Correspondence

A Fast Labeled Multi-Bernoulli Filter Using Belief Propagation

We propose a fast labeled multi-Bernoulli (LMB) filter that uses belief propagation for probabilistic data association. The complexity of our filter scales only linearly in the numbers of Bernoulli components and measurements, while the performance is comparable to or better than that of the Gibbs sampler-based LMB filter.

I. INTRODUCTION

The aim of multiobject tracking is to estimate the time-dependent number and states of multiple objects from noisy and cluttered sensor measurements [1]–[4]. Of particular interest are tracking algorithms that maintain track continuity, i.e., estimate entire trajectories of consecutive object states [1]–[3], [5]–[9]. Two noteworthy examples are the labeled multi-Bernoulli (LMB) filter [9]–[12], which formally includes track continuity in its basic formulation involving labels, and a "label-augmented" version of the track-oriented marginal multi-Bernoulli/Poisson (TOMB/P) filter [13], which was defined in [14] by heuristically introducing labels in the formulation of the (originally nonlabeled) TOMB/P filter. The LMB filter is an approximation of the more complex generalized labeled multi-Bernoulli (GLMB) filter [3], [5]–[8]. In particular, the complexity of a fast implementation of the LMB filter using the Gibbs sampler [10] scales quadratically in the number of Bernoulli components and linearly both in the number of measurements and in the number of samples.

Here, we propose a fast LMB filter with linear scaling. Our filter incorporates a variant of the belief propagation (BP) scheme for probabilistic data association proposed in [13], [15]. The use of this BP scheme within the LMB filter is enabled by a new derivation of the original LMB filter. In this derivation, the GLMB random finite set (RFS) is reformulated in terms of a joint association distribution, which

Manuscript received January 18, 2018; revised November 20, 2018 and May 8, 2019; released for publication August 14, 2019. Date of publication September 13, 2019; date of current version June 9, 2020.

DOI. No. 10.1109/TAES.2019.2941104

Refereeing of this contribution was handled by B.-N. Vo.

This work was supported in part by the Austrian Science Fund (FWF) under Grant P 32055-N31, Grant P 27370-N30, and Grant J 3886-N31 and in part by the Czech Science Foundation (GAČR) under Grant 17-19638S.

Authors' addresses: T. Kropfreiter and F. Hlawatsch are with the Institute of Telecommunications, TU Wien, Vienna, Austria and with Brno University of Technology, Brno, Czech Republic, E-mail: thomas.kropfreiter@tuwien.ac.at; franz.hlawatsch@tuwien.ac.at; F. Meyer is with the Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA, USA, E-mail: fmeyer@mit.edu. *(Corresponding author: Franz Hlawatsch.)*

is then approximated by the product of its marginals. The proposed fast LMB filter—which is different from the original LMB filter—is finally obtained by using a fast BP-based calculation of the marginal distributions. The complexity of our filter scales only linearly in the numbers of Bernoulli components, measurements, and BP iterations. Contrary to traditional LMB filter implementations based on Murty's algorithm or Markov chain Monte Carlo techniques, our BP-based LMB filter avoids the pruning of association information (GLMB components) in the update step. Thus, relevant association information that is discarded by traditional LMB filter implementations is preserved.

We will use the following notation. We denote vectors by small boldface letters (e.g., $\boldsymbol{x}$) and finite sets by capital letters (e.g., X, or, for labeled finite sets, $\tilde{X}$). For random quantities, we use a sans serif font, such as in $\boldsymbol{\mathsf{x}}$ or X. We write probability density functions (pdfs) as $f(\cdot)$ or $s(\cdot)$ and probability mass functions (pmfs) as $p(\cdot)$.

This paper is structured as follows. Section II provides an introduction to RFSs. The system model is described in Section III, and the original LMB filter is reviewed in Section IV. In Section V, we present our new derivation of the original LMB filter. Section VI reviews the BP scheme and describes the proposed fast LMB filter. Simulation results are reported in Section VII. A detailed derivation of the BP-based probabilistic data association algorithm used in our fast LMB filter is provided in an appendix. We note that some supplementary material—a pseudocode of our fast LMB filter algorithm and a comparison of the approximate marginal association probabilities arising in our fast LMB filter and in the original LMB filter—are available online [16].

II. RFS FUNDAMENTALS

An RFS $\mathsf{X} = \{\boldsymbol{\mathsf{x}}^{(1)}, \ldots, \boldsymbol{\mathsf{x}}^{(\mathsf{n})}\}$ is a random variable whose realizations X are finite sets $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)}\}$ of vectors $\boldsymbol{x}^{(i)} \in \mathbb{R}^{N_x}$. An RFS X can be described by its multiobject pdf $f(X)$ [3]. The *Bernoulli RFS* with existence probability r and "spatial pdf" $s(\boldsymbol{x})$ is empty with probability $1 - r$ and contains one element $\boldsymbol{\mathsf{x}} \sim s(\boldsymbol{x})$ with probability r. The *multi-Bernoulli RFS* is the union of statistically independent Bernoulli RFSs. The *Poisson RFS* has a cardinality distribution $\rho(n) \triangleq \Pr\{|\mathsf{X}| = n\}$ that is a Poisson pmf with mean μ, and its elements $\boldsymbol{\mathsf{x}} \in \mathsf{X}$ are independent and identically distributed (iid) according to a spatial pdf $f(\boldsymbol{x})$. The product $\lambda(\boldsymbol{x}) = \mu f(\boldsymbol{x})$ is referred to as probability hypothesis density (PHD).

In a labeled RFS $\tilde{\mathsf{X}}$, each element $\tilde{\boldsymbol{\mathsf{x}}}$ is a tuple $(\boldsymbol{\mathsf{x}}, \mathsf{l}) \in \mathbb{R}^{N_x} \times \mathbb{L}$, where $\mathbb{L}$ is a countable set [5], [6]. The labels $l^{(j)}$ of any realization $\tilde{X} = \{(\boldsymbol{x}^{(1)}, l^{(1)}), \ldots, (\boldsymbol{x}^{(n)}, l^{(n)})\}$ are distinct. We denote by $\mathcal{L}(\tilde{X}) \triangleq \{l^{(1)}, \ldots, l^{(n)}\}$ the set of all labels of $\tilde{X}$. The *LMB RFS* is a multi-Bernoulli RFS where for any realization $\tilde{X}$, each single-vector set $\{\boldsymbol{x}\}$ corresponding to a Bernoulli component $X^{(j)}$ is augmented by a distinct label $l \in \mathbb{L}^* \subseteq \mathbb{L}$. Using the labeling scheme of [9], the same label l is assigned to each state realization $\boldsymbol{x}$ of a given Bernoulli RFS $\mathsf{X}^{(j)}$. We will thus index the Bernoulli RFSs directly by their labels l, i.e., they are denoted $\mathsf{X}^{(l)}$, $l \in \mathbb{L}^*$, where $\mathbb{L}^* \subseteq \mathbb{L}$ denotes the finite set of assigned labels [9]. The LMB RFS $\tilde{\mathsf{X}}$ is then specified by the set of existence probabilities and spatial distributions, $\{(r^{(l)}, s^{(l)}(\boldsymbol{x}))\}_{l \in \mathbb{L}^*}$. Its multiobject pdf [5], [6] evaluated for a realization $\tilde{X}$ with label set $\mathcal{L}(\tilde{X}) \subseteq \mathbb{L} = \{l^{(1)}, \ldots, l^{(J)}\}$ and cardinality $|\tilde{X}| \leq J$ is [9]

$$f(\tilde{X}) = \Delta(\tilde{X}) \left(\prod_{l' \in \mathbb{L}^* \setminus \mathcal{L}(\tilde{X})} \left(1 - r^{(l')}\right) \right) \times \prod_{(\boldsymbol{x}, l) \in \tilde{X}} 1_{\mathbb{L}^*}(l)\, r^{(l)} s^{(l)}(\boldsymbol{x}). \quad (1)$$

Here, $\Delta(\tilde{X})$ is one if the labels of $\tilde{X}$ are distinct and zero otherwise, and $1_{\mathbb{L}^*}(l)$ is one if $l \in \mathbb{L}^*$ and zero otherwise. A generalization of the LMB RFS is the *GLMB RFS* [5], which will be considered in Section IV. A detailed description of the GLMB RFS and its relation to the LMB RFS can be found in [3], [5], [9].

III. SYSTEM MODEL

Following [9], we model the object states at discrete time k by an LMB RFS $\tilde{\mathsf{X}}_k$ with parameters $\{(r_k^{(l)}, s^{(l)}(\boldsymbol{x}_k))\}_{l \in \mathbb{L}_k^*}$, where $\mathbb{L}_k^* \subseteq \mathbb{L}_k = \{1, \ldots, k\} \times \mathbb{N}$. Following [5], the total label space $\mathbb{L}_k$ evolves according to $\mathbb{L}_k = \mathbb{L}_{k-1} \cup \mathbb{L}_k^{\mathrm{B}}$, where $\mathbb{L}_k^{\mathrm{B}} = \{k\} \times \mathbb{N}$ and $\mathbb{L}_{k-1} \cap \mathbb{L}_k^{\mathrm{B}} = \emptyset$. An object with preceding state $(\boldsymbol{x}_{k-1}, l)$ survives with probability $p_{\mathrm{S}}(\boldsymbol{x}_{k-1}, l)$, in which case its new state $\boldsymbol{\mathsf{x}}_k$ (without the label) is distributed according to some transition pdf $f(\boldsymbol{x}_k | \boldsymbol{x}_{k-1}, l)$ and the label l is preserved. The states of different objects evolve independently. Accordingly, the multiobject state of the survived objects at time k is an LMB RFS $\tilde{\mathsf{X}}_k^{\mathrm{S}}$ with parameters $\{(p_{\mathrm{S}}(\boldsymbol{x}_{k-1}, l), f(\boldsymbol{x}_k | \boldsymbol{x}_{k-1}, l))\}_{l \in \mathbb{L}_{k-1}^*}$. Newborn objects are modeled by an LMB RFS $\tilde{\mathsf{X}}_k^{\mathrm{B}}$ that is independent of $\tilde{\mathsf{X}}_k^{\mathrm{S}}$, given $\tilde{X}_{k-1}$, and parametrized by $\{(r_{\mathrm{B},k}^{(l)}, s_{\mathrm{B}}^{(l)}(\boldsymbol{x}_k))\}_{l \in \mathbb{L}_k^{\mathrm{B}*}}$, where $\mathbb{L}_k^{\mathrm{B}*} \subseteq \mathbb{L}_k^{\mathrm{B}}$ and $\mathbb{L}_k^{\mathrm{B}*} \cap \mathbb{L}_{k-1}^* = \emptyset$. The overall multiobject state of all objects, $\tilde{\mathsf{X}}_k = \tilde{\mathsf{X}}_k^{\mathrm{S}} \cup \tilde{\mathsf{X}}_k^{\mathrm{B}}$, is again an LMB RFS. The corresponding new label set is $\mathbb{L}_k^* = \mathbb{L}_{k-1}^* \cup \mathbb{L}_k^{\mathrm{B}*}$. The multiobject state transition model just described specifies the multiobject state transition pdf $f(\tilde{X}_k | \tilde{X}_{k-1})$ [5, Section IV-D].

