Distributed Estimation with Information-Seeking Control in Agent Networks

Florian Meyer, Student Member, IEEE, Henk Wymeersch, Member, IEEE, Markus Fröhle, and Franz Hlawatsch, Fellow, IEEE

Abstract

We introduce a distributed, cooperative framework and method for Bayesian estimation and control in decentralized agent networks. Our framework combines joint estimation of time-varying global and local states with information-seeking control optimizing the behavior of the agents. It is suited to nonlinear and non-Gaussian problems and, in particular, to location-aware networks. For cooperative estimation, a combination of belief propagation message passing and likelihood consensus is used. For cooperative control, the negative posterior joint entropy of all states is maximized via a gradient ascent. The estimation layer provides the control layer with probabilistic information in the form of sample-based representations of probability distributions. Simulation results demonstrate intelligent behavior of the agents and excellent estimation performance for a simultaneous self-tracking and target tracking problem.

Index Terms

Agent networks, distributed estimation, distributed control, information-seeking control, distributed target tracking, cooperative localization, belief propagation, message passing, likelihood consensus, sequential estimation.

I. INTRODUCTION

Recent research on distributed estimation and control in mobile agent networks [1]–[4] has been motivated by location-aware applications including environmental and agricultural monitoring [5], health-care monitoring [6], target tracking [7], pollution source localization [8], chemical plume

F. Meyer and F. Hlawatsch are with the Institute of Telecommunications, Vienna University of Technology, Vienna, Austria (e-mail: {florian.meyer, franz.hlawatsch}@nt.tuwien.ac.at). H. Wymeersch and M. Fröhle are with the Department of Signals and Systems, Chalmers University of Technology, Gothenburg, Sweden (e-mail: {henkw, frohle}@chalmers.se). This work was supported by the Austrian Science Fund (FWF) under Grant S10603 within the National Research Network SISE and by the European Commission under ERC Grant No. 258418 (COOPNET), EU FP7 Marie Curie Initial Training Network MULTI-POS under Grant 316528, and the Newcom# Network of Excellence in Wireless Communications.
tracking [3], and surveillance [9]. The agents in a mobile agent network are generally equipped with sensors, wireless communication interfaces, a processing unit, and actuators, all together forming a cyber-physical system [10] with a tight coupling between sensing, computing, communication, and control. A common task in mobile agent networks is seeking information, either about external phenomena or about the network itself. In turn, this task relies on estimation (quantifying, fusing, and disseminating information) and control (configuring the network to increase information). These aspects have traditionally been treated separately, as reviewed below. A common theme in previous works is the reliance on position information for estimation and/or control.

Estimation methods for mobile agent networks (our focus will be on distributed Bayesian estimation) address estimation of common global states [7], [11], [12], estimation of local states [13]–[15], or combined estimation of local and global states [16], [17]. In the first case, the agents obtain local measurements with respect to external objects or the surrounding environment, which are fused across the network. Global fusion methods that require only local communication include consensus [18] and gossip [19]. Example applications are distributed target tracking [7], cooperative exploration of the environment [12], and chemical plume tracking [3]. In the second case (estimation of local states), the agents cooperate such that each agent is able to better estimate its own local state. Here, the dimensionality of the total state grows with the network size, which leads to more complex factorizations of the joint posterior probability density function (pdf). When the factor graph [20] of this factorization matches the network topology, efficient message passing methods for distributed inference can be used, such as the belief propagation (BP) [20] and mean field [21] methods. Example applications are cooperative localization [13], synchronization [14], and simultaneous localization and synchronization [15]. In the third case (estimation of both global states and local states), a message passing algorithm can be combined with a network-wide information dissemination technique. An example application is cooperative simultaneous self-localization and target tracking [16], [17].

Whereas the works cited above do not consider control tasks, combined estimation and control in mobile agent networks has been treated by the control community. We will limit our discussion to information-seeking control. Here, the agents’ control objective is to maximize the joint information of all agents about global and/or local states. Suitable measures of information include negative posterior entropy [22], mutual information [22], and scalar-valued functions of the Fisher information matrix [23]. In particular, the determinant, trace, and spectral norm of the Fisher information matrix were considered in [24], where the control objective is to maximize the information related to the positions of the agents and of a target. The maximization of negative posterior entropy was considered in [11], [12], and [25]. In [25], a central controller steers agents with known positions along the gradient of negative posterior entropy to optimally sense a global state. A distributed solution for global state estimation was proposed in [11] based on a pairwise neighboring-agents approximation of mutual
information and in [12] by using a consensus algorithm.

In this paper, we present a unified Bayesian framework and method for (i) distributed, cooperative joint estimation of time-varying global and local states and (ii) distributed, cooperative information-seeking control. Our framework and method are suited to nonlinear and non-Gaussian problems, and require only communication with neighboring agents. In addition, they are scalable and able to cope with a changing network topology. Thereby, they are particularly suited to localization and tracking tasks in location-aware scenarios involving mobile networks and nonlinear models.

For distributed estimation, following [17], we use a combination of BP message passing, average consensus, and sample-based representations of the involved probability distributions. For distributed control, we define a global (holistic) objective function as the negative posterior entropy of all states in the network at the next time step conditioned on all measurements at the next time step. This objective function is optimized jointly by all agents via a gradient ascent. This reduces to the evaluation of local gradients at each agent, which is performed by using Monte Carlo integration based on the sample representations provided by the estimation stage and a distributed calculation of the joint (networkwide) likelihood function. Our method extends [11] and [12] in the following respects: (i) it constitutes a more general information-maximizing control framework based on BP for sequential estimation problems where the states may be time-varying; (ii) it includes estimation of the own (controlled) states of the agents; and (iii) the factorization structure of the considered estimation problem is more complex and possibly loopy.

This paper is organized as follows. In Section II, the system model is described and the joint estimation and control problem is formulated. Section III reviews joint local and global state estimation [17]. In Section IV, we introduce the proposed gradient-based controller. The computation of the gradient is discussed in Section V. Section VI considers two special cases of the joint estimation and control framework. Finally, in Section VII, we present simulation results demonstrating the performance of our method for a simultaneous self-tracking and target tracking problem.

II. SYSTEM MODEL AND PROBLEM FORMULATION

We consider a network of mobile agents $k \in \mathcal{A}$ as shown in Fig. I. The set of all agents, $\mathcal{A}$, consists of the set of cooperative agents (CAs), $\mathcal{C} \subseteq \mathcal{A}$, and the set of “targets,” $\mathcal{T} = \mathcal{A} \setminus \mathcal{C}$. Here, a target may be anything that does not cooperate and cannot be controlled, such as a noncooperative agent or a relevant feature of the environment. We will typically use the indices $k \in \mathcal{A}$, $l \in \mathcal{C}$, and $m \in \mathcal{T}$ to denote a generic agent, a CA, and a target, respectively. A block diagram of the overall “signal processing system” is shown in Fig. II for a CA (including the estimation and control layers of the proposed method) and for a target. This system is described below.
1) Agent States and Agent Dynamics: The state of agent $k \in A$ at discrete time $n \in \{0, 1, \ldots\}$ is denoted by the vector $x_k^{(n)}$. For example, in a localization scenario, $x_k^{(n)}$ may consist of the current position and motion-related quantities such as velocity, acceleration, and angular velocity [26]. The states evolve according to

$$x_k^{(n+1)} = \begin{cases} g_k(x_k^{(n-1)}, u_k^{(n)}, q_k^{(n)}), & k \in \mathcal{C} \\ g_l(x_l^{(n-1)}, q_l^{(n)}), & k \in \mathcal{T} \end{cases}$$  

where $g_k(\cdot)$ is a possibly nonlinear function, $q_k^{(n)}$ is process (driving) noise, and $u_k^{(n)} \in U_k$ is a deterministic control vector with domain $U_k$ that controls the $k$th CA. (Note that also targets may have control variables. However, as these are hidden from the CAs, we will subsume any control for target $m$ in the noise $q_m^{(n)}$.) For the derivation of the controller, we assume that for $l \in \mathcal{C}$, $g_l(x_l^{(n-1)}, u_l^{(n)}, q_l^{(n)})$ is bijective with respect to $x_l^{(n-1)}$ and differentiable with respect to $u_l^{(n)}$. The
statistical relation between $x_k^{(n-1)}$ and $x_k^{(n)}$ as defined by (1) can also be described by the state-transition pdf $f(x_k^{(n)} | x_l^{(n-1)}; u_l^{(n)})$ for $l \in C$ and $f(x_m^{(n)} | x_m^{(n-1)})$ for $m \in T$.

2) Network Topology: The communication and measurement topology of the network is described by the neighborhood sets $C_l^{(n)}$, $T_l^{(n)}$, and $A_l^{(n)}$ as follows. CA $l$ is able to receive data via a communication link from CA $l'$ if $l' \in C_l^{(n)} \subseteq C \setminus \{l\}$. In addition, CA $l$ is able to acquire a measurement $y_{l,l'}^{(n)}$ relative to CA $l'$ if $l' \in C_l^{(n)}$. This relation is symmetric, i.e., $l' \in C_l^{(n)}$ implies $l \in C_{l'}^{(n)}$. Furthermore, CA $l \in C$ acquires a measurement $y_{l,m}^{(n)}$ relative to target $m$ if $m \in T_l^{(n)} \subseteq T$. The targets are noncooperative in that they do not communicate and do not acquire any measurements. We also define $A_l^{(n)} \triangleq C_l^{(n)} \cup T_l^{(n)}$. Finally, the set $C_l^{(n)}$ contains all CAs measuring target $m$, i.e., all $l \in C$ with $m \in T_l^{(n)}$. The sets $C_l^{(n)}$, $T_l^{(n)}$, $A_l^{(n)}$, and $C_m^{(n)}$ are generally time-dependent. An example of a communication and measurement topology is given in Fig. 1. It is assumed that CA $l$ knows the state evolution models $g_k(\cdot)$, the pdfs of the process noises $f(q_k^{(n)})$, and the initial prior pdfs of the agent states $f(x_k^{(0)})$ for $k \in \{l\} \cup C_l \cup T$.

3) Sensor Measurements: We consider “pairwise” measurements $y_{l,k}^{(n)}$ that depend on the state $x_l^{(n)}$ of a measuring CA $l \in C$ and the state $x_k^{(n)}$ of a measured agent (CA or target) $k \in A_l^{(n)}$ according to

$$y_{l,k}^{(n)} = d_l(x_l^{(n)}, x_k^{(n)}, v_{l,k}^{(n)}), \quad l \in C, \; k \in A_l^{(n)},$$

where $d_l(\cdot)$ is a possibly nonlinear function and $v_{l,k}^{(n)}$ is measurement noise. Both $d_l(\cdot)$ and the pdfs $f(v_{l,k}^{(n)})$, $k \in A_l^{(n)}$ are assumed to be known to CA $l$. An example is the scalar measurement

$$y_{l,k}^{(n)} = \|x_l^{(n)} - x_k^{(n)}\| + v_{l,k}^{(n)},$$

where $x_k^{(n)}$ equals the position of agent $k$ and, hence, $\|x_l^{(n)} - x_k^{(n)}\|$ is the spatial distance between agents $l$ and $k$. The statistical relation between the measurement $y_{l,k}^{(n)}$ and the involved states $x_l^{(n)}$ and $x_k^{(n)}$ is also described by the local likelihood function $f(y_{l,k}^{(n)} | x_l^{(n)}, x_k^{(n)})$. For the derivation of the controller, $f(y_{l,k}^{(n)} | x_l^{(n)}, x_k^{(n)})$ is assumed differentiable with respect to $x_l^{(n)}$.

4) Problem Formulation: The following two tasks are to be performed at each time $n$:

1) Each CA $l \in C$ estimates the states $x_k^{(n)}$, $k \in \{l\} \cup T$ (i.e., its own local state and the states of all targets) from prior information and all past and present measurements in the network.

The assumption that CAs acquire measurements with respect to each other if and only if they are able to communicate is introduced to simplify the notation. The methods proposed in this paper can be generalized in a straightforward manner to scenarios where the measurement neighborhoods are subsets of the communication neighborhoods [12].

The proposed framework can be easily extended to self-measurements (measurements that involve only the own state) and cluster measurements (measurements that involve the states of several other agents).
2) Each CA is controlled such that the negative posterior entropy of all states in the network at the next time, conditioned on all measurements in the network at the next time, is maximized.

