TECHNISCHE UNIVERSITÄT WIEN
Vienna University of Technology

Dissertation

Navigation and Tracking in Networks: Distributed Algorithms for Cooperative Estimation and Information-Seeking Control

Ausgeführt zum Zwecke der Erlangung des akademischen Grades eines Doktors der technischen Wissenschaften

unter der Leitung von
Ao. Univ.-Prof. Dr. Franz Hlawatsch
Institute of Telecommunications

von
Dipl.-Ing. Florian Meyer
Bernardgasse 4/11, 1070 Wien, Österreich

Wien, im März 2015 _____

Die Begutachtung dieser Arbeit erfolgte durch:

1. **Ao. Univ.-Prof. Dr. Franz Hlawatsch**
   Institute of Telecommunications
   Technische Universität Wien

2. **Prof. Dr. Henk Wymeersch**
   Department of Signals and Systems
   Chalmers University of Technology

Acknowledgements

I would like to gratefully and sincerely thank Prof. Franz Hlawatsch for his guidance, understanding, and patience.

Furthermore, I would like to thank Prof. Henk Wymeersch for his valuable contributions that significantly improved the quality of my work.

I would like to express my special appreciation to all colleagues and cooperation partners, for many helpful and important discussions and interesting joint work.

I am also very grateful to all my family members for their constant inspiration and encouragement. Finally, I want to thank Monika, for all her love and support.

Abstract

Navigating unmanned autonomous vehicles (UAVs) and tracking objects are key prerequisites for an effective and safe operation of UAVs in real-world scenarios. To be able to perform complicated tasks autonomously, UAVs are often organized in networked teams. Due to the decentralized structure of the network, the potentially high mobility of the UAVs, and the high accuracy and robustness required for a fully autonomous operation, navigation and tracking in such *mobile agent networks* are difficult tasks. In large agent networks, centralized algorithms for solving these tasks are impractical since they are typically not scalable and not robust to agent failure. Therefore, to leverage the full potential of agent networks, there is a need for efficient distributed (decentralized) algorithms for navigation and tracking. This thesis presents the following contributions to the art of distributed navigation and tracking.

A powerful technique for cooperative navigation in agent networks is nonparametric belief propagation (NBP) message passing. NBP-based cooperative navigation is highly accurate and fully distributed, but it suffers from a high computational complexity and significant communication requirements. In this thesis, we propose a *dimension-augmented* reformulation of belief propagation (BP) message passing. This reformulation allows the application of an arbitrary technique for sequential Bayesian estimation (e.g., extended Kalman filter, sigma point filter, cubature Kalman filter, or belief condensation filter) to BP message passing. We use dimension-augmented BP to derive a new improved NBP algorithm. This algorithm differs from the conventional NBP algorithm in that it employs an efficient scheme for particle-based message multiplication whose complexity scales only linearly (rather than quadratically) with the number of particles. In addition, we use dimension-augmented BP to develop the *sigma point BP* (SPBP) message passing scheme for cooperative navigation. SPBP is a new low-complexity approximation of BP that extends the sigma point filter (aka unscented Kalman filter) to cooperative estimation problems. SPBP is characterized by very low communication requirements, since only a mean vector and a covariance matrix are communicated between neighboring agents. Our simulation results show that for cooperative navigation, SPBP can outperform conventional NBP while requiring significantly less computation and communication.

As a second contribution, we extend BP-based cooperative navigation to the case that some agents in the network are noncooperative in that they do not communicate and perform computations. For this problem of *cooperative simultaneous navigation and tracking* (CoSNAT), we develop a particle-based BP message passing algorithm. This algorithm is, to the best of the author's knowledge, the first method for CoSNAT in a fully dynamic setting. A key feature of the

proposed CoSNAT algorithm is a bidirectional probabilistic information transfer between the navigation and tracking stages, which allows uncertainties in one stage to be taken into account by the other stage and thereby improves the performance of both stages. The algorithm is fully distributed, i.e., communication is only performed between neighboring agents in the network and no complicated communication protocol is required. Simulation results demonstrate significant improvements in navigation and tracking performance compared to separate cooperative navigation and distributed tracking.

Finally, we present a distributed information-seeking control scheme that aims to move the agents in such a way that their measurements are maximally informative about the parameters (states) to be estimated. For information-seeking control, we define a global (holistic) objective function as the negative joint 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 optimization reduces to the evaluation of local gradients at each agent, which is performed by using Monte Carlo integration. The local gradients are based on particle representations of marginal posterior distributions that are provided by the estimation stage and a distributed calculation of the joint (networkwide) likelihood function. Simulation results demonstrate intelligent behavior of the agents and excellent estimation performance for cooperative navigation and for CoSNAT. In a cooperative navigation scenario with only one anchor present, mobile agents can localize themselves after a short time with an accuracy that is higher than the accuracy of the performed distance measurements.

Abbreviations

2D Two-Dimensional

ARMSE Average Root-Mean-Square Error

BP Belief Propagation

CA Cooperative Agent

CoSNAT Cooperative Simultaneous Navigation and Tracking

GLS Global Likelihood Function

GPS Global Positioning System

IMU Inertial Measurement Unit

MAP Maximum-a-Posteriori

MINT Multipath-Assisted Indoor Navigation and Tracking

MMSE Minimum Mean-Square Error

NBP Nonparametric Belief Propagation

NLOS Non-Line-of-Sight

PDF Probability Density Function

PR Particle Representation

RSS Received Signal Strength

SLAM Simultaneous Localization and Mapping

SLAT Simultaneous Localization and Tracking

SPAWN Sum-Product Algorithm Over a Wireless Network

SPBP Sigma Point Belief Propagation

UAV Unmanned Autonomous Vehicle

UWB Ultra Wideband

Contents

Chapter 1

Introduction

Navigation and *tracking* refer to the determination of position and possibly direction and motion parameters of an object such as a craft or vehicle. While in navigational techniques the navigator's position relative to known landmarks or patterns is determined, tracking refers to inferring the position of an external and moving object or phenomenon over time relative to the position of the sensing device or a global frame of reference [Bar-Shalom et al., 2002]. Navigating unmanned autonomous vehicles (UAVs) and tracking objects are key prerequisites for an effective and safe operation of UAVs in real-world scenarios. Promising application scenarios include autonomous driving, aerial sensor networks, and space and underwater robotics. To be able to perform challenging tasks autonomously, unmanned vehicles are often organized in networked teams and even swarms [Zachery et al., 2011, Stilwell and Bishop, 2000]. This is supported by the fact that hardware and software for unmanned vehicles are becoming inexpensive and easily available in the open-source community [Ross, 2014]. Due to the decentralized structure of the network, the potentially high mobility of the UAVs, and the high accuracy and robustness required for a fully autonomous operation, navigation and tracking in such *mobile agent networks* are difficult tasks offering many open research problems.

1.1 Technologies for Navigation and Tracking

For navigation, global navigation satellite systems (GNSS) such as the global positioning system (GPS) [Kaplan and Hegarty, 2005, Dardari et al., 2012] enable the determination of the position of persons or vehicles using mobile receivers. The obtained localization accuracy and robustness are acceptable for many applications, but can be compromised by multipath components or by the absence of visible satellites [Kaplan and Hegarty, 2005, Dardari et al., 2012]. In addition, GPS is not available indoors [Mazuelas et al., 2009, Meissner, 2014]. For more accurate and robust navigation information, measurements of inertial measurement units (IMUs) or of odometers can be integrated and fused with the GNSS information [Groves, 2008]. Promising approaches to indoor navigation include ranging based on received signal strength (RSS) [Mazuelas et al., 2009] or ultra wideband (UWB) [Dardari et al., 2012, Wymeersch et al., 2009] measurements. While typically multipath components and non-line-of-sight

(NLOS) measurements may compromise accuracy and robustness in UWB-based localization, a possible solution is provided by the emerging field of multipath-assisted indoor navigation and tracking (MINT) [Meissner, 2014]. MINT incorporates floorplan information and makes use of the additional localization information provided by NLOS measurements and multipath components to achieve higher navigation accuracy and robustness.

In tracking, by using a remote sensing device such as radar [Skolnik, 2008], sonar [Waite, 2001], or a camera system [Taj and Cavallaro, 2011], the positions of moving objects over time are determined. Typically, the position of the sensing device is perfectly known if the device has a fixed position or is irrelevant if the external object or phenomenon is tracked relative to the position of the sensing device itself [Bar-Shalom et al., 2002]. Tracking of multiple moving objects is often complicated by data association uncertainty [Bar-Shalom et al., 2009] and by the fact that the number of objects to be tracked is not known [Mahler, 2007]. Data association uncertainty occurs when the origin of the obtained measurements is uncertain, that is, the measurements are not necessarily related to the object or phenomenon of interest, and gets even more challenging when multiple targets are present and the measurement-to-target association is unknown. Recently, large-scale wireless sensor networks employing seismic or acoustic sensors have been developed for tracking objects in the environment [Li et al., 2002]. A key challenge in tracking using wireless sensor networks is the dissemination of data in the network [Hlinka et al., 2013].

1.2 Motivation and Problem Formulation

The individual agents in an agent network are generally equipped with sensors, wireless communication interfaces, a processing unit, and actuators, all together forming a cyber-physical system [Kim and Kumar, 2012] with a tight coupling between sensing, computing, communication, and control. Since the resources of a single agent are limited, complicated tasks can be performed only if multiple agents form a network and cooperate with each other. In large agent networks, centralized algorithms are impractical since they are typically not scalable and are not robust to agent failure. Therefore, to leverage the full potential of agent networks, there is a need for efficient distributed (decentralized) algorithms that do not rely on a fusion center. Due to the distributed nature of these algorithms, every agent contributes to a global task by communicating only with neighboring agents in the network.

Key tasks in decentralized, mobile agent networks are navigation of cooperative agents [Shen et al., 2012, Wymeersch et al., 2009] and tracking of external (noncooperative) objects or phenomena generally referred to as *targets* [Liu et al., 2007, Hlinka et al., 2013]. Recent research on navigation and tracking algorithms in mobile agent networks [Shima and Rasmussen, 2009, Dario et al., 2005, Corke et al., 2010, Bullo et al., 2009, Ko et al., 2010, Hlinka et al., 2013, Zhao and Nehorai, 2007, Nayak and Stojmenović, 2010, Aghajan and Cavallaro, 2009] has been motivated by location-aware applications including coordination of unmanned aerial [Shima and Rasmussen, 2009] and underwater [Dario et al., 2005] vehicles, environmen-

tal and agricultural monitoring [Corke et al., 2010], robotics [Bullo et al., 2009], health-care monitoring [Ko et al., 2010], target tracking [Hlinka et al., 2013], pollution source localization [Zhao and Nehorai, 2007], chemical plume tracking [Nayak and Stojmenović, 2010], and surveillance [Aghajan and Cavallaro, 2009].

In cooperative navigation, each cooperative agent (CA) measures quantities related to the position of neighboring CAs relative to its own position (e.g., involving distances or angles). By cooperating with other CAs and, in particular, by exchanging position-related information, each CA is able to estimate its own position. In distributed tracking, the CA measurements are related to the states of targets to be tracked. At each CA, estimates of the target states are cooperatively calculated from all CA measurements in a distributed way. While a large variety of distributed tracking algorithms are available (see [Hlinka et al., 2013] and references therein), existing practical implementations of cooperative navigation algorithms for mobile agent networks are limited to specific measurement models [Sathyan and Hedley, 2013]. Traditionally, cooperative navigation and distributed tracking are formulated as two separate *estimation* tasks, where estimation involves the quantification, fusion, and dissemination of information. However, these two tasks are strongly related since, ideally, a CA needs to know its position to be able to contribute to the tracking process. In addition, a smart *control* (moving CAs to different positions or adapting sensor characteristics) can be expected to lead to more informative measurements and, consequently, an improved estimation performance for both navigation and tracking. Since the estimation tasks involve the measurements of all CAs, the smart controller moving and adapting the individual CAs needs to be derived from a global objective function and implemented in a distributed way to be effective.

In this thesis, we address these issues and provide the following contributions (see Section 1.4 for a more detailed outline):

- A new class of distributed estimation algorithms for cooperative navigation is developed.
- These cooperative navigation algorithms are extended to include distributed tracking of noncooperative targets.
- A distributed, globally optimum controller that moves CAs in a way that is favorable for the joint navigation-and-tracking task is formulated.

1.3 State of the Art

As a relevant background and for further reference, we will summarize the state-of-the-art in distributed estimation and information-seeking control for navigation and tracking in networks.

1.3.1 Distributed Tracking Based on Sequential Estimation and Fusion Techniques

In distributed tracking, the CAs obtain local measurements with respect to external objects (targets) with time-varying positions, and estimate the states of these objects using all the measure-

ments that are available in the network while communicating only with neighboring CAs. For this task, typically, sequential Bayesian estimation is combined with a global fusion method that requires only local communication, such as consensus [Olfati-Saber et al., 2007] or gossip [Dimakis et al., 2010].

One practical method for sequential Bayesian estimation is the particle filter [Gordon et al., 1993, Ristic et al., 2004, Arulampalam et al., 2002], which uses particles to represent the probability density functions (PDFs) involved in the Bayesian recursion. Distributed particle filters (see [Hlinka et al., 2013] and references therein) are attractive since they are suited to arbitrary nonlinear and non-Gaussian systems. Examples include [Farahmand et al., 2011] and [Hlinka et al., 2014]. In [Farahmand et al., 2011], consensus algorithms are used to calculate global weights—reflecting the measurements of all CAs—at each CA. In [Hlinka et al., 2014], the *likelihood consensus* scheme is proposed for a distributed approximate computation of the global (all-CAs) likelihood function, which is used at each CA for calculating global weights.

An alternative to the particle filter is the sigma point filter, also known as unscented Kalman filter, which is a sequential Bayesian estimator for nonlinear systems that outperforms the extended Kalman filter [Anderson and Moore, 2005, Kay, 1993, Haykin, 2001] while being typically less complex than the particle filter [Julier and Uhlmann, 1997, Wan and van der Merwe, 2001]. Distributed implementations of the sigma point filter include [Mohammadi and Asif, 2011, Li and Jia, 2012].

1.3.2 Cooperative Navigation Using Message Passing Algorithms

In cooperative navigation, mobile CAs cooperate by performing measurements relative to other CAs and by exchanging position-related information, such that each CA is able to estimate its own local state (including the position) with improved accuracy. Here, the dimensionality of the total (all-CAs) state grows with the network size. In addition, the factorization of the joint posterior PDF is more complicated than in sequential estimation problems and therefore often described by a graphical model [Jordan, 1999, Wainwright and Jordan, 2008, Loeliger, 2004]. For these general—typically loopy—factor structures, an iterative message passing scheme can be used to (approximately) perform the marginalizations required for Bayesian inference [Kschischang et al., 2001]. Under mild assumptions, the factor graph [Loeliger, 2004] describing the factorization of the joint posterior PDF matches the network topology, and hence a fully distributed implementation of the iterative message passing scheme can easily be obtained [Wymeersch et al., 2009]. In addition, the complexity of iterative message passing schemes scales only linearly with the network size [Wymeersch et al., 2009].

The most widely used message passing scheme for navigation and tracking applications is belief propagation (BP). BP yields the true marginal posterior PDF if the factor graph is a tree. Sequential Bayesian estimation—used, e.g., in distributed tracking—is a special case of BP where the joint posterior PDF has a simple "sequential" factor structure corresponding to a tree-structured factor graph. Gaussian BP [Weiss and Freeman, 2001] and nonparametric BP (NBP) [Ihler et al., 2005] are computationally feasible variants of BP that extend the Kalman

filter [Anderson and Moore, 2005, Kay, 1993, Haykin, 2001] and the particle filter [Gordon et al., 1993, Ristic et al., 2004, Arulampalam et al., 2002], respectively to general factor structures. Gaussian BP assumes a linear, Gaussian system and uses Gaussian message representations, whereas NBP is suited to nonlinear, non-Gaussian systems due to its use of particle representations. Since in cooperative navigation the measurement models of the CAs are typically nonlinear, NBP is preferred over Gaussian BP. In particular, NBP for the localization of static CAs is proposed in [Ihler et al., 2005]. In [Wymeersch et al., 2009], a distributed BP message passing algorithm for cooperative navigation (cooperative localization of mobile CAs) called *sum-product algorithm over a wireless network* (SPAWN) is proposed. Furthermore, a nonparametric implementation of SPAWN and a low-complexity SPAWN scheme using a parametric message representation and censoring is considered for cooperative navigation in [Lien et al., 2012]. In [Savic and Zazo, 2012], a variant of NBP that uses a Gaussian approximation for the beliefs to reduce inter-CA communication is proposed. Finally, [Caceres et al., 2011] introduces a SPAWN-based cooperative navigation algorithm that fuses information from satellites and terrestrial wireless systems and employs a three-dimensional parametric message representation for reduced computation and communication.

As an alternative to BP, a message passing algorithm based on the mean field approximation is presented in [Pedersen et al., 2011] for cooperative localization of static CAs. Finally, a low-complexity and highly approximate method for cooperative navigation that does not employ a message passing scheme is presented in [Sathyan and Hedley, 2013]. This method is based on sequential Bayesian estimation and a linearized measurement equation.

1.3.3 Simultaneous Localization and Tracking (SLAT)

The framework of *simultaneous localization and tracking* (SLAT), introduced in [Taylor et al., 2006], constitutes a first step toward combining CA localization—however, not in a cooperative manner—and target tracking. In SLAT, static CAs simultaneously track a target and localize themselves in a not necessarily distributed (decentralized) manner. During runtime, SLAT algorithms use measurements of the distances between each CA and the target [Taylor et al., 2006], but not measurements of the distances between CAs. The SLAT problem is somewhat similar to the well-studied problem of *simultaneous localization and mapping* (SLAM) in robotics [Durrant-Whyte and Bailey, 2006]. In SLAT, in contrast to cooperative navigation, CAs are static and measurements of the distances between CAs are only used for initialization.

A centralized particle-based SLAT method using BP is proposed in [Savic and Wymeersch, 2013]. Distributed SLAT methods include a technique using sequential Bayesian estimation and communication via a junction tree [Funiak et al., 2006], iterative maximum likelihood methods [Kantas et al., 2012], and a method based on variational filtering [Teng et al., 2012].

1.3.4 Information-Seeking Control

The use of information measures for the control of the movement of a single CA or a network of CAs was introduced in [Burgard et al., 1997] and [Grocholsky, 2002], respectively. Here, the CAs' control objective is to maximize the joint information about a global state that is carried by the measurements of all CAs about a global state. Suitable measures of information include negative posterior entropy [Cover and Thomas, 2006], mutual information [Cover and Thomas, 2006], and scalar-valued functions of the Fisher information matrix [Kay, 1993]. In particular, the determinant, trace, and spectral norm of the Fisher information matrix are considered in [Morbidi and Mariottini, 2013], where the control objective is to maximize the information related to the positions of the CAs and of a target. The maximization of negative posterior entropy is considered in [Ryan et al., 2007, Hoffmann and Tomlin, 2010, Schwager et al., 2011, Julian et al., 2012, Atanasov et al., 2015]. In [Schwager et al., 2011], a central controller steers CAs with known positions along the gradient of negative posterior entropy with the goal of optimally sensing and estimating a static global state. A distributed solution for global state estimation is proposed in [Ryan et al., 2007, Hoffmann and Tomlin, 2010] based on a pairwise neighboring-CAs approximation of mutual information and in [Julian et al., 2012, Atanasov et al., 2015] by using a consensus algorithm.

1.4 Contribution and Outline

In this thesis, we propose computationally feasible distributed Bayesian estimation and information-seeking control algorithms for navigation and tracking in networks. In Chapters 2 and 3, estimation methods are presented while Chapter 4 focuses on information-seeking control. An outline of the thesis and a summary of our main contributions are provided in the following.

- The remainder of this chapter describes the system model that is used throughout this thesis.
- In **Chapter 2**, we first review BP message passing for cooperative navigation and its nonparametric implementation (i.e., NBP). This conventional implementation suffers from high computation and communication costs Therefore, for the case of two-dimensional (2D) position information and distance measurements between CAs, we propose a new hybrid nonparametric/parametric BP scheme. In this scheme, communication and computation costs are significantly reduced through the use of parametric representations of inter-CA messages (as introduced in [Lien et al., 2012], although the parameters are determined differently) and a simpler procedure to perform a particle-based message multiplication operation. Contrary to [Caceres et al., 2011], the introduced parametric message representation is 2D and distinguishes between unimodal, bimodal and multimodal beliefs.

  Furthermore, we propose a *dimension-augmented* reformulation of BP. This reformulation allows the systematic and straightforward application of an arbitrary sequential

Bayesian estimator (or Bayesian filter) to BP-based cooperative navigation. Examples of Bayesian filters that can be used for BP in this sense include the extended Kalman filter [Anderson and Moore, 2005, Kay, 1993, Haykin, 2001], the sigma point filter [Julier and Uhlmann, 1997, Wan and van der Merwe, 2001], the cubature Kalman filter [Arasaratnam and Haykin, 2009], and the belief condensation filter [Mazuelas et al., 2013]).

