end{bmatrix} \quad (26) \end{aligned}$$

It is clear now that different estimates, $\hat{\mathbf{x}}_R(k_o)$ and $\hat{\mathbf{x}}_R(k)$, are used in computing the Hessian matrix. This occurs because some of the states in \mathbf{x}_R are involved in measurements both in \mathcal{Z}_M and \mathcal{Z}_A . As a result of the above structure, it can

²Since we are interested in the information contained in the available measurements, the case without prior (i.e., $\mathbf{P}_{0|0} \rightarrow \infty$) is considered here.

be shown that the last column of the matrix in (24) does *not* belong in the nullspace of $\mathbf{A}(k)$ [7]. Instead, the nullspace of $\mathbf{A}(k)$ is spanned by only the first two columns of (24), which in turn shows that the rank of the Hessian in the SWF is *higher* than the rank of the Hessian of the batch MAP. Clearly, this difference is not desirable, since both estimators process the same measurements, and thus have access to the same amount of information.

IV. OBSERVABILITY CONSTRAINED (OC)-SWF

As seen from the preceding section, due to marginalization, the standard SWF possesses different parameter observability properties from the batch-MAP estimator, since its Hessian has a nullspace of lower dimension than that of the batch-MAP estimator. This implies that the standard SWF acquires spurious information along one direction of the state space (the one corresponding to global orientation), which can lead to inconsistency. To address this issue, we adopt the idea of observability-based rules for choosing linearization points that was originally proposed in our previous work [9], and develop a new observability constrained (OC)-SWF.

The key idea of the proposed approach is that the linearization points used in computing the Hessian matrix are selected so as to ensure that the Hessian has a nullspace of the same dimension as that of the batch MAP estimator (see (24)). Different approaches for selecting linearization points are possible to satisfy this observability condition. For example, the prior-linearization (PL)-SWF proposed in [7] employs a simple linearization scheme to achieve this goal based on [14]. Specifically, when computing the Hessian, it uses the prior estimates, $\hat{\mathbf{x}}_{\mathbf{R}}(k_o)$, instead of the current estimates $\hat{\mathbf{x}}_{\mathbf{R}}(k)$, for the states in $\mathbf{x}_{\mathbf{R}}$ that are connected to marginalized states. By doing so, it is guaranteed that the same estimate is used as the linearization point for each of these states. However, even though the PL-SWF typically performs substantially better than the standard SWF (see Section V), the prior estimates $\hat{\mathbf{x}}_{\mathbf{R}}(k_o)$ used as linearization points could be inaccurate, and thus can result in large linearization errors, which can degrade the estimator's performance. Therefore, in the proposed OC-SWF, we select the linearization points for the states $\mathbf{x}_{\mathbf{R}}$ and $\mathbf{x}_{\mathbf{N}}$ (i.e., the states that are still "active" in the minimization), in a way that not only ensures the correct dimension for the nullspace of the Hessian matrix, but also minimizes their difference from the current best available estimates (see [9]). This can be formulated as the following constrained minimization problem:³

$$\min_{\mathbf{x}_{\mathbf{R}}^*, \mathbf{x}_{\mathbf{N}}^*} \|\mathbf{x}_{\mathbf{R}}^* - \hat{\mathbf{x}}_{\mathbf{R}}(k)\|^2 + \|\mathbf{x}_{\mathbf{N}}^* - \hat{\mathbf{x}}_{\mathbf{N}}(k)\|^2 \quad (27)$$

$$\text{subject to } \mathbf{A}(k)\mathbf{N}_k = \mathbf{0} \quad (28)$$

In this formulation, \mathbf{N}_k is a design choice that defines the desired nullspace with correct dimension. Ideally, we would like to have the same nullspace as (24). However, this is not possible, as in the SWF some of the old states have

³For the clarity of presentation, hereafter the superscript (ℓ) is dropped, since, without loss of generality, we consider the $(\ell + 1)$ -th iteration in Gauss-Newton given that the results from the ℓ -th iteration are available.