We solve these two problems in a distributed and recursive manner. Our method consists of an estimation layer and a control layer, as shown in Fig. 2(a). In the estimation layer, CA $l$ computes an approximation of the marginal posterior pdfs of the states $x_k^{(n)}$, $k \in \{l\} \cup T$ given all the measurements and control vectors in the entire network, $y^{(1:n)}$ and $u^{(1:n)}$. In the control layer, CA $l$ uses these marginal posteriors and the statistical model to determine an optimal control variable $u_l^{(n+1)}$. In both layers, the CAs communicate with neighboring CAs.

III. Estimation Layer

The estimation layer performs distributed estimation of the local and global states by using the BP- and consensus-based method introduced in [17]. We will review this method in our present context. Let us denote by $x^{(n)} \triangleq [x_k^{(n)}]_{k \in A}$, $u^{(n)} \triangleq [u_l^{(n)}]_{l \in C}$, and $y^{(n)} \triangleq [y_{l,k}^{(n)}]_{l \in C, k \in A^{(n)}}$ the stacked vectors of, respectively, all states, control variables, and measurements at time $n$. Furthermore, let $x^{(1:n)} \triangleq [x^{(1)}; \ldots; x^{(n)}]^T$, $u^{(1:n)} \triangleq [u^{(1)}; \ldots; u^{(n)}]^T$, and $y^{(1:n)} \triangleq [y^{(1)}; \ldots; y^{(n)}]^T$. Each CA $l \in C$ estimates its local state $x_l^{(n)}$ and all the target states $x_m^{(n)}$, $m \in T$ from the measurements of all CAs up to time $n$, $y^{(1:n)}$. This estimation is based on the posteriors $f(x_l^{(n)}|y^{(1:n)}; u^{(1:n)})$, $k \in \{l\} \cup T$, which are marginals of the joint posterior $f(x^{(1:n)}|y^{(1:n)}; u^{(1:n)})$.

Using Bayes’ rule and common assumptions [13], the joint posterior can be factorized as

$$f(x^{(1:n)}|y^{(1:n)}; u^{(1:n)}) \propto \prod_{k \in A} \prod_{n=1}^N f(x_k^{(n)}) x_i^{(n-1)}|u_i^{(n')}) \times \prod_{m \in T} f(y_{m,k}^{(n')}|x_i^{(n')}, x_k^{(n')}) \prod_{m \in T} f(x_m^{(n')}|x_m^{(n-1)})$$

The marginal posterior of state $x_k^{(n)}$ is then given by

$$f(x_k^{(n)}|y^{(1:n)}; u^{(1:n)}) = \int f(x^{(1:n)}|y^{(1:n)}; u^{(1:n)}) \, dx_{\neq k,n}^{(1:n)}$$

where $x_{\neq k,n}^{(1:n)}$ denotes $x^{(1:n)}$ with $x_k^{(n)}$ removed. Using $f(x_k^{(n)}|y^{(1:n)}; u^{(1:n)})$, the minimum mean-square error (MMSE) estimator [23] of $x_k^{(n)}$ is obtained as

$$x_{k,\text{MMSE}}^{(n)} \triangleq \int x_k^{(n)} f(x_k^{(n)}|y^{(1:n)}; u^{(1:n)}) \, dx_k^{(n)}, \quad k \in A$$

A. Sequential Estimation

For a review of the sequential calculation of [4] proposed in [17], we now switch to the following simplified notation for the sake of readability. In the conditions of the various conditional pdfs, we omit the measurements up to time $n-1$, i.e., $y^{(1:n-1)}$, and the control vectors up to time $n$, i.e., $u^{(1:n)}$, because $y^{(1:n-1)}$ has already been observed and $u^{(1:n)}$ has already been determined; hence
sage passing and the likelihood consensus scheme [17]. This algorithm computes an approximation of (7) is provided by a distributed, cooperative algorithm that combines BP message passing and the inherent complexity of the marginalization process. A computationally feasible approach to (8) is given by [17, eq. (21)].

For sequential calculation of (6), CA $l \in \mathcal{C}$ employs a basic Bayesian recursive filtering method [13] consisting of a prediction step and a correction step. In the prediction step, CA $l \in \mathcal{C}$ computes a predictive posterior of its current state,

$$f(x_l) = \int f(x_l | x_l^-) f(x_l^-) \, dx_l^-.$$  

(5)

Here, $f(x_l)$ and $f(x_l^-)$ are short for $f(x_l^{(n)} | y^{(1:n-1)}; u^{(1:n)})$ and $f(x_l^{(n-1)} | y^{(1:n-1)}; u^{(1:n-1)})$, respectively. Furthermore, CA $l$ computes predictive posteriors of the target states

$$f(x_m) = \int f(x_m | x_m^-) f(x_m^-) \, dx_m^-,$$  

(6)

$m \in \mathcal{T}$. In the correction step, the CAs determine the marginal posteriors $f(x_l | y)$ and $f(x_m | y)$, $m \in \mathcal{T}$, which are given by

$$f(x_l | y) \propto \int \prod_{k' \in A} f(x_{k'}) \prod_{l' \in \mathcal{C}} f(y_{l',k} | x_{l'}; x_{k'}) \, dx_{\sim k}, \quad k \in \{l\} \cup \mathcal{T}.$$  

(7)

Here, $x_{\sim k}$ denotes $x$ with $x_k$ removed and $f(x_l | y)$ is short for $f(x_l^{(n)} | y^{(1:n)}; u^{(1:n)})$. As shown in Fig. 2, these marginal posteriors are handed over to the control layer, which determines the control input for the next time step, $u_l^+$.  

B. BP Message Passing and Likelihood Consensus

Calculation of $f(x_l | y)$ according to (7) is generally intractable, due to the reliance on nonlocal information and the inherent complexity of the marginalization process. A computationally feasible approximation of (7) is provided by a distributed, cooperative algorithm that combines BP message passing and the likelihood consensus scheme [17]. This algorithm computes an approximation (“belief”) $b(x_l) \approx f(x_l | y)$ using only communication with neighboring CAs $l' \in \mathcal{C}_l$, and with a complexity that scales only linearly with the number of states in the network, $|A|$.  

The belief of local state $x_l$ at message passing iteration $p \in \{1, \ldots, P\}$ is given by [17, eq. (21)]

$$b^{(p)}(x_l) \propto f(x_l) \prod_{l' \in \mathcal{C}_l} \int f(y_{l',l} | x_l, x_{l'}) b^{(p-1)}(x_{l'}) \, dx_{l'} \prod_{m \in \mathcal{T}} \int f(y_{l,m} | x_l, x_m) \psi_{m \rightarrow l}^{(p-1)}(x_m) \, dx_m,$$  

(8)

which is initialized as $b^{(0)}(x_l) = f(x_l)$. Similarly, the belief of target state $x_m$ at message passing iteration $p$ is given by [17, eq. (22)]

$$b^{(p)}(x_m) \propto f(x_m) \prod_{l \in \mathcal{C}_m} \int f(y_{l,m} | x_l, x_m) \psi_{l \rightarrow m}^{(p-1)}(x_l) \, dx_l,$$  

(9)

with initialization $b^{(0)}(x_m) = f(x_m)$. Here, $\psi_{m \rightarrow l}^{(p-1)}(x_m)$ and $\psi_{l \rightarrow m}^{(p-1)}(x_l)$ are the “extrinsic informations” from target $m$ to CA $l$ and from CA $l$ to target $m$, respectively, at the previous message passing iteration $p-1$. These extrinsic informations are calculated recursively as [17, eqs. (26) and (27)]
\[
\psi^{(p)}_{m \to l}(\mathbf{x}_m) = \frac{b^{(p)}(\mathbf{x}_m)}{\int f(\mathbf{y}_{l,m} | \mathbf{x}_l, \mathbf{x}_m) \psi^{(p-1)}_{l \to m}(\mathbf{x}_l) \, d\mathbf{x}_l}
\] (10)

\[
\psi^{(p)}_{l \to m}(\mathbf{x}_l) = \frac{b^{(p)}(\mathbf{x}_l)}{\int f(\mathbf{y}_{l,m} | \mathbf{x}_l, \mathbf{x}_m) \psi^{(p-1)}_{m \to l}(\mathbf{x}_m) \, d\mathbf{x}_m}
\] (11)

with initialization \(\psi^{(0)}_{m \to l}(\mathbf{x}_m) = f(\mathbf{x}_m)\) and \(\psi^{(0)}_{l \to m}(\mathbf{x}_l) = f(\mathbf{x}_l)\), respectively.

A sample-based distributed implementation of (9)–(11) has been proposed in [17]. A problem involved in (9) are not available at the CAs. However, as further described in [17], the likelihood consensus scheme [27] can be employed to provide an approximation of the functional form of these products (as a function of \(\mathbf{x}_m\)) to each CA using only local communication. For the calculations in the control layer—to be described in Sections IV and V—all CAs require a common set of measurements in the network at the next time \(n+1\) involved (10) are not yet known. Thus, with an abuse of notation, (9) is redefined as

\[
\mathbf{y}^{(n+1)} \triangleq \{ \mathbf{y}^{(n+1)}_{l,k} \}_{l \in \mathcal{C}, k \in \mathcal{A}^{(n+1)}_l}.
\] (12)

In the proposed control approach, each CA \(l \in \mathcal{C}\) determines its next control variable \(\mathbf{u}^{(n+1)}_l\) such that information about the next joint state \(\mathbf{x}^{(n+1)}\) given \(\mathbf{y}^{(1:n+1)}\) is maximized. We quantify this information by the negative conditional differential entropy [22, Chapter 8] of \(\mathbf{x}^{(n+1)}\) given \(\mathbf{y}^{(n+1)}\), with \(\mathbf{y}^{(1:n)}\) being an additional condition that has been observed previously and is thus fixed:

\[
-h(\mathbf{x}^{(n+1)} | \mathbf{y}^{(n+1)}, \mathbf{y}^{(1:n)}, \mathbf{u}^{(n+1)}) = \int \int f(\mathbf{x}^{(n+1)}, \mathbf{y}^{(n+1)} | \mathbf{y}^{(1:n)}, \mathbf{u}^{(n+1)}) \times \log f(\mathbf{x}^{(n+1)}, \mathbf{y}^{(n+1)}, \mathbf{y}^{(1:n)}, \mathbf{u}^{(n+1)}) \, d\mathbf{x}^{(n+1)} \, d\mathbf{y}^{(n+1)},
\] (13)
where \( \log \) denotes the natural logarithm. Note that \( h(x^{(n+1)}|y^{(n+1)}; y^{(1:n)}, u^{(n+1)}) \) depends on the random vectors \( x^{(n+1)} \) and \( y^{(n+1)} \), i.e., on their joint distribution but not on their values. Our notation indicates this fact by using a sans serif font for \( x^{(n+1)} \) and \( y^{(n+1)} \) in \( h(x^{(n+1)}|y^{(n+1)}; y^{(1:n)}, u^{(n+1)}) \).

According to expression (13), \(-h(x^{(n+1)}|y^{(n+1)}; y^{(1:n)}, u^{(n+1)})\) is a function of the control vector \( u^{(n+1)} \). This function will be denoted as \( D_h(u^{(n+1)}) \), i.e.,

\[
D_h(u^{(n+1)}) \triangleq -h(x^{(n+1)}|y^{(n+1)}; y^{(1:n)}, u^{(n+1)}),
\]

and it will be used as the objective function for control at each CA. This objective function is holistic in that it involves all the next states (of both the CAs and the targets), \( x^{(n+1)} \), and all the next measurements, \( y^{(n+1)} \). Instead of a full-blown maximization of \( D_h(u^{(n+1)}) \), we perform one step of a gradient ascent [30] at each time \( n \). Thus, \( u^{(n+1)} \) is determined as

\[
\hat{u}^{(n+1)} = u_r^{(n+1)} + c^{(n+1)} \nabla D_h(u^{(n+1)}),
\]

where \( u_r^{(n+1)} \) is a reference vector and \( c^{(n+1)} > 0 \) is a step size. The choice of \( u_r^{(n+1)} \) depends on the manner in which the local control vectors \( u_l^{(n+1)} \) (which are subvectors of \( u^{(n)} \)) enter into the state evolution functions \( g_l(x_l^{(n-1)}, u_l^{(n)}, q_l^{(n)}) \) in (1); two common choices are \( u_l^{(n+1)} = u_l^{(n)} \) and \( u_l^{(n+1)} = 0 \) (cf. Section VII-A).