We then use the proposed reformulation of BP to develop a new nonparametric (particle-based) implementation of BP with reduced complexity. This implementation will be used as a basis for the particle-based *cooperative simultaneous navigation and tracking* (CoSNAT) algorithm in Chapter 3. The main advantage of the proposed NBP algorithm is that its complexity scales only linearly—rather than quadratically as in conventional NBP [Ihler et al., 2005, Lien et al., 2012]—with the number of particles.

Finally, the presented dimension-augmented BP scheme is used to develop the *sigma point BP* (SPBP) message passing algorithm for cooperative navigation. SPBP is a new low-complexity approximation of BP that extends the sigma point filter—also known as unscented Kalman filter—to general factor structures. We demonstrate that the performance of SPBP can be similar to or even better than that of conventional NBP, at a far lower computation and communication cost. Indeed, SPBP is well suited to decentralized cooperative navigation because only a mean vector and a covariance matrix have to be communicated between neighboring sensors. Besides the cooperative navigation application considered here, we expect that the dimension-augmented reformulation of BP as well as the new implementations based on sigma points and particles will also be useful for other inference problems in agent networks. Simulation results demonstrate significant advantages of SPBP and of the new particle-based BP scheme over conventional NBP regarding performance, complexity, and communication cost.

The content of this chapter has been previously presented in [Meyer et al., 2014a, Meyer et al., 2014b].

- In **Chapter 3**, we introduce the framework of CoSNAT. This framework provides, for the first time, a consistent combination of cooperative navigation and distributed tracking in decentralized mobile agent networks. In CoSNAT, single or multiple targets are tracked by the mobile CAs while the CAs simultaneously localize themselves, using pairwise measurements between CAs and targets as well as between CAs. Thus, CoSNAT is different from SLAT in that it allows for CA mobility and uses pairwise measurements between the CAs also during runtime. We assume that the number of targets is known and the targets can be identified by the CAs. Even with this assumption, the fact that the CAs are mobile and their states (including their positions) are unknown causes the multi-target tracking problem to be much more challenging than in the setting where the CAs are static with known states. This is because the posterior distributions of the states of the individual targets and CAs are coupled through pairwise measurements, and thus all CA and target states should be estimated *jointly*. This joint estimation is performed

quite naturally by the proposed framework and algorithm due to the formulation of the entire estimation problem using a factor graph and the use of BP message passing, which transfers probabilistic information between all parts of the joint estimation problem.

First, distributed target tracking is briefly reviewed. Then, the BP message passing scheme for cooperative navigation reviewed at the beginning of Chapter 2 is extended to noncooperative targets, and a particle-based implementation of the resulting CoSNAT message passing scheme is developed. This algorithm is, to the best of the author's knowledge, the first method for CoSNAT in a fully dynamic setting. A key feature of the proposed CoSNAT algorithm is a "turbo-like" [Wymeersch, 2007] bidirectional probabilistic information transfer between the navigation and tracking stages, which allows uncertainties in one stage to be taken into account by the other stage and thereby improves the performance of both stages. A fundamental difficulty that has to be surmounted is the fact that, due to the noncooperative nature of the targets, certain messages needed to calculate the target beliefs are not available at the CAs. The proposed algorithm solves this problem by using a consensus scheme over the particle weights [Farahmand et al., 2011] for a distributed calculation of these lacking messages. The resulting coherent combination of BP and consensus may also be useful in other distributed inference problems involving noncooperative CAs. Simulation results demonstrate significant improvements in navigation and tracking performance compared to separate cooperative navigation and distributed tracking.

The content of this chapter has been previously presented in [Meyer et al., 2012, Meyer et al., 2013b, Meyer et al., 2014b].

- In **Chapter 4**, we extend the CoSNAT estimation framework developed in Chapter 3 to include cooperative information-seeking control. The goal of the proposed controller is to move the agents in such a way that their measurements are maximally informative about the states to be estimated. The resulting estimation-and-control framework and method are suited to nonlinear and non-Gaussian measurement and motion models, and they are distributed in that they require only communication with neighboring CAs. In addition, they can cope with a changing network topology.

  For distributed control, we define a global (holistic) objective function as the negative joint posterior entropy of all states in the network at the next time step conditioned on all measurements in the network at the next time step. This objective function is then optimized jointly by all CAs via a gradient ascent, which reduces to the evaluation of local gradients at each CA. These local evaluations are performed by using Monte Carlo integration based on the particle representations of marginal posterior PDFs that are provided by the estimation stage and a distributed calculation of the joint (networkwide) likelihood function.

  Our method advances beyond [Ryan et al., 2007, Hoffmann and Tomlin, 2010, Schwager et al., 2011, Julian et al., 2012, Atanasov et al., 2015] in the following respects: (i) it

constitutes a more general information-maximizing control framework based on BP for estimation problems involving multiple time-varying states; (ii) it includes estimation of the local (controlled) states of the CAs in addition to the target states, thus enabling its use in a wider range of applications including cooperative localization and CoSNAT. In particular, the methods proposed in [Ryan et al., 2007, Hoffmann and Tomlin, 2010, Schwager et al., 2011, Julian et al., 2012, Atanasov et al., 2015] do not use BP, do not allow for multiple time-varying states, and do not include estimation of local (controlled) states. Compared to the information-seeking controllers proposed in [Ryan et al., 2007, Hoffmann and Tomlin, 2010, Julian et al., 2012, Atanasov et al., 2015], where maximization of the negative posterior entropy reduces to maximization of the mutual information between measurements and states, the proposed controller includes an additional term that arises because the posterior entropy involves also the local (controlled) states of the CAs. Our simulation results demonstrate intelligent behavior of the CAs and excellent estimation performance for cooperative navigation and CoSNAT. In a cooperative navigation scenario with only one anchor present, mobile CAs can localize themselves after a short time with an accuracy that is higher than the accuracy of the performed distance measurements.

The content of this chapter has been previously presented in [Meyer et al., 2014c, Meyer et al., 2015].

- In **Chapter 5**, we summarize our contributions and suggest possible directions for future research.

1.5 System Model

We consider a decentralized network of mobile agents $k \in \mathcal{A}$, as illustrated in Fig. 1.1. The set of all agents, $\mathcal{A} \subseteq \mathbb{N}$, consists of the set of CAs, $\mathcal{S} \subseteq \mathcal{A}$, and the set of targets, $\mathcal{T} = \mathcal{A} \setminus \mathcal{S}$. (We will use the indices $k \in \mathcal{A}$, $l \in \mathcal{S}$, and $m \in \mathcal{T}$ to denote a generic agent, a CA, and a target, respectively.) The numbers of CAs and targets are assumed known. Targets are noncooperative in that they do not communicate, do not perform computations, and do not actively perform any measurements.

1.5.1 Agent States and Agent Dynamics

The *state* of agent $k \in \mathcal{A}$ at time $n \in \{0, 1, \ldots\}$, denoted $\mathbf{x}_{k,n}$, consists of the current position and, possibly, other parameters such as velocity, acceleration, bearing, and angular velocity [Li and Jilkov, 2003]. The states evolve according to

$$\mathbf{x}_{k,n} = \begin{cases} g_k(\mathbf{x}_{k,n-1}, \mathbf{u}_{k,n}, \mathbf{q}_{k,n}), & k \in \mathcal{S} \\ g_k(\mathbf{x}_{k,n-1}, \mathbf{q}_{k,n}), & k \in \mathcal{T}, \end{cases} \qquad n = 1, 2, \ldots, \tag{1.1}$$

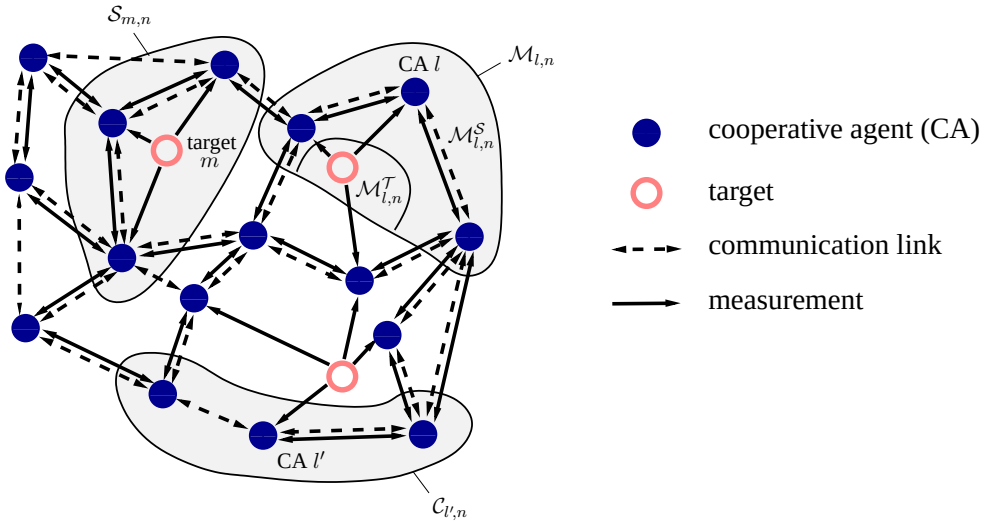


Figure 1.1: Agent network with CAs and targets. Also shown are the sets $\mathcal{M}_{l,n}$, $\mathcal{M}^{\mathcal{S}}_{l,n}$, and $\mathcal{M}^{\mathcal{T}}_{l,n}$ for a specific CA l, the set $\mathcal{C}_{l',n}$ for a specific CA l', and the set $\mathcal{S}_{m,n}$ for a specific target m.

where $g_k(\cdot)$ is a possibly nonlinear function, $\mathbf{q}_{k,n}$ is process (driving) noise, which has a PDF $f(\mathbf{q}_{k,n})$ and is independent of $\mathbf{x}_{k,n-1}$ and also independent across n and k, and $\mathbf{u}_{k,n} \in \mathcal{U}_k$ is a deterministic control vector that controls the kth CA. Since $\mathbf{u}_{k,n}$ is deterministic [Doucet et al., 2001], it is either completely unknown (before it is determined) or perfectly known (after being determined). 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 process noise $\mathbf{q}_{m,n}$. For the derivation of the controller in Chapter 4, we assume that for $l \in \mathcal{S}$, $g_l(\mathbf{x}_{l,n-1}, \mathbf{u}_{l,n}, \mathbf{q}_{l,n})$ is bijective with respect to $\mathbf{x}_{l,n-1}$ and differentiable with respect to $\mathbf{u}_{l,n}$. The statistical relation between $\mathbf{x}_{k,n}$ and $\mathbf{x}_{k,n-1}$ as defined by (1.1) and by the PDF $f(\mathbf{q}_{l,n})$ can also be described by the *state-transition PDF* $f(\mathbf{x}_{l,n} \big| \mathbf{x}_{l,n-1}; \mathbf{u}_{l,n})$ for $l \in \mathcal{S}$ and $f(\mathbf{x}_{m,n} \big| \mathbf{x}_{m,n-1})$ for $m \in \mathcal{T}$.

1.5.2 Network Topology

The communication and measurement topologies of the network are described by sets $\mathcal{C}_{l,n}$ and $\mathcal{M}_{l,n}$ as follows:

- CA $l \in \mathcal{S}$ is able to communicate with CA l' if $l' \in \mathcal{C}_{l,n} \subseteq \mathcal{S} \setminus \{l\}$. Communication is always symmetric, i.e., $l' \in \mathcal{C}_{l,n}$ implies $l \in \mathcal{C}_{l',n}$, and the communication graph of the network formed by the CAs is assumed to be a connected graph.

- CA $l \in \mathcal{S}$ acquires a measurement $\mathbf{y}_{l,k;n}$ relative to agent (CA or target) $k \in \mathcal{A}$ if $k \in \mathcal{M}_{l,n} \subseteq \mathcal{A} \setminus \{l\}$.

- We also define $\mathcal{M}^{\mathcal{S}}_{l,n} \triangleq \mathcal{M}_{l,n} \cap \mathcal{S}$ and $\mathcal{M}^{\mathcal{T}}_{l,n} \triangleq \mathcal{M}_{l,n} \cap \mathcal{T}$, i.e., the subsets of $\mathcal{M}_{l,n}$ containing only CAs and only targets, respectively.

- In addition, we introduce $\mathcal{S}_{m,n} \triangleq \{l \in \mathcal{S} \,|\, m \in \mathcal{M}_{l,n}^{\mathcal{T}}\}$, i.e., the set of CAs that acquire measurements relative to target m. Note that $m \in \mathcal{M}_{l,n}^{\mathcal{T}}$ if and only if $l \in \mathcal{S}_{m,n}$.

- We assume that $\mathcal{M}_{l,n}^{\mathcal{S}} \subseteq \mathcal{C}_{l,n}$, i.e., if CA l acquires a measurement relative to CA l', it is able to communicate with CA l'. The sets $\mathcal{C}_{l,n}$, $\mathcal{M}_{l,n}$, $\mathcal{M}_{l,n}^{\mathcal{S}}$, $\mathcal{M}_{l,n}^{\mathcal{T}}$, and $\mathcal{S}_{m,n}$ may be time-dependent.

An example of communication and measurement topologies is given in Fig. 1.1.

1.5.3 Sensor Measurements

We consider "pairwise" measurements $\mathbf{y}_{l,k;n}$ that depend on the states $\mathbf{x}_{l,n}$, $l \in \mathcal{S}$ and $\mathbf{x}_{k,n}$, $k \in \mathcal{M}_{l,n}$ according to

$$\mathbf{y}_{l,k;n} = d_k(\mathbf{x}_{l,n}, \mathbf{x}_{k,n}, \mathbf{v}_{l,k;n})\,, \qquad l \in \mathcal{S},\ k \in \mathcal{M}_{l,n}, \quad n = 1, 2, \ldots\,. \tag{1.2}$$

Here, $d_k(\cdot)$ is a possibly nonlinear function, and $\mathbf{v}_{l,k;n}$ is measurement noise, which has a PDF $f(\mathbf{v}_{l,k;n})$ and is independent of $\mathbf{x}_{l,n}$ and $\mathbf{x}_{k,n}$ and also independent across l, k, and n. An example is the scalar "noisy distance" measurement

$$y_{l,k;n} = \|\tilde{\mathbf{x}}_{l,n} - \tilde{\mathbf{x}}_{k,n}\| + v_{l,k;n}\,, \qquad l \in \mathcal{S},\ k \in \mathcal{M}_{l,n}, \quad n = 1, 2, \ldots\,, \tag{1.3}$$

where $\tilde{\mathbf{x}}_{k,n}$ represents the position of agent k (this is part of the state $\mathbf{x}_{k,n}$). Note that measurements $\mathbf{y}_{l,k;n}$ exist between a CA $l \in \mathcal{S}$ and another CA $k \in \mathcal{M}_{l,n}^{\mathcal{S}}$, and between a CA $l \in \mathcal{S}$ and a target $k \in \mathcal{M}_{l,n}^{\mathcal{T}}$. The statistical dependence of $\mathbf{y}_{l,k;n}$ on $\mathbf{x}_{l,n}$ and $\mathbf{x}_{k,n}$ as defined by (1.2) and by the PDF $f(\mathbf{v}_{l,k;n})$ is described by the *local likelihood function* $f(\mathbf{y}_{l,k;n}|\mathbf{x}_{l,n}, \mathbf{x}_{k,n})$. We assume that targets can be identified by the CAs, i.e., target-to-measurement assignments are known. This requires a certain degree of coordination: for example, if RSS measurements are employed, the targets have to use different frequency bands or different time slots, which have to be known to the CAs; if localization using UWB radios is employed, the identities of the UWB radios mounted on the targets have to be known to the CAs.

1.5.4 Assumptions

We will make the following commonly used assumptions, which are reasonable in many practical scenarios [Wymeersch et al., 2009]. Hereafter, we denote by

$$\mathbf{u}_n \triangleq [\mathbf{u}_{l,n}]_{l\in\mathcal{S}}\,, \quad \mathbf{x}_n \triangleq [\mathbf{x}_{k,n}]_{k\in\mathcal{A}}\,, \quad \mathbf{y}_n \triangleq [\mathbf{y}_{l,k;n}]_{l\in\mathcal{S},\, k\in\mathcal{M}_{l,n}} \tag{1.4}$$

the sets of all control vectors, states, and measurements, respectively at time n. Furthermore, we define

$$\mathbf{u}_{1:n} \triangleq [\mathbf{u}_1^{\mathrm{T}}, \ldots, \mathbf{u}_n^{\mathrm{T}}]^{\mathrm{T}}, \qquad \mathbf{x}_{1:n} \triangleq [\mathbf{x}_1^{\mathrm{T}}, \ldots, \mathbf{x}_n^{\mathrm{T}}]^{\mathrm{T}}, \qquad \mathbf{y}_{1:n} \triangleq [\mathbf{y}_1^{\mathrm{T}}, \ldots, \mathbf{y}_n^{\mathrm{T}}]^{\mathrm{T}}.$$

(A1) All agent states are independent *a priori* at time $n=0$, i.e.,

$$f(\mathbf{x}_0) = \prod_{k\in\mathcal{A}} f(\mathbf{x}_{k,0}).$$

(A2) All agent states evolve according to a memoryless walk, i.e.,

$$f(\mathbf{x}_{1:n}; \mathbf{u}_{1:n}) = f(\mathbf{x}_0) \prod_{n'=1}^{n} f(\mathbf{x}_{n'}|\mathbf{x}_{n'-1}; \mathbf{u}_{n'}).$$

This is a consequence of our assumption that the process noises $\mathbf{q}_{k,n}$ in (1.1) are independent across time n.

(A3) The state transitions at the various agents $k \in \mathcal{A}$ are independent, i.e.,

$$f(\mathbf{x}_n|\mathbf{x}_{n-1}; \mathbf{u}_n) = \prod_{m\in\mathcal{T}} f(\mathbf{x}_{m,n}|\mathbf{x}_{m,n-1}) \prod_{l\in\mathcal{S}} f(\mathbf{x}_{l,n}|\mathbf{x}_{l,n-1}; \mathbf{u}_{l,n}).$$

This is a consequence of our assumption that the process noises $\mathbf{q}_{k,n}$ in (1.1) are independent between the agents $k \in \mathcal{A}$.

(A4) The current measurements $\mathbf{y}_n$ are conditionally independent, given the current states $\mathbf{x}_n$, of all other states and of all past and future measurements, i.e.,

$$f(\mathbf{y}_n|\mathbf{x}_{0:\infty}, \mathbf{y}_{1:n-1}, \mathbf{y}_{n+1:\infty}) = f(\mathbf{y}_n|\mathbf{x}_n).$$

(A5) The current states $\mathbf{x}_n$ are conditionally independent, given the previous states $\mathbf{x}_{n-1}$, of all past measurements, i.e.,

$$f(\mathbf{x}_n|\mathbf{x}_{n-1}, \mathbf{y}_{1:n-1}; \mathbf{u}_n) = f(\mathbf{x}_n|\mathbf{x}_{n-1}; \mathbf{u}_n).$$

(A6) The measurement noises $\mathbf{v}_{l,k;n}$ and $\mathbf{v}_{l',k';n'}$ in (1.2) are conditionally independent unless $(l,k,n) = (l',k',n')$, and each measurement $\mathbf{y}_{l,k;n}$ depends only on the states $\mathbf{x}_{l,n}$ and $\mathbf{x}_{k,n}$. Together with (A4), this leads to the following factorization of the "total" (or "global") likelihood function:

$$f(\mathbf{y}_{1:n}|\mathbf{x}_{1:n}) = \prod_{n'=1}^{n} \prod_{l\in\mathcal{S}} \prod_{k\in\mathcal{M}_{l,n}} f(\mathbf{y}_{l,k;n'}|\mathbf{x}_{l,n'}, \mathbf{x}_{k,n'}).$$

In addition, we make the following assumptions. Each CA $l \in \mathcal{S}$ knows the functional forms of its own state-transition PDF and initial state PDF as well as of the state-transition PDFs and initial state PDFs of all targets, i.e., $f(\mathbf{x}_{k,n}|\mathbf{x}_{k,n-1})$ and $f(\mathbf{x}_{k,0})$ for $k \in \{l\} \cup \mathcal{T}$. Furthermore, all prior position and motion information is available in one global reference frame that is known to all CAs, all CAs can transmit in parallel, and the internal clocks of all CAs are synchronous

(see [Wu et al., 2011, Etzlinger et al., 2014, Meyer et al., 2013a, Etzlinger et al., 2013] for distributed clock synchronization algorithms).