been marginalized, and thus we do not maintain estimates for them. We next describe our choice of estimates used for constructing \mathbf{N}_k , and denote these estimates by the symbol " - ". Specifically, during the $(\ell + 1)$ -th Gauss-Newton iteration, we use the following estimates to construct the matrix \mathbf{N}_k : (i) For the new states, $\mathbf{x}_{\mathbf{N}}$, as well as those states in $\mathbf{x}_{\mathbf{R}}$ for which no prior exists, we use the estimates from the ℓ -th iteration, i.e., $\bar{\mathbf{x}}_i = \hat{\mathbf{x}}_i(k)$; (ii) For all marginalized states, $\mathbf{x}_{\mathbf{M}}$, as well as for states in $\mathbf{x}_{\mathbf{R}}$ for which a prior exists, we use the prior estimate, i.e., $\bar{\mathbf{x}}_i = \hat{\mathbf{x}}_i(k_o)$. By replacing the pertinent state estimates in (24) by the estimates selected above, $\bar{\mathbf{x}}_{0:k} = [\bar{\mathbf{x}}_{\mathbf{M}}^T \ \bar{\mathbf{x}}_{\mathbf{R}}^T \ \bar{\mathbf{x}}_{\mathbf{N}}^T]^T$, we obtain the desired nullspace, $\mathbf{N}_k = \mathbf{N}_k(\bar{\mathbf{x}}_{0:k})$.

By construction, the nullspace $\mathbf{N}_k(\bar{\mathbf{x}}_{0:k})$ always satisfies the equality $\mathbf{A}_1(k_o)\mathbf{N}_k = \mathbf{0}$. Thus, the condition $\mathbf{A}(k)\mathbf{N}_k(\bar{\mathbf{x}}_{0:k}) = \mathbf{0}$ can be written as (see (25)):

$$\begin{aligned} \mathbf{A}_2(k)\mathbf{N}_k &= \mathbf{0} \\ \Rightarrow \left(\sum_{\kappa=k_m}^{k-1} \Phi_{\kappa}^T \mathbf{Q}_{\kappa}^{-1} \Phi_{\kappa} + \sum_{\mathbf{z}_{ij} \in \mathcal{Z}_{\mathbf{A}}} \mathbf{H}_{ij}^T \mathbf{R}_{ij}^{-1} \mathbf{H}_{ij} \right) \mathbf{N}_k &= \mathbf{0} \\ \Rightarrow \begin{cases} \Phi_{\kappa} \mathbf{N}_k = \mathbf{0}, & \forall \kappa = k_m, \dots, k-1 \\ \mathbf{H}_{ij} \mathbf{N}_k = \mathbf{0}, & \forall \mathbf{z}_{ij} \in \mathcal{Z}_{\mathbf{A}} \end{cases} \end{aligned} \quad (29)$$

Using the structure of the Jacobians Φ_{κ} and \mathbf{H}_{ij} (see (5) and (9)) and that of the matrix \mathbf{N}_k (24), the above constraints (29) can be written as follows [12]:

$$\Phi_{\kappa} \mathbf{N}_k = \mathbf{0} \Rightarrow \mathbf{p}_{R_{\kappa}}^* - \bar{\mathbf{p}}_{R_{\kappa}} + \bar{\mathbf{p}}_{R_{\kappa+1}} - \mathbf{p}_{R_{\kappa+1}}^* = \mathbf{0} \quad (30)$$

$$\mathbf{H}_{ij} \mathbf{N}_k = \mathbf{0} \Rightarrow \mathbf{p}_{R_j}^* - \bar{\mathbf{p}}_{R_j} + \bar{\mathbf{p}}_{L_i} - \mathbf{p}_{L_i}^* = \mathbf{0} \quad (31)$$