Since \( u^{(n+1)} = [u_l^{(n+1)}]_{l \in C} \), we have \( \nabla D_h(u^{(n+1)}) = \left[ \frac{\partial D_h(u^{(n+1)})}{\partial u_l^{(n+1)}} \right]_{l \in C} \), and thus the gradient ascent (15) with respect to \( u^{(n+1)} \) is equivalent to local gradient ascents at the individual CAs \( l \) with respect to the local control vectors \( u_l^{(n+1)} \). At CA \( l \), the local gradient ascent is performed as

\[
\hat{u}_l^{(n+1)} = u_r,l^{(n+1)} + c_l^{(n+1)} \frac{\partial D_h(u^{(n+1)})}{\partial u_l^{(n+1)}} |_{u_l^{(n+1)}=u_l^{(n+1)}},
\]

where \( u_r,l^{(n+1)} \) is the part of \( u_r^{(n+1)} \) that corresponds to CA \( l \) (we have \( u_r^{(n+1)} = [u_r,l^{(n+1)}]_{l \in C} \)). The local step sizes \( c_l^{(n+1)} \) are constrained by the condition \( u_l^{(n+1)} \in U_l \) for given domains \( U_l \). In practice, this condition can be easily satisfied by an appropriate scaling of the \( c_l^{(n+1)} \). Note that, as in (12), we use different local step sizes \( c_l^{(n+1)} \) at the individual CAs \( l \). This heuristic modification is made to account for the possibly different domains \( U_l \) and to avoid the necessity of reaching a consensus on a common step size across all the CAs; it was observed to yield good results.

### B. Expansion of the Objective Function

A central contribution of this paper is a distributed sample-based technique for calculating the gradients \( \frac{\partial D_h(u^{(n+1)})}{\partial u_l^{(n+1)}} |_{u_l^{(n+1)}=u_l^{(n+1)}} \) in (16). As a starting point for developing this technique, we next derive an expansion of the objective function \( D_h(u^{(n+1)}) \). We will use the following simplified notation. We do not indicate the conditioning on \( y^{(1:n)} \) and \( u^{(1:n)} \) because at time \( n + 1 \), \( y^{(1:n)} \) has already been observed and \( u^{(1:n)} \) has already been determined; hence both are fixed. (Note that in
Section III we did not indicate the conditioning on \( y^{(1:n-1)} \), rather than \( y^{(1:n)} \). Also, we suppress the time index \( n \) and designate variables at time \( n+1 \) by the superscript “+”. For example, we write \( h(x^+; y^+; u^+) \) instead of \( h(x^{(n+1)}|y^{(n+1)}; y^{(1:n)}; u^{(1:n+1)}) \).

For calculating the gradient, following [11] and [12], we disregard the unknown driving noise \( q \) in (1) for \( k = l \in \mathcal{C} \) by formally replacing it with its expectation, \( \hat{q} \triangleq \int q f(q) \, dq. \) We can then rewrite (1) for \( k = l \in \mathcal{C} \) (with \( n \) replaced by \( n+1 \)) as

\[
x^+_i = g_t(x_i, u^+_i, q^+_i) = \hat{g}_t(x_i, u^+_i), \quad l \in \mathcal{C}.
\]

(17)

As shown in Appendix A, the conditional differential entropy in (14) can be expressed as

\[
h(x^+|y^+; u^+) = h(x_C, x^+_T|y^+; u^+) + \sum_{l \in \mathcal{C}} G_l(u^+_l),
\]

where \( x_C \triangleq [x_l]_{l \in \mathcal{C}}, x^+_T \triangleq [x^+_m]_{m \in \mathcal{T}}, \) and

\[
G_l(u^+_l) \triangleq \int f(x_l) \log |J_{\hat{g}_l}(x_l; u^+_l)| \, dx_l, \quad \text{with} \quad J_{\hat{g}_l}(x_l; u^+_l) \triangleq \det \frac{\partial \hat{g}_l(x_l, u^+_l)}{\partial x_l}.
\]

(19)

The first term on the right-hand side of (18) can be decomposed as [22] Chapter 8

\[
h(x_C, x^+_T|y^+; u^+) = h(x_C, x^+_T) - I(x_C, x^+_T; y^+; u^+).
\]

(20)

Here, \( I(x_C, x^+_T; y^+; u^+) \) denotes the mutual information between \( (x_C, x^+_T) \) and \( y^+ \) (with \( u^+ \) being a deterministic parameter), which is given by [22] Chapter 8

\[
I(x_C, x^+_T; y^+; u^+) = \int \int f(x_C, x^+_T, y^+; u^+) \log \frac{f(x_C, x^+_T, y^+; u^+)}{f(x_C, x^+_T)f(y^+; u^+)} \, dx_C \, dx_T \, dy^+.
\]

(21)

Note that \( h(x_C, x^+_T) \) in (20) does not depend on \( u^+ \), since neither the CA states \( x_C \) nor the future target states \( x^+_T \) are controlled by the future control variable \( u^+ \). We explicitly express the dependence of \( I(x_C, x^+_T; y^+; u^+) \) on \( u^+ \) by defining the function

\[
D_l(u^+) \triangleq I(x_C, x^+_T; y^+; u^+).
\]

(22)

Combining (14), (18), (20), and (22) then yields the following expansion of the objective function:

\[
D_h(u^+) = -h(x_C, x^+_T) + D_l(u^+) - \sum_{l' \in \mathcal{C} \setminus \{l\}} G_{l'}(u^+_l).
\]

(23)

This entails the following expansion of the gradient in (16):

\[
\frac{\partial D_h(u^+)}{\partial u^+_l} = \frac{\partial D_l(u^+)}{\partial u^+_l} - \frac{\partial G_l(u^+_l)}{\partial u^+_l}.
\]

(24)

In the next section, we will develop sample-based techniques for calculating \( \frac{\partial D_l(u^+)}{\partial u^+_l} \bigg|_{u^+=u^+_l} \) and \( \frac{\partial G_l(u^+_l)}{\partial u^+_l} \bigg|_{u^+=u^+_l} \). The calculation of \( \frac{\partial D_l(u^+)}{\partial u^+_l} \bigg|_{u^+=u^+_l} \) is cooperative and distributed; it requires communication with neighboring CAs \( l' \in \mathcal{C} \). The calculation of \( \frac{\partial G_l(u^+_l)}{\partial u^+_l} \bigg|_{u^+=u^+_l} \) is noncooperative and performed locally at each CA \( l \). Both calculations use the samples of relevant marginal posteriors that were computed by the estimation layer.
V. Calculation of the Gradients

A. Gradient of $D_I(u^+)$

The mutual information in (21) can be rewritten as

$$D_I(u^+) = \int \int f(y^+ | x^c, x^+_T; u^+) f(x^c, x^+_T) \log \frac{f(y^+ | x^c, x^+_T; u^+)}{f(y^+; u^+)} \ dx^c \ dx^+_T \ dy^+.$$  

Invoking [12] Th. 1, we obtain

$$\frac{\partial D_I(u^+)}{\partial u_l^+} = \int \int \frac{\partial f(y^+ | x^c, x^+_T; u^+)}{\partial u_l^+} f(x^c, x^+_T) \log \frac{f(y^+ | x^c, x^+_T; u^+)}{f(y^+; u^+)} \ dx^c \ dx^+_T \ dy^+. \quad (25)$$

Next, we will develop a Monte Carlo approximation of $\frac{\partial D_I(u^+)}{\partial u_l^+} \bigg|_{u^+=u_l^+}$ that uses importance sampling and is based on a factorization of the likelihood function $f(y^+ | x^c, x^+_T; u^+)$. Subsequently, we will address the distributed computation of this approximation.

1) Factorization of the Likelihood Function for $CA \ l$. The likelihood function $f(y^+ | x^c, x^+_T; u^+)$ involved in (25) can be written as

$$f(y^+ | x^c, x^+_T; u^+) = \prod_{l \in \mathcal{C}} \prod_{l' \in \mathcal{C}} f(y^+_{l,l'} | x_l, x_{l'}; u_l^+, u_{l'}^+) \prod_{m \in \mathcal{T}_l} f(y^+_{l,m} | x_l, x_m^+; u_l^+). \quad (26)$$

Using (17), the local likelihood functions involved in (26) are expressed as

$$f(y^+_{l,l'} | x_l, x_{l'}; u_l^+, u_{l'}^+) = f(y^+_{l,l'} | x_l^+, x_{l'}^+) \bigg|_{x_l^\prime = \hat{g}_l(x_l, u_l^+), x_{l'}^\prime = \hat{g}_{l'}(x_{l'}, u_{l'}^+)}, \quad l \in \mathcal{C}, \ l' \in \mathcal{C}_l$$

$$f(y^+_{l,m} | x_l, x_m^+; u_l^+) = f(y^+_{l,m} | x_l^+, x_m^+) \bigg|_{x_m^\prime = \hat{g}_m(x_l, u_l^+)}, \quad l \in \mathcal{C}, \ m \in \mathcal{T}_l.$$  

We can rewrite (26) for an arbitrary $l \in \mathcal{C}$ as

$$f(y^+ | x^c, x^+_T; u^+) = \alpha_l(y^+, x^c, x^+_T, u^+) \beta_l(y^+, x^c, x^+_T, u^+), \quad (27)$$

with

$$\alpha_l(y^+, x^c, x^+_T, u^+) \triangleq \prod_{l' \in \mathcal{C}_l} f(y^+_{l,l'} | x_l, x_{l'}; u_l^+, u_{l'}^+) \prod_{m \in \mathcal{T}_l} f(y^+_{l,m} | x_l, x_m^+; u_l^+), \quad (28)$$

$$\beta_l(y^+, x^c, x^+_T, u^+) \triangleq \prod_{l' \in \mathcal{C} \setminus \{l\}} \prod_{l' \in \mathcal{C}_l} f(y^+_{l,l'} | x_l, x_{l'}; u_l^+, u_{l'}^+) \prod_{m' \in \mathcal{T}_{l'}} f(y^+_{l,m'} | x_l, x_{m'}^+; u_l^+). \quad (29)$$

Here, $\alpha_l(y^+, x^c, x^+_T, u^+)$ depends on the local control vector $u_l^+$ whereas $\beta_l(y^+, x^c, x^+_T, u^+)$ does not.

2) Monte Carlo Approximation: In Appendix [3] it is shown that a Monte Carlo (i.e., sample-based) approximation of (25) evaluated at $u^+=u_l^+$ is given by

$$\frac{\partial D_I(u^+)}{\partial u_l^+} \bigg|_{u^+=u_l^+} \approx \frac{1}{JJ'} \sum_{j=1}^J \sum_{j'=1}^{J'} \frac{1}{\alpha_l(y^{+(j,j')}, x^{+(j,j')}_c, x^{+(j,j')}_T, u_l^+)} \frac{\partial \alpha_l(y^{+(j,j')}, x^{+(j,j')}_c, x^{+(j,j')}_T, u_l^+)}{\partial u_l^+} \bigg|_{u^+=u_l^+}$$

$$\times \log \frac{\alpha_l(y^{+(j,j')}, x^{+(j,j')}_c, x^{+(j,j')}_T, u_l^+) \beta_l(y^{+(j,j')}, x^{+(j,j')}_c, x^{+(j,j')}_T, u_l^+)}{f(y^{+(j,j')}; u^+=u_l^+)}, \quad (30)$$

August 19, 2014 DRAFT
with
\[
f(y^{(j,j')}; u^+ = u_j^+) \approx \frac{1}{J} \sum_{j' = 1}^{J} \alpha_j(y^{(j,j')}, x_c^{(j')}, x_T^{(j')}, u^+) \beta_j(y^{(j,j')}, x_c^{(j')}, x_T^{(j')}, u^+). \tag{31}
\]
Here, \(y^{(j,j')}, x_c^{(j')}\), and \(x_T^{(j')}\) are samples of \(y^+, x_c^+\), and \(x_T^+\), respectively that are drawn from the importance density \(q(y^+, x_c^+, x_T^+; u^+ = u_j^+) \triangleq f(x_c^+, x_T^+; f(y^+, x_c^+, x_T^+; u^+ = u_j^+) via the following two-stage procedure. First, samples \(\{ (x_c^{(j)}, x_T^{(j)}) \}_{j=1}^{J} \) are drawn from
\[
f(x_c^+, x_T^+) = \prod_{l' \in C} f(x_{l'}^+) \prod_{m \in T} f(x_m^+). \tag{32}
\]
(This factorization expresses the conditional statistical independence of the \(x_{l'}, l' \in C\) and the \(x_m^+, m \in T\) given \(y^{(1:n)}\). This is a common approximation, which was introduced in \([13]\) and is also used in the estimation layer \([17]\).) Then, for each sample \( (x_c^{(j)}, x_T^{(j)}) \), samples \(\{ y^{(j,j')} \}_{j' = 1}^{J'}\) are drawn from the conditional pdf \(f(y^+ | x_c^{(j)}, x_T^{(j)}; u^+ = u_j^+) = \alpha_j(y^+, x_c^{(j)}, x_T^{(j)}, u^+) \beta_j(y^+, x_c^{(j)}, x_T^{(j)}, u^+)\) (cf. \(27\)). The distributed calculation of these samples will be discussed in Sections V-A3 and V-A4 and in Appendices C and D. Finally, we note that using \(28\), one easily obtains a (rather unwieldy) expression of the derivative \(\frac{\partial \ln f(y^+, x_c^+, x_T^+, u^+)}{\partial x_i} \) occurring in \(30\). This expression involves the factors in \(28\) and the derivatives \(\frac{\partial f(y^+, x_c^+, x_T^+, u^+)}{\partial x_i} \) for \(l' \in C, m \in T, l \).)