Chapter 2

Cooperative Navigation

BP message passing [Jordan, 1999, Wainwright and Jordan, 2008] is a well-established technique for cooperative localization and navigation [Ihler et al., 2005, Wymeersch et al., 2009, Lien et al., 2012], and it is also the basis for the algorithms developed in this thesis. Therefore, this chapter starts with a review of this BP scheme and of its nonparametric implementation (i.e., NBP). Next, for the case of 2D position information and distance measurements between CAs, a new hybrid nonparametric/parametric BP scheme is proposed. In this scheme, communication and computation costs are significantly reduced through the use of parametric representations of inter-CA messages. Then, the dimension-augmented reformulation of BP, which allows the development of a new class of efficient algorithms for cooperative navigation, is proposed. This reformulation is used to derive two specific BP-based estimation algorithms. The first is a new reduced-complexity nonparametric implementation of BP. The second, SPBP, extends the sigma point filter to loopy BP and is characterized by very low computation and communication costs. The proposed dimension-augmented reformulation of BP as well as the resulting new algorithms based on particles and sigma points can be easily extended to more general system models; in particular, they are not limited to "pairwise" measurements. They are also suited to other decentralized cooperative inference problems besides cooperative navigation.

In this chapter and in the subsequent Chapter 3, we do not allow the CAs $l \in \mathcal{S}$ to be controlled. Therefore, we suppress the control variables $\mathbf{u}_{l,n}$ in the notation, i.e., we write $\mathbf{x}_{l,n} = g_l(\mathbf{x}_{l,n-1}, \mathbf{q}_{l,n})$, $l \in \mathcal{S}$ for the state evolution model and $f(\mathbf{x}_{l,n}|\mathbf{x}_{l,n-1})$ for the state transition PDF (cf. (1.1)).

2.1 Review: Cooperative Navigation Using Belief Propagation

In this section, the BP scheme for cooperative navigation [Wymeersch et al., 2009, Lien et al., 2012] and a possible particle-based implementation [Ihler et al., 2005, Lien et al., 2012] is reviewed. In Bayesian cooperative navigation, each CA $l \in \mathcal{S}$ estimates its own current state

$\mathbf{x}_{l,n}$ at time n, from the past and present pairwise measurements of all CAs,

$$\mathbf{y}_{1:n}^{\mathcal{S}} \triangleq \left[\mathbf{y}_{l_1,l_2;n'}\right]_{l_1\in\mathcal{S},\, l_2\in\mathcal{M}_{l_1,n'}^{\mathcal{S}},\, n'\in\{1,\ldots,n\}}.$$

In particular, the minimum mean-square error (MMSE) estimator and the maximum-a-posteriori (MAP) estimator of $\mathbf{x}_{l,n}$ are given by [Kay, 1993]

$$\hat{\mathbf{x}}_{l,n}^{\text{MMSE}} \triangleq \int \mathbf{x}_{l,n} f(\mathbf{x}_{l,n}|\mathbf{y}_{1:n}^{\mathcal{S}})\,\mathrm{d}\mathbf{x}_{l,n} \tag{2.1}$$

$$\hat{\mathbf{x}}_{l,n}^{\text{MAP}} \triangleq \underset{\mathbf{x}_{l,n}}{\arg\max}\; f(\mathbf{x}_{l,n}|\mathbf{y}_{1:n}^{\mathcal{S}})\,. \tag{2.2}$$

2.1.1 Cooperative Navigation Message Passing Scheme

By using Bayes' rule and assumptions (A1)–(A6), mentioned in Section 1.5, the joint posterior PDF of the set of all CA states, $\mathbf{x}_{1:n}^{\mathcal{S}} \triangleq [\mathbf{x}_{l,n'}]_{l\in\mathcal{S},\, n'\in\{1,\ldots,n\}}$, is obtained up to an irrelevant constant factor as

$$f(\mathbf{x}_{1:n}^{\mathcal{S}}|\mathbf{y}_{1:n}^{\mathcal{S}}) \propto \prod_{l\in\mathcal{S}} f(\mathbf{x}_{l,0}) \prod_{n'=1}^{n} f(\mathbf{x}_{l,n'}|\mathbf{x}_{l,n'-1}) \prod_{l'\in\mathcal{M}_{l,n'}^{\mathcal{S}}} f(\mathbf{y}_{l,l';n'}|\mathbf{x}_{l,n'},\mathbf{x}_{l',n'})\,. \tag{2.3}$$

CAs with an informative prior PDF

$$f(\mathbf{x}_{l,n}) = \int f(\mathbf{x}_{l,0}) \prod_{n'=1}^{n} \left[f(\mathbf{x}_{l,n'}|\mathbf{x}_{l,n'-1})\,\mathrm{d}\mathbf{x}_{l,n'-1}\right] \tag{2.4}$$

for all n are referred to as anchors.

Calculating the marginal posterior PDFs $f(\mathbf{x}_{l,n}|\mathbf{y}_{1:n}^{\mathcal{S}})$ involved in the MMSE or MAP estimator expressions (2.1) of (2.2) by direct marginalization of $f(\mathbf{x}_{1:n}^{\mathcal{S}}|\mathbf{y}_{1:n}^{\mathcal{S}})$ is infeasible because it involves integration in spaces whose dimension grows with time and network size. To reduce dimensionality a BP scheme can be used to take advantage of the temporal and spatial independence structure expressed by the factorization in (2.3). Explicit integration can be avoided by employing a particle-based implementation or using sigma points.

More specifically, approximate marginal posterior PDFs ("beliefs") $b(\mathbf{x}_{l,n}) \approx f(\mathbf{x}_{l,n}|\mathbf{y}_{1:n}^{\mathcal{S}})$, $l\in\mathcal{S}$ can be obtained by executing a distributed iterative BP message passing scheme [Wymeersch et al., 2009]. This is based on the factor graph [Loeliger, 2004] corresponding to the factorization of $f(\mathbf{x}_{1:n}^{\mathcal{S}}|\mathbf{y}_{1:n}^{\mathcal{S}})$ in (2.3), which is shown in Fig. 2.1. Because this factor graph is loopy, BP schemes can only provide an approximate marginalization. However, for cooperative navigation, the beliefs obtained with BP are known to be quite accurate [Wymeersch et al., 2009].

At each time n, P message passing iterations are performed. The iterated belief of CA $l\in\mathcal{S}$

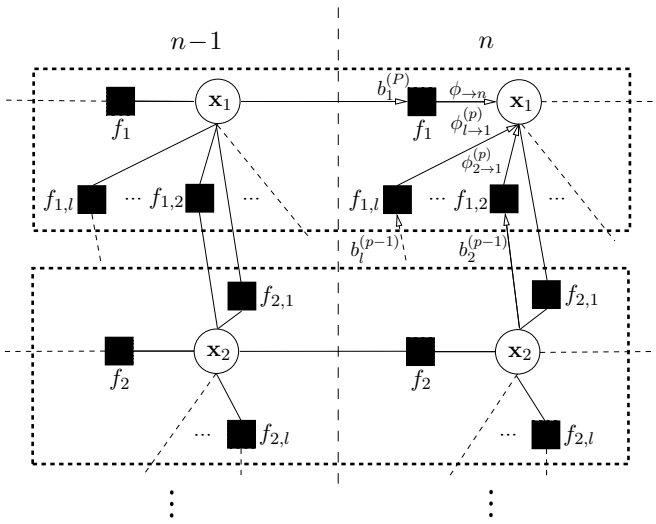


Figure 2.1: Cooperative navigation factor graph showing the states of CAs $l = 1, 2$ at time instants $n-1$ and n; time indices are omitted for simplicity. The short notation $f_l \triangleq f(\mathbf{x}_{l,n'}|\mathbf{x}_{l,n'-1})$, $f_{l,l'} \triangleq f(\mathbf{y}_{l,l';n'}|\mathbf{x}_{l,n'}, \mathbf{x}_{l',n'})$, $b_l^{(p)} \triangleq b^{(p)}(\mathbf{x}_{l,n'})$, $n' \in \{1, \ldots, n\}$ is used. Each dotted box corresponds to a CA $l \in \mathcal{S}$; calculations within the box are performed locally at that CA. Edges between dotted boxes imply communication between CAs. Only messages and beliefs involved in the computation of $b^{(p)}(\mathbf{x}_{1,n})$ are shown.

at time n and message passing iteration $p \in \{1, \ldots, P\}$ is calculated as

$$b^{(p)}(\mathbf{x}_{l,n}) \;\propto\; \phi_{\to n}(\mathbf{x}_{l,n}) \prod_{l' \in \mathcal{M}_{l,n}^{\mathcal{S}}} \phi_{l'\to l}^{(p)}(\mathbf{x}_{l,n})\,. \tag{2.5}$$

Here, the "prediction message" $\phi_{\to n}(\mathbf{x}_{l,n})$ is calculated from the state-transition PDF $f(\mathbf{x}_{l,n}|\mathbf{x}_{l,n-1})$ and the final belief at time $n-1$, $b^{(P)}(\mathbf{x}_{l,n-1})$, as

$$\phi_{\to n}(\mathbf{x}_{l,n}) \;=\; \int f(\mathbf{x}_{l,n}|\mathbf{x}_{l,n-1})\, b^{(P)}(\mathbf{x}_{l,n-1})\, \mathrm{d}\mathbf{x}_{l,n-1}\,, \tag{2.6}$$

and the "measurement messages" $\phi_{l'\to l}^{(p)}(\mathbf{x}_{l,n})$ are calculated as

$$\phi_{l'\to l}^{(p)}(\mathbf{x}_{l,n}) \;=\; \int f(\mathbf{y}_{l,l';n}|\mathbf{x}_{l,n}, \mathbf{x}_{l',n})\, b^{(p-1)}(\mathbf{x}_{l',n})\, \mathrm{d}\mathbf{x}_{l',n}\,. \tag{2.7}$$

These messages and beliefs are depicted in Fig. 2.1. Note that the iterative scheme (2.5)–(2.7) is initialized by setting $b^{(0)}(\mathbf{x}_{l,n}) = \phi_{\to n}(\mathbf{x}_{l,n})$ for all $l \in \mathcal{S}$.

Two remarks are in order for a better understanding of the message passing schedule (2.5)–(2.7) which is also known as SPAWN [Wymeersch et al., 2009]. First, the factor graph has also loops between different times $\ldots, n-1,\ n,\ n+1, \ldots$. Thus, in contrast to sequential state estimation in a tracking problem, where the factor graph is tree-like (to be considered in Section 3.1 and Fig. 3.1), $b^{(p)}(\mathbf{x}_{l,n})$ could be improved by sending messages backward and forward in time. However, in practice, for low complexity, communication , memory requirements as well

as a reduced latency, messages are sent only forward in time and iterative message passing is done at each time individually. As a consequence, the message ("extrinsic information") from the variable $\mathbf{x}_{l,n-1}$ to the factor $f(\mathbf{x}_{l,n}|\mathbf{x}_{l,n-1})$ equals the belief $b^{(P)}(\mathbf{x}_{l,n-1})$, and $\phi_{\to n}(\mathbf{x}_{l,n})$ in (2.6) (for n fixed) remains unchanged during all message passing iterations. Second, as no information from the factor $f(\mathbf{y}_{l,l';n}|\mathbf{x}_{l,n},\mathbf{x}_{l',n})$ is used in the calculation of $b^{(p-1)}(\mathbf{x}_{l',n})$ according to (2.5) and (2.7), $b^{(p-1)}(\mathbf{x}_{l',n})$ in (2.7) is the extrinsic information with respect to $f(\mathbf{y}_{l,l';n}|\mathbf{x}_{l,n},\mathbf{x}_{l',n})$. As explained in more detail in [Wymeersch et al., 2009] and [Lien et al., 2012], this message passing schedule significantly reduce the computational complexity of the standard BP message passing scheme. In addition, also the amount of communication between CAs is reduced since (as discussed presently) beliefs in SPAWN can be broadcast, whereas the exchange of extrinsic information in standard BP requires point-to-point communication between CAs.

Contrary to classical sequential Bayesian filtering [Doucet et al., 2001], which only exploits the temporal independence structure of the inference problem, BP message passing (2.5)–(2.7) also exploits the spatial independency structure. As a consequence, the overall computational complexity scales linearly both in the number of time steps and in the number of CAs. In addition, with BP message passing, the beliefs $b^{(p)}(\mathbf{x}_{l,n})$ at each CA $l \in \mathcal{S}$ can be calculated in a fully *distributed* manner using only local communication with neighbors. Each CA l broadcasts its own current belief $b^{(p)}(\mathbf{x}_{l,n})$ (calculated according to (2.5)) to all CAs l_1 for which $l \in \mathcal{M}^{\mathcal{S}}_{l_1,n}$ and uses the beliefs received from the neighboring CAs, $b^{(p)}(\mathbf{x}_{l',n})$ for $l' \in \mathcal{M}^{\mathcal{S}}_{l,n}$, and the local pairwise measurements $\mathbf{y}_{l,l';n}$, $l' \in \mathcal{M}^{\mathcal{S}}_{l,n}$ to calculate the corresponding measurement messages for the next message passing iteration, $\phi^{(p+1)}_{l'\to l}(\mathbf{x}_{l,n})$ for $l' \in \mathcal{M}^{\mathcal{S}}_{l,n}$, according to (2.7). The messages $\phi^{(p+1)}_{l'\to l}(\mathbf{x}_{l,n})$ and $\phi_{\to n}(\mathbf{x}_{l,n})$ (see (2.6)) are then used by CA l to calculate the new $b^{(p+1)}(\mathbf{x}_{l,n})$ according to (2.5), etc. Note that, since each CA $l \in \mathcal{S}$ is only interested in its own position $\mathbf{x}_{l,n}$, the corresponding belief $b^{(p)}(\mathbf{x}_{l,n})$ only needs to be stored (temporarily) at CA l. We note that, as loopy BP in general [Wymeersch et al., 2012], the BP scheme (2.5)–(2.7) exhibits accurate estimates but suffers from overconfident beliefs.

The remaining step is to avoid the high complexity of evaluating the integrals in (2.6) and (2.7) by a particle-based or using sigma point implementation.

2.1.2 NBP for Cooperative Navigation

We first review NBP [Ihler et al., 2005] for cooperative navigation [Lien et al., 2012]. NBP uses a particle representation (PR) of beliefs and messages. In a cooperative navigation scenario, it provides fast convergence and high accuracy [Wymeersch et al., 2009]. NBP consist of three basics operations—message filtering, message multiplication, and estimation—which are reviewed in what follows.

Message Filtering

PR-based calculation of (2.6) and (2.7) amounts to the generic problem of obtaining a PR of

$$\phi(\mathbf{x}') = \int c(\mathbf{x}', \mathbf{x})\, b(\mathbf{x})\, \mathrm{d}\mathbf{x}$$

from a PR $\left\{\left(\mathbf{x}^{(j)}, w^{(j)}\right)\right\}_{j=1}^{J}$ of $b(\mathbf{x})$. Here, $c(\mathbf{x}', \mathbf{x})$ is (proportional to) a conditional PDF $f(\mathbf{x}'|\mathbf{x})$. If we are able to obtain a PR $\left\{\left(\mathbf{x}^{(j)}, \mathbf{x}'^{(j)}, w^{(j)}\right)\right\}_{j=1}^{J}$ of $c(\mathbf{x}', \mathbf{x})\, b(\mathbf{x})$, then $\left\{\left(\mathbf{x}'^{(j)}, w^{(j)}\right)\right\}_{j=1}^{J}$ constitutes a PR of $\phi(\mathbf{x}')$ [Ristic et al., 2004]. Typically, $\mathbf{x}' = r(\mathbf{x}, \mathbf{q})$ with a known function $r(\cdot, \cdot)$ and a random vector $\mathbf{q}$. Particles $\left\{\left(\mathbf{x}^{(j)}, \mathbf{x}'^{(j)}, w^{(j)}\right)\right\}_{j=1}^{J}$ corresponding to $c(\mathbf{x}', \mathbf{x})\, b(\mathbf{x})$ can then be obtained by first drawing particles $\left\{(\mathbf{x}^{(j)}, \mathbf{q}^{(j)})\right\}_{j=1}^{J}$ from $f(\mathbf{x}, \mathbf{q})$ and then calculating $\mathbf{x}'^{(j)} = r\big(\mathbf{x}^{(j)}, \mathbf{q}^{(j)}\big)$, $j \in \{1, \ldots, J\}$.

Message Multiplication

A PR $\left\{\left(\mathbf{x}_{l,n}^{(j)}, w_{l,n}^{(j)}\right)\right\}_{j=1}^{J}$ of $b^{(p)}(\mathbf{x}_{l,n}) \propto \phi_{\to n}(\mathbf{x}_{l,n}) \prod_{l' \in \mathcal{M}_{l,n}^{\mathcal{S}}} \phi_{l' \to l}^{(p)}(\mathbf{x}_{l,n})$ in (2.5) can be obtained using importance sampling [Ristic et al., 2004]. We first draw particles $\left\{\mathbf{x}_{l,n}^{(j)}\right\}_{j=1}^{J}$ from the proposal distribution $q(\mathbf{x}_{l,n}) = \phi_{\to n}(\mathbf{x}_{l,n})$. Corresponding weights $\left\{w_{l,n}^{(j)}\right\}_{j=1}^{J}$ are then obtained by calculating

$$\tilde{w}_{l,n}^{(j)} \propto \frac{b^{(p)}(\mathbf{x}_{l,n}^{(j)})}{q(\mathbf{x}_{l,n}^{(j)})},$$

i.e.,

$$\tilde{w}_{l,n}^{(j)} = \prod_{l' \in \mathcal{M}_{l,n}^{\mathcal{S}}} \phi_{l' \to l}^{(p)}(\mathbf{x}_{l,n}^{(j)}), \tag{2.8}$$

and normalizing, i.e.,

$$w_{l,n}^{(j)} = \frac{\tilde{w}_{l,n}^{(j)}}{W_{l,n}} \qquad \text{with} \qquad W_{l,n} = \sum_{j'=1}^{J} \tilde{w}_{l,n}^{(j')}. \tag{2.9}$$

Since only a PR of $\phi_{l' \to l}^{(p)}(\mathbf{x}_{l,n})$ is available, we substitute a kernel estimate[1] $\hat{\phi}_{l' \to l}^{(p)}(\mathbf{x}_{l,n}^{(j)})$ for $\phi_{l' \to l}^{(p)}(\mathbf{x}_{l,n})$ in (2.8), and thus obtain

$$\tilde{w}_{l,n}^{(j)} = \prod_{l' \in \mathcal{M}_{l,n}^{\mathcal{S}}} \hat{\phi}_{l' \to l}^{(p)}(\mathbf{x}_{l,n}^{(j)}). \tag{2.11}$$

[1] We note that a kernel estimate of a message $\phi(\mathbf{x})$ is calculated from a PR $\left\{\left(\mathbf{x}^{(j)}, w^{(j)}\right)\right\}_{j=1}^{J}$ of $\phi(\mathbf{x})$ as

$$\hat{\phi}(\mathbf{x}) = \sum_{j=1}^{J} w^{(j)} K(\mathbf{x} - \mathbf{x}^{(j)}), \tag{2.10}$$

with some kernel function $K(\mathbf{x})$.

This message multiplication operation is the most complex part of NBP; its complexity scales quadratically in the number of particles J.

Estimation

Finally, from the PR $\left\{\left(\mathbf{x}_{l,n}^{(j)}, w_{l,n}^{(j)}\right)\right\}_{j=1}^{J}$ of $b^{(p)}(\mathbf{x}_{l,n})$, an approximation of the estimate $\hat{\mathbf{x}}_{l,n}^{\text{MMSE}}$ in (2.1) is obtained as

$$\hat{\mathbf{x}}_{l,n}^{\text{MMSE}} \approx \sum_{j=1}^{J} w_{l,n}^{(j)} \mathbf{x}_{l,n}^{(j)} \,. \tag{2.12}$$

Alternatively, an approximation of the MAP estimator $\hat{\mathbf{x}}_{l,n}^{\text{MAP}}$ in (2.2) can be derived from the PR as described in [Driessen and Boers, 2008].

2.2 Parametric Message Representations

A significant limitation of the nonparametric implementation of (2.5)–(2.7) described in the previous section are the high computation and communication costs. Communication requirements are high, because the exchange of probability distributions (beliefs) between different CAs is done by transmitting directly the J particles that represent these probability distributions. The main bottleneck in terms of computational complexity is the message multiplication procedure in Section 2.1.2, which scales quadratically in the number of particles J and makes the nonparametric implementation as described in the previous section infeasible for large J.

In this section, we propose an alternative implementation of the cooperative navigation message passing scheme (2.5)–(2.7) for the special case of a 2D setting and measurements of the distance are performed among CAs. More specifically, in this section for each CA $l \in \mathcal{S}$ the "position-part" $\tilde{\mathbf{x}}_{l,n}$ of the state $\mathbf{x}_{l,n}$ is 2D and the general measurement model (1.2) is replaced by

$$y_{l,l';n} \;=\; \|\tilde{\mathbf{x}}_{l,n} - \tilde{\mathbf{x}}_{l',n}\| + v_{l,l';n} \qquad\qquad l' \in \mathcal{M}_{l,n}^{\mathcal{S}}. \tag{2.13}$$

The proposed alternative implementation achieves significant savings in both communications and computation through the use of parametric measurement messages (as introduced in [Lien et al., 2012], although the parameters are determined differently) and a more efficient procedure to perform the costly message multiplication operation. In the following, we describe the alternative and efficient message multiplication procedure, that is performed at time n, CA l and message passing iteration p using parametric measurement messages. It is assumed that the drawing of particles from $m_{\to n}(\tilde{\mathbf{x}}_{l,n})$ in (3.11) using particles representing $b^{(P)}(\tilde{\mathbf{x}}_{l,n-1})$ has already be performed using message filtering (see Section 2.1.2).