Therefore, the problem (27)-(28) can be simplified as:

$$\min_{\mathbf{x}_{\mathbf{R}}^*, \mathbf{x}_{\mathbf{N}}^*} \|\mathbf{x}_{\mathbf{R}}^* - \hat{\mathbf{x}}_{\mathbf{R}}(k)\|^2 + \|\mathbf{x}_{\mathbf{N}}^* - \hat{\mathbf{x}}_{\mathbf{N}}(k)\|^2 \quad (32)$$

$$\text{s.t. } \begin{cases} \mathbf{p}_{R_{\kappa}}^* - \bar{\mathbf{p}}_{R_{\kappa}} + \bar{\mathbf{p}}_{R_{\kappa+1}} - \mathbf{p}_{R_{\kappa+1}}^* = \mathbf{0}, & \forall \kappa = k_m, \dots, k-1 \\ \mathbf{p}_{R_j}^* - \bar{\mathbf{p}}_{R_j} + \bar{\mathbf{p}}_{L_i} - \mathbf{p}_{L_i}^* = \mathbf{0}, & \forall \mathbf{z}_{ij} \in \mathcal{Z}_{\mathbf{A}} \end{cases} \quad (33)$$

We now derive an analytical solution to the constrained minimization problem (32)-(33). In particular, the approach of Lagrangian multipliers [15] is employed. The Lagrangian function is constructed as follows:

$$\begin{aligned} \mathcal{L} &= \|\mathbf{x}_{\mathbf{R}}^* - \hat{\mathbf{x}}_{\mathbf{R}}(k)\|^2 + \|\mathbf{x}_{\mathbf{N}}^* - \hat{\mathbf{x}}_{\mathbf{N}}(k)\|^2 \quad (34) \\ &+ \sum_{\kappa=k_m}^{k-1} \boldsymbol{\mu}_{\kappa}^T \left(\mathbf{p}_{R_{\kappa}}^* - \bar{\mathbf{p}}_{R_{\kappa}} + \bar{\mathbf{p}}_{R_{\kappa+1}} - \mathbf{p}_{R_{\kappa+1}}^* \right) \\ &+ \sum_{(i,j), \mathbf{z}_{ij} \in \mathcal{Z}_{\mathbf{A}}} \boldsymbol{\lambda}_{ij}^T \left(\mathbf{p}_{R_j}^* - \bar{\mathbf{p}}_{R_j} + \bar{\mathbf{p}}_{L_i} - \mathbf{p}_{L_i}^* \right) \end{aligned}$$

By setting the derivatives with respect to the state and

Lagrangian-multiplier variables equal to zero, we have:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{p}_{R_\kappa}^*} = \begin{cases} 2(\mathbf{p}_{R_\kappa}^* - \hat{\mathbf{p}}_{R_\kappa|k}) + \boldsymbol{\mu}_\kappa + \sum_{i, \mathbf{z}_{i\kappa} \in \mathcal{Z}_A} \boldsymbol{\lambda}_{i\kappa} = \mathbf{0}, & \text{if } \kappa = k_m \\ 2(\mathbf{p}_{R_\kappa}^* - \hat{\mathbf{p}}_{R_\kappa|k}) - \boldsymbol{\mu}_{\kappa-1} + \sum_{i, \mathbf{z}_{i\kappa} \in \mathcal{Z}_A} \boldsymbol{\lambda}_{i\kappa} = \mathbf{0}, & \text{if } \kappa = k \\ 2(\mathbf{p}_{R_\kappa}^* - \hat{\mathbf{p}}_{R_\kappa|k}) + \boldsymbol{\mu}_\kappa - \boldsymbol{\mu}_{\kappa-1} + \sum_{i, \mathbf{z}_{i\kappa} \in \mathcal{Z}_A} \boldsymbol{\lambda}_{i\kappa} = \mathbf{0}, & \text{else} \end{cases} \quad (35)$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{p}_{L_i}^*} = 2(\mathbf{p}_{L_i}^* - \hat{\mathbf{p}}_{L_i|k}) - \sum_{j, \mathbf{z}_{ij} \in \mathcal{Z}_A} \boldsymbol{\lambda}_{ij} = \mathbf{0} \quad (36)$$

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\mu}_\kappa} = \mathbf{p}_{R_\kappa}^* - \bar{\mathbf{p}}_{R_\kappa} + \bar{\mathbf{p}}_{R_{\kappa+1}} - \mathbf{p}_{R_{\kappa+1}}^* = \mathbf{0} \quad (37)$$