3) Quasi-centralized Processing: For a distributed computation of \(30\) and \(31\), we will present two alternative approaches. First, a simple “quasi-centralized” approach is as follows. As a result of the estimation layer, samples \(\{x_k^{(j)}\}_{j=1}^{J} \sim f(x_k)\), \(k \in \{l\} \cup T\) are available at CA \(l\) (see \(7\)), noting that \(f(x_k)\) was denoted \(f(x_k | y)\) there. A flooding algorithm \([32]\) is now used to make available to each CA \(l\) the reference vectors \(u_{l,t}^+\) and the samples \(\{x_{l'}^{(j)}\}_{j = 1}^{J} \sim f(x_{l'})\) of all the other CAs \(l' \in C \setminus \{l\}\). In addition, CA \(l\) locally calculates predictive marginal posteriors for all target states via the following prediction step (which is \(6\) with \(n\) replaced by \(n + 1\)):
\[
f(x_m^+) = \int f(x_m^+ | x_m) f(x_m) \, dx_m, \quad m \in T. \tag{33}
\]
A sample-based implementation of \(33\) using the samples \(\{x_m^{(j)}\}_{j=1}^{J}, m \in T\) produced by the estimation layer (which are available at CA \(l\) and yielding samples \(\{x_m^{+} (j)\}_{j=1}^{J} \sim f(x_m^+), m \in T\) is described in \([17]\). Because all states \(x_k\) are conditionally independent given \(y^{(1:n)}\) (see \([32]\)), samples \(\{ (x_c^{(j)}, x_T^{(j)}) \}_{j=1}^{J} \sim f(x_c^+, x_T^+)\) can now be obtained by simple stacking operations, i.e., \(x_c^{(j)} = [x_l^{(j)}]_{l' \in C}\) and \(x_T^{(j)} = [x_m^{(j)}]_{m \in T}\). Finally, for each \( (x_c^{(j)}, x_T^{(j)})\), CA \(l\) computes samples \(\{y^{(j,j')}\}_{j' = 1}^{J'} \sim f(y^+ | x_c^{(j)}, x_T^{(j)}; u^+ = u_j^+)\) as described in Appendix C.

Using the samples \( (x_c^{(j)}, x_T^{(j)})\) and \(y^{(j,j')}, j = 1, \ldots, J, j' = 1, \ldots, J'\), as well as the reference vectors \(u_{l,t}^+, l' \in C\), the gradient \(\frac{\partial D_i(u_{l,t}^+)}{\partial u_j} \) can be computed locally at CA \(l\) according to \(30\) and \(31\). Note, however, that this quasi-centralized approach presumes that each CA \(l\) knows the
state evolution models (1) and the measurement models (2) of all the other CAs \( l' \in C \setminus \{l\} \).

The communication cost of the quasi-centralized scheme, in terms of the number of real values transmitted by each CA, is \((JM + M_u)W \approx JM W\). Here, \( M \) and \( M_u \) are the dimensions of the \( x_i \) and the \( u_i \), respectively, and \( W \) depends on the network size and topology and is bounded as \( 1 \leq W \leq |C| \). Thus, the number of transmissions scales linearly with \( J \) and does not depend on \( J' \). If the network formed by the CAs is fully connected, i.e., \( C = \{l\} \cup C_l \), then all the samples \( \{ (x_{l,i}^{(j)}, x_{l,i}^{(j')}) \}_{j=1}^J \) can be obtained without flooding; CA \( l \) simply broadcasts its reference vector \( u_{i,l}^+ \) and its samples \( \{ x_{l,i}^{(j')} \}_{j=1}^J \sim f(x_i) \) to all the other CAs in the network and receives their reference vectors and samples. Here, the number of real values transmitted by each CA is only \( JM + M_u \).

4) Distributed Processing: A truly distributed computation of (30) and (31) that avoids the use of flooding and does not require each CA to know the state evolution and measurement models of all the other CAs can be performed as follows. First, CA \( l \) broadcasts its own samples \( \{ x_{l,i}^{(j)} \}_{j=1}^J \sim f(x_i) \) calculated in the estimation layer to all neighboring CAs \( l' \in C_l \), and it receives samples \( \{ x_{l,i}^{(j')} \}_{j=1}^J \sim f(x_i) \) from all neighbors \( l' \in C_l \). In addition, CA \( l \) locally calculates samples \( \{ x_{m,i}^{(j)} \}_{j=1}^J \sim f(x_i) \) for all \( m \in T_l \) via the prediction step (33) (with \( T \) replaced by \( T_l \)), using the sample-based implementation described in [17]. Thus, after the stacking operations \( x_{C_l}^{(j)} = [x_{l,i}^{(j)}]_{l' \in C_l} \) and \( x_{C_l}^{(j)} = [x_{m,i}^{(j)}]_{m \in T_l} \), samples \( \{ (x_{l,i}^{(j)}, x_{C_l}^{(j)}, x_{C_{l'}}^{(j)}) \}_{j=1}^J \sim f(x_i, x_{C_l}, x_{C_{l'}}^+) \) are available at CA \( l \).

A key aspect for a distributed computation of (30) and (31) is the (non)availability of the quantities \( \alpha_l(y^{+(j,j')}, x_{C,l}^{(j')}, x_{T,l}^{(j')}, u_{l}^+) \) and \( \beta_l(y^{+(j,j')}, x_{C,l}^{(j')}, x_{T,l}^{(j')}, u_{l}^+) \) at CA \( l \). The factors of \( \alpha_l(y^{+}, x_{C,l}^{+}, x_{T,l}^{+}, u_{l}^+) \) (see (28)) correspond to measurements to be acquired by CA \( l \); they are known to CA \( l \) since the own state evolution model and measurement model are known to CA \( l \). In fact, it is easily verified using (28) that \( \alpha_l(y^{+}, x_{C,l}^{+}, x_{T,l}^{+}, u_{l}^+) = f(\tilde{y}_{l}^+) = f(\tilde{y}_{l}^+) = f(\tilde{y}_{l}^+) \) at CA \( l \), where \( u_{l}^+ \triangleq [u_{l,v}^+]_{v \in C_l} \) and

\[
\tilde{y}_{l}^+ \triangleq [y_{l,k}^+]_{k \in A_l} = [d_{l}(x_{l,k}^+, x_{l,k}^+, v_{l,k}^+)]_{k \in A_l}.
\] (34)

Note that \( \tilde{y}_{l}^+ \) comprises the measurements acquired by CA \( l \) at the next time (except that \( A_l \) is replaced with \( A_l \) since it is not known at the current time \( n \)). Thus, we conclude that \( \alpha_l(y^{+}, x_{C,l}^{+}, x_{T,l}^{+}, u_{l}^+) \) is available at CA \( l \). On the other hand, the factors of \( \beta_l(y^{+}, x_{C,l}^{+}, x_{T,l}^{+}, u_{l}^+) \) (see (29)) correspond to measurements to be acquired by other CAs; they are not available at CA \( l \) since, typically, the respective state evolution and measurement models are not known to CA \( l \). Therefore, \( \beta_l(y^{+}, x_{C,l}^{+}, x_{T,l}^{+}, u_{l}^+) \) is not available at CA \( l \).

We will now present a distributed computation of (30) and (31) that avoids the use of flooding and does not require each CA to know the state evolution and measurement models of all the other CAs \( l' \in C \setminus \{l\} \).
and
\[
F_{j,j',j''} = \frac{1}{|C|} \sum_{l \in C} F_{j,j',j''}^{(l)},
\]
for \(j = 1, \ldots, J, j' = 1, \ldots, J', \) and \(j'' = 1, \ldots, J'' \). To compute \(F_{j,j',j''}^{(l)} \) in (36), CA \(l \) only needs samples \(\{Y_{l}^{(j,j')}\}_{j'=1}^{J'} \sim f(Y_{l}^{(j,j')} | x_{C_l}^{(j)}, x_{R}^{(j)}; u_{i,l}^{+} = u_{i}, C_l) \) and the reference vectors \(u_{i,l}^{+} \) for \(l' \in C_l \).

The computation of the samples \(\{Y_{l}^{(j,j')}\}_{j'=1}^{J'} \) is described in Appendix A. The \(u_{i,l}^{+} \) can be obtained at CA \(l \) through communication with the neighbor CAs \(l' \in C_l \).

Once the \(F_{j,j',j''}^{(l)} \) have been calculated at CA \(l \), their averages \(F_{j,j',j''} \) in (37) can be computed in a distributed way by using \(J^2J' \) parallel instances of an average consensus or gossip scheme [18], [19]. These schemes are iterative and use only communication between neighboring CAs (i.e., each CA \(l \in C \) transmits data to each neighbor \(l' \in C_l \)); they are are initialized at each CA \(l \) with \(F_{j,j',j''}^{(l)} \). After convergence of the consensus or gossip scheme, \(F_{j,j',j''} \) and, hence, \(\beta_l(x^{(j,j')}, x_{C_l}^{(j')}, x_{R}^{(j')}, u_{i}^{+}) \) for all \(j, j', j'' \) is available at each CA \(l \).

At this point, CA \(l \) has available all the information required to evaluate \(\alpha_l(x^{(j,j')}, x_{C_l}^{(j')}, x_{R}^{(j')}, u_{i}^{+}) \) and \(\frac{\partial}{\partial u_{i}} f(x^{(j,j')}, x_{C_l}^{(j')}, x_{R}^{(j')}, u_{i}^{+}) \bigg|_{u_{i}=u_{i}^{+}} \) and an approximation of \(\beta_l(x^{(j,j')}, x_{C_l}^{(j')}, x_{R}^{(j')}, u_{i}^{+}) \) has been provided by the consensus or gossip scheme, for \(j = 1, \ldots, J, j' = 1, \ldots, J', \) and \(j'' = 1, \ldots, J \).

Therefore, CA \(l \) is now able to evaluate (30) and (31). In the course of the overall distributed computation, CA \(l \) transmits \(J^2J'|C_l| R + J M + M_u \approx J^2J'|C_l|R \) real values, where \(R \) is the number of iterations used for one instance of the consensus or gossip scheme. A large \(R \) corresponds to a high degree of convergence of the consensus or gossip scheme, which means that local data is disseminated over large distances in the network. Because the control vector of a given CA might not be strongly affected by information far away CAs, a small \(R \) is often sufficient for good performance. Nevertheless, because the communication requirements are proportional to \(J^2J' \), they are typically higher than those of the flooding approach discussed in Section V-A.3

B. Gradient of \(G_l(u_{i}^{+}) \)

Next, we consider the second gradient in the expansion (24), i.e., using (19),
\[
\frac{\partial G_l(u_{i}^{+})}{\partial u_{i}^{+}} \bigg|_{u_{i}=u_{i}^{+}} = \frac{\partial}{\partial u_{i}^{+}} \left( \int f(x_l) \log |J_{\beta_l}(x_l; u_{i}^{+})| \, dx_l \right) \bigg|_{u_{i}=u_{i}^{+}}
\]

We assume that for each value of \(x_l, J_{\beta_l}(x_l; u_{i}^{+}) \) is differentiable with respect to \(u_{i}^{+} \) at \(u_{i}^{+} \) and nonzero for all \(u_{i}^{+} \) in some (arbitrarily small) neighborhood of \(u_{i}^{+} \); note that this implies that \(J_{\beta_l}(x_l; u_{i}^{+}) \neq 0 \) and that also \(|J_{\beta_l}(x_l; u_{i}^{+})| \) is differentiable with respect to \(u_{i}^{+} \) at \(u_{i}^{+} \). We then have
\[
\frac{\partial G_l(u_{i}^{+})}{\partial u_{i}^{+}} \bigg|_{u_{i}=u_{i}^{+}} = \int f(x_l) \frac{\partial \log |J_{\beta_l}(x_l; u_{i}^{+})|}{\partial u_{i}^{+}} \bigg|_{u_{i}=u_{i}^{+}} \, dx_l
\]
\[
= \int f(x_l) \left[ \frac{1}{|J_{\beta_l}(x_l; u_{i}^{+})|} \frac{\partial |J_{\beta_l}(x_l; u_{i}^{+})|}{\partial u_{i}^{+}} \right] \bigg|_{u_{i}=u_{i}^{+}} \, dx_l.
\]

(38)
Based on the samples \( \{ x_i^{(j)} \}_{j=1}^J \sim f(x_i) \), which were calculated in the estimation layer, a Monte Carlo approximation of (38) is obtained as

\[
\frac{\partial G_l(u_i^+)}{\partial u_i^+} \bigg|_{u_i^+ = u_i} \approx \frac{1}{J} \sum_{j=1}^J \frac{1}{|J_{g_l}(x_i^{(j)}; u_i^+)|} \frac{\partial |J_{g_l}(x_i^{(j)}; u_i^+)|}{\partial u_i^+} \bigg|_{u_i^+ = u_i}.
\] (39)

For many practically relevant state evolution models (17), the computation of \( \frac{\partial G_l(u_i^+)}{\partial u_i^+} \bigg|_{u_i^+ = u_i} \) can be avoided altogether or \( \frac{\partial G_l(u_i^+)}{\partial u_i^+} \bigg|_{u_i^+ = u_i} \) can be calculated in closed form, without a sample-based approximation. Some examples are considered in the following.