A sensor generates M_k measurements $\boldsymbol{\mathsf{z}}_k^{(1)}, \ldots, \boldsymbol{\mathsf{z}}_k^{(\mathsf{M}_k)}$, which are modeled by an RFS $\mathsf{Z}_k = \{\boldsymbol{\mathsf{z}}_k^{(1)}, \ldots, \boldsymbol{\mathsf{z}}_k^{(\mathsf{M}_k)}\}$. An object with state $(\boldsymbol{x}_k, l)$ is detected by the sensor with probability $p_{\mathrm{D}}(\boldsymbol{x}_k, l)$, in which case it generates a measurement $\boldsymbol{\mathsf{z}}_k$ according to some conditional pdf (likelihood function) $f(\boldsymbol{z}_k | \boldsymbol{x}_k, l)$. Hence, the object-originated measurements are modeled by a multi-Bernoulli RFS $\mathsf{Z}_k^{\mathrm{O}}$ parametrized by $\{(p_{\mathrm{D}}(\boldsymbol{x}_k, l), f(\boldsymbol{z}_k | \boldsymbol{x}_k, l))\}_{l \in \mathbb{L}_k^*}$. Clutter-originated measurements are modeled by a Poisson RFS $\mathsf{Z}_k^{\mathrm{C}}$ that is independent of $\mathsf{Z}_k^{\mathrm{O}}$, given $\tilde{X}_k$, and parametrized by some mean parameter μ_{C} and spatial pdf $f_{\mathrm{C}}(\boldsymbol{z}_k)$ and, thus, PHD $\lambda_{\mathrm{C}}(\boldsymbol{z}_k) = \mu_{\mathrm{C}} f_{\mathrm{C}}(\boldsymbol{z}_k)$. The overall measurement RFS is given by $\mathsf{Z}_k = \mathsf{Z}_k^{\mathrm{O}} \cup \mathsf{Z}_k^{\mathrm{C}}$. The multiobject measurement model described above specifies the multiobject likelihood function $f(Z_k|\tilde{X}_k)$ [5, Section IV-C].

IV. REVIEW OF THE LMB FILTER

The LMB filter propagates the posterior multiobject pdf $f(\tilde{X}_k|Z_{1:k})$, where $Z_{1:k} \triangleq (Z_1, \ldots, Z_k)$. In the *prediction step* at time k, the preceding posterior pdf $f(\tilde{X}_{k-1}|Z_{1:k-1})$, which is of LMB form, is converted into a “predicted” posterior pdf $f(\tilde{X}_k|Z_{1:k-1})$. This pdf is again of LMB form, with parameters $\{(r_{k|k-1}^{(l)}, s_{k|k-1}^{(l)}(\boldsymbol{x}_k))\}_{l\in\mathbb{L}_k^*}$. Expressions of these parameters are provided in [9].

The *update step* converts $f(\tilde{X}_k|Z_{1:k-1})$ into $f(\tilde{X}_k|Z_{1:k})$. Consider the mapping $\theta_k : L \to \{0, 1, \ldots, M_k\}$ with $L \in \mathcal{F}(\mathbb{L}_k^*)$, where $\mathcal{F}(\mathbb{L}_k^*)$ is the set of all subsets of $\mathbb{L}_k^*$. Here, $\theta_k(l) = m \in \{1, \ldots, M_k\}$ indicates that object state $(\boldsymbol{x}_k, l)$ is associated with measurement m and $\theta_k(l) = 0$ indicates that it is not associated with any measurement. Let Θ_L denote the set of all mappings θ_k that describe *admissible* associations, i.e., assigning at most one measurement to the same object and no measurement to more than one object. As shown in [5] and [9], $f(\tilde{X}_k|Z_{1:k})$ is of GLMB form and can be expressed as

$$f(\tilde{X}_k|Z_{1:k}) = \Delta(\tilde{X}_k) \sum_{L\in\mathcal{F}(\mathbb{L}_k^*)} \sum_{\theta_k\in\Theta_L} w^{(L,\theta_k)}\, \delta_L(\mathcal{L}(\tilde{X}_k)) \times \prod_{(\boldsymbol{x}_k,l)\in\tilde{X}_k} s^{(l,\theta_k(l))}(\boldsymbol{x}_k). \tag{2}$$

Here, $\delta_L(\mathcal{L}(\tilde{X}_k))$ is one if $L = \mathcal{L}(\tilde{X}_k)$ and zero otherwise. Further, up to a normalization factor,

$$w^{(L,\theta_k)} \propto \left(\prod_{l'\in\mathbb{L}_k^*\setminus L} \left(1 - r_{k|k-1}^{(l')}\right) \right) \prod_{l\in L} r_{k|k-1}^{(l)}\, \eta^{(l,\theta_k(l))} \tag{3}$$

for $L \in \mathcal{F}(\mathbb{L}_k^*)$, where $\eta_k^{(l,m)}$ equals $B_k^{(l)} \triangleq \int (1 - p_{\mathrm{D}}(\boldsymbol{x}_k, l)) s_{k|k-1}^{(l)}(\boldsymbol{x}_k) \mathrm{d}\boldsymbol{x}_k$ for $m = 0$ and $C_k^{(l)}(\boldsymbol{z}_k^{(m)})/\lambda_{\mathrm{C}}(\boldsymbol{z}_k^{(m)})$ with $C_k^{(l)}(\boldsymbol{z}_k^{(m)}) \triangleq \int f(\boldsymbol{z}_k^{(m)}|\boldsymbol{x}_k, l)\, p_{\mathrm{D}}(\boldsymbol{x}_k, l)\, s_{k|k-1}^{(l)}(\boldsymbol{x}_k) \mathrm{d}\boldsymbol{x}_k$ for $m \in \{1, \ldots, M_k\}$. Finally,

$$s^{(l,m)}(\boldsymbol{x}_k) = \begin{cases} \big(1 - p_{\mathrm{D}}(\boldsymbol{x}_k, l)\big) s_{k|k-1}^{(l)}(\boldsymbol{x}_k)/B_k^{(l)}, & m = 0 \\ f(\boldsymbol{z}_k^{(m)}|\boldsymbol{x}_k, l) p_{\mathrm{D}}(\boldsymbol{x}_k, l) s_{k|k-1}^{(l)}(\boldsymbol{x}_k)/C_k^{(l)}(\boldsymbol{z}_k^{(m)}), & m \in \{1, \ldots, M_k\}. \end{cases} \tag{4}$$

In the LMB filter [9], the GLMB pdf in (2) is approximated by an LMB pdf (1) with existence probabilities

$$r_k^{(l)} = \sum_{L\in\mathcal{F}(\mathbb{L}_k^*)} \sum_{\theta_k\in\Theta_L} 1_L(l) w^{(L,\theta_k)} \tag{5}$$

and spatial pdfs

$$s^{(l)}(\boldsymbol{x}_k) = \frac{1}{r_k^{(l)}} \sum_{L\in\mathcal{F}(\mathbb{L}_k^*)} \sum_{\theta_k\in\Theta_L} 1_L(l) w^{(L,\theta_k)} s^{(l,\theta_k(l))}(\boldsymbol{x}_k) \tag{6}$$

for $l \in \mathbb{L}_k^*$.

V. A NEW DERIVATION OF THE LMB FILTER

We first show that the LMB filter can be derived by reformulating the GLMB posterior pdf in (2) in terms of a joint association pmf and approximating that pmf by the product of its marginals. A similar approach was used for the TOMB/P filter to obtain a multi-Bernoulli pdf from an unlabeled multi-Bernoulli mixture pdf [13]. We start by rewriting (2) as

$$f(\tilde{X}_k|Z_{1:k}) = \Delta(\tilde{X}_k) \sum_{\theta_k\in\Theta_{\mathcal{L}(\tilde{X}_k)}} w^{(\mathcal{L}(\tilde{X}_k),\theta_k)} \times \prod_{(\boldsymbol{x}_k,l)\in\tilde{X}_k} 1_{\mathbb{L}_k^*}(l) s^{(l,\theta_k(l))}(\boldsymbol{x}_k). \tag{7}$$

In (2), the factor $\delta_L(\mathcal{L}(\tilde{X}_k))$ with $L \in \mathcal{F}(\mathbb{L}_k^*)$ ensured that the labels of $\tilde{X}_k$, i.e., $\mathcal{L}(\tilde{X}_k)$, are from the set $\mathbb{L}_k^*$; this is now expressed by $\prod_{(\boldsymbol{x}_k,l)\in\tilde{X}_k} 1_{\mathbb{L}_k^*}(l)$. Next, instead of using the mapping θ_k to describe the object-measurement associations [9], [10], we introduce the association vector $\mathbf{a}_k$ with elements $\mathrm{a}_k^{(l)} \in \{-1, 0, \ldots, M_k\}$, where $l \in \mathbb{L}_k^*$. Here, $\mathrm{a}_k^{(l)} = m \in \{1, \ldots, M_k\}$ indicates that object state $(\boldsymbol{x}_k, l)$ is associated with measurement m, $\mathrm{a}_k^{(l)} = 0$ indicates that it is not associated with any measurement, and $\mathrm{a}_k^{(l)} = -1$ indicates that it does not exist, i.e., $(\boldsymbol{x}_k, l) \notin \tilde{X}_k$. Let $\mathcal{A}_k$ denote the set of admissible association vectors $\boldsymbol{a}_k$. Just as an admissible mapping θ_k in Section IV, an admissible association vector $\boldsymbol{a}_k$ assigns at most one measurement to the same object and no measurement to more than one object. We can now rewrite (7) as

$$f(\tilde{X}_k|Z_{1:k}) = \Delta(\tilde{X}_k) \sum_{\boldsymbol{a}_k\in\mathcal{A}_k} \varphi(\boldsymbol{a}_k, \tilde{X}_k) w_{\boldsymbol{a}_k} \times \prod_{(\boldsymbol{x}_k,l)\in\tilde{X}_k} 1_{\mathbb{L}_k^*(l)} s^{(l,a_k^{(l)})}(\boldsymbol{x}_k). \tag{8}$$