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\lambda}_{ij}} = \mathbf{p}_{R_j}^* - \bar{\mathbf{p}}_{R_j} + \bar{\mathbf{p}}_{L_i} - \mathbf{p}_{L_i}^* = \mathbf{0} \quad (38)$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{x}_{\text{other}}^*} = 2(\mathbf{x}_{\text{other}}^* - \hat{\mathbf{x}}_{\text{other}}(k)) = \mathbf{0} \quad (39)$$

where $\mathbf{x}_{\text{other}}$ denotes all the state variables except the ones involved in (35)-(38). Solving (35), (36), and (39) yields the following optimal solutions:

$$\mathbf{p}_{R_\kappa}^* = \hat{\mathbf{p}}_{R_\kappa|k} - \frac{1}{2} \left[\delta \boldsymbol{\mu}_\kappa + \sum_{i, \mathbf{z}_{i\kappa} \in \mathcal{Z}_A} \boldsymbol{\lambda}_{i\kappa} \right] \quad (40)$$

$$\mathbf{p}_{L_i}^* = \hat{\mathbf{p}}_{L_i|k} + \frac{1}{2} \left[\sum_{j, \mathbf{z}_{ij} \in \mathcal{Z}_A} \boldsymbol{\lambda}_{ij} \right] \quad (41)$$

$$\mathbf{x}_{\text{other}}^* = \hat{\mathbf{x}}_{\text{other}}(k) \quad (42)$$

where

$$\delta \boldsymbol{\mu}_\kappa = \begin{cases} \boldsymbol{\mu}_\kappa, & \text{if } \kappa = k_m \\ -\boldsymbol{\mu}_{\kappa-1}, & \text{if } \kappa = k \\ \boldsymbol{\mu}_\kappa - \boldsymbol{\mu}_{\kappa-1}, & \text{else} \end{cases}$$

Substituting (40)-(42) into (37) and (38) yields the following linear equations in terms of the Lagrangian multipliers:

$$\begin{aligned} & \Delta \boldsymbol{\mu}_\kappa + \sum_{i, \mathbf{z}_{i\kappa} \in \mathcal{Z}_A} \boldsymbol{\lambda}_{i\kappa} - \sum_{i, \mathbf{z}_{i(\kappa+1)} \in \mathcal{Z}_A} \boldsymbol{\lambda}_{i(\kappa+1)} \\ &= 2(\hat{\mathbf{p}}_{R_\kappa|k} - \bar{\mathbf{p}}_{R_\kappa} + \bar{\mathbf{p}}_{R_{\kappa+1}} - \hat{\mathbf{p}}_{R_{\kappa+1}|k}) \end{aligned} \quad (43)$$

$$\begin{aligned} & \delta \boldsymbol{\mu}_\kappa + \sum_{i, \mathbf{z}_{i\kappa} \in \mathcal{Z}_A} \boldsymbol{\lambda}_{i\kappa} + \sum_{j, \mathbf{z}_{ij} \in \mathcal{Z}_A} \boldsymbol{\lambda}_{ij} \\ &= 2(\hat{\mathbf{p}}_{R_\kappa|k} - \bar{\mathbf{p}}_{R_\kappa} + \bar{\mathbf{p}}_{L_i} - \hat{\mathbf{p}}_{L_i|k}) \end{aligned} \quad (44)$$

where

$$\Delta \boldsymbol{\mu}_\kappa = \begin{cases} 2\boldsymbol{\mu}_\kappa - \boldsymbol{\mu}_{\kappa+1}, & \text{if } \kappa = k_m \\ 2\boldsymbol{\mu}_\kappa - \boldsymbol{\mu}_{\kappa-1}, & \text{if } \kappa = k - 1 \\ -\boldsymbol{\mu}_{\kappa-1}, & \text{if } \kappa = k \\ 2\boldsymbol{\mu}_\kappa - \boldsymbol{\mu}_{\kappa-1} - \boldsymbol{\mu}_{\kappa+1}, & \text{else} \end{cases}$$