1) \( J_{g_l}(x_l; u_l^+) \) does not depend on \( u_l^+ \): In this case, \( \frac{\partial G_l(u_i^+)}{\partial u_i^+} = 0 \). An important example is the “linear additive” state evolution model (cf. (17)) \( \tilde{g}_l(x_l; u_l^+) = Ax_l + \zeta(u_l^+) \) with an arbitrary matrix \( A \) and function \( \zeta(\cdot) \) of suitable dimensions. Here, we obtain \( J_{g_l}(x_l; u_l^+) = \det A \) and thus \( \frac{\partial G_l(u_i^+)}{\partial u_i^+} = 0 \). A second important example is the velocity motion model [33, Section 5.3].

Here, the local state \( x_l \) is the pose of a robot, which consists of the 2D position \( (x_{l,1}, x_{l,2}) \) and the orientation \( \theta_l \), and the control vector \( u_l \) consists of the translational velocity \( v_l \) and the rotational velocity \( \omega_l \). The state evolution model is given by

\[
\tilde{g}_l(x_l; u_l^+) = \begin{bmatrix} x_{l,1} + v_l^+ \cos(\theta_l) + \omega_l^+ \theta_l \\ x_{l,2} + v_l^+ \sin(\theta_l) + \omega_l^+ \end{bmatrix}.
\]

Here, \( J_{g_l}(x_l; u_l^+) = 1 \) and thus \( \frac{\partial G_l(u_i^+)}{\partial u_i^+} = 0 \).

2) \( J_{g_l}(x_l; u_l^+) \) does not depend on \( x_l \): If \( J_{g_l}(x_l; u_l^+) = J_{g_l}(u_l^+) \), then (19) simplifies to \( G_l(u_i^+) = \log |J_{g_l}(u_l^+)| \). Thus, we have

\[
\frac{\partial G_l(u_i^+)}{\partial u_i^+} = \frac{1}{|J_{g_l}(u_l^+)|} \frac{\partial |J_{g_l}(u_l^+)|}{\partial u_i^+},
\]

which can be calculated in closed form.

C. Summary of the Control Layer

The operations performed in the control layer are summarized in Fig. [3] for the quasi-centralized method discussed in Section V-A3 and in Fig. [4] for the distributed method discussed in Section V-A4.

VI. Special Cases

Next, we discuss two special cases: distributed, cooperative estimation of local states (i.e., there are no targets) and distributed, cooperative estimation of global states (i.e., the local states are known).
\{x_k^{(j)}\}_{j=1}^{J} \sim f(x_k), \ k \in \{l\} \cup T \quad \text{(from estimation layer)}

Flooding algorithm provides $u_{l,l'}^+$ and $\{x_{l'}^{(j)}\}_{j=1}^{J} \sim f(x_{l'}), \ l' \in C \setminus \{l\}$

Prediction step maps $\{x_{m}^{(j)}\}_{j=1}^{J} \sim f(x_m), \ m \in T \quad \text{(see (33) and (17))}$

At this point, samples $\{ (x_k^{(j)}, x_{l}^{(j)}, x_{l}^{(j)}) \}_{j=1}^{J}$ are available at CA $l$

\begin{align*}
\text{Computation of } & \frac{\partial D_i(u^+_l)}{\partial u_l} \bigg|_{u_l = u_l^+} \quad \text{(see (30) and (31))} \\
\text{Computation of } & \frac{\partial G_i(u^+_l)}{\partial u_l} \bigg|_{u_l = u_l^+} \quad \text{(see (39))} \\
\end{align*}

Evaluation of (24) and (16)

Fig. 3. Flow chart of the quasi-centralized implementation of the control layer at CA $l$ (see Section V-A3).

\begin{align*}
\{x_k^{(j)}\}_{j=1}^{J} \sim f(x_k), \ k \in \{l\} \cup T_l \quad \text{(from estimation layer)}

\text{Communication with CAs } l' \in C_l \text{ provides } u_{l,l'}^+ \text{ and } \{x_{l'}^{(j)}\}_{j=1}^{J} \sim f(x_{l'}), \ l' \in C_l

\text{Prediction step maps } \{x_{m}^{(j)}\}_{j=1}^{J} \sim f(x_m), \ m \in T_l \quad \text{(see (33) and (17))}

\text{At this point, samples } \{ (x_l^{(j)}, x_{l}^{(j)}, x_{l}^{(j)}) \}_{j=1}^{J} \text{ are available at CA } l

\begin{align*}
\text{Calculation of } & \left\{ x^{(j,j',j'')} \right\}_{j'=1}^{J} \sim f\left( y^+ | x_l^{(j)}, x_{l}^{(j)}, x_{l}^{(j')}; u^+_l = u^+_i C_l \right) \\
\text{for each } (x_l^{(j)}, x_{l}^{(j)}, x_{l}^{(j)}) & \text{, } j = 1, \ldots, J \quad \text{(see Appendix D)}
\end{align*}

\begin{align*}
\text{Average consensus or gossip algorithm provides } & \text{(see (35))} \\
\beta_l \left( y^{(j,j',j'')} | x_l^{(j)}, x_{l}^{(j)}, x_{l}^{(j')}; u^+_l = u^+_i C_l \right), \ j = 1, \ldots, J, \ j' = 1, \ldots, J', \ j'' = 1, \ldots, J

\{x_l^{(j)}\}_{j=1}^{J} \sim f(x_l) \quad \text{(from estimation layer)}

\begin{align*}
\text{Computation of } & \frac{\partial D_i(u^+_l)}{\partial u_l} \bigg|_{u_l = u_l^+} \quad \text{(see (30) and (31))} \\
\text{Computation of } & \frac{\partial G_i(u^+_l)}{\partial u_l} \bigg|_{u_l = u_l^+} \quad \text{(see (39))} \\
\end{align*}

Evaluation of (24) and (16)

Fig. 4. Flow chart of the distributed implementation of the control layer at CA $l$ (see Section V-A4).
A. Cooperative Estimation of Local States

Here, we assume that there are no targets, and thus the task considered is only the distributed, cooperative estimation of the local states.

1) Estimation Layer: The marginal posteriors corresponding to the targets are no longer calculated. In the calculation of the marginal posterior of CA $l$, the correction step (7) simplifies to

$$f(x_l|y) \propto \int \prod_{l' \in \mathcal{C}} f(x_{l'}) \prod_{l_i \in \mathcal{C}_{l'}} f(y_{l_i,l'}|x_{l_i}, x_{l'}) \, dx_{-l},$$

while the prediction step (5) remains unchanged. A feasible and, typically, accurate approximation of $f(x_l|y)$ in (40) can be obtained by evaluating

$$b^{(p)}(x_l) \propto f(x_l) \prod_{l' \in \mathcal{C}} f(y_{l,l'}|x_l, x_{l'}) b^{(p-1)}(x_{l'}) \, dx_{l'}$$

for iteration index $p = 1, \ldots, P$, where $b^{(0)}(x_{l'}) = f(x_{l'})$, $l' \in \mathcal{C}_l$. This amounts to the BP-based sum-product algorithm over a wireless network (SPAWN) presented in [13]. All quantities involved in (41) are locally available at CA $l$ or can be made available by communicating only with the neighboring CAs $l' \in \mathcal{C}_l$. A sample-based implementation of (41) is discussed in [34] and [35].

2) Control Layer: Since there are no targets, the component $D_l(u^+) = I(x_C, x_T^+; y^+; u^+)$ of the objective function in (23) simplifies to $D_l(u^+) = I(x_C; y^+; u^+)$. The expression of the gradient of $D_l(u^+)$ in (30) and (31) here simplifies as well because $\alpha_l(y^+, x_C, x_T^+, u^+) = \alpha_l(y^+, x_C, u^+)$ and $\beta_l(y^+, x_C, x_T^+, u^+) = \beta_l(y^+, x_C, u^+)$ (according to (28) and (29), since $T = \emptyset$); furthermore, sampling from $f(x_C, x_T^+)$ (see Sections V-A3 and V-A4) reduces to sampling from $f(x_C)$.

B. Cooperative Estimation of Global States

Next, we discuss the case where the local states of the CAs are known, and thus our task is only the distributed, cooperative estimation of the target states.

1) Estimation Layer: The marginal posteriors corresponding to the CAs are no longer calculated, and the correction step (7) in the calculation of the marginal posterior of the $m$th target simplifies to

$$f(x_m|y) \propto f(x_m) \prod_{l \in \mathcal{C}_m} f(y_{l,m}|x_l, x_m),$$

where $f(x_m)$ is calculated according to (6). A computationally feasible approximation of sequential state estimation as given by (42) and (6) is provided by the particle filter [36].

The product of local likelihood functions $\prod_{l \in \mathcal{C}_m} f(y_{l,m}|x_l, x_m)$ is not available at the CAs. However, as in Section III-B the likelihood consensus scheme can be employed to provide an approximation of the functional form of $\prod_{l \in \mathcal{C}_m} f(y_{l,m}|x_l, x_m)$ (as a function of $x_m$) to each CA using only
local communication [27]. Alternatively, an average consensus or gossip algorithm can be executed for each sample weight [28]. For the calculations in the control layer (described presently), a common set of samples is required at each CA. This can be ensured by additionally using, e.g., a max-consensus and providing all CAs with the same seed for random number generation [28], [29].

2) Control Layer: Since there are no unknown CA states, the objective function in (23) simplifies in that $D_I(u^+) = I(x^+_T; y^+; u^+)$ and $G_l(u^+_l) = 0$. The expression of the gradient of $D_I(u^+)$ in (30) and (31) simplifies because $\alpha_l(y^+, x_C, x^+_T, u^+) = \alpha_l(y^+, x^+_C, x^+_T)$; furthermore, sampling from $f(x_C, x^+_T)$ reduces to sampling from $f(x^+_T)$.

This special case was previously considered in [12]. More specifically, [12] studied estimation of one static global state and proposed a distributed, gradient-based, information-seeking controller and a sample-based implementation. Our present work extends [12] to time-varying local and global states and cooperative estimation tasks with possibly loopy factorization structure.

VII. SIMULATION RESULTS

We demonstrate the performance of the proposed method for three different localization scenarios. In Section VII-B we study the behavior of the controller by considering noncooperative self-tracking of four mobile CAs based on distance measurements relative to an anchor. In Section VII-C we consider cooperative self-tracking of three mobile CAs. Finally, in Section VII-D two mobile CAs perform cooperative simultaneous self-tracking and tracking of a target. Simulation source files and animated plots are available at http://www.nt.tuwien.ac.at/about-us/staff/florian-meyer/.