Here, $\varphi(\boldsymbol{a}_k, \tilde{X}_k) = 1$ for all $\boldsymbol{a}_k$ with $a_k^{(l)} = -1$ for $l \in \mathbb{L}_k^* \setminus \mathcal{L}(\tilde{X}_k)$ and $a_k^{(l)} \in \{0, \ldots, M_k\}$ for $l \in \mathcal{L}(X_k)$, and $\varphi(\boldsymbol{a}_k, \tilde{X}_k) = 0$ otherwise; this factor reduces the sum over all $\boldsymbol{a}_k \in \mathcal{A}_k$ in (8) to the sum over all corresponding mappings $\theta_k \in \Theta_{\mathcal{L}(\tilde{X}_k)}$ in (7). Furthermore, the weights $w_{\boldsymbol{a}_k}$ can be expressed as

$$w_{\boldsymbol{a}_k} \propto \prod_{l\in\mathbb{L}_k^*} \beta_k^{(l,a_k^{(l)})}, \quad \boldsymbol{a}_k \in \mathcal{A}_k \tag{9}$$

where the “association weights” $\beta_k^{(l,a_k^{(l)})}$ are defined as $r_{k|k-1}^{(l)} \eta^{(l,a_k^{(l)})}$ for $a_k^{(l)} \in \{0, \ldots, M_k\}$ and as $1 - r_{k|k-1}^{(l)}$ for $a_k^{(l)} = -1$ [cf. (3)]. Finally, $s^{(l,a_k^{(l)})}(\boldsymbol{x}_k)$ in (8) equals $s^{(l,\theta_k(l))}(\boldsymbol{x}_k)$ (with $\theta_k(l)$ replaced by $a_k^{(l)}$) because $s^{(l,-1)}(\boldsymbol{x}_k)$ does not occur in (8) (recall that $a_k^{(l)} = -1$ implies $(\boldsymbol{x}_k, l) \notin \tilde{X}_k$). In contrast to the weights $w^{(L,\theta_k)}$ in (2), the $w_{\boldsymbol{a}_k}$ do not depend on $\mathcal{L}(\tilde{X}_k)$. They are normalized in that $\sum_{\boldsymbol{a}_k\in\mathcal{A}_k} w_{\boldsymbol{a}_k} = 1$. Expressions (5) and (6) can now be reformulated in terms of $\boldsymbol{a}_k$ as

$$r_k^{(l)} = \sum_{\boldsymbol{a}_k \in \mathcal{A}_k^{(l)}} w_{\boldsymbol{a}_k} \tag{10}$$

$$s^{(l)}(\boldsymbol{x}_k) = \frac{1}{r_k^{(l)}} \sum_{\boldsymbol{a}_k \in \mathcal{A}_k^{(l)}} w_{\boldsymbol{a}_k} s^{(l, a_k^{(l)})}(\boldsymbol{x}_k) \tag{11}$$

where $\mathcal{A}_k^{(l)} \triangleq \{\boldsymbol{a}_k \in \mathcal{A}_k : a_k^{(l)} \in \{0, \ldots, M_k\}\}$. This is possible because (5) and (6) contain only terms involving $w^{(L,\theta_k)}$ with L such that $l \in L$; this can be expressed via $\boldsymbol{a}_k$ by removing all $\boldsymbol{a}_k$ with $a_k^{(l)} = -1$ from $\mathcal{A}_k$, which results in $\mathcal{A}_k^{(l)}$.

With this reformulation, we can interpret the weights $w_{\boldsymbol{a}_k}$ as the pmf of the association vector $\mathbf{a}_k$. More precisely, we define the pmf of $\mathbf{a}_k$ as $p(\boldsymbol{a}_k) = w_{\boldsymbol{a}_k}$ for $\boldsymbol{a}_k \in \mathcal{A}_k$ and $p(\boldsymbol{a}_k) = 0$ otherwise. We can then rewrite (8) as

$$f(\tilde{X}_k | Z_{1:k}) = \Delta(\tilde{X}_k) \sum_{\boldsymbol{a}_k \in \mathcal{M}_k^{|\mathbb{L}_k^*|}} \varphi(\boldsymbol{a}_k, \tilde{X}_k) p(\boldsymbol{a}_k) \times \prod_{(\boldsymbol{x}_k, l) \in \tilde{X}_k} 1_{\mathbb{L}_k^*}(l) s^{(l, a_k^{(l)})}(\boldsymbol{x}_k) \tag{12}$$

with $\mathcal{M}_k \triangleq \{-1, 0, \ldots, M_k\}$. Note that $\sum_{\boldsymbol{a}_k \in \mathcal{A}_k}$ in (8) can be replaced by $\sum_{\boldsymbol{a}_k \in \mathcal{M}_k^{|\mathbb{L}_k^*|}}$ since $p(\boldsymbol{a}_k) = 0$ for $\boldsymbol{a}_k \in \mathcal{M}_k^{|\mathbb{L}_k^*|} \setminus \mathcal{A}_k$. Next, we approximate $p(\boldsymbol{a}_k)$ by the product of its marginals, i.e.,

$$p(\boldsymbol{a}_k) \approx \prod_{l \in \mathbb{L}_k^*} p\big(a_k^{(l)}\big), \quad \boldsymbol{a}_k \in \mathcal{M}_k^{|\mathbb{L}_k^*|} \tag{13}$$

where

$$p\big(a_k^{(l)}\big) = \sum_{\boldsymbol{a}_k^{\sim l} \in \mathcal{M}_k^{|\mathbb{L}_k^*|-1}} p(\boldsymbol{a}_k). \tag{14}$$

(Here, $\boldsymbol{a}_k^{\sim l}$ denotes the vector $\boldsymbol{a}_k$ with the lth component, $a_k^{(l)}$, removed.) Inserting (13) into (12) yields

$$f(\tilde{X}_k | Z_{1:k}) \approx \Delta(\tilde{X}_k) \sum_{\boldsymbol{a}_k \in \mathcal{M}_k^{|\mathbb{L}_k^*|}} \left(\prod_{l' \in \mathbb{L}_k^*} p\big(a_k^{(l')}\big) \right) \varphi(\boldsymbol{a}_k, \tilde{X}_k) \times \prod_{(\boldsymbol{x}_k, l) \in \tilde{X}_k} 1_{\mathbb{L}_k^*}(l) s^{(l, a_k^{(l)})}(\boldsymbol{x}_k). \tag{15}$$

Using $\sum_{\boldsymbol{a}_k \in \mathcal{M}_k^{|\mathbb{L}_k^*|}} = \sum_{a_k^{(1)} \in \mathcal{M}_k} \cdots \sum_{a_k^{(|\mathbb{L}_k^*|)} \in \mathcal{M}_k}$, splitting $\prod_{l' \in \mathbb{L}_k^*} p(a_k^{(l')})$ as $(\prod_{l' \in \mathbb{L}_k^* \setminus \mathcal{L}(\tilde{X}_k)} p(a_k^{(l')})) \prod_{l \in \mathcal{L}(\tilde{X}_k)} p(a_k^{(l)})$, evaluating $\varphi(\boldsymbol{a}_k, \tilde{X}_k)$, and grouping terms, we obtain

$$f(\tilde{X}_k | Z_{1:k}) \approx \Delta(\tilde{X}_k) P(\tilde{X}_k) \prod_{(\boldsymbol{x}_k, l) \in \tilde{X}_k} 1_{\mathbb{L}_k^*}(l) \times \sum_{a_k^{(l)}=0}^{M_k} p\big(a_k^{(l)}\big) s^{(l, a_k^{(l)})}(\boldsymbol{x}_k) \tag{16}$$

with $P(\tilde{X}_k) \triangleq \prod_{l \in \mathbb{L}_k^* \setminus \mathcal{L}(\tilde{X}_k)} p(a_k^{(l)} = -1)$. Comparing expression (16) with (1), we conclude that it is an LMB pdf with existence probabilities

$$r_k^{(l)} = 1 - p(a_k^{(l)} = -1) = \sum_{a_k^{(l)}=0}^{M_k} p(a_k^{(l)}) \tag{17}$$

and spatial pdfs

$$s^{(l)}(\boldsymbol{x}_k) = \frac{1}{r_k^{(l)}} \sum_{a_k^{(l)}=0}^{M_k} p(a_k^{(l)}) s^{(l, a_k^{(l)})}(\boldsymbol{x}_k). \tag{18}$$

Finally, we show that (17) and (18) are identical to (5) and (6), respectively. Inserting (14) into (17), we obtain

$$r_k^{(l)} = \sum_{a_k^{(l)}=0}^{M_k} \sum_{\boldsymbol{a}_k^{\sim l} \in \mathcal{M}_k^{|\mathbb{L}_k^*|-1}} p(\boldsymbol{a}_k) = \sum_{\boldsymbol{a}_k \in \mathcal{A}_k^{(l)}} w_{\boldsymbol{a}_k} \tag{19}$$

where the last expression follows because $p(\boldsymbol{a}_k)$ equals $w_{\boldsymbol{a}_k}$ for $\boldsymbol{a}_k \in \mathcal{A}_k$ and 0 otherwise. Thus, (17) is identical to (10) and, hence, to (5). Similarly, it can be verified that (18) is identical to (11) and, hence, to (6). This shows that our LMB approximation (15) of the GLMB posterior pdf $f(\tilde{X}_k | Z_{1:k})$ in (2) is equivalent to the LMB approximation underlying the original LMB filter [9].

VI. A NEW FAST LMB FILTER

We now leverage the new formulation of the LMB filter derived above for a reduction of complexity. More specifically, slightly adapting the algorithm of [15], we present a fast BP-based approximate calculation of the marginal association probabilities $p(a_k^{(l)})$ involved in (17) and (18). First, we review the general framework of factor graphs and BP [17].