2.2.1 Extracting the Belief Parameters

Consider $b^{(p-1)}(\tilde{\mathbf{x}}_{l',n})$, i.e., the belief of the 2D position $\tilde{\mathbf{x}}_{l',n}$ of CA $l' \in \mathcal{S}$ at time n and message passing iteration $p-1$. This 2D function is either (i) unimodal if the CA is well localized; (ii) bimodal with two modes if the CA is localized with ambiguity; or (iii) multimodal (e.g.,

annularly shaped) if the CA is poorly localized [Wymeersch et al., 2009]. To reduce communications, the number of parameters to be transmitted, we approximate the belief $b^{(p-1)}(\tilde{\mathbf{x}}_{l',n})$ by a Gaussian $\mathcal{N}(\tilde{\boldsymbol{\mu}}_{l',n}, \tilde{\mathbf{C}}_{l',n})$ if it is unimodal and by a mixture of two Gaussians $\mathcal{N}\big(\tilde{\boldsymbol{\mu}}^{(1)}_{l',n}, \tilde{\mathbf{C}}^{(1)}_{l',n}\big)$ and $\mathcal{N}\big(\tilde{\boldsymbol{\mu}}^{(2)}_{l',n}, \tilde{\mathbf{C}}^{(2)}_{l',n}\big)$ with equal weights if it is bimodal. The respective means and covariances are then transmitted to the navigation partners.

We propose the following procedure for extracting the belief parameters at CA l'. First, CA l' derives particles $\big\{\tilde{\mathbf{x}}^{(j)}_{l',n}\big\}_{j=1}^{J}$ representing $b^{(p-1)}(\tilde{\mathbf{x}}_{l',n})$, which are obtained from the particles representing $b^{(p-1)}(\mathbf{x}_{l',n})$ simply by discarding the irrelevant entries in each particle vector (recall that $\tilde{\mathbf{x}}_{l',n}$ is a subvector of $\mathbf{x}_{l',n}$). Next, CA l' uses a clustering algorithm such as K-means [Gan et al., 2007] to partition the set of particles $\big\{\tilde{\mathbf{x}}^{(j)}_{l',n}\big\}_{j=1}^{J}$ into two disjoint subsets $\big\{\tilde{\mathbf{x}}^{(j)}_{l',n}\big\}_{j\in\mathcal{J}_1}$ and $\big\{\tilde{\mathbf{x}}^{(j)}_{l',n}\big\}_{j\in\mathcal{J}_2}$, and it calculates the Fisher linear discriminant [Gan et al., 2007] (denoted D) for that partition. Also, a mean $\boldsymbol{\mu}_{l',n}$ and a covariance matrix $\mathbf{C}_{l',n}$ are computed for each particle subset, i.e.,

$$\tilde{\boldsymbol{\mu}}^{(i)}_{l',n} = \frac{1}{|\mathcal{J}_i|}\sum_{j\in\mathcal{J}_i}\tilde{\mathbf{x}}^{(j)}_{l',n}, \quad \tilde{\mathbf{C}}^{(i)}_{l',n} = \frac{1}{|\mathcal{J}_i|}\sum_{j\in\mathcal{J}_i}\tilde{\mathbf{x}}^{(j)}_{l',n}\tilde{\mathbf{x}}^{(j)\mathrm{T}}_{l',n} - \tilde{\boldsymbol{\mu}}^{(i)}_{l',n}\tilde{\boldsymbol{\mu}}^{(i)\mathrm{T}}_{l',n},$$

for $i \in \{1,2\}$. If D is above a threshold T and $\|\tilde{\boldsymbol{\mu}}^{(1)}_{l',n} - \tilde{\boldsymbol{\mu}}^{(2)}_{l',n}\| > 4\sigma_v$, the clustering result is accepted and, thus, the bimodal Gaussian mixture model is adopted for $b^{(p-1)}(\tilde{\mathbf{x}}_{l',n})$. Otherwise, the clustering is rejected and a single mean $\tilde{\boldsymbol{\mu}}_{l',n}$ and covariance matrix $\tilde{\mathbf{C}}_{l',n}$ are determined from the total particle set $\big\{\tilde{\mathbf{x}}^{(j)}_{l',n}\big\}_{j=1}^{J}$. Then, if $(\tilde{\mathbf{C}}_{l',n})_{1,1} + (\tilde{\mathbf{C}}_{l',n})_{2,2} < 10\sigma_v^2$, the unimodal Gaussian model $\mathcal{N}(\tilde{\boldsymbol{\mu}}_{l',n}, \tilde{\mathbf{C}}_{l',n})$ is adopted, otherwise $b^{(p-1)}(\tilde{\mathbf{x}}_{l',n})$ is considered multimodal.

2.2.2 Calculating the Parametric Measurement Messages

After all belief parameters have been transmitted, as discussed above, each CA $l \in \mathcal{S}$ at time n knows (approximate representations of) the beliefs $b^{(p-1)}(\tilde{\mathbf{x}}_{l',n})$ of its localization partners $l' \in \mathcal{M}^{\mathcal{S}}_{l,n}$.

Based on its knowledge of the beliefs $b^{(p-1)}(\tilde{\mathbf{x}}_{l',n})$ for $l' \in \mathcal{M}^{\mathcal{S}}_{l,n}$, CA l next calculates a particle representation of its own belief $b^{(p)}(\mathbf{x}_{l,n})$ by implementing (2.5)–(2.7) as will be described in Section 2.2.3. Because this calculation requires closed-form expressions of the measurement messages $m^{(p)}_{l'\to l}(\tilde{\mathbf{x}}_{l,n})$, $l' \in \mathcal{M}^{\mathcal{S}}_{l,n}$, we use the parametric message representations introduced in [Lien et al., 2012]. More specifically, if CA l' is localized, i.e., $b^{(p-1)}(\tilde{\mathbf{x}}_{l',n})$ is unimodal, we set

$$m^{(p)}_{l'\to l}(\tilde{\mathbf{x}}_{l,n}) \propto \exp\left(-\frac{\big(y_{l',l;n} - \|\tilde{\mathbf{x}}_{l,n} - \tilde{\boldsymbol{\mu}}_{l',n}\|\big)^2}{2r_{l',l;n}}\right). \tag{2.14}$$

The shape of this message is an annulus about the midpoint $\tilde{\boldsymbol{\mu}}_{l',n}$ with nominal radius $y_{l',l;n}$; the radial width about the nominal radius is determined by $r_{l',l;n}$. If CA l' is localized with ambiguity, i.e., $b^{(p-1)}(\tilde{\mathbf{x}}_{l',n})$ is bimodal, we set $m^{(p)}_{l'\to l}(\tilde{\mathbf{x}}_{l,n})$ equal to the sum of two annuli that are located about the midpoints $\tilde{\boldsymbol{\mu}}^{(1)}_{l',n}$ and $\tilde{\boldsymbol{\mu}}^{(2)}_{l',n}$ and have equal nominal radius $y_{l',l;n}$ and

possibly different width parameters $r^{(1)}_{l',l;n}$ and $r^{(2)}_{l',l;n}$. Finally, if CA l' is poorly localized, i.e., $b^{(p-1)}(\tilde{\mathbf{x}}_{l',n})$ is multimodal, CA l ignores localization partner $l' \in \mathcal{M}^{\mathcal{S}}_{l,n}$ by setting the corresponding message $m^{(p)}_{l'\to l}(\tilde{\mathbf{x}}_{l,n})$ to a constant value.

It remains to determine the width parameter(s) for the first two cases. Let us first consider the unimodal representation (2.14). If CA l' is an anchor CA, we have $b^{(p-1)}(\tilde{\mathbf{x}}_{l',n}) = \delta(\tilde{\mathbf{x}}_{l',n} - \tilde{\boldsymbol{\mu}}_{l',n})$, and the parametric message (2.14) with width parameter $r_{l',l;n} = \sigma_v^2$ is the exact result of the expression in exactly equal to (2.7). Otherwise, let

$$\rho_{l,l';n}(\tilde{\mathbf{x}}_{l',n}) \triangleq \left\| \hat{\tilde{\mathbf{x}}}^{(p-1)}_{l,n} - \tilde{\mathbf{x}}_{l',n} \right\| \tag{2.15}$$

be the distance of the estimate of $\tilde{\mathbf{x}}_{l,n}$ the position of CA l at message passing iteration $p-1$, denoted $\hat{\tilde{\mathbf{x}}}^{(p-1)}_{l,n}$, from $\tilde{\mathbf{x}}_{l',n}$. A good approximation of (2.7) is then obtained by choosing $r_{l',l,n}$ as

$$r_{l',l;n} = \mathbf{h}^{\mathrm{T}}_{l,l';n} \tilde{\mathbf{C}}_{l',n} \mathbf{h}_{l,l';n} + \sigma_v^2 \,, \tag{2.16}$$

where $\mathbf{h}_{l,l';n} = \Delta_{\tilde{\boldsymbol{\mu}}_{l',n}} \rho_{l,l';n}(\mathbf{x}_{l',n})$ is the gradient of $\rho_{l,l';n}(\tilde{\mathbf{x}}_{l',n})$ evaluated at $\tilde{\boldsymbol{\mu}}_{l',n}$ [Sathyan and Hedley, 2013]. This result for $r_{l',l;n}$ is obtained via a linear approximation of the "reduced" measurement equation $y'_{l,l';n} = \rho_{l,l';n}(\tilde{\mathbf{x}}_{l',n}) + v_{l,l';n}$ around $\tilde{\boldsymbol{\mu}}_{l',n}$ [Sathyan and Hedley, 2013]. More specifically, $r_{l',l;n}$ is the variance of

$$\rho_{l,l';n}(\tilde{\boldsymbol{\mu}}_{l',n}) + \mathbf{h}^{\mathrm{T}}_{l,l';n}(\tilde{\mathbf{x}}_{l',n} - \tilde{\boldsymbol{\mu}}_{l',n}) + v_{l,l';n}. \tag{2.17}$$

Note that now the radial width of the annularly shaped message $m^{(p)}_{l'\to l}(\tilde{\mathbf{x}}_{l,n})$ in (2.14) characterized by $r_{l',l,n}$ is influenced by both the uncertainty in the l'th CA position, expressed by $\mathbf{h}^{\mathrm{T}}_{l,l';n} \tilde{\mathbf{C}}_{l',n} \mathbf{h}_{l,l';n}$, and the measurement variance σ_v^2.

For the bimodal representation, we choose the two width parameters as in (2.16), i.e.,

$$r^{(i)}_{l',l;n} = \mathbf{h}^{(i)\mathrm{T}}_{l,l';n} \tilde{\mathbf{C}}^{(i)}_{l',n} \mathbf{h}^{(i)}_{l,l';n} + \sigma_v^2 \tag{2.18}$$

for $i \in \{1,2\}$, where $\mathbf{h}^{(i)}_{l,l';n}$ is the gradient of $\rho_{l,l';n}(\tilde{\mathbf{x}}_{l',n})$ evaluated at $\tilde{\boldsymbol{\mu}}^{(i)}_{l',n}$. An example of a bimodal Gaussian belief $b^{(p-1)}(\tilde{\mathbf{x}}_{l',n})$ and corresponding bi-annular message $m^{(p)}_{l'\to l}(\tilde{\mathbf{x}}_{l,n})$ is shown in Fig. 2.2.

2.2.3 Updating the Beliefs

With all messages $m^{(p)}_{l'\to l}(\tilde{\mathbf{x}}_{l,n})$, $l' \in \mathcal{M}^{\mathcal{S}}_{l,n}$ determined, an approximation of the functional form of $\prod_{l'\in\mathcal{M}^{\mathcal{S}}_{l,n}} m^{(p)}_{l'\to l}(\tilde{\mathbf{x}}_{l,n})$ is available at CA $l \in \mathcal{S}$. Thus, CA l is able to calculate a particle representation of its updated belief $b^{(p)}(\mathbf{x}_{l,n})$ according to (2.5).

This calculation is done by means of importance sampling [Doucet et al., 2001], using the prediction message $m_{\to n}(\mathbf{x}_{l,n})$ as proposal density: particles $\left\{\mathbf{x}^{(j)}_{l,n}\right\}_{j=1}^{J}$ are drawn from

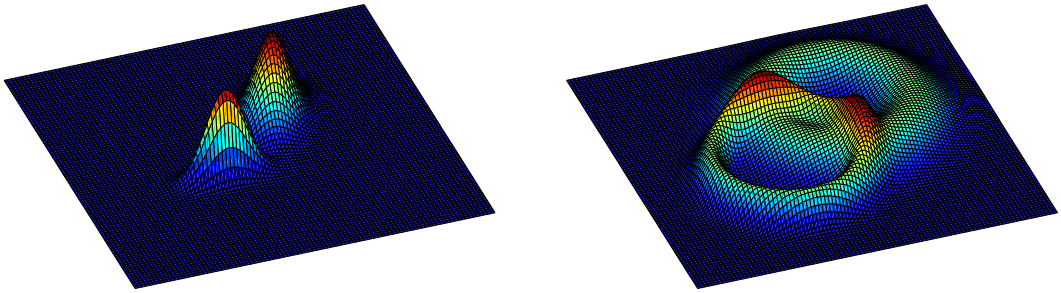


Figure 2.2: Example of a bimodal Gaussian belief $b^{(p-1)}(\tilde{\mathbf{x}}_{l,n})$ (left) and the corresponding bi-annular message $m^{(p)}_{l\to k}(\tilde{\mathbf{x}}_{k,n})$ (right).

$m_{\to n}(\mathbf{x}_{l,n})$, and associated weights unnormalized weights are obtained as

$$\tilde{w}^{(j)}_{l,n} = \prod_{l'\in\mathcal{M}_{l,n}} m^{(p)}_{l'\to l}(\tilde{\mathbf{x}}^{(j)}_{l,n}). \tag{2.19}$$

This procedure is followed by a normalization and a resampling step [Doucet et al., 2001] to obtain equally weighted particles for the belief $b^{(p)}(\mathbf{x}_{l,n})$.

2.2.4 Computation and Communication Requirements

The transmission of means and covariances among neighboring navigation partners, requires the transmission of $2+3=5$ real numbers in the unimodal case and of 10 real numbers in the bimodal case. If $b^{(p-1)}(\tilde{\mathbf{x}}_{l',n})$ is multimodal, no belief parameters are transmitted, because a poorly localized CA cannot provide useful information to its navigation partners. Therefore, the communication cost is reduced by about an order of magnitude, since in the conventional nonparametric algorithms typically hundreds particles have to be transmitted [Ihler et al., 2005, Lien et al., 2012]. Furthermore, it can easily be shown, that the calculation of parameters as described in Sections 2.2.1 and 2.2.2 and as well as the evaluation of parametric messages in (2.19) scales only linearly with the number of particles J, rather than quadratically as in the case of the conventional nonparametric algorithm [Ihler et al., 2005, Lien et al., 2012].

2.3 Dimension-Augmented Reformulation of BP Message Passing

In this section, we introduce the dimension-augmented reformulation of BP message passing. The key idea is to reformulate the BP operations in higher-dimensional spaces, which allows the application of many well studied filter for sequential Bayesian estimation (e.g. extended Kalman filter [Kay, 1993], sigma point filter [Julier and Uhlmann, 1997, Wan and van der Merwe, 2001], cubature Kalman filter [Arasaratnam and Haykin, 2009], the particle filter [Gordon et al., 1993, Ristic et al., 2004, Doucet et al., 2001] and the Gaussian belief condensation filter [Mazuelas

et al., 2013]) to BP-based cooperative navigation. In the following Sections 2.5 and 2.4, we will use the proposed dimension-augmented formulation to develop a low-complexity alternative to NBP and SPBP which is a extension of the sigma point filter to more general—possibly loopy—graphical models.

Let us first introduce the "composite vectors" $\bar{\mathbf{x}}_{l,n} \triangleq [\mathbf{x}_{l',n}]_{l' \in \{l\} \cup \mathcal{M}_{l,n}^{\mathcal{S}}}$ and $\bar{\mathbf{y}}_{l,n} \triangleq [\mathbf{y}_{l,l',n}]_{l' \in \mathcal{M}_{l,n}^{\mathcal{S}}}$. Now, using (2.7) in (2.5), one readily obtains

$$b^{(p)}(\mathbf{x}_{l,n}) = \int b^{(p)}(\bar{\mathbf{x}}_{l,n}) \mathrm{d}\bar{\mathbf{x}}_{l,n}^{\sim l}\,, \tag{2.20}$$

where

$$b^{(p)}(\bar{\mathbf{x}}_{l,n}) \propto f(\bar{\mathbf{y}}_{l,n}|\bar{\mathbf{x}}_{l,n})\, f^{(p-1)}(\bar{\mathbf{x}}_{l,n}), \tag{2.21}$$

$$f^{(p-1)}(\bar{\mathbf{x}}_{l,n}) \propto \phi_{\to n}(\mathbf{x}_{l,n}) \prod_{l' \in \mathcal{M}_{l,n}^{\mathcal{S}}} b^{(p-1)}(\mathbf{x}_{l',n}), \tag{2.22}$$

$$f(\bar{\mathbf{y}}_{l,n}|\bar{\mathbf{x}}_{l,n}) = \prod_{l' \in \mathcal{M}_{l,n}^{\mathcal{S}}} f(\mathbf{y}_{l,l';n}|\mathbf{x}_{l,n}, \mathbf{x}_{l',n}), \tag{2.23}$$

$$\phi_{\to n}(\mathbf{x}_{l,n}) = \int f(\mathbf{x}_{l,n}|\mathbf{x}_{l,n-1})\, b^{(P)}(\mathbf{x}_{l,n-1})\, \mathrm{d}\mathbf{x}_{l,n-1}\,, \tag{2.24}$$

and $\mathrm{d}\bar{\mathbf{x}}_{l,n}^{\sim l} \triangleq \prod_{l' \in \mathcal{M}_{l,n}^{\mathcal{S}}} \mathrm{d}\mathbf{x}_{l',n}$.

Note that $f^{(p-1)}(\bar{\mathbf{x}}_{l,n})$ can be interpreted as a "iterated prior PDF" and $f(\bar{\mathbf{y}}_{l,n}|\bar{\mathbf{x}}_{l,n})$ is the likelihood function corresponding to the composite observation model

$$\bar{\mathbf{y}}_{l,n} = D_{l,n}(\bar{\mathbf{x}}_{l,n}, \bar{\mathbf{v}}_{l,n})$$

where $D_{l,n}(\bar{\mathbf{x}}_{l,n}, \bar{\mathbf{v}}_{l,n}) \triangleq \big[\big(d_l(\mathbf{x}_{l,n}, \mathbf{x}_{l',n}, \mathbf{v}_{l,l',n})\big]_{l' \in \mathcal{M}_{l,n}^{\mathcal{S}}}$ and $\bar{\mathbf{v}}_{l,n} \triangleq \big[\mathbf{v}_{l,l',n}\big]_{l' \in \mathcal{M}_{l,n}^{\mathcal{S}}}$.

2.3.1 Sequential Filtering for Cooperative Navigation

The main advantage of the dimension-augmented reformulation is that well-known filters for sequential Bayesian estimation such as the extended Kalman filter [Kay, 1993], the sigma point filter [Julier and Uhlmann, 1997, Wan and van der Merwe, 2001], the cubature Kalman filter [Arasaratnam and Haykin, 2009], the particle filter [Gordon et al., 1993, Ristic et al., 2004, Doucet et al., 2001] and the Gaussian belief condensation filter [Mazuelas et al., 2013]), can be directly applied to the BP scheme (2.5)–(2.7) for cooperative navigation. The only constrain is that the used filter is based on an approximate representation of the belief for which "marginalizing out" variables is a trivial operation. This is the case for e.g. a particle-based or a Gaussian representation. After choosing a suitable filter, the evaluation of (2.20)–(2.24) (which is equivalent to the evaluation of (2.5)–(2.7)) at CA $l \in \mathcal{S}$ is performed as follows:

Prediction: Calculating the approximate representation of the prediction message $\phi_{\to n}(\mathbf{x}_{l,n})$ according to (2.24) is performed equivalently as in the prediction step of the chosen filter. After

receiving an approximate representation of $b^{(p-1)}(\mathbf{x}_{l',n})$ from neighboring CAs $l' \in \mathcal{M}_{l,n}^{\mathcal{S}}$, an approximate representation of $f^{(p-1)}(\bar{\mathbf{x}}_{l,n})$ in (2.22) is then available at CA l.

High-Dimensional Measurement Update: Using Equation (2.21), high-dimensional measurement update is performed equivalently as in the measurement update step of the chosen filter; this involves the likelihood function $f(\bar{\mathbf{y}}_{l,n}|\bar{\mathbf{x}}_{l,n})$ defined in (2.23). The result of this high dimensional measurement update operations is an approximate representation of the "stacked" belief $b^{(p)}(\bar{\mathbf{x}}_{l,n})$.