A. Simulation Setup

The following aspects of the simulation setup are common to all three scenarios. The states of the CAs consist of their 2D position, i.e., $x^{(n)}_l \triangleq [x^{(n)}_{l,1}, x^{(n)}_{l,2}]^T$. In addition to the mobile CAs, there is one anchor CA (indexed by $l = 1$), i.e., a static CA that broadcasts its own (true) position to the mobile CAs but does not perform any measurements. The CA network is fully connected if not indicated otherwise. The states of the mobile CAs evolve independently according to (43)

$$x^{(n)}_l = x^{(n-1)}_l + u^{(n)}_l + q^{(n)}_l, \quad n = 1, 2, \ldots$$

Here, $q^{(n)}_l \in \mathbb{R}^2$ is zero-mean Gaussian with independent and identically distributed entries, i.e., $q^{(n)}_l \sim \mathcal{N}(0, \sigma^2_q I)$ with $\sigma^2_q = 10^{-3}$, and $q^{(n)}_l$ and $q^{(n)}_{l'}$ are independent unless $(l, n) = (l', n')$. The domain $U_l$ of the control vector $u^{(n)}_l$ is defined by the norm constraint $\|u^{(n)}_l\| \leq u^\text{max}_l$. In the initialization of the algorithms, at time $n = 0$, we use a state prior that is uniform on $[-200, 200] \times [-200, 200]$.
The mobile CAs acquire distance measurements according to (3), i.e., $y_{l,k}^{(n)} = \|x_i^{(n)} - x_k^{(n)}\| + \nu_{l,k}^{(n)}$, where the measurement noise $\nu_{l,k}^{(n)}$ is independent across $l$, $k$, and $n$ and Gaussian with variance

$$\sigma_{l,k}^{(n)2} = \begin{cases} \sigma_0^2, & \|x_i^{(n)} - x_k^{(n)}\| \leq d_0 \\ \sigma_0^2 \left( \frac{\|x_i^{(n)} - x_k^{(n)}\|}{d_0} - 1 \right)^\kappa + 1, & \|x_i^{(n)} - x_k^{(n)}\| > d_0 \end{cases} \quad (44)$$

That is, the measurement noise variance stays constant up to some distance $d_0$ and then increases polynomially with some exponent $\kappa$. This is a simple model for time-of-arrival distance measurements [37]. We set $\sigma_0^2 = 50$ and $\kappa = 2$ and, if not stated otherwise, $d_0 = 50$.

In the estimation layer, we use $J = 3600$ samples. A resampling step is performed to avoid weight degeneracy [38]. Resampling transforms weighted samples $\{(x_k^{(n)(j)}, w_k^{(n)(j)})\}_{j=1}^{J}$ representing the belief $b(x_k^{(n)})$ into nonweighted samples $\{x_k^{(n)(j)}\}_{j=1}^{J}$. We use a somewhat nonorthodox type of resampling that helps move samples to positions with high probability mass, thereby reducing the number of samples needed. More specifically, at every $L$th time step $n$, we sample from a kernel approximation of the belief; at all other time steps, we perform standard systematic resampling [38]. The kernel approximation of the belief $b(x_k^{(n)})$ is obtained as [39]

$$\tilde{b}(x_k^{(n)}) = \sum_{j=1}^{J} w_k^{(n)(j)} K(x_k^{(n)} - x_k^{(n)(j)}) ,$$

with the Gaussian kernel $K(x) = (2\pi \sigma_K^2)^{-1} \exp\left(-\|x\|^2/(2 \sigma_K^2)\right)$. Here, the variance $\sigma_K^2$ is chosen as (cf. [34, 39]) $\sigma_K^2 = J^{1/3} \text{tr}(C_k^{(n)})/2$ if $\text{tr}(C_k^{(n)}) < 2\sigma_0^2$ and $\sigma_K^2 = \sigma_0^2$ otherwise, where $\text{tr}(C_k^{(n)})$ denotes the trace of the weighted sample covariance matrix defined as

$$C_k^{(n)} = \sum_{j=1}^{J} w_k^{(n)(j)} x_k^{(n)(j)} x_k^{(n)(j)\top} - \mu_k^{(n)} \mu_k^{(n)\top} , \quad \text{where} \quad \mu_k^{(n)} = \sum_{j=1}^{J} w_k^{(n)(j)} x_k^{(n)(j)} .$$

We choose $L = 40$ if $\text{tr}(C_k^{(n)}) < 80$, $L = 20$ if $80 \leq \text{tr}(C_k^{(n)}) < 1000$, and $L = 10$ if $\text{tr}(C_k^{(n)}) \geq 1000$; this choice led to good results in our simulation setting.

We employ a censoring scheme [35] to reduce the number of particles and avoid numerical problems during the first time steps, where the mobile CAs still have uninformative beliefs. More specifically, only CAs $l$ with $\text{tr}(C_l^{(n)}) < 10$ are used as localization partners by neighboring CAs and (in our third scenario) are involved in localizing the target. In the control layer, this censoring scheme corresponds to the following strategy: as long as CA $l$ is not localized (i.e., $\text{tr}(C_l^{(n)}) < 10$), its objective function is $D_h(u_l^{(n+1)}) \triangleq -h(y_{l(y_1^{(n+1)})}^{(n+1)}; y_{l(y_1^{(n)})}^{(n+1)}; u_l^{(n+1)})$, i.e., the negative differential entropy of only the own state conditioned on only the own measurement relative to the anchor CA, $y_{l(y_1^{(n)})}^{(n+1)}$.

The gradient ascent in the controller (see (16)) uses the reference points $u_{l(t)}^{(n)} = 0$, which are consistent with the state evolution model (43), and step sizes $\epsilon_l^{(n)}$ chosen such that $\|u_l^{(n)}\| = \epsilon_l^{(n)}$.
Thus, the controller moves every mobile CA \( l \in \mathcal{C} \) with maximum nominal speed (determined by \( u_{l}^{\text{max}} \)) in the direction of maximum local increase of the objective function. The number of samples used in the control layer is \( JJ' = 60000 \), with \( J = 1200 \) and \( J' = 50 \).

### B. Noncooperative Self-Tracking

In order to study the behavior of the controller, we consider four mobile CAs \( l = 2, 3, 4, 5 \) that perform self-tracking without any cooperation during 300 time steps \( n \). The mobile CAs measure their distance to the static anchor CA \( (l = 1) \), which is located at position \([0, 0]^T\), but they do not measure any distances between themselves. Their measurement models use different values of \( d_0 \), namely, \( d_0 = 20, 50, 100, \) and 100 for \( l = 2, 3, 4, \) and 5, respectively. The mobile CAs start at position \([100, 0]^T\) and move with identical nominal speed determined by \( u_{l}^{\text{max}} = 1 \). The objective function for the control of CAs 2, 3, and 4 is \( D_h(x_l^{(n+1)}|y_l^{(1:n)}, y_l^{(n+1)}, u_l^{(n+1)}) \). CA 5 is not controlled; it randomly chooses a direction at \( n = 1 \) and then moves in that direction with constant nominal speed determined by \( u_{l}^{\text{max}} = 1 \). Fig. 5 shows an example of the trajectories of the four mobile CAs. These trajectories are quite different because of the different values of \( d_0 \) and the fact that CA 5 is not controlled. CA 4, after an initial turn, is roughly localized in the sense that the shape of its marginal posterior has changed from an annulus to only a segment of an annulus. Thereafter, CA 4 turns around the anchor, which is reasonable in view of the single distance measurement available at each time \( n \) and the fact that, since \( d_0 = 100 \), the measurement noise cannot be decreased by approaching the anchor. CA 3 (with \( d_0 = 50 \)) initially approaches the anchor. At a distance of 50 to the anchor, the measurement noise cannot be decreased any more, and thus CA 3 turns around the anchor without approaching it further. A similar behavior is exhibited by CA 2 (with \( d_0 = 20 \)).

Fig. 6 shows the self-localization root-mean-square errors (RMSEs) of the four mobile CAs.
RMSEs were determined at each time \( n \) by averaging over 300 simulation runs. As can be seen, the three CAs performing information seeking-control \( (l = 2, 3, 4) \) are fairly well localized after about 100 time steps. CA 2 (with \( d_0 = 20 \)) takes longer to localize itself than CAs 3 and 4 since, prior to reaching a distance of 20 to the anchor, it has a larger noise variance (see (44)). The performance of CA 3 and CA 4 is almost identical; the larger noise variance of CA 3 during the initial time steps is compensated by a smaller turning radius once a distance of 50 to the anchor has been reached. CA 5 is unable to localize itself, due to the lack of intelligent control.

C. Cooperative Self-Tracking

Next, we study the proposed method for cooperative self-tracking with information-seeking control (abbreviated as C–C). There are three mobile CAs \( l = 2, 3, 4 \) with different start points \( ([−50, 0]^T, [0, −50]^T, \text{ and } [0, 70]^T \text{ for } l = 2, 3, \text{ and } 4, \text{ respectively}) \) and different nominal speeds \( (u_{\text{max}}^l = 1, 0.3, \text{ and } 0.1 \text{ for } l = 2, 3, \text{ and } 4, \text{ respectively}) \). The mobile CAs measure their distances to a static anchor \( l = 1 \) located at \( [−60, 0]^T \) and to each other, using \( d_0 = 50 \). Example trajectories are shown in Fig. 7.

For comparison, we also consider noncooperative self-tracking with information-seeking control as studied in Section VII-B (abbreviated as N–C). Finally, we consider another scheme (abbreviated as C–N) where the CAs cooperate in the estimation layer but no intelligent control is performed. Here, each CA randomly chooses a direction and then moves in that direction with constant nominal speed determined by \( u_{\text{max}}^l \).

Fig. 8 shows the self-localization RMSEs of the three schemes, which were determined by averaging over the three mobile CAs and over 300 simulation runs. It is seen that the RMSEs of the two reference schemes N–C and C–N decrease only very slowly whereas, after about 70 time steps, the RMSE of the proposed C–C scheme decreases rather quickly to a low value. This behavior can be explained as follows. Without cooperation (N–C) or without intelligent control (C–N), CAs 3 and 4 need a long
time to localize themselves because they are slow and initially far away from the anchor. On the other hand, CA 2 localizes itself very quickly because it is fast and initially close to the anchor. With cooperation and control (C–C), CA 2 moves in such a way that it supports the self-localization of the two other CAs. In fact, as shown by Fig. 7, CA 2 first localizes itself by starting to turn around the anchor and then makes a sharp turn to approach CAs 3 and 4, which helps them localize themselves. This demonstrates the function and benefits of cooperative estimation and control.

D. Cooperative Self-Tracking and Target Tracking

Finally, we consider cooperative simultaneous self-tracking and target tracking. Two mobile CAs \( l = 2, 3 \) starting at position \( [20, 20]^T \) and \( [-10, -10]^T \), respectively and with nominal speed determined by \( u_l^{\text{max}} = 1 \) cooperatively localize and track themselves and a mobile target. There is also a static anchor \( l = 1 \) at position \( [-50, 0]^T \). The target state \( x_m^{(n)} = x_4^{(n)} \) consists of position and velocity, i.e.,
Fig. 9. Example trajectories for cooperative simultaneous self-tracking and target tracking with information-seeking control (C–C scheme). The initial CA positions are indicated by bullets, the initial target position by a cross, and the anchor position by a star.

$$x_4^{(n)} \triangleq [x_4^{(n)}, x_4^{(n)}, \dot{x}_4^{(n)}, \dot{x}_4^{(n)}]^T.$$ The target state evolves according to

$$x_4^{(n)} = Gx_4^{(n-1)} + Wq_i^{(n)}, \quad n = 1, 2, \ldots ,$$

where

$$G = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \quad W = \begin{pmatrix} 0.5 & 0 & 0.5 \\ 0 & 0 & 1 \\ 1 & 0 \\ 0 & 1 \end{pmatrix},$$

and $$q_i^{(n)} \in \mathbb{R}^2$$ is zero-mean Gaussian with independent and identically distributed entries, i.e., $$q_i^{(n)} \sim \mathcal{N}(0, \tilde{\sigma}_q^2 I)$$ with $$\tilde{\sigma}_q^2 = 10^{-5}$$, and with $$q_i^{(n)}$$ and $$q_i^{(n')}$$ independent unless $$n = n'$$. The target trajectory is initialized with position $$[x_{4,1}^{(0)}, x_{4,2}^{(0)}]^T = [50, 0]^T$$ and velocity $$[\dot{x}_{4,1}^{(0)}, \dot{x}_{4,2}^{(0)}]^T = [0.05, 0.05]^T$$. In the initialization of the algorithms, we use a target position prior that is uniform on $$[-200, 200] \times [-200, 200]$$ and a target velocity prior that is Gaussian with mean $$[0, 0]^T$$ and covariance matrix $$\text{diag} \{10^{-1}, 10^{-1}\}$$. Fig. 9 shows an example of CA and target trajectories obtained with the proposed method of cooperative localization with information-seeking control (C–C). One can observe that the two CAs first start turning around the anchor to localize themselves and then approach the target. Finally, at a distance of 50 to the target, where further approaching the target would no longer decrease the measurement noise, the CAs spread out to achieve a geometric formation that is favorable for cooperatively localizing and tracking the target.

As before, we compare our C–C method with two reference methods, namely, noncooperative localization with information-seeking control (N–C) and cooperative localization with fixed, randomly chosen directions of movement (C–N). Fig. 10 shows the self-localization RMSEs and target localization RMSEs of the three schemes, which were determined by averaging over the two CAs and over 100 simulation runs. The following observations can be made:
Fig. 10. Performance of three different methods for simultaneous self-localization and target localization: (a) Self-localization RMSE, (b) target-localization RMSE.