A. Review of Belief Propagation

Consider J discrete random variables a_j, $j = 1, \ldots, J$. We would like to calculate the marginal pmfs $p(a_j)$ from the joint pmf $p(\boldsymbol{a})$ with $\mathbf{a} = [\mathrm{a}_1 \cdots \mathrm{a}_J]^{\mathrm{T}}$. However, often a direct marginalization is computationally infeasible.

Using BP (or, equivalently, the *sum-product algorithm* [17]), the marginalizations yielding the pmfs $p(a_j)$, $j = 1, \ldots, J$ can be performed—at least approximately—in an efficient manner if $p(\boldsymbol{a})$ factorizes according to

$$p(\boldsymbol{a}) \propto \prod_{q=1}^{Q} \psi_q(\boldsymbol{a}^{(q)}). \tag{20}$$

Here, each argument $\boldsymbol{a}^{(q)}$ comprises certain variables a_j. The factorization (20) can be represented by a factor graph, in which each variable a_j is represented by a variable node, each factor $\psi_q(\cdot)$ is represented by a factor node, and variable node "a_j" and factor node "ψ_q" are adjacent, i.e., connected by an edge, if the variable a_j is an argument of the factor $\psi_q(\cdot)$ and, thus, part of $\boldsymbol{a}^{(q)}$. Fig. 1 considers the case where $\mathbf{a} = [\mathrm{a}_1\ \mathrm{a}_2]^{\mathrm{T}}$ and shows the factor graph representing the factorization

$$p(\boldsymbol{a}) \propto \psi_1(a_1) \psi_2(a_1, a_2) \psi_3(a_2). \tag{21}$$

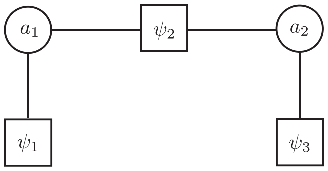

Fig. 1. Factor graph representing the factorization of the pmf $p(\boldsymbol{a}) \propto \psi_1(a_1)\psi_2(a_1, a_2)\psi_3(a_2)$, with $\mathbf{a} = [\mathrm{a}_1\ \mathrm{a}_2]^{\mathrm{T}}$. Variable nodes are depicted as circles and factor nodes as squares.

BP is a message passing algorithm where each node in the factor graph passes messages to the adjacent nodes. More specifically, consider a variable node "a_j" and an adjacent factor node "ψ_q", i.e., the variable a_j is part of the argument $\boldsymbol{a}^{(q)}$ of $\psi_q(\boldsymbol{a}^{(q)})$. Then, the message passed from factor node "ψ_q" to variable node "a_j" is given by

$$\phi^{(\psi_q \to a_j)}(a_j) = \sum_{\boldsymbol{a}_{\sim j}} \psi_q(\boldsymbol{a}^{(q)}) \prod_{j' \in \mathcal{J}_q \setminus \{j\}} \eta^{(a_{j'} \to \psi_q)}(a_{j'}) \quad (22)$$

where $\mathcal{J}_q$ denotes the neighborhood set of factor node "ψ_q" (i.e., the set of indices j of all variable nodes "a_j" that are adjacent to factor node "ψ_q"), $\sum_{\boldsymbol{a}_{\sim j}}$ denotes summation with respect to all variables $a_{j'}$, $j' \in \mathcal{J}_q$ except a_j, and $\eta^{(a_{j'} \to \psi_q)}(a_{j'})$ is the message passed from variable node "$a_{j'}$" to factor node "ψ_q" (to be explained presently). For example, the message passed from factor node "ψ_2" to variable node "a_2" in Fig. 1 is $\phi^{(\psi_2 \to a_2)}(a_2) = \sum_{a_1} \psi_2(a_1, a_2)\eta^{(a_1 \to \psi_2)}(a_1)$. The message $\eta^{(a_j \to \psi_q)}(a_j)$ passed from variable node "a_j" to factor node "ψ_q" is given by the product of the messages passed to variable node "a_j" from all adjacent factor nodes except "ψ_q", i.e.,

$$\eta^{(a_j \to \psi_q)}(a_j) = \prod_{q' \in \mathcal{Q}_j \setminus \{q\}} \phi^{(\psi_{q'} \to a_j)}(a_j) \quad (23)$$

where the neighborhood set $\mathcal{Q}_j$ comprises the set of indices q of all factor nodes "ψ_q" that are adjacent to variable node "a_j". For example, in Fig. 1, the message passed from variable node "a_2" to factor node "ψ_3" is $\eta^{(a_2 \to \psi_3)}(a_2) = \phi^{(\psi_2 \to a_2)}(a_2)$. This message passing process is started at variable nodes with only one edge, which pass a constant message, and/or factor nodes with only one edge, which pass the corresponding factor. We note that BP can also be applied to functions involving continuous variables; the only difference is that in (22) the respective summations are replaced with integrations.

When all messages have been passed as described above, then for each variable node "a_j", a *belief* $\tilde{p}(a_j)$ is calculated as the product of all incoming messages (passed from all adjacent factor nodes) followed by a normalization such that $\sum_{a_j} \tilde{p}(a_j) = 1$. For example, in Fig. 1,

$$\tilde{p}(a_2) \propto \phi^{(\psi_2 \to a_2)}(a_2)\phi^{(\psi_3 \to a_2)}(a_2). \quad (24)$$

If the factor graph is a tree, then the obtained belief $\tilde{p}(a_j)$ is exactly equal to the marginal pmf $p(a_j)$. On the other hand, if the factor graph contains cycles (loops), BP is usually applied in an iterative manner, and the beliefs $\tilde{p}(a_j)$ are only approximations of the respective marginal pmfs $p(a_j)$. In these iterative "loopy BP" schemes, there is no canonical order in which the messages should be calculated, and different orders may lead to different beliefs. For each node, the outgoing message can be calculated as soon as the incoming messages involved in (22) or (23) are available. The choice of an order (schedule) of message calculation provides a certain flexibility in the design of BP-based inference algorithms.

B. BP-Based Probabilistic Data Association

We now present a fast BP-based algorithm for calculating approximations of the marginal association probabilities $p(a_k^{(l)})$, $l \in \mathbb{L}_k^*$ involved in (17) and (18). This algorithm is a variant[1] of the BP scheme for probabilistic data association proposed in [15]. We recall that $\boldsymbol{a}_k \in \mathcal{M}_k^{|\mathbb{L}_k^*|}$ and, further, that $p(\boldsymbol{a}_k) = w_{\boldsymbol{a}_k}$ for $\boldsymbol{a}_k \in \mathcal{A}_k \subseteq \mathcal{M}_k^{|\mathbb{L}_k^*|}$ and $p(\boldsymbol{a}_k) = 0$ otherwise. Using (9), we can then express the joint association pmf $p(\boldsymbol{a}_k)$ as

$$p(\boldsymbol{a}_k) \propto \Psi(\boldsymbol{a}_k) \prod_{l \in \mathbb{L}_k^*} \beta_k^{(l, a_k^{(l)})}, \quad \boldsymbol{a}_k \in \mathcal{M}_k^{|\mathbb{L}_k^*|} \quad (25)$$

where $\Psi(\boldsymbol{a}_k) = 1$ if $\boldsymbol{a}_k \in \mathcal{A}_k$ and $\Psi(\boldsymbol{a}_k) = 0$ otherwise. Note that $\Psi(\boldsymbol{a}_k)$ enforces the admissibility (defined in Section V) of the association described by $\boldsymbol{a}_k$. Without $\Psi(\boldsymbol{a}_k)$, (25) would describe the probability of "independent" single-object associations, and in the resulting algorithm, each object would be tracked without taking into account the presence of other objects. This would produce track losses when objects are in close proximity.

Following [15], we introduce the alternative association vector $\mathbf{b}_k$ with elements $\mathrm{b}_k^{(m)}$, $m \in \{1, \ldots, M_k\}$, where $\mathrm{b}_k^{(m)} = l \in \mathbb{L}_k^*$ indicates that measurement m is associated with object state $(\boldsymbol{x}_k, l)$ and $\mathrm{b}_k^{(m)} = 0$ indicates that measurement m is not associated with any object state. We can reformulate the joint association pmf $p(\boldsymbol{a}_k)$ in terms of *both* $\mathbf{a}_k$ and $\mathbf{b}_k$. Indeed, analogously to (25), we can express $p(\boldsymbol{a}_k, \boldsymbol{b}_k)$ as

$$p(\boldsymbol{a}_k, \boldsymbol{b}_k) \propto \Psi(\boldsymbol{a}_k, \boldsymbol{b}_k) \prod_{l \in \mathbb{L}_k^*} \beta_k^{(l, a_k^{(l)})}. \quad (26)$$

Here, the admissibility of $\boldsymbol{a}_k$ and $\boldsymbol{b}_k$ is enforced by the factor

$$\Psi(\boldsymbol{a}_k, \boldsymbol{b}_k) = \prod_{l \in \mathbb{L}_k^*} \prod_{m=1}^{M_k} \Psi_{l,m}(a_k^{(l)}, b_k^{(m)}) \quad (27)$$

where $\Psi_{l,m}(a_k^{(l)}, b_k^{(m)}) = 0$ if either $a_k^{(l)} = m$ and $b_k^{(m)} \neq l$ or $a_k^{(l)} \neq m$ and $b_k^{(m)} = l$, and $\Psi_{l,m}(a_k^{(l)}, b_k^{(m)}) = 1$ otherwise.

[1] The algorithm in [15] is not suited in our context because it presupposes that the number of objects is known. The related algorithm in [13] is not suited either because it combines the association weights for object nonexistence, $\beta_k^{(l,-1)}$, and those for a missed detection, $\beta_k^{(l,0)}$, into common association weights and also includes association weights for objects that are detected for the first time.