Marginalization: For the approximate representation of the "stacked" belief $b^{(p)}(\bar{\mathbf{x}}_{l,n})$, marginalization in (2.20) is performed to obtain an approximate representation of the belief $b^{(p)}(\mathbf{x}_{l,n})$. If the chosen filter uses a Gaussian, a mixture of Gaussian, or particles for the representation of beliefs, the marginalization in (2.20) is trivial and "for free" in the sense that it can be performed without any cost in terms of computational complexity. (In the case of a particle representation $\{\bar{\mathbf{x}}_{l,n}^{(j)}\}_{j=1}^{J}$ of the belief $b^{(p)}(\bar{\mathbf{x}}_{l,n})$, $\mathbf{x}_{l,n}^{(j)}$ can be obtained by discarding from $\bar{\mathbf{x}}_{l,n}^{(j)}$ all vectors $\mathbf{x}_{l',n}^{(j)}$ with $l' \neq l$, i.e., by discarding $[\mathbf{x}_{l',n}^{(j)}]_{l' \in \mathcal{M}_{l,n}^{\mathcal{S}}}$. If $b^{(p)}(\bar{\mathbf{x}}_{l,n})$ is represented by a Gaussian with mean $\boldsymbol{\mu}_{b(\bar{\mathbf{x}}_{l,n})}^{(p)}$ and covariance matrix $\mathbf{C}_{b(\bar{\mathbf{x}}_{l,n})}^{(p)}$, the mean and covariance related to $\mathbf{x}_{l,n}$ can be extracted from $\boldsymbol{\mu}_{b(\bar{\mathbf{x}}_{l,n})}^{(p)}$ and $\mathbf{C}_{b(\bar{\mathbf{x}}_{l,n})}^{(p)}$ by discarding all entries related to $\mathbf{x}_{l',n}$ with $l' \neq l$.)

In Sections 2.4 and 2.5 we will develop new algorithms for cooperative navigation by combining the bootstrap particle filter [Gordon et al., 1993] and the sigma point filter [Julier and Uhlmann, 1997, Wan and van der Merwe, 2001] with this sequential scheme.

2.3.2 Curse of Dimensionality, Censoring, and Algorithm Tuning

As described in the last section, in the dimension-augmented reformulation of BP message passing, at CA $l \in \mathcal{S}$ the measurement updated step is performed for the augmented state $\bar{\mathbf{x}}_{l,n}$, which has a dimensionality that depends on the number of neighbors $\mathcal{M}_{l,n}$. This means the dimensionality of the message multiplication operation in the proposed reformulation is larger than that of the conventional formulation described in Section 2.1.1; furthermore it depends on the network topology and is thus time-dependent. At first sight this might be problematic for two reasons: First, since the complexity of measurement updated strongly depends on the dimensionality of the problem, the complexity of the resulting algorithms might be prohibitive [Doucet et al., 2001, Daum and Huang, 2003]. Second, algorithm parameters which depend on the dimensionally of the state (like the the number of particles J in a particle-based algorithm) are difficult to tune.

These concerns can be addressed by the fact that in localization scenarios only a small number of neighbors is needed to obtain a high localization accuracy; an increase of over a certain number leads to very little or even no improvement in localization accuracy [Das and Wymeersch, 2012, Savic and Zazo, 2012]. For this reason, in dense networks, censoring schemes [Das and Wymeersch, 2012, Savic and Zazo, 2012], are typically employed to keep the number of neighbors used for calculating the belief small. Therefore, a maximum number of neighbors can be fixed to a certain value by a censoring scheme, and the algorithm based on the dimension-

augmented reformulation of BP can then be tuned to that value. (Selections strategy for censoring can be found in [Das and Wymeersch, 2012, Savic and Zazo, 2012].) Surprisingly, as numerically demonstrated in Section 2.6 in terms of average run time, in moderately dense networks, the new algorithms based on the dimension augmented reformulation are significantly less complex then the existing nonparametric algorithm for cooperative navigation reviewed in Section 2.1.2.

2.4 A New Particle-Based Implementation of BP

We now use the proposed reformulation of BP to derive a new low-complexity nonparametric (particle-based) implementation of BP with a reduced complexity that will also be the basis for the particle-based CoSNAT algorithm presented in the next Chapter 3. The main difference to the conventional NBP algorithm [Ihler et al., 2005, Lien et al., 2012] is that it employs an efficient scheme for particle-based message multiplication which avoids kernel density estimate and whose complexity scales only linearly (rather than quadratically) with the number of particles. We note that an alternative message multiplication scheme that also avoids the use of kernel density estimates and whose complexity is linear in the number of particles was proposed in [Briers et al., 2005]. The method in [Briers et al., 2005] constructs an approximate importance function in order to calculate weighted particles for beliefs and messages. Our approach is different since the importance function is formed simply by "stacking" incoming beliefs, and the calculation of particles and weights for incoming messages is avoided.

2.4.1 Statement of Low-Complexity NBP

The low-complexity alternative to nonparametric BP using the dimension-augmented reformulation can be obtained as described in what follows: First, we rewrite (2.20) as

$$b^{(p)}(\mathbf{x}_{l,n}) = \int b^{(p)}(\bar{\mathbf{x}}_{l,n})\,\mathrm{d}\bar{\mathbf{x}}_{l,n}^{\sim l}\,, \tag{2.25}$$

where

$$b^{(p)}(\bar{\mathbf{x}}_{l,n}) \propto f(\bar{\mathbf{y}}_{l,n}|\bar{\mathbf{x}}_{l,n})\, f^{(p-1)}(\bar{\mathbf{x}}_{l,n}).$$

Based on (2.25), we obtain a PR $\left\{\left(\mathbf{x}_{l,n}^{(j)}, w_{l,n}^{(j)}\right)\right\}_{j=1}^{J}$ of $b^{(p)}(\mathbf{x}_{l,n})$ from a PR $\left\{\left(\bar{\mathbf{x}}_{l,n}^{(j)}, w_{l,n}^{(j)}\right)\right\}_{j=1}^{J}$ of $b^{(p)}(\bar{\mathbf{x}}_{l,n})$, which is obtained via importance sampling using

$$f^{(p-1)}(\bar{\mathbf{x}}_{l,n}) \triangleq \phi_{\to n}(\mathbf{x}_{l,n}) \prod_{l' \in \mathcal{M}_{l,n}^{\mathcal{S}}} b^{(p-1)}(\mathbf{x}_{l',n})$$

as proposal distribution. There is no need to explicitly draw particles $\left\{\bar{\mathbf{x}}_{l,n}^{(j)}\right\}_{j=1}^{J}$ from $f^{(p-1)}(\bar{\mathbf{x}}_{l,n})$ because such particles are already available: more specifically, they can be obtained simply by stacking the particles $\left\{\mathbf{x}_{l,n}^{(j)}\right\}_{j=1}^{J}$ representing the prediction message $\phi_{\to n}(\mathbf{x}_{l,n})$ (which were calculated by means of message filtering as described in Section 2.1.2) and the parti-

cles $\{\mathbf{x}_{l',n}^{(j)}\}_{j=1}^{J}$ representing the beliefs $b^{(p-1)}(\mathbf{x}_{l',n})$, $l' \in \mathcal{M}_{l,n}^{\mathcal{S}}$ (which were received from neighboring CAs). Using these particles $\{\bar{\mathbf{x}}_{l,n}^{(j)}\}_{j=1}^{J}$, we then obtain weights $w_{l,n}^{(j)}$ by calculating nonnormalized weights

$$\tilde{w}_{l,n}^{(j)} \propto b^{(p)}(\bar{\mathbf{x}}_{l,n}^{(j)}) / f^{(p-1)}(\bar{\mathbf{x}}_{l,n}^{(j)}),$$

i.e.,

$$\tilde{w}_{l,n}^{(j)} = \; f(\bar{\mathbf{y}}_{l,n}|\bar{\mathbf{x}}_{l,n}^{(j)}) = \prod_{l' \in \mathcal{M}_{l,n}^{\mathcal{S}}} f(\mathbf{y}_{l,l';n}|\mathbf{x}_{l,n}^{(j)}, \mathbf{x}_{l',n}^{(j)}), \tag{2.26}$$

and normalizing.

The set $\{(\bar{\mathbf{x}}_{l,n}^{(j)}, w_{l,n}^{(j)})\}_{j=1}^{J}$ is a PR of $b^{(p)}(\bar{\mathbf{x}}_{l,n})$. Hence, $\{(\mathbf{x}_{l,n}^{(j)}, w_{l,n}^{(j)})\}_{j=1}^{J}$ is a PR of

$$b^{(p)}(\mathbf{x}_{l,n}) \propto \int b^{(p)}(\bar{\mathbf{x}}_{l,n}) \, \mathrm{d}\bar{\mathbf{x}}_{l,n}^{\sim l}. \tag{2.27}$$

(This is because $\mathbf{x}_{l,n}^{(j)}$ can be obtained by discarding from $\bar{\mathbf{x}}_{l,n}^{(j)}$ all vectors $\mathbf{x}_{l',n}^{(j)}$ with $l' \neq l$, i.e., by discarding $\big[\mathbf{x}_{l',n}^{(j)}\big]_{l' \in \mathcal{M}_{l,n}^{\mathcal{S}}}$, which is the Monte Carlo implementation of the marginalization $b^{(p)}(\mathbf{x}_{l,n}) \propto \int b^{(p)}(\bar{\mathbf{x}}_{l,n}) \, \mathrm{d}\bar{\mathbf{x}}_{l,n}^{\sim l}$.)

Finally, a resampling [Ristic et al., 2004] is performed to obtain equally weighted particles representing $b^{(p)}(\mathbf{x}_{l,n})$. These particles are broadcast to all neighboring CAs $l' \in \mathcal{M}_{l,n}^{\mathcal{S}}$, where they are used to calculate the beliefs $b^{(p+1)}(\mathbf{x}_{l',n})$. This algorithm avoids kernel density estimation for the measurement messages.

2.4.2 Computation and Communication Requirements

In the following discussion of the computation and communication for the alternative particle-based BP algorithm, we assume for simplicity that all states $\mathbf{x}_{l,n}$, $l \in \mathcal{S}$ have identical dimension L at all times. Particle-based filtering algorithms in general suffers from an exponential scaling of the computational complexity in the dimension L, which is known as curse of dimensionality [Doucet et al., 2001, Daum and Huang, 2003]. However, as analyzed in [Daum and Huang, 2003] the curse of dimensionality is avoided if the proposal $f^{(p-1)}(\bar{\mathbf{x}}_{l,n})$ strongly resembles the belief $b^{(p)}(\bar{\mathbf{x}}_{l,n})$. If this is the case for our alternative particle-based BP algorithm can not be answered in general, since it depends mainly on the variance of the process noise $\mathbf{q}_{l',n}$ for all $l' \in \{l\} \cup \mathcal{M}_{l,n}$ (c.f. (1.1)) and on the availability of informative beliefs $b^{(p)}(\bar{\mathbf{x}}_{l',n-1})$, $l' \in \{l\} \cup \mathcal{M}_{l,n}$ from the prior time step $n-1$.

Furthermore, it is straightforward that the computational complexity of the evaluation for all J particles in (2.26) scales as $\mathcal{O}\big(\big|\mathcal{M}_{l,n}^{\mathcal{S}}\big| J\big)$, i.e., only linearly in the number of particles J. (The conventional NBP described in Section 2.1.2 scales quadratically in the number of particles J.) The dimension of the distribution $b^{(p)}(\bar{\mathbf{x}}_{l,n})$ involved in the importance sampling scheme is $\big(\big|\mathcal{M}_{l,n}^{\mathcal{S}}\big| + 1\big) L$, and thus higher than that of $b^{(p)}(\mathbf{x})$ in the case of NBP (cf. (2.8)). Nevertheless, we will see in Section 2.6 that for the simulated setting where the number of neighbors $\big|\mathcal{M}_{l,n}^{\mathcal{S}}\big|$ is moderately large, using a similar number of particles J as for conventional NBP reviewed in Section 2.1.2 yields a comparable accuracy at a strongly reduced runtime.

The communication requirements of both NBP methods are equal and relatively high. More specifically, at time n and message passing iteration p, CA l receives JL real values corresponding to the particles representing the belief $b^{(p-1)}(\bar{\mathbf{x}}_{l',n})$ from all $l' \in \mathcal{M}_{l,n}^{\mathcal{S}}$(this is needed to calculate the belief $b^{(p)}(\mathbf{x}_{l,n})$), and it broadcasts JL real values representing the belief $b^{(p-1)}(\bar{\mathbf{x}}_{l',n})$. If the measurement model in (2.38) involves only substates $\boldsymbol{\lambda}_{l,n}$ of the states $\mathbf{x}_{l,n}$ only subparticles corresponding to $\boldsymbol{\lambda}_{l,n}$ have to be transmitted. Due to the higher dimensional problem, in the new NBP scheme, the number of particles J is typically higher compared to standard NBP. Therefore, the lower complexity comes at the cost of higher communication requirements (this drawback can however be avoided by using a parametric message representation for communication [Savic and Zazo, 2012]).

2.5 Sigma Point Belief Propagation (SPBP)

In this section, we present SPBP, a new low-complexity approximation of BP which is a promising approach for cooperative navigation. It is based on the dimension-augmented reformulation of BP and extends the sigma point filter to general factor structures. We start by reviewing the basic principles of sigma points and state then the SPBP algorithm using the reformulated BP message passing equations from Section 2.3. SPBP is is particular interesting for cooperative navigation due to its low computation and communication requirements

2.5.1 Review: Sigma Point Basics

In this section we denote by $\mathbf{x} \in \mathbb{R}^L$ a general (non-Gaussian) random vector whose mean $\boldsymbol{\mu}_{\mathbf{x}} = \mathsf{E}\{\mathbf{x}\}$ and covariance matrix $\mathbf{C}_{\mathbf{x}} = \mathsf{E}\{(\mathbf{x}-\boldsymbol{\mu}_{\mathbf{x}})(\mathbf{x}-\boldsymbol{\mu}_{\mathbf{x}})^{\mathrm{T}}\}$ are known, and a transformed random vector $\mathbf{z} = \Gamma(\mathbf{x})$, where $\Gamma(\cdot)$ is a generally nonlinear function. Sigma points $\{\mathbf{x}^{(j)}\}_{j=0}^{2L}$ and corresponding weights $\{w_{\mathrm{m}}^{(j)}\}_{j=0}^{2L}$ and $\{w_{\mathrm{c}}^{(j)}\}_{j=0}^{2L}$ are chosen such that the weighted sample mean

$$\tilde{\boldsymbol{\mu}}_{\mathbf{x}} = \sum_{j=0}^{2L} w_{\mathrm{m}}^{(j)} \mathbf{x}^{(j)}$$

and weighted sample covariance matrix

$$\tilde{\mathbf{C}}_{\mathbf{x}} = \sum_{j=0}^{2L} w_{\mathrm{c}}^{(j)} (\mathbf{x}^{(j)} - \tilde{\boldsymbol{\mu}}_{\mathbf{x}})(\mathbf{x}^{(j)} - \tilde{\boldsymbol{\mu}}_{\mathbf{x}})^{\mathrm{T}} \tag{2.28}$$

are exactly equal to $\boldsymbol{\mu}_{\mathbf{x}}$ and $\mathbf{C}_{\mathbf{x}}$, respectively. Closed-form expressions of the sigma points and weights are provided in [Wan and van der Merwe, 2001]. The spread of the sigma points around the mean $\boldsymbol{\mu}_{\mathbf{x}}$ can be adjusted via tuning parameters, whose choice depends on the dimension L of $\mathbf{x}$ [Julier and Uhlmann, 1997, Wan and van der Merwe, 2001]. Next, each sigma point is propagated through $\Gamma(\cdot)$, resulting in $\mathbf{z}^{(j)} = \Gamma\big(\mathbf{x}^{(j)}\big)$, $j \in \{0, \ldots, 2L\}$ ("unscented transformation"). The set $\big\{\big(\mathbf{x}^{(j)}, \mathbf{z}^{(j)}, w_{\mathrm{m}}^{(j)}, w_{\mathrm{c}}^{(j)}\big)\big\}_{j=0}^{2L}$ then represents the joint second-order statistics of $\mathbf{x}$ and $\mathbf{z}$ in an approximate manner. In particular, $\boldsymbol{\mu}_{\mathbf{z}}$, $\mathbf{C}_{\mathbf{z}}$, and $\mathbf{C}_{\mathbf{xz}} = \mathsf{E}\{(\mathbf{x}-\boldsymbol{\mu}_{\mathbf{x}})(\mathbf{z}-\boldsymbol{\mu}_{\mathbf{z}})^{\mathrm{T}}\}$ are

approximated by

$$\tilde{\boldsymbol{\mu}}_{\mathbf{z}} = \sum_{j=0}^{2L} w_{\mathrm{m}}^{(j)} \mathbf{z}^{(j)} \tag{2.29}$$

$$\tilde{\mathbf{C}}_{\mathbf{z}} = \sum_{j=0}^{2L} w_{\mathrm{c}}^{(j)} (\mathbf{z}^{(j)} - \tilde{\boldsymbol{\mu}}_{\mathbf{z}})(\mathbf{z}^{(j)} - \tilde{\boldsymbol{\mu}}_{\mathbf{z}})^{\mathrm{T}} \tag{2.30}$$

$$\tilde{\mathbf{C}}_{\mathbf{xz}} = \sum_{j=0}^{2L} w_{\mathrm{c}}^{(j)} (\mathbf{x}^{(j)} - \tilde{\boldsymbol{\mu}}_{\mathbf{x}})(\mathbf{z}^{(j)} - \tilde{\boldsymbol{\mu}}_{\mathbf{z}})^{\mathrm{T}}. \tag{2.31}$$

It has been shown in [Julier and Uhlmann, 1997] and [Wan and van der Merwe, 2001] that these approximations are at least as accurate as those resulting from a linearization (first-order Taylor series approximation) of $\Gamma(\cdot)$. Note also that the number $2L + 1$ of sigma points grows linearly with the dimension of $\mathbf{x}$ and is typically much smaller than the number of random samples in a particle representation.

Next, we consider the use of sigma points for Bayesian estimation of a random vector $\mathbf{x}$ from an observed vector

$$\mathbf{y} = \mathbf{z} + \mathbf{n}\,, \quad \text{with } \mathbf{z} = \Gamma(\mathbf{x})\,.$$

Here, the noise [2] $\mathbf{n}$ is statistically independent of $\mathbf{x}$ and generally non-Gaussian, with zero mean and known covariance matrix $\mathbf{C}_{\mathbf{n}}$. Bayesian estimation relies on the posterior PDF

$$f(\mathbf{x}|\mathbf{y}) \propto f(\mathbf{y}|\mathbf{x}) f(\mathbf{x})\,, \tag{2.32}$$

where $f(\mathbf{y}|\mathbf{x})$ is the likelihood function and $f(\mathbf{x})$ is the prior PDF. Direct calculation of (2.32) is usually infeasible. An important exception is the case where $\mathbf{x}$ and $\mathbf{n}$ are Gaussian random vectors and $\Gamma(\mathbf{x}) = \Gamma\mathbf{x}$ with some known matrix Γ. Then $f(\mathbf{x}|\mathbf{y})$ is also Gaussian, and the posterior mean $\boldsymbol{\mu}_{\mathbf{x}|\mathbf{y}}$ and posterior covariance matrix $\mathbf{C}_{\mathbf{x}|\mathbf{y}}$ can be calculated as

$$\boldsymbol{\mu}_{\mathbf{x}|\mathbf{y}} = \boldsymbol{\mu}_{\mathbf{x}} + \mathbf{K}(\mathbf{y} - \boldsymbol{\mu}_{\mathbf{z}})\,, \quad \mathbf{C}_{\mathbf{x}|\mathbf{y}} = \mathbf{C}_{\mathbf{x}} - \mathbf{K}(\mathbf{C}_{\mathbf{z}} + \mathbf{C}_{\mathbf{n}})\mathbf{K}^{\mathrm{T}}, \tag{2.33}$$

where

$$\boldsymbol{\mu}_{\mathbf{z}} = \Gamma\boldsymbol{\mu}_x\,, \quad \mathbf{C}_{\mathbf{z}} = \Gamma\mathbf{C}_{\mathbf{x}}\Gamma^{\mathrm{T}} \tag{2.34}$$

and

$$\mathbf{K} = \mathbf{C}_{\mathbf{xz}}(\mathbf{C}_{\mathbf{z}} + \mathbf{C}_{\mathbf{n}})^{-1}, \quad \text{with } \mathbf{C}_{\mathbf{xz}} = \mathbf{C}_{\mathbf{x}}\Gamma^{\mathrm{T}}. \tag{2.35}$$

These expressions are used in the measurement update step of the Kalman filter [Haykin, 2001]. The minimum mean-square error estimate of $\mathbf{x}$ is given by $\boldsymbol{\mu}_{\mathbf{x}|\mathbf{y}}$, and a characterization of the accuracy of estimation by $\mathbf{C}_{\mathbf{x}|\mathbf{y}}$.