- The self-localization performance of C–N is very poor: after an initial decrease, the RMSE slowly increases. In fact, typically, no cooperation is actually taking place, since the CAs are unable to localize themselves and thus each CA is censored by the respective other CA. The self-localization RMSEs of C–C and N–C decrease rather quickly to a low value. They are very similar, which can be explained as follows. Because both CAs move with the same nominal speed, they localize themselves approximately in the same manner. Therefore, as long as the CAs are not localized, no cooperation takes place due to censoring, and after they are localized, no further gain can be achieved by cooperation.

- The target localization RMSEs of the three methods are initially equal to 50 and slowly increase during the first 40 time steps. Indeed, due to the censoring scheme, the CAs start localizing the target only when they are localized themselves. Therefore, during the first 40 time steps, no measurements of the distance to the target are used by the CAs and the CAs’ target position estimation is solely based on the prior distribution, which is uniform. This leads to a target position estimate of $[0, 0]^T$ and in turn (since the target is initially located at $[50, 0]^T$) to an initial target localization RMSE of 50 at time $n = 1$. During the first 40 time steps, the RMSE slowly increases since the target is slowly moving to the upper right corner. The RMSE of C–N continues to increase in this manner even after $n = 40$ since with C–N, the CAs are never localized and therefore never start localizing the target. For C–C and N–C (both employing information-seeking control), after $n = 40$, the RMSE first increases and then decreases. The RMSE of C–C increases less and decreases sooner and more quickly than that of N–C, which again shows the benefits of cooperative estimation.

The initial increase and subsequent decrease of the target localization RMSE observed with C–C
and N–C after \( n = 40 \) can be explained as follows. After the CAs localized themselves and start localizing the target, the target position posterior at a given CA is roughly annularly shaped, with the center of the annulus being the CA position. (This position is equal to the turning point of the respective CA trajectory in Fig. 9.) The resulting target position estimate is located at that center. Thus, it is more distant from the true target position than the estimate \([0, 0]^T\) that was obtained when the CA was not yet localized and the target position posterior was still uniform. As the CAs approach the target, the target position posterior becomes unimodal and the target can be localized, resulting in a decrease of the target localization RMSE.

**VIII. Conclusion**

We proposed a Bayesian framework and method for distributed estimation with information-seeking control in agent networks. Distributed, cooperative, sequential estimation is performed for time-varying global states (related to noncooperative targets or features of the environment) and/or time-varying local states (related to individual cooperative agents), using a combination of belief propagation message passing and likelihood consensus. The distributed, cooperative control seeks to optimize the behavior of the cooperative agents by maximizing the negative posterior entropy of the agent states via a gradient ascent. A probabilistic information transfer from the estimation layer to the control layer enables effective control strategies and thus leads to excellent estimation performance.

A major advantage of the proposed approach is its generality. Our method relies on general state evolution and measurement models, an information-theoretic objective function for control, and sample-based representations of probability distributions. These characteristics make it suitable for nonlinear and non-Gaussian systems, such as those arising in location-aware networks. Numerical simulations for a simultaneous agent self-tracking and target tracking problem demonstrated intelligent behavior of the cooperative agents and a resulting improvement of estimation performance.

Possible directions for future research include an extension of the myopic (i.e., optimizing only one time step ahead) controller to a receding horizon [40]; this can be expected to improve the performance in scenarios with multiple time-varying global states. Furthermore, the complexity and communication cost of the proposed method could be reduced by introducing Gaussian or Gaussian mixture approximations [41] and using cubature points [42] instead of random samples.

**APPENDIX A**

**Proof of Equation (18)**

We will use the following transformation rule for differential entropy [43, Equation 18]: For a continuous random vector \( \mathbf{a} \) and a transformed random vector of identical dimension \( \mathbf{b} = g(\mathbf{a}) \),
where \( g(\cdot) \) is a bijective differentiable function with Jacobian determinant \( J_y(a) = \det \frac{\partial g(a)}{\partial a} \).

\[
    h(b) = h(a) + e(a), \quad \text{with} \quad e(a) \triangleq \int f(a) \log |J_y(a)| \, da. \tag{45}
\]

The conditional differential entropy \( h(x^+|y^+; u^+) \) can be expanded as \[22\] Chapter 8

\[
    h(x^+|y^+; u^+) = h(x^+, y^+; u^+) - h(y^+; u^+). \tag{46}
\]

The vector \( x^+ \) consists of \( x_l^+ \) and \( x_{A \setminus \{l\}}^+ \triangleq [x_k^+]_{k \in A \setminus \{l\}} \), and there is \( x_l^+ = \tilde{g}_l(x_l, u_l^+) \) (see (17)). Thus, the first term on the right-hand side of (46) can be expressed as \( h(x^+, y^+; u^+) = h(\tilde{g}_l(x_l, u_l^+), x_{A \setminus \{l\}}^+, y^+; u^+) \). Applying the transformation rule (45) to the “extended state evolution mapping” \( \tilde{g}_l^+ : [x_l^+, x_{A \setminus \{l\}}^+, y^+]^T \mapsto [(\tilde{g}_l(x_l, u_l^+))^T, x_{A \setminus \{l\}}^+, y^+]^T \), we then obtain

\[
    h(x^+, y^+; u^+) = h(x_l, x_{A \setminus \{l\}}^+, y^+; u_l^+) + e(x_l, x_{A \setminus \{l\}}^+, y^+; u_l^+), \tag{47}
\]

where

\[
    e(x_l, x_{A \setminus \{l\}}^+, y^+; u_l^+) \triangleq \int \int \int f(x_l, x_{A \setminus \{l\}}^+, y^+) \log |J_{\tilde{g}_l}(x_l, x_{A \setminus \{l\}}^+, y^+; u_l^+)| \, dx_l \, dx_{A \setminus \{l\}}^+ \, dy^+. \tag{48}
\]

Here, \( J_{\tilde{g}_l}(x_l, x_{A \setminus \{l\}}^+, y^+; u_l^+) \) is the Jacobian determinant of \( \tilde{g}_l(x_l, x_{A \setminus \{l\}}^+, y^+; u_l^+) \). It is easily seen that \( J_{\tilde{g}_l}(x_l, x_{A \setminus \{l\}}^+, y^+; u_l^+) = J_{\tilde{g}_l}(x_l; u_l^+) \), and thus we obtain further

\[
    e(x_l, x_{A \setminus \{l\}}^+, y^+; u_l^+) = \int \int \left[ \int f(x_l, x_{A \setminus \{l\}}^+, y^+) \, dx_{A \setminus \{l\}}^+ \, dy^+ \right] \log |J_{\tilde{g}_l}(x_l; u_l^+)| \, dx_l
    = \int f(x_l) \log |J_{\tilde{g}_l}(x_l; u_l^+)| \, dx_l
    = G_l(u_l^+). \tag{49}
\]

Inserting (48) into (47) and the resulting expression of \( h(x^+, y^+; u^+) \) into (46) gives

\[
    h(x^+|y^+; u^+) = h(x_l, x_{A \setminus \{l\}}^+, y^+; u^+) + G_l(u_l^+) - h(y^+; u^+). \tag{49}
\]

Next, we repeat this transformation procedure but apply it to the term \( h(x_l, x_{A \setminus \{l\}}^+, y^+; u^+) \) in (49) instead of \( h(x^+, y^+; u^+) \). Consider an arbitrary \( l' \in C \setminus \{l\} \), and note that \( x_{A \setminus \{l\}}^+ \) consists of \( x_{l'}^+ \) and \( x_{A \setminus \{l,l'\}}^+ \triangleq [x_k^+]_{k \in A \setminus \{l,l'\}} \), where \( x_k^+ = \tilde{g}_k(x_k, u_k^+) \) according to (17). Proceeding as above and inserting the resulting expression of \( h(x_l, x_{A \setminus \{l,l'\}}^+, y^+; u^+) \) into (49) yields

\[
    h(x^+|y^+; u^+) = h(x_l, x_{l'}, x_{A \setminus \{l,l'\}}^+, y^+; u^+) + G_{l'}(u_{l'}^+) + G_l(u_l^+) - h(y^+; u^+). \tag{50}
\]

We continue this procedure in a recursive fashion, splitting off CA state vectors from \( x_{A \setminus \{l,l'\}}^+ \) until only the target states (contained in \( x_{A \setminus \{l,l'\}}^+ \)) are left, and applying the transformation rule at each recursion. In the end, we obtain

\[
    h(x^+|y^+; u^+) = h(x_C, x_{A \setminus \{l,l'\}}^+, y^+; u^+) + \sum_{l \in C} G_l(u_l^+) - h(y^+; u^+). \tag{51}
\]
Finally, Equation (18) is obtained by noting that
\[ h(x_C, x_T^+; y^+; u^+) = h(x_C, x_T^+ | y^+; u^+) + h(y^+; u^+). \]

**APPENDIX B**

**DERIVATION OF (30) AND (31)**

1) Derivation of (30): Using (27) in (25) and recalling that \( \beta_l(y^+, x_C, x_T^+, u^+) \) does not depend on \( u_l^+ \) (see (29)) yields

\[
\frac{\partial D_l(u^+)}{\partial u_l^+} = \int \int f(x_C, x_T^+) \beta_l(y^+, x_C, x_T^+, u^+ \frac{\partial \alpha_l(y^+, x_C, x_T^+, u^+)}{\partial u_l^+} \times \log_\frac{\alpha_l(y^+, x_C, x_T^+, u^+)}{\alpha_l(y^+, x_C, x_T^+, u^+)} \beta_l(y^+, x_C, x_T^+, u^+) d\xi d\eta d \eta^+. \quad (50)
\]

Setting \( u^+ = u_l^+ \), and multiplying and dividing the integrand in (50) by \( \alpha_l(y^+, x_C, x_T^+, u^+), \) we obtain further

\[
\frac{\partial D_l(u^+)}{\partial u_l^+} \bigg|_{u^+ = u_l^+} = \int \int q(y^+, x_C, x_T^+; u^+ = u_l^+) \frac{1}{\alpha_l(y^+, x_C, x_T^+, u_l^+)} \frac{\partial \alpha_l(y^+, x_C, x_T^+, u_l^+)}{\partial u_l^+} \bigg|_{u_l^+ = u_l^+} \times \log_\frac{\alpha_l(y^+, x_C, x_T^+, u_l^+)}{\alpha_l(y^+, x_C, x_T^+, u_l^+)} \beta_l(y^+, x_C, x_T^+, u_l^+) d\xi d\eta d \eta^+, \quad (51)
\]

where

\[
q(y^+, x_C, x_T^+; u^+ = u_l^+) = f(x_C, x_T^+) f(y^+ | x_C, x_T^+; u^+ = u_l^+) = f(x_C, x_T^+, y^+; u^+ = u_l^+).
\]

Then, (30) is recognized to be a Monte Carlo approximation of (51) that is obtained by performing importance sampling (31) using \( q(y^+, x_C, x_T^+; u^+ = u_l^+) \) as importance density, i.e., the samples \( y^{+}, x^C, x_T^+ \) occurring in (30) are drawn from \( q(y^+, x_C, x_T^+; u^+ = u_l^+) \). Using (27), this importance density can be expressed as

\[
q(y^+, x_C, x_T^+; u^+ = u_l^+) = f(x_C, x_T^+) f(y^+ | x_C, x_T^+; u^+ = u_l^+) = f(x_C, x_T^+, y^+; u^+ = u_l^+).
\]

The second expression, \( f(x_C, x_T^+) f(y^+ | x_C, x_T^+; u^+ = u_l^+) \), underlies the two-stage sampling procedure described in Section V-A2.