In order to determine the Lagrangian multipliers, $\boldsymbol{\mu}_\kappa$ and $\boldsymbol{\lambda}_{ij}$, we stack equations (43)-(44) for all the measurements (constraints) into matrix-vector form and solve the resulting linear system. Once the Lagrangian multipliers are specified, the

optimal linearization points can be obtained based on (40)-(42). Subsequently, the Jacobian and Hessian matrices are computed using the optimal linearization points, and then the standard Gauss-Newton steps are carried out (see Section II-C). It should be pointed out that, as compared to the standard SWF and the PL-SWF, the OC-SWF only requires an additional computational overhead of linearly solving for the Lagrangian multipliers, which in general is cubic in the number of active proprioceptive and exteroceptive measurements.

V. SIMULATION RESULTS

A series of Monte Carlo simulations were conducted under different conditions, in order to validate the capability of the proposed OC-SWF to improve estimation performance. The metrics used to evaluate the estimator's performance were: (i) the average root mean squared error (RMSE), and (ii) the normalized (state) estimation error squared (NEES) [8]. Both metrics are computed by averaging over all Monte Carlo runs. The average RMSE provides us with a concise metric of the accuracy of a given estimator, while the NEES is a metric for evaluating the estimator's consistency. By studying both the RMSE and NEES, we obtain a comprehensive picture of the estimator's performance.

In simulation tests presented in this section, we conducted 50 Monte Carlo simulations, and compared four different estimators: (i) the batch-MAP estimator, (ii) the standard SWF, (iii) the PL-SWF [7], and (iv) the proposed OC-SWF. In the simulation setup, a robot with a simple 3-wheel (2 active and 1 caster) kinematic model moves on a planar surface, at a constant velocity of $v = 0.5$ m/sec. The two active wheels are equipped with encoders, which measure their revolutions and provide measurements of velocity (i.e., right and left wheel velocities, v_r and v_l , respectively), with standard deviation equal to $\sigma = 1\%$ for each wheel. These measurements are used to obtain linear and rotational velocity measurements for the robot, which are given by:

$$v = \frac{v_r + v_l}{2}, \quad \omega = \frac{v_r - v_l}{a}$$

where $a = 0.5$ m is the distance between the active wheels. The standard deviation of the linear and rotational velocity measurement noise is thus equal to $\sigma_v = \frac{\sigma}{\sqrt{2}}$ and $\sigma_\omega = \frac{\sqrt{2}\sigma}{a}$, respectively. We considered a SLAM scenario where a robot moves along a circular trajectory of total length of about 500 m, and measures bearing angles to landmarks that lie within its sensing range of 10 m. There are 50 landmarks in total which are randomly generated along the robot trajectory. This can arise, for example, in the case in which a robot moves inside corridors and tracks its position and corners (landmarks) using a monocular camera. At each time step, approximately 10 landmarks are visible. In the SWFs we chose to maintain a sliding window comprising 20 robot poses and at most 10 active landmarks. To ensure a fair comparison among the SWF algorithms, all three of them process the same data and maintain the same states in their windows. In this simulation, the landmarks to be marginalized are chosen such that at least two "old"