In the general case of nonlinear $\Gamma(\cdot)$, the basic approximation underlying the extended

[2] Note that for simplicity we here use additive noise. As described in [Wan and van der Merwe, 2001], the reviewed procedure can easily be extended to a general noise model.

Kalman filter [Haykin, 2001] is obtained by using (essentially) (2.33)–(2.35) with Γ being the Jacobian matrix resulting from a linearization of $\Gamma(\cdot)$. A more accurate alternative is to approximate $\boldsymbol{\mu}_{\mathbf{x}|\mathbf{y}}$ and $\mathbf{C}_{\mathbf{x}|\mathbf{y}}$ by means of sigma points. For this, we use (2.33) and the first equation in (2.35), with $\boldsymbol{\mu}_{\mathbf{z}}$, $\mathbf{C}_{\mathbf{z}}$, and $\mathbf{C}_{\mathbf{xz}}$ replaced by the sigma point approximations $\tilde{\boldsymbol{\mu}}_{\mathbf{z}}$, $\tilde{\mathbf{C}}_{\mathbf{z}}$, and $\tilde{\mathbf{C}}_{\mathbf{xz}}$ in (2.29)–(2.31). This gives

$$\tilde{\boldsymbol{\mu}}_{\mathbf{x}|\mathbf{y}} = \boldsymbol{\mu}_{\mathbf{x}} + \tilde{\mathbf{K}}(\mathbf{y} - \tilde{\boldsymbol{\mu}}_{\mathbf{z}})\,, \quad \tilde{\mathbf{C}}_{\mathbf{x}|\mathbf{y}} = \mathbf{C}_{\mathbf{x}} - \tilde{\mathbf{K}}(\tilde{\mathbf{C}}_{\mathbf{z}} + \mathbf{C}_{\mathbf{n}})\tilde{\mathbf{K}}^{\mathrm{T}}, \tag{2.36}$$

with $\tilde{\mathbf{K}} = \tilde{\mathbf{C}}_{\mathbf{xz}}(\tilde{\mathbf{C}}_{\mathbf{z}} + \mathbf{C}_{\mathbf{n}})^{-1}$. We thus obtain the following approximate sigma point implementation of (2.32).

Step 1: Sigma points and weights $\big\{\big(\mathbf{x}^{(j)}, w_{\mathrm{m}}^{(j)}, w_{\mathrm{c}}^{(j)}\big)\big\}_{j=0}^{2L}$ are calculated from $\boldsymbol{\mu}_{\mathbf{x}}$ and $\mathbf{C}_{\mathbf{x}}$ [Wan and van der Merwe, 2001].

Step 2: The transformed sigma points $\mathbf{z}^{(j)} = \Gamma\big(\mathbf{x}^{(j)}\big)$, $j \in \{0, \ldots, 2L\}$ are calculated.

Step 3: From $\big\{\big(\mathbf{x}^{(j)}, \mathbf{z}^{(j)}, w_{\mathrm{m}}^{(j)}, w_{\mathrm{c}}^{(j)}\big)\big\}_{j=0}^{2L}$, the means and covariances $\tilde{\boldsymbol{\mu}}_{\mathbf{z}}$, $\tilde{\mathbf{C}}_{\mathbf{z}}$, and $\tilde{\mathbf{C}}_{\mathbf{xz}}$ in (2.29)–(2.31) and, in turn, $\tilde{\boldsymbol{\mu}}_{\mathbf{x}|\mathbf{y}}$ and $\tilde{\mathbf{C}}_{\mathbf{x}|\mathbf{y}}$ in (2.36) are calculated.

2.5.2 Statement of the SPBP Algorithm

We will now derive the SPBP algorithm for cooperative navigation. For simplicity, we perform this derivation using additive noise in both measurement model and motion model[3]. More specifically, instead of the motion model in (1.1) and the measurement model in (1.2) we use

$$\mathbf{x}_{l,n} = g_l(\mathbf{x}_{l,n-1}) + \mathbf{q}_{l,n}\,, \quad l \in \mathcal{S}\,, \tag{2.37}$$

and

$$\mathbf{y}_{l,l';n} = d_l(\mathbf{x}_{l,n}, \mathbf{x}_{l',n}) + \mathbf{v}_{l,l';n}\,, \quad l \in \mathcal{S},\ l' \in \mathcal{M}_{l,n}^{\mathcal{S}}\,, \tag{2.38}$$

respectively.

Prediction

First, we discuss how a approximate mean $\boldsymbol{\mu}_{\mathbf{x}_{l,n}}$ and covariance matrix $\mathbf{C}_{\mathbf{x}_{l,n}}$ corresponding to the normalized prediction message $\phi_{\to n}(\mathbf{x}_{l,n})$ in (2.6) can be obtained from the approximate mean $\boldsymbol{\mu}_{b(\mathbf{x}_{l,n-1})}^{(P)}$ and covariance matrix $\mathbf{C}_{b(\mathbf{x}_{l,n-1})}^{(P)}$ corresponding to $b^{(P)}(\mathbf{x}_{l,n-1})$ at CA $l \in \mathcal{S}$. Note, that this procedure is identical to the prediction step in the sigma point filter [Wan and van der Merwe, 2001]. The following steps are now performed for prediction (note that the first three steps are analogous to those in Section 2.5.1).

Step 1: Sigma points and weights $\big\{\big(\mathbf{x}_{l,n}^{(j)}, w_{\mathrm{m}}^{(j)}, w_{\mathrm{c}}^{(j)}\big)\big\}_{j=0}^{2L_{l,n}}$ corresponding to $b^{(P)}(\mathbf{x}_{l,n-1})$ are calculated from $\boldsymbol{\mu}_{b(\mathbf{x}_{l,n-1})}^{(P)}$ and $\mathbf{C}_{b(\mathbf{x}_{l,n-1})}^{(P)}$ [Wan and van der Merwe, 2001].

Step 2: The transformed sigma points $\mathbf{z}_{l,n}^{(j)} = g\big(\mathbf{x}_{l,n}^{(j)}\big)$, $j \in \{0, \ldots, 2L_{l,n}\}$ are calculated.

[3]The proposed algorithm can easily extended to a general noise model [Wan and van der Merwe, 2001]

Step 3: From $\big\{\big(\mathbf{z}_{l,n}^{(j)}, w_{\mathrm{m}}^{(j)}, w_{\mathrm{c}}^{(j)}\big)\big\}_{j=0}^{2L_{l,n}}$, the mean and covariance matrix $\boldsymbol{\mu}_{\mathbf{z}_{l,n}}^{(p)}$ and $\mathbf{C}_{\mathbf{z}_{l,n}}^{(p)}$ are calculated as in (2.29) and (2.30).

Step 4: Finally, the approximate mean $\boldsymbol{\mu}_{\mathbf{x}_{l,n}}$ and covariance matrix $\mathbf{C}_{\mathbf{x}_{l,n}}$ corresponding to $\phi_{\to n}(\mathbf{x}_{l,n})$ are obtained as

$$\boldsymbol{\mu}_{\mathbf{x}_{l,n}} = \boldsymbol{\mu}_{\mathbf{z}_{l,n}} + \boldsymbol{\mu}_{\mathbf{q}} \qquad\qquad \mathbf{C}_{\mathbf{x}_{l,n}} = \mathbf{C}_{\mathbf{z}_{l,n}} + \mathbf{C}_{\mathbf{q}}.$$

Here, $\boldsymbol{\mu}_{\mathbf{q}}$ and $\mathbf{C}_{\mathbf{q}}$ are mean and covariance matrix of the driving noise $\mathbf{q}_{l,n}$, respectively. Note that in case the motion model is linear, i.e., (2.37) can be written as $\mathbf{x}_{l,n} = \mathbf{G}_l\mathbf{x}_{l,n-1} + \mathbf{q}_{l,n}$, Steps 1–4 can be avoided and we directly get

$$\boldsymbol{\mu}_{\mathbf{x}_{l,n}} = \mathbf{G}_l\boldsymbol{\mu}_{\mathbf{x}_{l,n-1}} + \boldsymbol{\mu}_{\mathbf{q}} \qquad\qquad \mathbf{C}_{\mathbf{x}_{l,n}} = \mathbf{G}_l\mathbf{C}_{\mathbf{x}_{l,n-1}}\mathbf{G}_l^{\mathrm{T}} + \mathbf{C}_{\mathbf{q}}.$$

Measurement Update

Now, to develop an sigma-point-based approximate calculation of $b^{(p)}(\mathbf{x}_{l,n})$, we rewrite (2.20) as

$$b^{(p)}(\mathbf{x}_{l,n}) = \int b^{(p)}(\bar{\mathbf{x}}_{l,n})\,\mathrm{d}\bar{\mathbf{x}}_{l,n}^{\sim l}\,, \tag{2.39}$$

where

$$b^{(p)}(\bar{\mathbf{x}}_{l,n}) \propto f(\bar{\mathbf{y}}_{l,n}|\bar{\mathbf{x}}_{l,n})\,f^{(p-1)}(\bar{\mathbf{x}}_{l,n}). \tag{2.40}$$

Note that the dimension of $\bar{\mathbf{x}}_{l,n}$ is $\bar{L}_{l,n} \triangleq L_{l,n} + \sum_{l'\in|\mathcal{M}_l^{\mathcal{S}}|} L_{l',n}$, where $L_{l,n}$ denotes the dimension of $\mathbf{x}_{l,n}$. Because the expression (2.40) of the "composite belief" $b^{(p)}(\bar{\mathbf{x}}_{l,n})$ is analogous to (2.32), we can obtain an approximate sigma point representation of $b^{(p)}(\bar{\mathbf{x}}_{l,n})$ in a similar way as we obtained an approximate sigma points representation of $f(\mathbf{x}|\mathbf{y})$ in Section 2.5.1. We first specify the "neighbor" set of CA $l \in \mathcal{S}$ as $\mathcal{M}_{l,n}^{\mathcal{S}} = \{l_1, l_2, \ldots, l_{|\mathcal{M}_{l,n}^{\mathcal{S}}|}\}$ and define a mean vector and a covariance matrix corresponding to the "composite prior" $f^{(p-1)}(\bar{\mathbf{x}}_{l,n}) \propto \phi_{\to n}(\mathbf{x}_{l,n})\prod_{l'\in\mathcal{M}_{l,n}^{\mathcal{S}}} b^{(p-1)}(\mathbf{x}_{l',n})$:

$$\boldsymbol{\mu}_{\bar{\mathbf{x}}_{l,n}}^{(p-1)} \triangleq \Big(\boldsymbol{\mu}_{\mathbf{x}_{l,n}}^{\mathrm{T}}\ \boldsymbol{\mu}_{l_1',n}^{(p-1)\mathrm{T}}\ \boldsymbol{\mu}_{l_2',n}^{(p-1)\mathrm{T}} \cdots \boldsymbol{\mu}_{l'_{|\mathcal{M}_{l,n}^{\mathcal{S}}|},n}^{(p-1)\mathrm{T}}\Big)^{\mathrm{T}} \tag{2.41}$$

$$\mathbf{C}_{\bar{\mathbf{x}}_{l,n}}^{(p-1)} \triangleq \mathrm{diag}\Big\{\mathbf{C}_{\mathbf{x}_{l,n}}, \mathbf{C}_{l_1',n}^{(p-1)}, \mathbf{C}_{l_2',n}^{(p-1)}, \ldots, \mathbf{C}_{l'_{|\mathcal{M}_{l,n}^{\mathcal{S}}|},n}^{(p-1)}\Big\}. \tag{2.42}$$

Here, we interpreted $\prod_{l'\in\mathcal{M}_{l,n}^{\mathcal{S}}} b^{(p-1)}(\mathbf{x}_{l',n})$ as the product of the PDFs of statistically independent random variables; furthermore, $\boldsymbol{\mu}_{l_i',n}^{(p-1)}$ and $\mathbf{C}_{l_i',n}^{(p-1)}$ are the mean and covariance matrix of $b^{(p-1)}(\mathbf{x}_{l_i',n})$; and $\mathrm{diag}\{\cdot\}$ denotes the block diagonal matrix whose diagonal blocks are the listed matrices. The following steps are now performed for measurement update (note that the first three steps are analogous to those in Section 2.5.1).

Step 1: Sigma points and weights $\big\{\big(\bar{\mathbf{x}}_{l,n}^{(j)}, w_{\mathrm{m}}^{(j)}, w_{\mathrm{c}}^{(j)}\big)\big\}_{j=0}^{2\bar{L}_{l,n}}$ corresponding to $f^{(p-1)}(\bar{\mathbf{x}}_{l,n})$ are calculated from $\boldsymbol{\mu}_{\bar{\mathbf{x}}_{l,n}}^{(p-1)}$ and $\mathbf{C}_{\bar{\mathbf{x}}_{l,n}}^{(p-1)}$ [Wan and van der Merwe, 2001]. (Note that the dimension

and number of the sigma points depend on the number of neighbors $|\mathcal{M}_{l,n}^{\mathcal{S}}|$, and thus the tuning parameters that adjust the spread of the sigma points should be adapted to $|\mathcal{M}_{l,n}^{\mathcal{S}}|$.)

Step 2: The transformed sigma points $\bar{\mathbf{z}}_{l,n}^{(j)} = D_{l,n}\big(\bar{\mathbf{x}}_{l,n}^{(j)}\big)$, $j \in \{0, \ldots, 2\bar{L}_{l,n}\}$ are calculated.

Step 3: From $\big\{\big(\bar{\mathbf{x}}_{l,n}^{(j)}, \bar{\mathbf{z}}_{l,n}^{(j)}, w_{\mathrm{m}}^{(j)}, w_{\mathrm{c}}^{(j)}\big)\big\}_{j=0}^{2\bar{L}_{l,n}}$, the means and covariances $\boldsymbol{\mu}_{\bar{\mathbf{z}}_{l,n}}^{(p)}$, $\mathbf{C}_{\bar{\mathbf{z}}_{l,n}}^{(p)}$, and $\mathbf{C}_{\bar{\mathbf{x}}_{l,n}\bar{\mathbf{z}}_{l,n}}^{(p)}$ are calculated as in (2.29)–(2.31). Subsequently, $\boldsymbol{\mu}_{b(\bar{\mathbf{x}}_{l,n})}^{(p)}$ and $\mathbf{C}_{b(\bar{\mathbf{x}}_{l,n})}^{(p)}$ (the sigma point approximations of the mean and covariance matrix of $b^{(p)}(\bar{\mathbf{x}}_{l,n})$) are calculated as in (2.36), using $\boldsymbol{\mu}_{\bar{\mathbf{z}}_{l,n}}^{(p)}$, $\mathbf{C}_{\bar{\mathbf{z}}_{l,n}}^{(p)}$, and $\mathbf{C}_{\bar{\mathbf{x}}_{l,n}\bar{\mathbf{z}}_{l,n}}^{(p)}$ instead of $\boldsymbol{\mu}_{\mathbf{z}}$, $\mathbf{C}_{\mathbf{z}}$, and $\mathbf{C}_{\mathbf{xz}}$, respectively.

Step 4: From $\boldsymbol{\mu}_{b(\bar{\mathbf{x}}_{l,n})}^{(p)}$ and $\mathbf{C}_{b(\bar{\mathbf{x}}_{l,n})}^{(p)}$, the elements related to $\mathbf{x}_{l,n}$ are extracted (this corresponds to the marginalization (2.39)). More specifically, the approximate mean $\boldsymbol{\mu}_{b(\mathbf{x}_{l,n})}^{(p)}$ and covariance matrix $\mathbf{C}_{b(\mathbf{x}_{l,n})}^{(p)}$ of the "marginal belief" $b^{(p)}(\mathbf{x}_{l,n})$ are given by the first $L_{l,n}$ elements of $\boldsymbol{\mu}_{b(\bar{\mathbf{x}}_{l,n})}^{(p)}$ and the upper-left $L_{l,n}\times L_{l,n}$ submatrix of $\mathbf{C}_{b(\bar{\mathbf{x}}_{l,n})}^{(p)}$, respectively (cf. the stacked structure of $\boldsymbol{\mu}_{\bar{\mathbf{x}}_{l,n}}^{(p-1)}$ in (2.41) and the block structure of $\mathbf{C}_{\bar{\mathbf{x}}_{l,n}}^{(p-1)}$ in (2.42)).

2.5.3 Computation and Communication Requirements

As in Section 2.4.2 for NBP, in the following discussion of the computation and communication requirements of the proposed SPBP algorithm, we assume for simplicity that all states $\mathbf{x}_{l,n}$, $l \in \mathcal{S}$ have identical dimension L at all times. Similar to the sigma point filter [Wan and van der Merwe, 2001], SPBP requires the computation of the square root of the $\bar{L}\times\bar{L}$ matrices $\mathbf{C}_{\bar{\mathbf{x}}_{l,n}}^{(p-1)}$ to calculate the sigma points $\bar{\mathbf{x}}_{l,n}^{(j)}$ in Step 1. This is the most complex part of the SPBP algorithm. An efficient computation of the matrix square root uses the Cholesky decomposition [Press et al., 1992], whose complexity is cubic in $\bar{L} = |\mathcal{M}_{l,n}^{\mathcal{S}}|(L+1)$. Thus, the complexity of SPBP at one CA l and one time step n is cubic in $\big|\mathcal{M}_{l,n}^{\mathcal{S}}\big|$ and, also, in the number of sigma points (which is $2\bar{L}+1$). The complexity of NBP is linear in $\big|\mathcal{M}_{l,n}^{\mathcal{S}}\big|$. Furthermore, it is quadratic in the number of particles [Lien et al., 2012] for the conventional NBP and it is linear in the number of particles for the low-complexity alternative presented in Section 2.4. However, in NBP the number of particles is usually much higher than the number of sigma points in SPBP. Moreover, the quadratic and cubic complexity terms of the Cholesky decomposition are rather small (about $\bar{L}^3/6$ multiplications, $\bar{L}^2/2$ divisions, and $\bar{L}$ square root operations are used [Press et al., 1992]). Therefore, as also investigate in the next Section 2.6 for cooperative navigation, SPBP is significantly less complex than the conventional NBP implementation and its complexity is comparable to that of the alternative BP implementation in Section 2.4.

SPBP has very low communication requirements. Because the beliefs $b^{(p)}(\mathbf{x}_{l,n})$ are represented by a mean vector and a covariance matrix, at most $L + \frac{L(L+1)}{2} = \frac{L(L+3)}{2}$ real values per message passing iteration $p \in \{1, \ldots, P\}$ have to be transmitted from CA l to neighboring CAs, rather than hundreds or thousands of particles in NBP. More specifically, at message passing iteration p, CA l receives $\boldsymbol{\mu}_{l'}^{(p-1)}$ and $\mathbf{C}_{l'}^{(p-1)}$ from all $l' \in \mathcal{M}_{l,n}^{\mathcal{S}}$ (this is needed to calculate $\boldsymbol{\mu}_{\bar{\mathbf{x}}_l}^{(p-1)}$ and $\mathbf{C}_{\bar{\mathbf{x}}_l}^{(p-1)}$, see (2.41) and (2.42)), and it broadcasts $\boldsymbol{\mu}_{l}^{(p-1)}$ and $\mathbf{C}_{l}^{(p-1)}$ to all $l' \in \mathcal{M}_{l,n}^{\mathcal{S}}$. These communications are a precondition for Step 1 of the SPBP algorithm. If the measurement model in (2.38) involves only substates $\boldsymbol{\lambda}_{l,n}$ of the states $\mathbf{x}_{l,n}$, only the mean and covariance

matrix corresponding to $\boldsymbol{\lambda}_{l,n}$ have to be transmitted.

2.6 Simulation Results

We simulated a decentralized, cooperative navigation scenario using a network of $|\mathcal{S}|=5$ CAs, of which three are mobile and two are static anchors, i.e., CAs with perfect position information. The state of mobile CA $l \in \{1,2,3\}$ at time $n = 1,\ldots,100$ consists of the position $\boldsymbol{\lambda}_{l,n} \triangleq (x_{1,l,n}\ x_{2,l,n})^{\mathrm{T}}$ and the velocity, i.e., $\mathbf{x}_{l,n} \triangleq (x_{1,l,n}\ x_{2,l,n}\ \dot{x}_{1,l,n}\ \dot{x}_{2,l,n})^{\mathrm{T}}$. Each mobile CA moves within a field of size 50×50, performs distance measurements relative to all other CAs, communicates the mean and covariance matrix of its current position to all other CAs, and estimates its own state. We assume that each mobile CA is able to associate its measurements with the individual CAs. Each anchor CA $l \in \{4,5\}$ communicates its own (true) position $\bar{\boldsymbol{\lambda}}_l$. The distance measurement of mobile CA $l \in \{1,2,3\}$ relative to CA l' at time n is (cf. (2.38))

$$y_{l,l';n} = \begin{cases} \|\boldsymbol{\lambda}_{l,n} - \boldsymbol{\lambda}_{l',n}\| + v_{l,l';n}\,, & l' \in \{1,2,3\}\setminus\{l\} \\ \|\boldsymbol{\lambda}_{l,n} - \bar{\boldsymbol{\lambda}}_{l'}\| + v_{l,l';n}\,, & l' \in \{4,5\}\,, \end{cases}$$

where $v_{l,l';n}$ is zero-mean Gaussian measurement noise with variance $\sigma_n^2 = 1$.