2) Derivation of (31): We have

\[
f(y^+; u^+ = u_l^+) = \int \int f(y^+ | x_C, x_T^+; u^+ = u_l^+) f(x_C, x_T^+) d\xi d\eta. \quad (52)
\]

Using samples \( \{ (x_C^{(j)}, x_T^{+ (j)}) \}_{j=1}^J \sim f(x_C, x_T^+) \) (see Section V-A2), a Monte Carlo approximation of (52) is obtained as

\[
f(y^+; u^+ = u_l^+) \approx \frac{1}{J} \sum_{j=1}^J f(y^+ | x_C^{(j)}, x_T^{+ (j)}; u^+ = u_l^+).
\]

Evaluating this for \( y^+ = y^{+ (j)} \) (again see Section V-A2) and inserting (27) yields (31).
APPENDIX C

DRAWING SAMPLES FROM $f(\mathbf{y}^+|\mathbf{x}_c^+, \mathbf{x}_T^+; \mathbf{u}^+ = \mathbf{u}_c^+)$

We consider the quasi-centralized setting of Section V.A3. As discussed there, samples $\{\mathbf{x}_l^+(j)\}_{j=1}^J$ combined with (12) and (2), the composite measurement vector $\mathbf{y}^+$ can be written as

$$\mathbf{y}^+ = \left[ d_l(\mathbf{x}_l^+, \mathbf{x}_k^+, \mathbf{v}_{l,k}^+) \right]_{l \in \mathcal{C}, k \in \mathcal{A}_l}. \tag{53}$$

First, CA $l$ obtains samples $\{\mathbf{x}_l^+(j)\}_{j=1}^J \sim f(\mathbf{x}_l^+), l' \in \mathcal{C}$ and $\{\mathbf{x}_m^+(j)\}_{j=1}^J \sim f(\mathbf{x}_m^+), m \in \mathcal{T}$ are available at CA $l$, and it is assumed that the state evolution and measurement models of all CAs $l' \in \mathcal{C}$ are known to CA $l$. We start by noting that by combining (12) and (2), the desired samples $\{\mathbf{y}_l^+(j)\}_{j=1}^J \sim f(\mathbf{y}_l^+|\mathbf{x}_c^+, \mathbf{x}_T^+; \mathbf{u}^+ = \mathbf{u}_c^+) \sim f(\mathbf{y}_l^+|\mathbf{x}_c^+, \mathbf{x}_T^+) \sim f(\mathbf{y}_l^+|\mathbf{x}_c^+, \mathbf{x}_T^+, \mathbf{u}^+ = \mathbf{u}_c^+)$ can be calculated by carrying out the steps of Appendix C with obvious modifications—in particular, $\mathbf{y}^+$ is replaced by $\tilde{\mathbf{y}}_l^+$, $\mathcal{C}$ by $\{\mathcal{I}\} \cup \mathcal{C}_l$, and $\mathcal{T}$ by $\mathcal{T}_l$.

APPENDIX D

DRAWING SAMPLES FROM $f(\tilde{\mathbf{y}}_l^+|\mathbf{x}_l^+(j), \mathbf{x}_c^+, \mathbf{x}_T^+; \mathbf{u}_c^+ = \mathbf{u}_{l,c}^+)$

In the distributed setting of Section V.A3 samples $\{\mathbf{x}_l^+(j)\}_{j=1}^J \sim f(\mathbf{x}_l^+), l' \in \{\mathcal{I}\} \cup \mathcal{C}_l$ and $\{\mathbf{x}_m^+(j)\}_{j=1}^J \sim f(\mathbf{x}_m^+), m \in \mathcal{T}_l$ are available at CA $l$. We recall from (34) that $\tilde{\mathbf{y}}_l^+ = \left[ d_l(\mathbf{x}_l^+, \mathbf{x}_k^+, \mathbf{v}_{l,k}^+) \right]_{k \in \mathcal{A}_l}$. Based on the analogy of this expression to (53), the desired samples $\{\tilde{\mathbf{y}}_l^+(j,j')\}_{j'=1}^{J'} \sim f(\tilde{\mathbf{y}}_l^+|\mathbf{x}_l^+(j), \mathbf{x}_c^+, \mathbf{x}_T^+; \mathbf{u}_c^+ = \mathbf{u}_{l,c}^+) \sim f(\tilde{\mathbf{y}}_l^+|\mathbf{x}_l^+(j), \mathbf{x}_c^+, \mathbf{x}_T^+, \mathbf{u}_c^+ = \mathbf{u}_{l,c}^+)$ can be calculated by carrying out the steps of Appendix C with obvious modifications—in particular, $\mathbf{y}^+$ is replaced by $\tilde{\mathbf{y}}_l^+$, $\mathcal{C}$ by $\{\mathcal{I}\} \cup \mathcal{C}_l$, and $\mathcal{T}$ by $\mathcal{T}_l$.

REFERENCES

[1] T. Shima and S. Rasmussen, UAV Cooperative Decision and Control: Challenges and Practical Approaches. Philadelphia, PA: SIAM, 2009.
[2] F. Bullo, J. Cortés, and S. Martínez, Distributed Control of Robotic Networks: A Mathematical Approach to Motion Coordination Algorithms. Princeton, NJ: Princeton University Press, 2009.
[3] A. Nayak and I. Stojmenović, Wireless Sensor and Actuator Networks: Algorithms and Protocols for Scalable Coordination and Data Communication. Hoboken, NJ: Wiley, 2010.

[4] F. Zhao and L. J. Guibas, Wireless Sensor Networks: An Information Processing Approach. Amsterdam, The Netherlands: Morgan Kaufmann, 2004.

[5] P. Corke, T. Wark, R. Jurdak, W. Hu, P. Valencia, and D. Moore, “Environmental wireless sensor networks,” Proc. IEEE, vol. 98, no. 11, pp. 1903–1917, 2010.

[6] J. Ko, C. Lu, M. B. Srivastava, J. A. Stankovic, A. Terzis, and M. Welsh, “Wireless sensor networks for healthcare,” Proc. IEEE, vol. 98, no. 11, pp. 1947–1960, 2010.

[7] O. Hlinka, F. Hlawatsch, and P. M. Djuric, “Distributed particle filtering in agent networks: A survey, classification, and comparison,” IEEE Signal Process. Mag., vol. 30, no. 1, pp. 61–81, 2013.

[8] T. Zhao and A. Nehorai, “Distributed sequential Bayesian estimation of a diffusive source in wireless sensor networks,” IEEE Trans. Signal Process., vol. 55, no. 4, pp. 1511–1524, 2007.

[9] H. Aghajan and A. Cavallaro, Multi-Camera Networks: Principles and Applications. Burlington, MA: Academic Press, 2009.

[10] K.-D. Kim and P. R. Kumar, “Cyber physical systems: A perspective at the centennial,” Proc. IEEE, vol. 100, pp. 1287–1308, 2012.

[11] G. M. Hoffmann and C. J. Tomlin, “Mobile sensor network control using mutual information methods and particle filters,” IEEE Trans. Autom. Control, vol. 55, no. 1, pp. 32–47, 2010.

[12] B. J. Julian, M. Angermann, M. Schwager, and D. Rus, “Distributed robotic sensor networks: An information-theoretic approach,” Int. J. Robot. Res., vol. 31, pp. 1134–1154, Sep. 2012.

[13] H. Wymeersch, J. Lien, and M. Z. Win, “Cooperative localization in wireless networks,” Proc. IEEE, vol. 97, pp. 427–450, Feb. 2009.

[14] Y.-C. Wu, Q. M. Chaudhari, and E. Serpedin, “Clock synchronization of wireless sensor networks,” IEEE Signal Process. Mag., vol. 28, Jan. 2011.

[15] F. Meyer, B. Etzlinger, F. Hlawatsch, and A. Springer, “A distributed particle-based belief propagation algorithm for cooperative simultaneous localization and synchronization,” in Proc. Asilomar Conf. Sig., Syst., Comput., Pacific Grove, CA, pp. 527–531, Nov. 2013.

[16] J. Teng, H. Snoussi, C. Richard, and R. Zhou, “Distributed variational filtering for simultaneous sensor localization and target tracking in wireless sensor networks,” IEEE Trans. Veh. Technol., vol. 61, no. 5, pp. 2305–2318, 2012.

[17] F. Meyer, O. Hlinka, H. Wymeersch, E. Riegler, and F. Hlawatsch, “Cooperative simultaneous localization and tracking in mobile agent networks,” IEEE Trans. Signal Process., 2014, submitted. Available online: http://arxiv.org/abs/1403.1824.

[18] R. Olfati-Saber, J. A. Fax, and R. M. Murray, “Consensus and cooperation in networked multi-agent systems,” Proc. IEEE, vol. 95, no. 1, pp. 215–233, 2007.

[19] A. G. Dimakis, S. Kar, J. M. F. Moura, M. G. Rabbat, and A. Scaglione, “Gossip algorithms for distributed signal processing,” Proc. IEEE, vol. 98, pp. 1847–1864, Nov. 2010.

[20] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, “Factor graphs and the sum-product algorithm,” IEEE Trans. Inf. Theory, vol. 47, pp. 498–519, Feb. 2001.

[21] C. M. Bishop, Pattern Recognition and Machine Learning. New York, NY: Springer, 2006.

[22] T. M. Cover and J. A. Thomas, Elements of Information Theory. New York, NY: Wiley, 2006.

[23] S. M. Kay, Fundamentals of Statistical Signal Processing: Estimation Theory. Upper Saddle River, NJ: Prentice-Hall, 1993.

[24] F. Morbidi and G. L. Mariottini, “Active target tracking and cooperative localization for teams of aerial vehicles,” IEEE Trans. Control Syst. Technol., vol. 21, no. 5, pp. 1694–1707, 2013.
[25] M. Schwager, P. Dames, D. Rus, and V. Kumar, “A multi-robot control policy for information gathering in the presence of unknown hazards,” in Proc. ISRR-11, Flagstaff, AZ, Aug. 2011.

[26] X. R. Li and V. P. Jilkov, “Survey of maneuvering target tracking. Part I: Dynamic models,” IEEE Trans. Aerosp. Electron. Syst., vol. 39, pp. 1333–1364, Oct. 2003.

[27] O. Hlinka, F. Hlawatsch, and P. M. Djuric, “Consensus-based distributed particle filtering with distributed proposal adaptation,” IEEE Trans. Signal Process., vol. 62, pp. 3029–3041, Jun. 2014.

[28] V. Savic, H. Wymeersch, and S. Zazo, “Belief consensus algorithms for fast distributed target tracking in wireless sensor networks,” Signal Processing, vol. 95, pp. 149–160, 2014.

[29] C. Lindberg, L. S. Muppirisetty, K.-M. Dahlén, V. Savic, and H. Wymeersch, “MAC delay in belief consensus for distributed tracking,” in Proc. IEEE WPNC-13, Dresden, Germany, Mar. 2013.

[30] R. Fletcher, Practical Methods of Optimization. New York, NY: Wiley, 1987.

[31] A. Doucet, N. De Freitas, and N. Gordon, Sequential Monte Carlo Methods in Practice. New York, NY: Springer, 2001.

[32] H. Lim and C. Kim, “Multicast tree construction and flooding in wireless ad hoc networks,” in Proc. MSWIM ’00, New York, NY, pp. 61–68, Aug. 2000.

[33] S. Thrun, W. Burgard, and D. Fox, Probabilistic Robotics. Cambridge, MA: MIT Press, 2005.

[34] A. T. Ihler, J. W. Fisher, R. L. Moses, and A. S. Willsky, “Nonparametric belief propagation for self-localization of sensor networks,” IEEE J. Sel. Areas Commun., vol. 23, pp. 809–819, Apr. 2005.

[35] J. Lien, J. Ferner, W. Srichavengsup, H. Wymeersch, and M. Z. Win, “A comparison of parametric and sample-based message representation in cooperative localization.” Int. J. Navig. Observ., 2012.

[36] B. Ristic, S. Arulampalam, and N. Gordon, Beyond the Kalman Filter: Particle Filters for Tracking Applications. Norwood, MA: Artech House, 2004.

[37] G. E. Garcia, L. S. Muppirisetty, E. M. Schiller, and H. Wymeersch, “On the trade-off between accuracy and delay in cooperative UWB localization: Performance bounds and scaling laws,” IEEE Trans. Wireless Commun., vol. 13, pp. 4574–4585, Aug. 2014.

[38] R. Douc and O. Cappe, “Comparison of resampling schemes for particle filtering,” in Proc. ISPA-05, Zagreb, Croatia, pp. 64–69, Sep. 2005.

[39] B. W. Silverman and P. J. Green, Density Estimation for Statistics and Data Analysis. London, UK: Chapman and Hall, 1986.

[40] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert, “Constrained model predictive control: Stability and optimality,” Automatica, vol. 36, pp. 789–814, Jun. 2000.

[41] D. L. Alspach and H. W. Sorenson, “Nonlinear Bayesian estimation using Gaussian sum approximations,” IEEE Trans. Autom. Control, vol. 17, pp. 439–448, Aug. 1972.

[42] I. Arasaratnam and S. Haykin, “Cubature Kalman filters,” IEEE Trans. Autom. Control, vol. 54, pp. 1254–1269, Jun. 2009.

[43] Y. Deville and A. Deville, “Exact and approximate quantum independent component analysis for qubit uncoupling.” in Proc. IVA/ICA ’12, Tel Aviv, Israel, pp. 58–65, Mar. 2012.