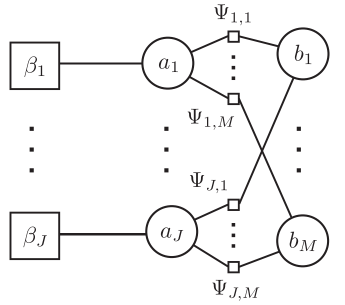

Fig. 2. Factor graph representing the factorization (26), (27). The following short notations are used: $\beta_j \triangleq \beta_k^{(l',m')}$, $a_j \triangleq a_k^{(l')}$, $b_m \triangleq b_k^{(m)}$, $\Psi_{j,m} \triangleq \Psi_{l',m}(a_k^{(l')}, b_k^{(m)})$, $M \triangleq M_k$, and $J = |\mathbb{L}_k^*|$, with $l' \triangleq l^{(j)}$ and $m' \triangleq a_k^{(l^{(j)})}$. Here, $l^{(j)} \in \mathbb{L}_k^* = \{l^{(1)}, \ldots, l^{(J)}\}$.

The vector $\mathbf{b}_k$ does not carry any additional association information compared to the vector $\mathbf{a}_k$. However, as discussed in [14] and [15], the redundant formulation of the joint association pmf using $\mathbf{a}_k$ and $\mathbf{b}_k$ in parallel, as given by (26) and (27), enables a fast method for BP-based probabilistic data association. On a more general level, the introduction of additional random variables that are redundant in that they deterministically depend on existing random variables (such as $\mathbf{b}_k$, which deterministically depends on $\mathbf{a}_k$) is a common means of expanding factor graphs [17]. In many cases, using BP on the expanded graph is more computationally efficient than using BP on the original graph. In our case, the introduction of the redundant association vector $\mathbf{b}_k$ results in the expression (27) of the admissibility constraint, which has the important property that it *completely factorizes* into individual components indexed by $(l, m) \in \mathbb{L}_k^* \times \{1, \ldots, M_k\}$. As we will show next, this complete factorization allows us to devise a fast algorithm for probabilistic data association.

The factorization (26), (27) can be represented by the factor graph [17] shown in Fig. 2. Then, still following [15], approximations of the marginal association pmfs $p(a_k^{(l)})$ and $p(b_k^{(m)})$ can be obtained via iterative BP message passing.[2] In BP iteration $i \in \{1, \ldots, I\}$, a message $\zeta_k^{[i](l\to m)}$ is passed from variable node "$a_k^{(l)}$" via factor node "$\Psi_{l,m}(a_k^{(l)}, b_k^{(m)})$" to variable node "$b_k^{(m)}$", and a message $\nu_k^{[i](m\to l)}$ is passed from variable node "$b_k^{(m)}$" via factor node "$\Psi_{l,m}(a_k^{(l)}, b_k^{(m)})$" to variable node "$a_k^{(l)}$". Using (22) and (23), it is shown in the appendix that these messages are given by

$$\zeta_k^{[i](l\to m)} = \frac{\beta_k^{(l,m)}}{\beta_k^{(l,-1)} + \beta_k^{(l,0)} + \sum_{\substack{m'=1\\ m'\neq m}}^{M_k} \beta_k^{(l,m')}\nu_k^{[i-1](m'\to l)}} \tag{28}$$

$$\nu_k^{[i](m\to l)} = \frac{1}{1 + \sum_{l'\in\mathbb{L}_k^*\setminus\{l\}} \zeta_k^{[i](l'\to m)}} \tag{29}$$

for $l \in \mathbb{L}_k^*$ and $m \in \{1, \ldots, M_k\}$. The recursion established by these two equations is initialized by $\nu_k^{[0](m\to l)} = 1$. After the final iteration $i = I$, approximations of the marginal association pmfs $p(a_k^{(l)})$, $l \in \mathbb{L}_k^*$ are provided by the beliefs at the respective variable nodes "$a_k^{(l)}$" in Fig. 2. As shown in the appendix, these beliefs are obtained as

$$\tilde{p}(a_k^{(l)} = m) = \begin{cases} \beta_k^{(l,m)}/D_k^{(l)}, & m \in \{-1, 0\} \\ \beta_k^{(l,m)}\nu_k^{[I](m\to l)}/D_k^{(l)}, & m \in \{1, \ldots, M_k\} \end{cases} \tag{30}$$

where $D_k^{(l)} \triangleq \beta_k^{(l,-1)} + \beta_k^{(l,0)} + \sum_{m'=1}^{M_k} \beta_k^{(l,m')}\nu_k^{[I](m'\to l)}$. Similarly, an approximation of $p(b_k^{(m)} = 0)$—to be used in Section VII—is obtained as

$$\tilde{p}(b_k^{(m)} = 0) = \frac{1}{1 + \sum_{l\in\mathbb{L}_k^*} \zeta_k^{[I](l\to m)}}. \tag{31}$$

The proposed fast LMB filter is finally obtained by using the BP-based approximate calculation of the $p(a_k^{(l)})$ according to (28)–(30) in the update equations (17) and (18). We emphasize that our fast LMB filter is different from the LMB filters in [9], [10] because the underlying BP-based approximation is different from the approximations employed in [9] or [10]. A pseudocode of our LMB filter is provided in [16], and a MATLAB implementation is available at https://github.com/ThoKro/BP-LMB.git

C. Complexity Analysis

The complexity of the fast Gibbs sampler-based LMB filter recently proposed in [10] is $\mathcal{O}(P|\mathbb{L}_k^*|^2 M_k)$, where P is the number of samples used in the Gibbs sampler and, as before, $|\mathbb{L}_k^*|$ and M_k are the numbers of Bernoulli components and measurements, respectively. By contrast, the complexity of our proposed LMB filter is $\mathcal{O}(I|\mathbb{L}_k^*|M_k)$, where I is the number of BP iterations. The linear scaling in $|\mathbb{L}_k^*|$ improves on the quadratic scaling exhibited by the Gibbs sampler-based LMB filter. The second difference is that P is replaced by I. A typical value of I (to be used in our simulations in Section VII) is 20. As we will demonstrate in Section VII, for scenarios with a high clutter rate and/or a large number of objects, P has to be chosen much higher than 20 in order for the tracking performance of the Gibbs sampler-based LMB filter to be similar to that of our BP-based LMB filter.

Some algorithmic aspects affecting the complexity and performance of the BP-based and Gibbs sampler-based LMB filters are discussed in Section VII-B.

[2] We note that, as studied in [18], approximations of the $p(a_k^{(l)})$ can also be calculated by running the BP algorithm on a factor graph containing only the variable nodes "$a_k^{(l)}$" and a factor node representing the admissibility constraint factor $\psi(\boldsymbol{a}_k)$. However, these approximations are inferior to those obtained by running the BP algorithm on the factor graph of Fig. 2 [18].

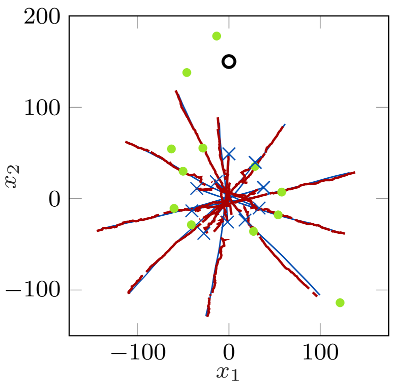

Fig. 3. Example of true trajectories for parameter setting PS1 (blue lines; starting points indicated by blue crosses), as well as of trajectories estimated by the proposed BP-LMB filter (red lines) and measurements acquired at time $k = 100$ (green dots). The black circle indicates the sensor position.

VII. NUMERICAL STUDY

A. Simulation Setup

We consider a two-dimensional tracking scenario [19], where a sensor is located at $\boldsymbol{p} = [p_1 \;\; p_2]^{\mathrm{T}} = [0 \;\; 150]^{\mathrm{T}}$. The sensor has a measurement range of 300 and the region of interest (ROI) is equal the sensor's field of view, which is the disk determined by the sensor's measurement range. We consider two different parameter settings dubbed PS1 and PS2. Ten (PS1) or twenty (PS2) objects appear before $k = 30$ and disappear after $k = 140$. The object states consist of position and velocity, i.e., $\boldsymbol{x}_k = [x_{1,k} \;\, x_{2,k} \;\, \dot{x}_{1,k} \;\, \dot{x}_{2,k}]^{\mathrm{T}}$. They evolve according to the nearly constant velocity model [20, Section 6.3.2] with an iid Gaussian driving process of variance $\sigma_u^2 = 10^{-4}$. We employ the trajectory generation scheme of [19], according to which all objects move toward the point (0,0), come close to each other there around time $k = 60$, and separate again afterwards. A detailed description of this trajectory generation scheme can be found in [19], and a realization of the object trajectories is shown in Fig. 3. The sensor is characterized by the nonlinear range-bearing measurement model

$$\boldsymbol{z}_k = [\rho(\boldsymbol{x}_k) \quad \phi(\boldsymbol{x}_k)]^{\mathrm{T}} + \boldsymbol{v}_k. \tag{32}$$

Here, $\rho(\boldsymbol{x}_k) \triangleq \|\boldsymbol{x}'_k - \boldsymbol{p}\|$, where $\boldsymbol{x}'_k \triangleq [x_{1,k} \;\, x_{2,k}]^{\mathrm{T}}$ denotes the position of an object, and $\phi(\boldsymbol{x}_k) \triangleq \tan^{-1}(\frac{x_{1,k}-p_1}{x_{2,k}-p_2})$. Furthermore, $\boldsymbol{v}_k$ is iid Gaussian measurement noise with independent components and component standard deviations $\sigma_\rho = 2$ and $\sigma_\phi = 1^\circ$. The clutter pdf $f_{\mathrm{C}}(\boldsymbol{z}_k)$ is uniform (in polar coordinates) on the ROI, and the mean parameter μ_{C} is 10 (PS1) or 50 (PS2). Objects are detected by the sensor with probability $p_{\mathrm{D}} = 0.5$.