The states of the mobile CAs evolve independently according to [Li and Jilkov, 2003]

$$\mathbf{x}_{l,n} = \mathbf{G}\mathbf{x}_{l,n-1} + \mathbf{W}\mathbf{q}_{l,n}, \quad n = 1,2,\ldots. \tag{2.43}$$

Here, the matrices $\mathbf{G}$ and $\mathbf{W}$ are given by

$$\mathbf{G} = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \quad \mathbf{W} = \begin{pmatrix} 0.5 & 0 \\ 0 & 0.5 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}, \tag{2.44}$$

and the driving noise vectors $\mathbf{q}_{l,n} \in \mathbb{R}^2$ are Gaussian, i.e., $\mathbf{q}_{l,n} \sim \mathcal{N}(\mathbf{0}, \sigma_q^2\mathbf{I})$, with variance $\sigma_q^2 = 10^{-4}$. In the generation of the state sequences, this recursive evolution of the $\mathbf{x}_{l,n}$ was initialized with $\mathbf{x}_{1,0} = (0\ 0\ 0.1\ 0.5)^{\mathrm{T}}$, $\mathbf{x}_{2,0} = (25\ 50\ 0.25\ -0.4)^{\mathrm{T}}$, and $\mathbf{x}_{3,0} = (50\ 0\ -.5\ 0.2)^{\mathrm{T}}$. The anchor CAs are located at $\bar{\boldsymbol{\lambda}}_4 = (0\ 25)^{\mathrm{T}}$ and $\bar{\boldsymbol{\lambda}}_5 = (50\ 25)^{\mathrm{T}}$ for all n. Fig. 2.3 shows the network topology used for the simulations and example realizations of the mobile CA trajectories. In the simulation of the various cooperative localization algorithms, for the mobile CAs, we used the initial prior PDF $f(\mathbf{x}_{l,0}) = \mathcal{N}(\boldsymbol{\mu}_{l,0}, \mathbf{C}_{l,0})$. Here, $\mathbf{C}_{l,0} = \mathrm{diag}\{1,1,0.01,0.01\}$ represents the uncertainty in knowing $\mathbf{x}_{l,0}$, and $\boldsymbol{\mu}_{l,0}$ is a random hyperparameter that was randomly sampled (for each simulation run) from $\mathcal{N}(\mathbf{x}_{l,0}, \mathbf{C}_{l,0})$. For the anchor CAs, the true positions were used. The number of message passing iterations p at each time n was set to $P = 2$.

We compare the proposed conventional NBP algorithm (NBP-1) with the proposed alternative NBP algorithm (NBP-2) and the SPBP algorithm (using 25 sigma points). All three

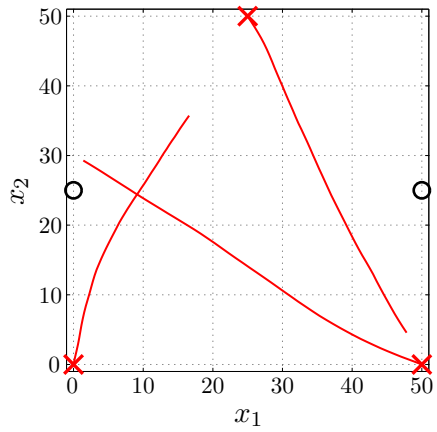


Figure 2.3: Network topology used for the simulations and example realizations of the trajectories of the three mobile CA. Initial mobile CA positions are indicated by crosses, and anchor positions are indicated by circles.

methods use the same message passing scheme (2.5)–(2.7) known as SPAWN. The NBP methods use $J = 500$, or $J = 5000$ particles. In NBP-1, the bandwidth of the Gaussian kernels was equal to the measurement noise variance $\sigma_n^2 = 1$ [Ihler et al., 2005]. Fig. 2.4 shows the simulated average root-mean-square error (ARMSE) of the various methods for $n = 1, \ldots, 100$. This error was determined by averaging over the three mobile CAs and 1000 simulation runs.

It is seen that, for $J = 500$, SPBP outperforms the two NBP methods. However, if the number of particles used in the NBP methods is increased to $J = 5000$ NBP-1 outperforms SPBP and NBP-2 performs equally well than SPBP. Note that, the performance advantages of NBP over SPBP are expected to be larger in the case of a stronger nonlinear measurement model or a nonlinear motion model.

The average runtime of our SPBP implementation on an Intel Xeon X5650 CPU, for all 100 time steps of one simulation run, was 0.81s. The average runtime of NBP-1 was 10.38s, and 842.02s (for 500, and 5000 particles, respectively), that of NBP-2 was 0.27s, and 1.23s. Thus, the improved performance of NBP-1 using 5000 particles over SPBP come at the cost of a dramatically increased computational complexity. Finally, it is interesting to see, that SPBP and NBP-2, which perform equally well, do also have a similar computational complexity.

With SPBP, since our measurement model involves only the two-dimensional position $\boldsymbol{\lambda}_{l,n}$, each mobile CA broadcasts the mean vector and covariance matrix of $b^{(p)}(\boldsymbol{\lambda}_{n,l}) = \int\int b^{(p)}(\mathbf{x}_{n,l})$ $\mathrm{d}\dot{x}_{1,n,l}\mathrm{d}\dot{x}_{2,n,l}$ at each message passing iteration p, corresponding to $2 + 3 = 5$ real values. By contrast, for the NBP methods with 500, and 5000 particles, the number of real values broadcast by each mobile CA at each message passing iteration is 1000, and 10000, respectively. Thus, SPBP requires significantly less communications than the NBP methods. However, the drawback of higher communication requirements of NBP methods can be avoided by using a parametric message representation for communication [Savic and Zazo, 2012]. (In all three methods, each anchor CA broadcasts its position, corresponding to two real values; however,

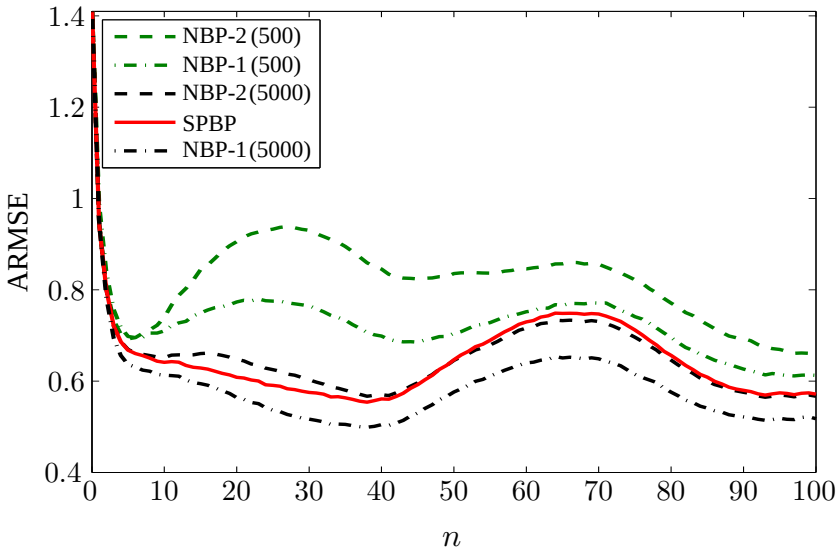


Figure 2.4: ARMSE of the simulated self-localization algorithms versus time n.

this is a preparatory step that is executed only once.)

Chapter 3

Cooperative Simultaneous Navigation and Tracking (CoSNAT)

This chapter introduces a BP-based framework and methodology for CoSNAT, which, for the first time, provide a consistent combination of cooperative navigation and distributed tracking in decentralized mobile agent networks. In CoSNAT, mobile CAs track single or multiple targets while simultaneously localizing themselves. This is based on pairwise measurements between CAs and targets as well as between CAs.

We start by reviewing distributed target tracking. Then, we propose an extension of the BP message passing scheme for cooperative navigation discussed in Chapter 2 to noncooperative targets. Finally, we develop a distributed particle-based implementation of the resulting CoSNAT message passing scheme. This algorithm is the first method for CoSNAT in a fully dynamic setting. The key feature of the proposed CoSNAT algorithm is a bidirectional probabilistic information transfer between the navigation and tracking stages. In this way, uncertainties in one stage can be taken into account by the other stage, which results in an improved performance of both navigation and tracking.

3.1 Review: Distributed Tracking Using Particle Filtering

Besides cooperative navigation using particle-based BP message passing as discussed in the last Section 2, a second methodological component of our CoSNAT algorithm is distributed tracking using data dissemination techniques. This component will therefore be reviewed next.

In distributed tracking, at time n, the CAs $l' \in \mathcal{S}_{m,n}$ acquire measurements $\mathbf{y}_{l',m;n}$ associated with target $m \in \mathcal{T}$. Each CA $l \in \mathcal{S}$ then estimates all target states $\mathbf{x}_{m,n}$, $m \in \mathcal{T}$ from the past and present measurements of all CAs $l' \in \mathcal{S}_{m,n}$ up to time n, $\mathbf{y}_{m,1:n} \triangleq \left[\mathbf{y}_{l',m;n}\right]_{l' \in \mathcal{S}_{m,n},\, n' \in \{1,\ldots,n\}}$. This is done, e.g., by means of the MMSE estimator

$$\hat{\mathbf{x}}_{m,n}^{\mathrm{MMSE}} \triangleq \int \mathbf{x}_{m,n} f(\mathbf{x}_{m,n} | \mathbf{y}_{m,1:n}; \mathbf{x}_{1:n}^{\mathcal{S}}) \mathrm{d}\mathbf{x}_{m,n}\,, \quad m \in \mathcal{T}. \tag{3.1}$$

(Alternatively, MAP estimation similar to (2.2) can be used.) The estimate (3.1) also involves

the set of CA states up to time n, $\mathbf{x}_{1:n}^{\mathcal{S}} = [\mathbf{x}_{l,n'}]_{l\in\mathcal{S},\, n'\in\{1,\ldots,n\}}$, which normally would have to be estimated separately using a cooperative navigation method. However, the distributed tracking method reviewed in the following merely assumes that each CA $l \in \mathcal{S}$ knows its own position in the global reference frame. Note that this assumption of known CA positions will be lifted in the cooperative navigation and tracking setting.

3.1.1 Consensus-based Particle Filter for Distributed Tracking

The statistical relationship between the set of all measurements involving target m, $\mathbf{y}_{m,n} \triangleq [\mathbf{y}_{l,m;n}]_{l\in\mathcal{S}_{m,n}}$, and the target state $\mathbf{x}_{m,n}$ is described by the *global likelihood function* (GLF)

$$G_{m,n}(\mathbf{x}_{m,n}) \triangleq f(\mathbf{y}_{m,n}|\mathbf{x}_{m,n};\mathbf{x}_n^{\mathcal{S}}) = \prod_{l\in\mathcal{S}_{m,n}} f(\mathbf{y}_{l,m;n}|\mathbf{x}_{m,n};\mathbf{x}_{l,n})\,, \tag{3.2}$$

where assumption (A6) was used in the last step. Note that here $\mathbf{x}_{l,n}$ is the known position of CA l. Based on assumptions (A2)–(A5), the posterior PDF involved in (3.1) can be calculated sequentially according to [Ristic et al., 2004]

$$f(\mathbf{x}_{m,n}|\mathbf{y}_{m,1:n};\mathbf{x}_{1:n}^{\mathcal{S}}) \propto G_{m,n}(\mathbf{x}_{m,n}) \int f(\mathbf{x}_{m,n}|\mathbf{x}_{m,n-1}) \times f(\mathbf{x}_{m,n-1}|\mathbf{y}_{m,1:n-1};\mathbf{x}_{1:n-1}^{\mathcal{S}})\, d\mathbf{x}_{m,n-1}\,. \tag{3.3}$$

A feasible approximation of sequential state estimation as given by (3.1) and (3.3) is provided by the particle filter (PF) [Ristic et al., 2004]. The PF uses a PR $\left\{\left(\mathbf{x}_{m,n}^{(j)}, w_{m,n}^{(j)}\right)\right\}_{j=1}^{J}$ of $f(\mathbf{x}_{m,n}|\mathbf{y}_{m,1:n};\mathbf{x}_{1:n}^{\mathcal{S}})$, from which an approximation of the MMSE estimate (3.1) can be obtained (cf. (2.12)).

The weights $w_{m,n}^{(j)}$ are calculated by evaluating the GLF $G_{m,n}(\mathbf{x}_{m,n})$ at the particles $\mathbf{x}_{m,n}^{(j)}$ [Ristic et al., 2004]. Following [Farahmand et al., 2011] and [Savic et al., 2014], this evaluation can be performed in a distributed manner by employing J parallel instances of an average consensus or gossip scheme [Olfati-Saber et al., 2007, Dimakis et al., 2010]. These schemes are iterative and update an "internal state" at each iteration; they require only communication with neighboring CAs. Let $\zeta_{l,m;n}^{(j;i)}$ denote the internal states of CA $l \in \mathcal{S}$ at iteration $i \in \{1,\ldots,C\}$. Each internal state is initialized as

$$\zeta_{l,m;n}^{(j;0)} = \begin{cases} \log f(\mathbf{y}_{l,m;n}|\mathbf{x}_{m,n}^{(j)};\mathbf{x}_{l,n})\,, & l \in \mathcal{S}_{m,n} \\ 0\,, & l \notin \mathcal{S}_{m,n}\,. \end{cases} \tag{3.4}$$

If the network's communication graph is connected, then after convergence of the consensus or gossip algorithm, i.e., as $i \to \infty$, the internal state would equal the average of all the initial values in the agent network, i.e.,

$$\lim_{i\to\infty} \zeta_{l,m;n}^{(j;i)} = \frac{1}{|\mathcal{S}|} \sum_{l\in\mathcal{S}_{m,n}} \log f(\mathbf{y}_{l,m;n}|\mathbf{x}_{m,n}^{(j)};\mathbf{x}_{l,n})\,. \tag{3.5}$$

Therefore, if the number C of iterations is sufficiently large, then due to (3.2), a good approximation of $G_{m,n}(\mathbf{x}_{m,n}^{(j)})$ can be obtained at each CA by $G_{m,n}(\mathbf{x}_{m,n}^{(j)}) \approx \exp\big(|\mathcal{S}|\zeta_{l,m;n}^{(j;C)}\big)$. Because perfect consensus on the weights is required (to guarantee identical particles at all CAs) but cannot generally be obtained with a finite number C of iterations, in addition a max-consensus scheme [Farahmand et al., 2011] is used. This scheme computes the exact maximum of all values using only local communication; it converges in I iterations, where I denotes the diameter of the CA network. Furthermore, the pseudo-random number generators of all CAs have to be initialized with the same seed at time $n = 0$; this ensures that they are in identical states at all times.

Alternatively, the likelihood consensus scheme [Hlinka et al., 2012, Hlinka et al., 2013] can be employed to approximate the functional form of $G_{m,n}^{(p)}(\mathbf{x}_{m,n})$ at each CA using only local communication. The local likelihood functions are approximated by an expansion into a given set of basis functions; then consensus over the expansion coefficients is employed. This typically requires less communications than the "consensus on the weights" scheme. However, the computational effort is larger since each CA has to solve a least squares problem to calculate its coefficients [Hlinka et al., 2012, Hlinka et al., 2013], and a more informative initial prior is required for good performance.

3.1.2 Message Passing Interpretation

For later reference, we note that sequential Bayesian distributed tracking according to (3.2) and (3.3) is equivalent to running BP on the factor graph shown in Fig. 3.1 [Loeliger, 2004]. Because of the tree structure of this graph, BP is performed noniteratively, i.e., the message passing procedure (2.5)–(2.7) is performed only once to calculate $b(\mathbf{x}_{m,n})$. Furthermore, $b(\mathbf{x}_{m,n})$ is exactly equal to $f(\mathbf{x}_{m,n}|\mathbf{y}_{m,1:n}; \mathbf{x}_{1:n}^{\mathcal{S}})$. We have (cf. (2.5))

$$b(\mathbf{x}_{m,n}) \propto \phi_{\to n}(\mathbf{x}_{m,n}) \prod_{l \in \mathcal{S}_{m,n}} \phi_{l \to m}(\mathbf{x}_{m,n}), \tag{3.6}$$

where $\phi_{\to n}(\mathbf{x}_{m,n})$ is calculated similarly to (2.6). The messages $\phi_{l\to m}(\mathbf{x}_{m,n})$, $l \in \mathcal{S}_{m,n}$ need not be calculated using (2.7) because they equal the local likelihood functions $f(\mathbf{y}_{l,m;n}|\mathbf{x}_{m,n}; \mathbf{x}_{l,n})$. (This follows from our assumption that each CA $l \in \mathcal{S}_{m,n}$ knows its own true state, and will be shown in Section 3.2.) The messages and beliefs involved in the calculation of $b(\mathbf{x}_{m,n})$ are depicted in Fig. 3.1. We emphasize that messages entering or leaving a target variable node in the factor graph in Fig. 3.1 do not imply that there occurs any communication involving targets. As mentioned earlier, the targets are noncooperative.

The PF is a particle implementation of (3.3) and a special case of NBP. Particles $\big\{\mathbf{x}_{m,n}^{(j)}\big\}_{j=1}^{J}$ representing $\phi_{\to n}(\mathbf{x}_{m,n})$ (cf. (3.6)) are drawn by performing message filtering as described in Section 2.1.2. Furthermore, using importance sampling with $\phi_{\to n}(\mathbf{x}_{m,n})$ as proposal distribu-

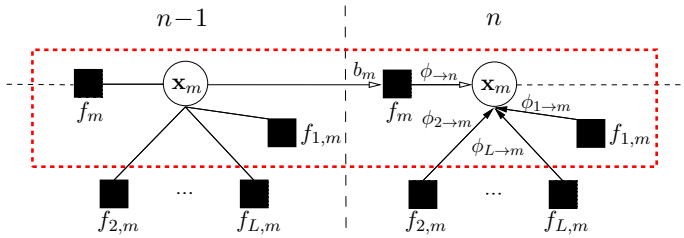


Figure 3.1: Distributed tracking factor graph for target $m \in \mathcal{T}$, involving CAs $l \in \mathcal{S}_{m,n} = \{1, 2, \ldots, L\}$. The time instants $n-1$ and n are shown; time indices are omitted for simplicity. The short notation $f_m \triangleq f(\mathbf{x}_{m,n'}|\mathbf{x}_{m,n'-1})$, $f_{l,m} \triangleq f(\mathbf{y}_{l,m;n'}|\mathbf{x}_{l,n'}, \mathbf{x}_{m,n'})$, $b_m \triangleq b(\mathbf{x}_{m,n'})$, $n' \in \{1, \ldots, n\}$ is used. Factors inside the dotted box correspond to calculations performed by CA $1 \in \mathcal{S}_{m,n}$; factors outside the box imply communication with other CAs. Only messages and beliefs involved in the computation of $b(\mathbf{x}_{m,n})$ are shown. Edges with non-filled arrowheads depict particle-based messages and beliefs, while edges with filled arrowheads depict messages involved in the consensus scheme.

tion, corresponding weights $\{w_{m,n}^{(j)}\}_{j=1}^{J}$ are obtained by evaluating the message product

$$\prod_{l \in \mathcal{S}_{m,n}} \phi_{l\to m}(\mathbf{x}_{m,n}) = \prod_{l \in \mathcal{S}_{m,n}} f(\mathbf{y}_{l,m;n}|\mathbf{x}_{m,n}; \mathbf{x}_{l,n}) \tag{3.7}$$

at these particles. This message multiplication is simpler than that of Section 2.1.2 because no kernel estimates are required.

Since each CA is interested in all target states $\mathbf{x}_{m,n}$, $m \in \mathcal{T}$, the corresponding beliefs $b(\mathbf{x}_{m,n})$ are stored (temporarily) at all CAs $l \in \mathcal{S}$ in parallel.

3.2 CoSNAT Message Passing Scheme

The CoSNAT message passing scheme developed in this section combines the cooperative navigatio and distributed tracking message passing schemes reviewed in Sections 2.1 and 3.1, respectively. A distributed CoSNAT algorithm based on this message passing scheme will be presented in Section 3.3.