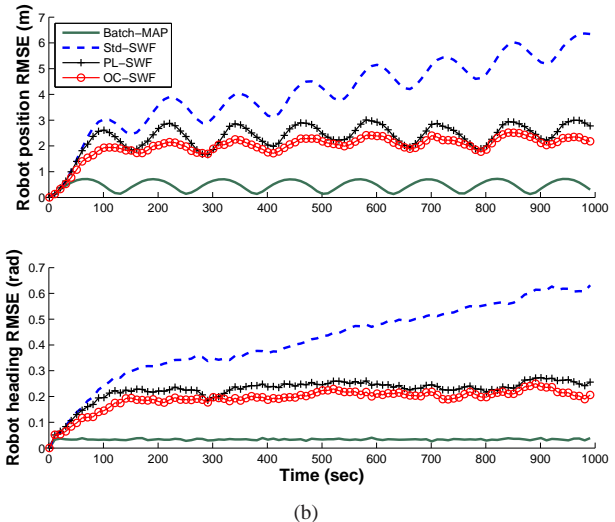
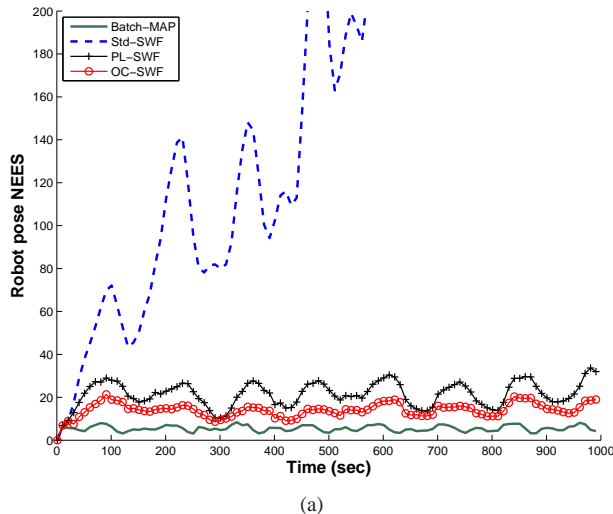


Fig. 1. Monte-Carlo simulation results: (a) NEEs of the latest robot pose, (b) average RMSE of the latest robot pose (position and orientation). It is clear that both the PL-SWF and OC-SWF perform significantly better than the standard SWF, in terms of both consistency (NEES) and accuracy (RMSE). Note also that the OC-SWF attains better performance than the PL-SWF.

Batch MAP	Std-SWF	PL-SWF	OC-SWF
RMSE for Landmark Position (m)			
0.5184	2.7449	2.6713	2.6235
NEES for Landmark Position			
3.7769	42.1306	12.3615	9.5719

TABLE I
LANDMARK POSITION ESTIMATION PERFORMANCE

landmarks always remain in the window, to ensure that the uncertainty does not continuously increase. The batch MAP estimator processes all measurements, and is used as the benchmark.

For the results presented here, we considered a case with relatively large measurement noise, compared to what is typically encountered in practice, since larger noise levels can lead to larger estimation errors, and thus less accurate linearization, which will make the effects of inconsistency more apparent. Specifically, the standard deviation of the bearing measurement noise was set to 10 deg. Fig. 1 shows the results for the robot pose based on the compared estimators, while Table I depicts the average NEEs and RMSE for the landmark positions (averaged over all the landmarks). First notice that as expected, the batch-MAP estimator attains the best performance, since it utilizes all the available information, while the SWFs discard the measurements belonging to the inactive measurement set, \mathcal{Z}_M , due to marginalization. More importantly, the two observability-constrained smoothers (i.e., PL-SWF and OC-SWF) perform substantially better than the standard SWF, in terms of both consistency (NEES) and accuracy (RMSE). This is attributed to the fact that the appropriate parameter observability properties are preserved in the proposed observability-based smoothing framework. We also note that the OC-SWF achieves better performance than the PL-SWF.

This is due to the fact that when the noise is large, the prior estimates used as linearization points in the PL-SWF are inaccurate (i.e., the linearization errors become significant), which degrades the estimator’s performance. In contrast, the OC-SWF employs, by construction, the optimal linearization points and thus yields better estimation accuracy.

VI. EXPERIMENTAL RESULTS

To experimentally validate the performance of the OC-SWF, the estimator was tested on the Victoria Park data set courtesy of Nebot and Guivant⁴. The experimental platform was a 4-wheeled vehicle equipped with a kinematic GPS, a laser sensor, and wheel encoders. The GPS system was used to provide ground truth for the robot position. Wheel encoders were used to provide odometric measurements, and propagation was carried out using the Ackerman model. In this particular application, since the most common feature in the environment were trees, the profiles of trees were extracted from the laser data, the centers of the trunks were then used as the point landmarks, and distance and bearing measurements to them were used for estimation [16].