We study the performance of the proposed BP-based LMB filter (briefly termed BP-LMB filter) in comparison to the Gibbs sampler-based LMB filter [10] (briefly Gibbs-LMB filter) and the fast BP-based version of the label-augmented TOMB/P filter [13]–[15] (briefly BP-TOMB/P filter). All filters use particle implementations [9], [21]. They represent the spatial pdf of each Bernoulli component by 1000 particles, prune components with existence probability below 10^{-3}, declare an object as detected if its existence probability exceeds 0.5, and use $p_{\mathrm{S}}(\boldsymbol{x}_{k-1}, l) = p_{\mathrm{S}}(\boldsymbol{x}_{k-1}) = 0.99$ and $p_{\mathrm{D}}(\boldsymbol{x}_k, l) = p_{\mathrm{D}}(\boldsymbol{x}_k) = 0.5$. With regard to the newborn objects, the Gibbs-LMB filter generates a new Bernoulli component for each measurement observed at the preceding time $k-1$ [10]; the existence probability of that Bernoulli component is initialized as μ_{B}/M_{k-1} with $\mu_{\mathrm{B}} = 0.1$. By contrast, the BP-LMB filter generates new Bernoulli components only for measurements $m \in \{1, \ldots, M_{k-1}\}$ with $\tilde{p}(b_{k-1}^{(m)} = 0) > 0.5$ (cf. (31)), i.e., for measurements that are considered not to be generated by an already existing object. The existence probabilities of these Bernoulli components are chosen as $(\mu_{\mathrm{B}}/M_{k-1})\tilde{p}(b_{k-1}^{(m)} = 0)$, $l \in \mathbb{L}_k^{\mathrm{B}*}$, with $\mu_{\mathrm{B}} = 0.1$. Here, each $l \in \mathbb{L}_k^{\mathrm{B}*}$ is associated with a measurement m for which $\tilde{p}(b_{k-1}^{(m)} = 0) > 0.5$. Both the Gibbs-LMB filter and the BP-LMB filter choose the spatial pdf of new Bernoulli components as

$$s_{\mathrm{B}}^{(l)}(\boldsymbol{x}_k) \propto \int f(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}) f(\boldsymbol{z}_{k-1}|x_{1,k-1}, x_{2,k-1}) \times f_{\mathrm{v}}(\dot{x}_{1,k-1}, \dot{x}_{2,k-1}) \mathrm{d}\boldsymbol{x}_{k-1} \tag{33}$$

for $l \in \mathbb{L}_k^{\mathrm{B}*}$, where $f(\boldsymbol{z}_{k-1}|x_{1,k-1}, x_{2,k-1})$ is the likelihood function corresponding to our measurement model (32) and $f_{\mathrm{v}}(\dot{x}_{1,k-1}, \dot{x}_{2,k-1})$ is the pdf of independent, zero-mean, Gaussian random variables $\dot{\mathrm{x}}_{1,k-1}$, $\dot{\mathrm{x}}_{2,k-1}$ with variance 0.25. The number P of samples used by the Gibbs sampler in the Gibbs-LMB filter is 100 or 1000; the resulting Gibbs-LMB filters will be referred to as Gibbs-LMB-100 and Gibbs-LMB-1000, respectively. In the BP-TOMB/P filter, the Poisson RFS modeling newborn objects has mean parameter $\mu_{\mathrm{B}} = 0.3$ and its spatial pdf is uniform on the ROI; furthermore, the PHD of the Poisson RFS of "undetected objects" is chosen as a k-dependent constant on the ROI, which is updated as discussed in [21] and initialized as $\mu_{\mathrm{U}} f_{\mathrm{U}}(\boldsymbol{x}_0)$ with $\mu_{\mathrm{U}} = 0.01$ and uniform $f_{\mathrm{U}}(\boldsymbol{x}_0)$. The BP-TOMB/P filter generates a new Bernoulli component for each measurement at time k. The BP-LMB and BP-TOMB/P filters use $I = 20$ BP iterations to calculate the approximate marginal probabilities.

B. Simulation Results

The example shown in Fig. 3 suggests that the proposed BP-LMB filter has excellent detection and estimation performance. For a quantitative evaluation of the average performance of the three filters, we use the Euclidean distance-based optimal subpattern assignment (OSPA) metric with cutoff parameter $c = 20$ and order $p = 1$ [22]. The OSPA metric penalizes both a deviation between the estimated and true numbers of objects and deviations between the estimated and true object states [22].

Fig. 4 shows the mean OSPA (MOSPA) error—averaged over 1000 simulation runs—versus time k for PS1 (10 objects, $\mu_{\mathrm{C}} = 10$) and PS2 (20 objects, $\mu_{\mathrm{C}} = 50$). For PS1, the BP-LMB, BP-TOMB/P, and Gibbs-LMB-1000 filters perform best and almost identically, closely followed

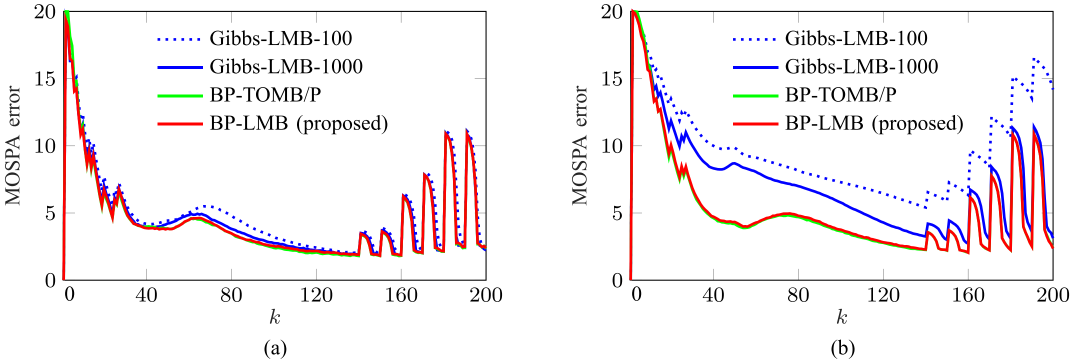

Fig. 4. MOSPA error versus time. (a) PS1. (b) PS2.

by the Gibbs-LMB-100 filter. For the more challenging setting PS2, the BP-LMB and BP-TOMB/P filters perform best and almost identically, whereas both Gibbs-LMB filters perform substantially worse: already the Gibbs-LMB-1000 filter has a significantly larger MOSPA error during a long time interval, and the Gibbs-LMB-100 filter has an even larger MOSPA error at almost all times. If the number of samples is increased beyond 1000 (not shown in Fig. 4), the MOSPA error of the Gibbs-LMB filter decreases, but this comes at the cost of a higher complexity.

The performance difference between the BP-LMB filter and the Gibbs-LMB filter for PS2 can be explained as follows. The Gibbs-LMB filter reduces complexity by pruning GLMB components with low weights in (2). As a consequence, the summations in the update equations (5) and (6) are performed only over the remaining (nonpruned) components. The pruning performed by the Gibbs-LMB filter can be equivalently formulated in terms of the association vector $\mathbf{a}_k$ introduced in Section V. In this formulation, the pruning is based on drawing samples $\tilde{\boldsymbol{a}}_k$ from the pmf $p(\boldsymbol{a}_k)$, where each $\tilde{\boldsymbol{a}}_k$ corresponds to one GLMB component. After sampling, all GLMB components that do not correspond to a sample $\tilde{\boldsymbol{a}}_k$ are pruned. In PS2, the large numbers of objects and clutter measurements lead to a large number of relevant GLMB components with significant pmf values. As a consequence, if the number of samples is small, some of the relevant GLMB components are necessarily pruned, which means that relevant association information is ignored by the Gibbs-LMB filter. This results in a reduced tracking performance of the Gibbs-LMB filter in PS2. By contrast, in the BP-LMB filter, the approximate calculation of the marginal association probabilities using BP completely avoids the pruning of components.

We conclude from Fig. 4 that for both PS1 and PS2, the proposed BP-LMB filter performs better than or similarly to the other filters. An interesting observation is the similarity of performance relative to the BP-TOMB/P filter. Indeed, a deeper analysis—which is beyond the scope of this work—shows that despite the differences in the underlying state and system models, the BP-LMB and BP-TOMB/P filters are quite similar algorithmically. The BP-TOMB/P filter differs from the BP-LMB filter mainly in that it models undetected objects by a Poisson RFS. This results in a higher complexity but, in the scenarios considered, does not yield a better tracking performance. However, we note that a more general (nonuniform) PHD representation of undetected objects in the BP-TOMB/P filter, e.g., using particles, may lead to a faster detection of newborn objects and, thus, a lower OSPA metric in scenarios with a nonuniform detection probability and/or a nonuniform birth process. On the other hand, this would result in a further increase in computational complexity.

TABLE I
Total Runtime and AP Runtime for PS2

Filter	Total runtime	AP runtime
BP-LMB (proposed)	0.4435 s	$2.28\cdot10^{-7}$ s
Gibbs-LMB-100	0.4888 s	$2.98\cdot10^{-5}$ s
BP-TOMB/P	0.8189 s	$3.48\cdot10^{-7}$ s
Gibbs-LMB-1000	1.7318 s	$3.56\cdot10^{-4}$ s

Table I lists the average runtimes of the different filters per time (k) step, referred to as “total runtimes,” as well as the average runtimes used for calculating one approximate marginal association probability, referred to as “AP runtimes.” The approximate marginal association probabilities are given by $\tilde{p}(a_k^{(l)} = m)$ in (30) for the proposed BP-LMB filter and similarly for the BP-TOMB/P filter, and analogous quantities are computed by the Gibbs-LMB filter using the Gibbs sampling algorithm. The runtimes were obtained for PS2, using a MATLAB implementation on an Intel quad-core i7-6600U CPU. The results for the total runtimes show that the proposed BP-LMB filter is here less complex than the BP-TOMB/P filter and the Gibbs-LMB-100 filter, and significantly less complex than the Gibbs-LMB-1000 filter. Furthermore, the AP runtimes of the BP-LMB and BP-TOMB/P filters are significantly lower than those of the Gibbs-LMB filter. Finally, as may be expected, the total and AP runtimes of the Gibbs-LMB filter increase with the number of Gibbs samples.

The observed lower runtimes of the BP-LMB filter compared to the Gibbs-LMB-1000 filter reflect also the linear scaling behavior of the BP algorithm compared to the quadratic scaling behavior of the Gibbs sampler-based calculation (cf. Section VI-C). On the other hand, the Gibbs-LMB filter employs a smaller number of Bernoulli components than the BP-LMB filter; this is a consequence of the reduction of the number of summation terms in (5) and (6) caused by the Gibbs sampling. However, this effect is counteracted by the fact that the complexity of the Gibbs-LMB filter scales quadratically in the number of Bernoulli components. This, together with the fact that the number P of samples used by the Gibbs sampler is considerably larger than the number I of BP iterations, causes the Gibbs-LMB filter to be more complex than the BP-LMB filter. Finally, the higher runtime of the BP-TOMB/P filter results from additional operations related to an explicit modeling of undetected objects and a different strategy for generating Bernoulli components.