In this test, we compared the same four estimators as in the preceding simulation: (i) the batch-MAP estimator, (ii) the standard SWF, (iii) the PL-SWF [7], and (iv) the proposed OC-SWF. Since in this experiment, both true landmark positions and true robot orientations were unavailable, we only compared the robot position estimation performance, which is shown in Fig. 2. Specifically, Fig. 2(a) depicts the trajectory estimates produced by the four estimators as compared to the GPS ground truth, while Fig. 2(b) shows the estimation errors of the robot position over time. Note that since the GPS had different frequency (up to 5 Hz) from the other exteroceptive sensors and its satellite signal

⁴The data set is available at: http://www-personal.acfr.usyd.edu.au/nebot/victoria_park.htm. Note that in order to ensure the comparison to the batch MAP estimator, we here considered the first half of the data set.

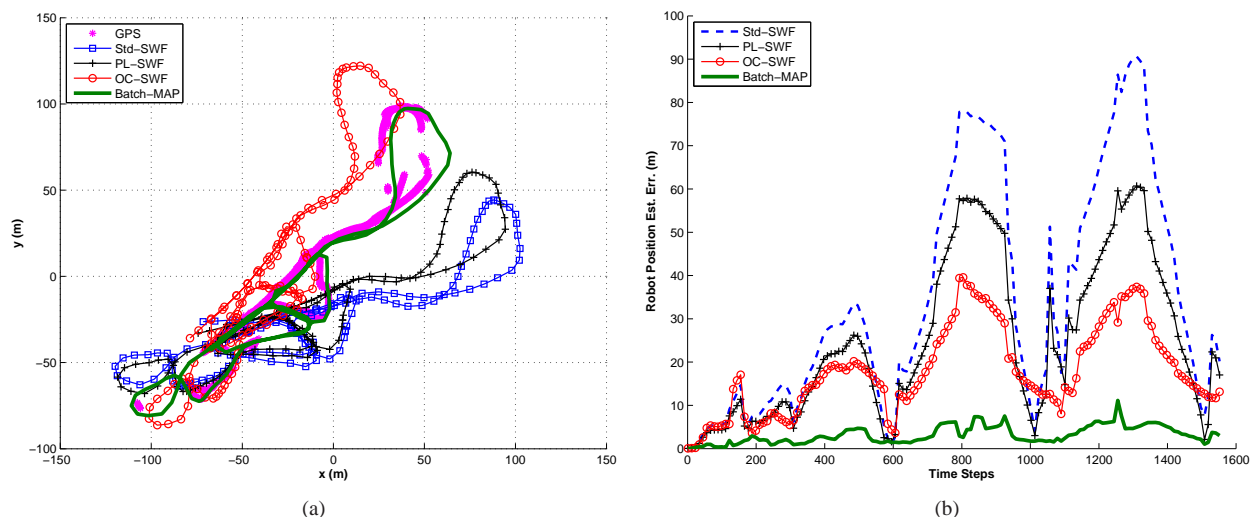


Fig. 2. Experimental results: (a) The robot trajectory estimates as compared to the GPS data, (b) estimation errors of robot position. It is clear that the OC-SWF performs more accurately than the standard SWF and the PL-SWF.

was not always available, we interpolated the estimates of the four compared estimators, and computed the estimation errors only at the times when the GPS was available. As evident from Fig. 2, the OC-SWF performs significantly better than the standard SWF and the PL-SWF [7]. These results, along with those of the simulations presented in the previous section, show that it is essential for an estimator to ensure appropriate observability properties in order to improve its performance.

VII. CONCLUSIONS AND FUTURE WORK

In this paper, we addressed the consistency issue of the standard SWF. Even though the SWF is an appealing smoothing algorithm well-suited for real-time applications where the effects of nonlinearity of the measurements are significant, it can suffer from inconsistency. In particular, due to marginalization, the standard SWF uses different estimates for the same states as linearization points when computing the Hessian matrix, which results in its Hessian having a nullspace of lower dimension than the batch-MAP estimator. This implies that the standard SWF acquires spurious information and thus may become inconsistent. To address this issue, we have introduced an observability-based smoothing framework, which extends the method presented in [9] for EKF to the case of the SWF. Specifically, we select the linearization points at which the Hessian is evaluated, so as to ensure that the nullspace of the Hessian is of the same dimension as that of the batch-MAP estimator, while minimizing the linearization errors. Both simulation and experimental results have shown that the proposed OC-SWF performs substantially better than the standard SWF as well as the PL-SWF [7], in terms of both accuracy and consistency. In our future work, we plan to extend this approach to the case of robots navigating in 3D.