We also simulated the BP-LMB, Gibbs-LMB, and BP-TOMB/P filters in scenarios with less clutter, fewer objects, and a higher detection probability. Our results (not shown here) demonstrate that in these scenarios, the number P of samples used by the Gibbs-LMB filter can be reduced without increasing the MOSPA error. This can result in an almost identical performance of the three filters and a similar complexity (runtime) of the BP-LMB and Gibbs-LMB filters, which is lower than that of the BP-TOMB/P filter. However, in challenging scenarios with a high clutter rate and/or a large number of (closely spaced) objects and/or a low detection probability, our results in Fig. 4 suggest that for a similar performance of the Gibbs-LMB filter to that of the BP-LMB filter, P has to be chosen much larger than I. This implies a larger number of summation terms in the approximations of (5) and (6) employed by the Gibbs-LMB filter, and explains why the resulting runtime is higher than that of the proposed BP-LMB filter.

VIII. CONCLUSION

We presented a new derivation of the LMB filter via an approximation of the joint association distribution by the product of its marginals. This derivation enabled the use of a belief propagation algorithm for calculating approximations of the marginal association distributions and, in turn, led to a new fast LMB filter. The computational complexity of the proposed filter scales only linearly in the number of Bernoulli components and the number of measurements. Furthermore, our simulation results showed that the proposed filter outperforms the Gibbs sampling-based LMB filter in scenarios with a high clutter rate and a large number of (close) objects.

APPENDIX

We derive the BP-based probabilistic data association algorithm presented in Section VI-B, more specifically the message and belief expressions (28)–(31). The derivation is analogous to that in [15], where approximate marginal association probabilities are calculated for a slightly different association problem.

A. Message Expressions (28) and (29)

At message passing iteration $i \in \{1, \ldots, I\}$, first a message $\eta_k^{[i](a_k^{(l)}\to\Psi_{l,m})}(a_k^{(l)})$ is passed from each variable node "$a_k^{(l)}$" to the adjacent factor node "$\Psi_{l,m}(a_k^{(l)}, b_k^{(m)})$" in Fig. 2. Let us write $\nu_k^{[i-1](\Psi_{l,m}\to l)}(a_k^{(l)})$ for $\phi_k^{[i-1](\Psi_{l,m}\to a_k^{(l)})}(a_k^{(l)})$, i.e., for the message passed from factor node "$\Psi_{l,m}(a_k^{(l)}, b_k^{(m)})$" to the adjacent variable node "$a_k^{(l)}$" at message passing iteration $i-1$. According to (23), we obtain

$$\eta_k^{[i](a_k^{(l)}\to\Psi_{l,m})}(a_k^{(l)}) = \beta_k^{(l,a_k^{(l)})} \prod_{\substack{m'=1\\ m'\neq m}}^{M_k} \nu_k^{[i-1](\Psi_{l,m'}\to l)}(a_k^{(l)}). \quad (34)$$

Then, a message $\phi_k^{[i](\Psi_{l,m}\to b_k^{(m)})}(b_k^{(m)})$ is passed from each factor node "$\Psi_{l,m}(a_k^{(l)}, b_k^{(m)})$" to the adjacent variable node "$b_k^{(m)}$". Let us write $\zeta_k^{[i](\Psi_{l,m}\to m)}(b_k^{(m)})$ for $\phi_k^{[i](\Psi_{l,m}\to b_k^{(m)})}(b_k^{(m)})$. According to (22), this message is given by

$$\begin{aligned}\zeta_k^{[i](\Psi_{l,m}\to m)}(b_k^{(m)}) = &\sum_{a_k^{(l)}=-1}^{M_k} \Psi_{l,m}(a_k^{(l)}, b_k^{(m)})\\ &\times \eta_k^{[i](a_k^{(l)}\to\Psi_{l,m})}(a_k^{(l)}).\end{aligned} \quad (35)$$

Inserting (34) in (35) results in

$$\begin{aligned}\zeta_k^{[i](\Psi_{l,m}\to m)}(b_k^{(m)}) = &\sum_{a_k^{(l)}=-1}^{M_k} \beta_k^{(l,a_k^{(l)})} \Psi_{l,m}(a_k^{(l)}, b_k^{(m)})\\ &\times \prod_{\substack{m'=1\\ m'\neq m}}^{M_k} \nu_k^{[i-1](\Psi_{l,m'}\to l)}(a_k^{(l)})\end{aligned} \quad (36)$$

for $l \in \mathbb{L}_k^*$ and $m \in \{1, \ldots, M_k\}$. In a similar manner, we obtain the following expression of the message that is passed from factor node "$\Psi_{l,m}(a_k^{(l)}, b_k^{(m)})$" to the adjacent variable node "$a_k^{(l)}$":

$$\begin{aligned}\nu_k^{[i](\Psi_{l,m}\to l)}(a_k^{(l)}) = &\sum_{b_k^{(m)}\in\{0\}\cup\mathbb{L}_k^*} \Psi_{l,m}(a_k^{(l)}, b_k^{(m)})\\ &\times \prod_{l'\in\mathbb{L}_k^*\setminus\{l\}} \zeta_k^{[i](\Psi_{l',m}\to m)}(b_k^{(m)})\end{aligned} \quad (37)$$

for $l \in \mathbb{L}_k^*$ and $m \in \{1, \ldots, M_k\}$.

Still following [15], the vector-valued messages (36) and (37)—vector-valued in the sense that there is one message value for each value of $b_k^{(m)}$ or $a_k^{(l)}$—can be simplified to scalar ones. Because of the admissibility constraint expressed by $\Psi_{l,m}(a_k^{(l)}, b_k^{(m)})$, each message comprises actually only two different values. Indeed, we have

$$\zeta_k^{[i](\Psi_{l,m}\to m)}(b_k^{(m)}) = \begin{cases}\zeta_{k,l,m}^{[i]}, & b_k^{(m)} = l\\ \zeta_{k,l,m}^{[i]\prime}, & b_k^{(m)} \neq l\end{cases} \quad (38)$$

where

$$\zeta^{[i]}_{k,l,m} = \beta_k^{(l,m)} \prod_{\substack{m'=1\\ m'\neq m}}^{M_k} \nu_k^{[i-1](\Psi_{l,m'}\to l)}(m) \tag{39}$$

$$\zeta^{[i]\prime}_{k,l,m} = \sum_{\substack{a_k^{(l)}=-1\\ a_k^{(l)}\neq m}}^{M_k} \beta_k^{(l,a_k^{(l)})} \prod_{\substack{m'=1\\ m'\neq m}}^{M_k} \nu_k^{[i-1](\Psi_{l,m'}\to l)}(a_k^{(l)}) \tag{40}$$

and similarly

$$\nu_k^{[i](\Psi_{l,m}\to l)}(a_k^{(l)}) = \begin{cases} \nu^{[i]}_{k,l,m}, & a_k^{(l)} = m \\ \nu^{[i]\prime}_{k,l,m}, & a_k^{(l)} \neq m \end{cases} \tag{41}$$

where

$$\nu^{[i]}_{k,l,m} = \prod_{l'\in\mathbb{L}_k^*\setminus\{l\}} \zeta_k^{[i](\Psi_{l',m}\to m)}(l) \tag{42}$$

$$\nu^{[i]\prime}_{k,l,m} = \sum_{b_k^{(m)}\in(\{0\}\cup\mathbb{L}_k^*)\setminus\{l\}} \prod_{l'\in\mathbb{L}_k^*\setminus\{l\}} \zeta_k^{[i](\Psi_{l',m}\to m)}(b_k^{(m)}). \tag{43}$$

We next normalize the messages according to $\bar{\zeta}_k^{[i](\Psi_{l,m}\to m)}(b_k^{(m)}) \triangleq \zeta_k^{[i](\Psi_{l,m}\to m)}(b_k^{(m)})/\zeta^{[i]\prime}_{k,l,m}$ and $\bar{\nu}_k^{[i](\Psi_{l,m}\to l)}(a_k^{(l)}) \triangleq \nu_k^{[i](\Psi_{l,m}\to l)}(a_k^{(l)})/\nu^{[i]\prime}_{k,l,m}$, which yields

$$\bar{\zeta}_k^{[i](\Psi_{l,m}\to m)}(b_k^{(m)}) = \begin{cases} \zeta^{[i]}_{k,l,m}/\zeta^{[i]\prime}_{k,l,m}, & b_k^{(m)} = l \\ 1, & b_k^{(m)} \neq l \end{cases} \tag{44}$$

and

$$\bar{\nu}_k^{[i](\Psi_{l,m}\to l)}(a_k^{(l)}) = \begin{cases} \nu^{[i]}_{k,l,m}/\nu^{[i]\prime}_{k,l,m}, & a_k^{(l)} = m \\ 1, & a_k^{(l)} \neq m. \end{cases} \tag{45}$$

Let us consider $\bar{\zeta}_k^{[i](\Psi_{l,m}\to m)}(b_k^{(m)})$ for $b_k^{(m)} = l$. Inserting (39) and (40) into (44) yields

$$\bar{\zeta}_k^{[i](\Psi_{l,m}\to m)}(l) = \frac{\beta_k^{(l,m)} \prod_{\substack{m'=1\\ m'\neq m}}^{M_k} \nu_k^{[i-1](\Psi_{l,m'}\to l)}(m)}{\sum_{\substack{a_k^{(l)}=-1\\ a_k^{(l)}\neq m}}^{M_k} \beta_k^{(l,a_k^{(l)})} \prod_{\substack{m'=1\\ m'\neq m}}^{M_k} \nu_k^{[i-1](\Psi_{l,m'}\to l)}(a_k^{(l)})}. \tag{46}$$

Substituting $\nu_k^{[i-1](\Psi_{l,m}\to l)}(a_k^{(l)}) = \bar{\nu}_k^{[i-1](\Psi_{l,m}\to l)}(a_k^{(l)})\, \nu^{[i-1]\prime}_{k,l,m}$, expression (46) becomes

$$\bar{\zeta}_k^{[i](\Psi_{l,m}\to m)}(l) = \frac{\beta_k^{(l,m)} \prod_{\substack{m'=1\\ m'\neq m}}^{M_k} \bar{\nu}_k^{[i-1](\Psi_{l,m'}\to l)}(m)}{\sum_{\substack{a_k^{(l)}=-1\\ a_k^{(l)}\neq m}}^{M_k} \beta_k^{(l,a_k^{(l)})} \prod_{\substack{m'=1\\ m'\neq m}}^{M_k} \bar{\nu}_k^{[i-1](\Psi_{l,m'}\to l)}(a_k^{(l)})}. \tag{47}$$

Finally, using the fact that according to (45), $\bar{\nu}_k^{[i-1](\Psi_{l,m'}\to l)}(a_k^{(l)}) = 1$ for $a_k^{(l)} \neq m'$, expression (47) simplifies to

$$\bar{\zeta}_k^{[i](\Psi_{l,m}\to m)}(l) = \frac{\beta_k^{(l,m)}}{\beta_k^{(l,-1)} + \beta_k^{(l,0)} + \sum_{\substack{m'=1\\ m'\neq m}}^{M_k} \beta_k^{(l,m')} \bar{\nu}_k^{[i-1](\Psi_{l,m'}\to l)}(m')}. \tag{48}$$

We also recall from (44) that $\bar{\zeta}_k^{[i](\Psi_{l,m}\to m)}(b_k^{(m)}) = 1$ for $b_k^{(m)} \neq l$.

Analogously, by inserting (42) and (43) into (45), we obtain for $\bar{\nu}_k^{[i](\Psi_{l,m}\to l)}(a_k^{(l)})$ with $a_k^{(l)} = m$

$$\bar{\nu}_k^{[i](\Psi_{l,m}\to l)}(m) = \frac{1}{1 + \sum_{l'\in\mathbb{L}_k^*\setminus\{l\}} \bar{\zeta}_k^{[i](\Psi_{l',m}\to m)}(l')}. \tag{49}$$

Furthermore, $\bar{\nu}_k^{[i](\Psi_{l,m}\to l)}(a_k^{(l)}) = 1$ for $a_k^{(l)} \neq m$.

Finally, using the shorthands $\zeta_k^{[i](l\to m)} \triangleq \bar{\zeta}_k^{[i](\Psi_{l,m}\to m)}(l)$ and $\nu_k^{[i](m\to l)} \triangleq \bar{\nu}_k^{[i](\Psi_{l,m}\to l)}(m)$, the two equations (48) and (49) are seen to be equivalent to the recursion given by (28) and (29).

B. Belief Expressions (30) and (31)

According to the general rule described in Section VI-A, the belief $\tilde{p}(a_k^{(l)})$ is obtained after the final iteration $i = I$ by calculating the product of all the incoming messages at variable node "$a_k^{(l)}$" in Fig. 2 and normalizing the resulting function. Thus

$$\tilde{p}(a_k^{(l)} = m) \propto \beta_k^{(l,m)} \prod_{m'=1}^{M_k} \nu_k^{[I](\Psi_{l,m'}\to l)}(m) \tag{50}$$

for $m \in \{-1, 0, 1, \ldots, M_k\}$. Substituting $\nu_k^{[I](\Psi_{l,m'}\to l)}(a_k^{(l)}) = \bar{\nu}_k^{[I](\Psi_{l,m'}\to l)}(a_k^{(l)})\, \nu^{[I]\prime}_{k,l,m'}$, we obtain further

$$\tilde{p}(a_k^{(l)} = m) \propto \beta_k^{(l,m)} \left(\prod_{m''=1}^{M_k} \nu^{[I]\prime}_{k,l,m''} \right) \prod_{m'=1}^{M_k} \bar{\nu}_k^{[I](\Psi_{l,m'}\to l)}(m)$$

$$\propto \beta_k^{(l,m)} \prod_{m'=1}^{M_k} \bar{\nu}_k^{[I](\Psi_{l,m'}\to l)}(m). \tag{51}$$

Since by (45) $\bar{\nu}_k^{[I](\Psi_{l,m'}\to l)}(m) = 1$ for all $m' \neq m$, expression (51) simplifies to

$$\tilde{p}(a_k^{(l)} = m) \propto \begin{cases} \beta_k^{(l,m)}, & m \in \{-1, 0\} \\ \beta_k^{(l,m)} \bar{\nu}_k^{[I](\Psi_{l,m}\to l)}(m), & m \in \{1, \ldots, M_k\}. \end{cases}$$

These expressions still need to be normalized, which amounts to division by $\beta_k^{(l,-1)} + \beta_k^{(l,0)} + \sum_{m=1}^{M_k} \beta_k^{(l,m)} \bar{\nu}^{[I](\Psi_{l,m}\to l)}(m)$. Finally, using the shorthand $\nu_k^{[I](m\to l)} \triangleq \bar{\nu}_k^{[I](\Psi_{l,m}\to l)}(m)$, we obtain (30).

Similarly, the belief $\tilde{p}(b_k^{(m)})$ is obtained after the final iteration $i = I$ as the normalized product of all the incoming messages at variable node "$b_k^{(m)}$" in Fig. 2. A derivation analogous to the one above leads to (31).

THOMAS KROPFREITER

TU Wien, Vienna, Austria and Brno University of Technology, Brno, Czech Republic

FLORIAN MEYER
Massachusetts Institute of Technology, Cambridge, MA, USA

FRANZ HLAWATSCH
TU Wien, Vienna, Austria and Brno University of Technology, Brno, Czech Republic

REFERENCES

[1] Y. Bar-Shalom, P. K. Willett, and X. Tian
Tracking and Data Fusion: A Handbook of Algorithms. Storrs, CT, USA: Yaakov Bar-Shalom, 2011.

[2] S. Challa, M. R. Morelande, D. Musicki, and R. Evans
Fundamentals of Object Tracking. Cambridge, UK: Cambridge Univ. Press, 2011.

[3] R. P. S. Mahler
Advances in Statistical Multisource-Multitarget Information Fusion. Boston, MA, USA: Artech House, 2014.

[4] W. Koch
Tracking and Sensor Data Fusion: Methodological Framework and Selected Applications. Berlin, Germany: Springer, 2014.

[5] B.-T. Vo and B.-N. Vo
Labeled random finite sets and multi-object conjugate priors
IEEE Trans. Signal Process., vol. 61, no. 13, pp. 3460–3475, Jul. 2013.

[6] B.-N. Vo, B.-T. Vo, and D. Phung
Labeled random finite sets and the Bayes multi-target tracking filter
IEEE Trans. Signal Process., vol. 62, no. 24, pp. 6554–6567, Dec. 2014.

[7] B.-N. Vo, B.-T. Vo, and H. G. Hoang
An efficient implementation of the generalized labeled multi-Bernoulli filter
IEEE Trans. Signal Process., vol. 65, no. 8, pp. 1975–1987, Apr. 2017.

[8] Y. Punchihewa, B.-N. Vo, and B.-T. Vo
A generalized labeled multi-Bernoulli filter for maneuvering targets
In *Proc. 19th Int. Conf. Inf. Fusion*, Heidelberg, Germany, Jul. 2016, pp. 980–986.

[9] S. Reuter, B.-T. Vo, B.-N. Vo, and K. Dietmayer
The labeled multi-Bernoulli filter
IEEE Trans. Signal Process., vol. 62, no. 12, pp. 3246–3260, Jun. 2014.

[10] S. Reuter, A. Danzer, M. Stübler, A. Scheel, and K. Granström
A fast implementation of the labeled multi-Bernoulli filter using Gibbs sampling
In *Proc. IEEE Intell. Vehicles Symp.*, Los Angeles, CA, USA, Jun. 2017, pp. 765–772.

[11] M. Beard, S. Reuter, K. Granström, B.-T. Vo, B.-N. Vo, and A. Scheel
Multiple extended target tracking with labeled random finite sets
IEEE Trans. Signal Process., vol. 64, no. 7, pp. 1638–1653, Apr. 2016.

[12] A. Danzer, S. Reuter, and K. Dietmayer
The adaptive labeled multi-Bernoulli filter
in *Proc. 19th Int. Conf. Inf. Fusion*, Heidelberg, Germany, Jul. 2016, pp. 1531–1538.

[13] J. L. Williams
Marginal multi-Bernoulli filters: RFS derivation of MHT, JIPDA, and association-based MeMBer
IEEE Trans. Aerosp. Electron. Syst., vol. 51, no. 3, pp. 1664–1687, Jul. 2015.

[14] F. Meyer *et al.*
Message passing algorithms for scalable multitarget tracking
Proc. IEEE, vol. 106, no. 2, pp. 221–259, Feb. 2018.

[15] J. L. Williams and R. Lau
Approximate evaluation of marginal association probabilities with belief propagation
IEEE Trans. Aerosp. Electron. Syst., vol. 50, no. 4, pp. 2942–2959, Oct. 2014.

[16] T. Kropfreiter, F. Meyer, and F. Hlawatsch
A fast labeled multi-Bernoulli filter using belief propagation — Supplementary material
2019, [Online]. Available: https://github.com/ThoKro/BP-LMB.git

[17] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger
Factor graphs and the sum-product algorithm
IEEE Trans. Inf. Theory, vol. 47, no. 2, pp. 498–519, Feb. 2001.

[18] J. L. Williams and R. A. Lau
Data association by loopy belief propagation
In *Proc. 13th Int. Conf. Inf. Fusion*, Edinburgh, U.K., Jul. 2010.

[19] F. Meyer, P. Braca, P. Willett, and F. Hlawatsch
A scalable algorithm for tracking an unknown number of targets using multiple sensors
IEEE Trans. Signal Process., vol. 65, no. 13, pp. 3478–3493, Jul. 2017.

[20] Y. B. Shalom, X.-R. Li, and T. Kirubarajan
Estimation With Applications to Tracking and Navigation. New York, NY, USA: Wiley, 2002.

[21] T. Kropfreiter, F. Meyer, and F. Hlawatsch
Sequential Monte Carlo implementation of the track-oriented marginal multi-Bernoulli/Poisson filter
In *Proc. 19th Int. Conf. Inf. Fusion*, Heidelberg, Germany, Jul. 2016, pp. 972–979.

[22] D. Schuhmacher, B.-T. Vo, and B.-N. Vo
A consistent metric for performance evaluation of multi-object filters
IEEE Trans. Signal Process., vol. 56, no. 8, pp. 3447–3457, Aug. 2008.