REFERENCES

- [1] S. Kay, *Fundamentals of Statistical Signal Processing, Vol. I - Estimation Theory*. Prentice Hall, 1993.
- [2] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, "Bundle adjustment – a modern synthesis," *Lecture Notes in Computer Science*, vol. 1883, pp. 298–375, Jan. 2000.
- [3] F. Dellaert and M. Kaess, "Square root SAM: Simultaneous localization and mapping via square root information smoothing," *Intl. Journal of Robotics Research*, vol. 25, no. 12, pp. 1181–1203, Dec. 2006.
- [4] G. Sibley, L. Matthies, and G. Sukhatme, "Sliding window filter with application to planetary landing," *Journal of Field Robotics*, vol. 27, no. 5, pp. 587–608, Sep./Oct. 2010.
- [5] P. S. Maybeck, *Stochastic Models, Estimation and Control*, ser. Mathematics in Science and Engineering. London: Academic Press, 1982, vol. 141-2.
- [6] A. Ranganathan, M. Kaess, and F. Dellaert, "Fast 3D pose estimation with out-of-sequence measurements," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, San Diego, CA, Oct. 29–Nov. 2, 2007, pp. 2486–2493.
- [7] T. Dong-Si and A. I. Mourikis, "Motion tracking with fixed-lag smoothing: Algorithm and consistency analysis," in *Proc. IEEE International Conference on Robotics and Automation*, Shanghai, China, May 9–13, 2011, pp. 5655–5662.
- [8] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with applications to tracking and navigation*. New York: Wiley, 2001.
- [9] G. P. Huang, A. I. Mourikis, and S. I. Roumeliotis, "Observability-based rules for designing consistent EKF SLAM estimators," *Intl. Journal of Robotics Research*, vol. 29, no. 5, pp. 502–528, Apr. 2010.
- [10] D. Nister, O. Naroditsky, and J. Bergen, "Visual odometry for ground vehicle applications," *Journal of Field Robotics*, vol. 23, no. 1, pp. 3–20, Jan. 2006.
- [11] A. Mourikis, N. Trawny, S. Roumeliotis, A. Johnson, A. Ansar, and L. Matthies, "Vision-aided inertial navigation for spacecraft entry, descent, and landing," *IEEE Transactions on Robotics*, vol. 25, no. 2, pp. 264–280, Apr. 2009.
- [12] G. P. Huang and S. I. Roumeliotis, "An observability constrained sliding window filter for SLAM," MARS Lab, University of Minnesota, Minneapolis, MN, Tech. Rep., Feb. 2011. [Online]. Available: www.cs.umn.edu/~ghuang/paper/TR_OCMMAP.pdf
- [13] G. Sibley, G. S. Sukhatme, and L. Matthies, "Constant time sliding window filter SLAM as a basis for metric visual perception," in *Proc. IEEE International Conference on Robotics and Automation Workshop*, Roma, Italy, Apr. 10–14, 2007.
- [14] G. P. Huang, A. I. Mourikis, and S. I. Roumeliotis, "Analysis and improvement of the consistency of extended Kalman filter-based SLAM," in *Proc. IEEE International Conference on Robotics and Automation*, Pasadena, CA, May 19–23, 2008, pp. 473–479.
- [15] D. P. Bertsekas, *Nonlinear Programming*. Athena Scientific, 1999.
- [16] J. E. Guivant and E. M. Nebot, "Optimization of the simultaneous localization and map building algorithm for real time implementation," *IEEE Transactions on Robotics and Automation*, vol. 17, no. 3, pp. 242–257, June 2001.