In CoSNAT, each CA $l \in \mathcal{S}$ estimates both its state $\mathbf{x}_{l,n}$ and all target states $\mathbf{x}_{m,n}$, $m \in \mathcal{T}$ from the *entire measurement set* $\mathbf{y}_{1:n} = [\mathbf{y}_{l',k;n'}]_{l' \in \mathcal{S},\, k \in \mathcal{M}_{l',n'}, n' \in \{1,\ldots,n\}}$, i.e., from the pairwise measurements between the CAs and those between the CAs and the targets up to time n. The MMSE estimator of the CA and target states is given by (remember that $\mathcal{S} \cup \mathcal{T} = \mathcal{A}$)

$$\hat{\mathbf{x}}_{k,n}^{\text{MMSE}} \triangleq \int \mathbf{x}_{k,n} f(\mathbf{x}_{k,n}|\mathbf{y}_{1:n}) d\mathbf{x}_{k,n}\,, \quad k \in \mathcal{A}\,. \tag{3.8}$$

Alternatively, MAP estimation (2.2) can be employed. In the cooperative simultanous navigation and tracking estimates (3.8), compared to the cooperative navigation estimates (2.1) and the distributed tracking estimates (3.1), the measurement set is extended in that it includes also the respective other measurements—i.e., the pairwise measurements between CAs and targets in the

CA state estimates and the pairwise measurements between CAs in the target state estimates. This is a major reason why cooperative navigation and tracking outperforms separate cooperative navigation–distributed tracking and SLAT. In fact, by using in the estimator (3.8) the PDF $f(\mathbf{x}_{k,n}|\mathbf{y}_{1:n})$, which involves the total set $\mathbf{y}_{1:n}$ of all present and past measurements available throughout the entire network, the inherent coupling between the cooperative navigation and distributed tracking tasks is exploited in an optimum manner.

The marginal posteriors $f(\mathbf{x}_{k,n}|\mathbf{y}_{1:n})$, $k \in \mathcal{A}$ in (3.8) can be obtained by marginalizing $f(\mathbf{x}_{1:n}|\mathbf{y}_{1:n})$. Using Bayes' rule and assumptions (A1)–(A6), one obtains the factorization

$$f(\mathbf{x}_{1:n}|\mathbf{y}_{1:n}) \propto \left[\prod_{k\in\mathcal{A}} f(\mathbf{x}_{k,0})\right] \prod_{n'=1}^{n} \left[\prod_{k_1\in\mathcal{A}} f(\mathbf{x}_{k_1,n'}|\mathbf{x}_{k_1,n'-1})\right] \times \prod_{l\in\mathcal{S}} \prod_{k_2\in\mathcal{M}_{l,n'}} f(\mathbf{y}_{l,k_2;n'}|\mathbf{x}_{l,n'}, \mathbf{x}_{k_2,n'}) \,.$$

The corresponding factor graph, shown in Fig. 3.2, is the cooperative navigation factor graph in Fig. 2.1 extended by the target states. In contrast to the distributed tracking factor graph in Fig. 3.1, the likelihood function related to measurements of a target, $f(\mathbf{y}_{l,m;n}|\mathbf{x}_{l,n}, \mathbf{x}_{m,n})$, is now a factor ($f_{l,m}$ in Fig. 3.2) between a target state and a CA state. These factors enable a "turbo-like" probabilistic information transfer [Wymeersch, 2007] between cooperative and distributed tracking (see Fig. 3.3), which is another reason for the superior performance of CoSNAT.

On the CoSNAT factor graph in Fig. 3.2, we run a modified BP message passing scheme. This is motivated by the advantages of BP that were discussed in the cooperative navigation context in Section 2.1.1. Again, since the factor graph is loopy, SPAWN and other BP schemes are suboptimum. However, as we will show in Section 3.5, BP provides accurate estimates.

In the modified BP scheme, the belief of agent node $l \in \mathcal{S}$ or $m \in \mathcal{T}$ at message passing iteration $p \in \{1, \ldots, P\}$ is given, up to a normalization factor, by (cf. (2.5) and (3.6))

$$b^{(p)}(\mathbf{x}_{l,n}) \propto \phi_{\to n}(\mathbf{x}_{l,n}) \prod_{k\in\mathcal{M}_{l,n}} \phi^{(p)}_{k\to l}(\mathbf{x}_{l,n}) \,, \quad l \in \mathcal{S} \,, \tag{3.9}$$

$$b^{(p)}(\mathbf{x}_{m,n}) \propto \phi_{\to n}(\mathbf{x}_{m,n}) \prod_{l\in\mathcal{S}_{m,n}} \phi^{(p)}_{l\to m}(\mathbf{x}_{m,n}) \,, \quad m \in \mathcal{T}, \tag{3.10}$$

with the prediction message (cf. (2.6))

$$\phi_{\to n}(\mathbf{x}_{k,n}) = \int f(\mathbf{x}_{k,n}|\mathbf{x}_{k,n-1})\, b^{(P)}(\mathbf{x}_{k,n-1})\, d\mathbf{x}_{k,n-1} \,, \quad k \in \mathcal{A} \tag{3.11}$$

and the measurement messages (cf. (2.7))

$$\phi^{(p)}_{k\to l}(\mathbf{x}_{l,n}) = \begin{cases} \int f(\mathbf{y}_{l,k;n}|\mathbf{x}_{l,n}, \mathbf{x}_{k,n})\, b^{(p-1)}(\mathbf{x}_{k,n})\, d\mathbf{x}_{k,n} & k \in \mathcal{M}^{\mathcal{S}}_{l,n}, \; l \in \mathcal{S} \\ \int f(\mathbf{y}_{l,k;n}|\mathbf{x}_{l,n}, \mathbf{x}_{k,n})\, \psi^{(p-1)}_{k\to l}(\mathbf{x}_{k,n})\, d\mathbf{x}_{k,n} & k \in \mathcal{M}^{\mathcal{T}}_{l,n}, \; l \in \mathcal{S} \end{cases} \tag{3.12}$$

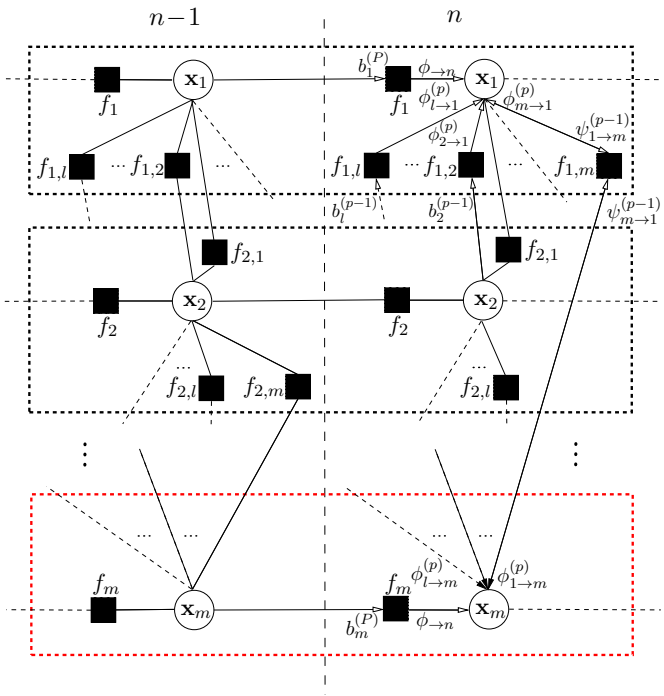


Figure 3.2: CoSNAT factor graph showing the states of CAs $l = 1, 2$ and of a target m at time instants $n-1$ and n; time indices are omitted for simplicity. The short notation $f_k \triangleq f(\mathbf{x}_{k,n'}|\mathbf{x}_{k,n'-1})$, $f_{l,k} \triangleq f(\mathbf{y}_{l,k;n'}|\mathbf{x}_{l,n'}, \mathbf{x}_{k,n'})$, $b_k^{(p)} \triangleq b^{(p)}(\mathbf{x}_{k,n'})$, $\psi_{k\to l}^{(p)} \triangleq \psi_{k\to l}^{(p)}(\mathbf{x}_{k,n'})$, $n' \in \{1, \ldots, n\}$ is used. The upper three (black) dotted boxes correspond to the cooperative navigation part; the bottom (red) dotted box corresponds to the distributed tracking part. Edges between black dotted boxes imply communication between CAs. Only messages and beliefs involved in the computation of $b^{(p)}(\mathbf{x}_{1,n})$ and $b^{(p)}(\mathbf{x}_{m,n})$ are shown. Edges with non-filled arrowheads depict particle-based messages and beliefs, while edges with filled arrowheads depict messages involved in the consensus scheme.

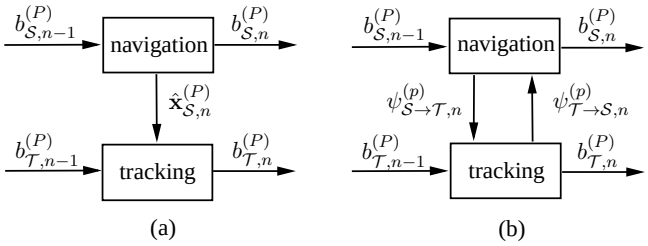


Figure 3.3: Block diagram of (a) separate navigation and tracking and (b) CoSNAT, with $b_{\mathcal{S},n}^{(P)} \triangleq \left\{b^{(P)}(\mathbf{x}_{l,n})\right\}_{l\in\mathcal{S}}$, $b_{\mathcal{T},n}^{(P)} \triangleq \left\{b^{(P)}(\mathbf{x}_{m,n})\right\}_{m\in\mathcal{T}}$, $\psi_{\mathcal{S}\to\mathcal{T},n}^{(p)} \triangleq \left\{\psi_{l\to m}^{(p)}(\mathbf{x}_{l,n})\right\}_{l\in\mathcal{S}, m\in\mathcal{M}_{l,n}^{\mathcal{T}}}$, $\psi_{\mathcal{T}\to\mathcal{S},n}^{(p)} \triangleq \left\{\psi_{m\to l}^{(p)}(\mathbf{x}_{m,n})\right\}_{m\in\mathcal{T}, l\in\mathcal{S}_{m,n}}$, and $\hat{\mathbf{x}}_{\mathcal{S},n}^{(P)} \triangleq \left[\hat{\mathbf{x}}_{l,n}^{(P)}\right]_{l\in\mathcal{S}}$. In separate navigation and tracking, only the final CA state estimates $\hat{\mathbf{x}}_{\mathcal{S},n}^{(P)}$ are transferred once from navigation to tracking. In CoSNAT, probabilistic information (the extrinsic information $\psi_{\mathcal{S}\to\mathcal{T},n}^{(p)}$ and $\psi_{\mathcal{T}\to\mathcal{S},n}^{(p)}$) is exchanged between navigation and tracking at each message passing iteration p.

(3.13)

and

$$\phi^{(p)}_{l\to m}(\mathbf{x}_{m,n}) = \int f(\mathbf{y}_{l,m;n}|\mathbf{x}_{l,n}, \mathbf{x}_{m,n})\, \psi^{(p-1)}_{l\to m}(\mathbf{x}_{l,n})\, d\mathbf{x}_{l,n} \qquad l{=}\mathcal{S}_{m,n},\ m{\in}\mathcal{T}. \tag{3.14}$$

Here, $\psi^{(p-1)}_{m\to l}(\mathbf{x}_{m,n})$ and $\psi^{(p-1)}_{l\to m}(\mathbf{x}_{l,n})$ (constituting the extrinsic information) are given by

$$\psi^{(p-1)}_{l\to m}(\mathbf{x}_{l,n}) \;=\; \phi_{\to n}(\mathbf{x}_{l,n}) \prod_{k\in\mathcal{M}_{l,n}\backslash\{m\}} \phi^{(p-1)}_{k\to l}(\mathbf{x}_{l,n}) \tag{3.15}$$

$$\psi^{(p-1)}_{m\to l}(\mathbf{x}_{m,n}) \;=\; \phi_{\to n}(\mathbf{x}_{m,n}) \prod_{l'\in\mathcal{S}_{m,n}\backslash\{l\}} \phi^{(p-1)}_{l'\to m}(\mathbf{x}_{m,n})\,. \tag{3.16}$$

For reasons discussed in Section 2.1.1, the prediction messages in (3.11) (cf. (2.6)) and the CA-related measurement messages in (3.12) (cf. (2.7)) differ from the standard BP message passing rules, i.e., the extrinsic information is equal to the belief. The messages and beliefs involved in the calculation of $b^{(p)}(\mathbf{x}_{l,n})$ and $b^{(p)}(\mathbf{x}_{m,n})$ are depicted in Fig. 3.2.

Again, messages entering or leaving a target variable node in Fig. 3.2 do not imply that there occurs any communication involving targets.

In the "pure distributed tracking" case considered in Section 3.1, CA $l \in \mathcal{S}_{m,n}$ knows its own true state, $\mathbf{x}^{\text{true}}_{l,n}$, and thus $\psi^{(p-1)}_{l\to m}(\mathbf{x}_{l,n})$ is replaced by $\delta(\mathbf{x}_{l,n} - \mathbf{x}^{\text{true}}_{l,n})$. Hence, (3.14) yields

$$\phi^{(p)}_{l\to m}(\mathbf{x}_{m,n}) = f(\mathbf{y}_{l,m;n}|\mathbf{x}_{m,n}; \mathbf{x}^{\text{true}}_{l,n}),$$

as was claimed in Section 3.1.2.

As in cooperative navigation, each CA belief $b^{(p)}(\mathbf{x}_{l,n})$ is stored (temporarily) only at the respective CA l. However, as each CA is also interested in all the target states $\mathbf{x}_{m,n}$, the corresponding beliefs $b^{(p)}(\mathbf{x}_{m,n})$ are also computed and stored (temporarily) at all CAs $l \in \mathcal{S}$ in parallel.

3.3 Distributed CoSNAT Algorithm

We will next devise a distributed CoSNAT algorithm that combines particle-based BP—i.e., a particle-based implementation of (3.9)–(3.16)—with consensus. This algorithm requires only local communication between neighboring CAs. The distributed calculation of the target beliefs, CA beliefs, and extrinsic information will be discussed in separate subsections.

3.3.1 Distributed Calculation of the Target Beliefs

Estimation of the target states $\mathbf{x}_{m,n}$, $m \in \mathcal{T}$ from $\mathbf{y}_{1:n}$ according to (3.8) essentially amounts to a computation of $f(\mathbf{x}_{m,n}|\mathbf{y}_{1:n})$. The following discussion describes the calculations associated with the red dotted box in Fig. 3.2.

The target belief $b^{(p)}(\mathbf{x}_{m,n})$, $p \in \{1, \ldots, P\}$ approximating $f(\mathbf{x}_{m,n}|\mathbf{y}_{1:n})$ is given, up to a

factor, by (see (3.10))

$$b^{(p)}(\mathbf{x}_{m,n}) \propto \phi_{\to n}(\mathbf{x}_{m,n})\, \Phi^{(p)}_{m,n}(\mathbf{x}_{m,n})\,, \tag{3.17}$$

with (recalling (3.11))

$$\phi_{\to n}(\mathbf{x}_{m,n}) = \int f(\mathbf{x}_{m,n}|\mathbf{x}_{m,n-1})\, b^{(P)}(\mathbf{x}_{m,n-1})\, d\mathbf{x}_{m,n-1} \tag{3.18}$$

and

$$\Phi^{(p)}_{m,n}(\mathbf{x}_{m,n}) \triangleq \prod_{l\in\mathcal{S}_{m,n}} \phi^{(p)}_{l\to m}(\mathbf{x}_{m,n})\,. \tag{3.19}$$

The key observation now is that expression (3.17) along with (3.18) is of the same form as the distributed tracking recursion (3.3), but with the GLF $G_{m,n}(\mathbf{x}_{m,n})$ replaced by the message product $\Phi^{(p)}_{m,n}(\mathbf{x}_{m,n})$. The belief $b^{(P)}(\mathbf{x}_{m,n-1})$ occurring in (3.18) was calculated by each CA at time $n-1$; using this belief, the CA is able to calculate the message $\phi_{\to n}(\mathbf{x}_{m,n})$ involved in (3.17). Regarding the message product $\Phi^{(p)}_{m,n}(\mathbf{x}_{m,n})$, the individual messages $\phi^{(p)}_{l\to m}(\mathbf{x}_{m,n})$ (cf. (3.19)) involve the extrinsic informations $\psi^{(p-1)}_{l\to m}(\mathbf{x}_{l,n})$ (see (3.14)); calculation of the latter will be discussed in Section 3.3.3. However, because the targets do not cooperate, at each CA at most one message $\phi^{(p)}_{l\to m}(\mathbf{x}_{m,n})$ is available (for a given m). Thus, the overall message product $\Phi^{(p)}_{m,n}(\mathbf{x}_{m,n})$ is not available at the CAs.

Particle-based Calculation of $b^{(p)}(\mathbf{x}_{m,n})$

A particle-based implementation of (3.17) can be obtained via importance sampling with proposal distribution $\phi_{\to n}(\mathbf{x}_{m,n})$. First, based on (3.18), particles $\{\mathbf{x}^{(j)}_{m,n}\}_{j=1}^{J}$ representing $\phi_{\to n}(\mathbf{x}_{m,n})$ are calculated from particles representing $b^{(P)}(\mathbf{x}_{m,n-1})$ by means of message filtering (cf. Section 2.1.2). Next, weights $\{w^{(j)}_{m,n}\}_{j=1}^{J}$ are calculated as

$$\tilde{w}^{(j)}_{m,n} \triangleq \Phi^{(p)}_{m,n}(\mathbf{x}^{(j)}_{m,n}) \tag{3.20}$$

followed by normalization. Finally, resampling is performed to obtain equally weighted particles representing $b^{(p)}(\mathbf{x}_{m,n})$. However, this particle-based implementation presupposes that the message product $\Phi^{(p)}_{m,n}(\mathbf{x}_{m,n})$ is available at the CAs. Therefore, in the following, we present a distributed scheme for the evaluation of $\Phi^{(p)}_{m,n}(\cdot)$ at the particles $\{\mathbf{x}^{(j)}_{m,n}\}_{j=1}^{J}$. This scheme requires only local communication.

Distributed Evaluation of $\Phi^{(p)}_{m,n}(\cdot)$

For a distributed computation of $\Phi^{(p)}_{m,n}(\mathbf{x}^{(j)}_{m,n})$, $j \in \{1,\ldots,J\}$, we employ J parallel instances of a consensus or gossip scheme [Olfati-Saber et al., 2007, Dimakis et al., 2010]. In analogy to (3.4), the internal states are initialized as

$$\zeta^{(j;0)}_{l,m;n} = \begin{cases} \log\, \phi^{(p)}_{l\to m}(\mathbf{x}^{(j)}_{m,n}), & l \in \mathcal{S}_{m,n} \\ 0 & l \notin \mathcal{S}_{m,n}\,. \end{cases} \tag{3.21}$$

Note that a closed-form approximation of $\phi_{l\to m}^{(p)}(\mathbf{x}_{m,n})$ can be obtained via a kernel estimate (see (2.10)) or via a parametric representation (see Sections 2.1.2 and 2.2.2) as well as [Caceres et al., 2011]). If the network's communication graph is connected, then after convergence of the consensus or gossip algorithm, the internal state would become

$$\lim_{i\to\infty} \zeta_{l,m;n}^{(j;i)} = \frac{1}{|\mathcal{S}|} \sum_{l\in\mathcal{S}_{m,n}} \log\, \phi_{l\to m}^{(p)}(\mathbf{x}_{m,n}^{(j)})\,.$$

Thus, for a sufficiently large number C of iterations, because of (3.19), a good approximation of $\Phi_{m,n}^{(p)}(\mathbf{x}_{m,n}^{(j)})$ is obtained at each CA by $\Phi_{m,n}^{(p)}(\mathbf{x}_{m,n}^{(j)}) \approx \exp\big(|\mathcal{S}|\,\zeta_{l,m;n}^{(j;C)}\big)$. Here, the number of CAs $|\mathcal{S}|$ can be determined in a distributed way by using another consensus or gossip algorithm at time $n = 0$ [Pham et al., 2009]. Furthermore, as explained in Section 3.1.1, an additional max-consensus scheme [Farahmand et al., 2011] has to be used to obtain perfect consensus on the weights $\tilde{w}_{m,n}^{(j)}$ in (3.20) and, in turn, identical particles at all CAs. The max-consensus converges in I iterations. Finally, the pseudo-random number generators of all CAs have to be initialized with the same seed at time $n = 0$. This distributed evaluation of $\Phi_{m,n}^{(p)}(\mathbf{x}_{m,n}^{(j)})$ requires only local communication: at each iteration, for each consensus, J real values are broadcast by each CA to neighboring CAs [Olfati-Saber et al., 2007, Dimakis et al., 2010].

As an alternative to this "consensus on the weights" scheme, the likelihood consensus scheme [Hlinka et al., 2012, Hlinka et al., 2013] can be employed to approximate the functional form of $\Phi_{m,n}^{(p)}(\mathbf{x}_{m,n})$ at the CAs, again using only local communication.

Probabilistic Information Transfer

According to (3.14), the messages $\phi_{l\to m}^{(p)}(\mathbf{x}_{m,n})$ occurring in $\Phi_{m,n}^{(p)}(\mathbf{x}_{m,n}) = \prod_{l\in\mathcal{S}_{m,n}} \phi_{l\to m}^{(p)}(\mathbf{x}_{m,n})$ involve the extrinsic informations $\psi_{l\to m}^{(p-1)}(\mathbf{x}_{l,n})$ of all CAs l observing target m, i.e., $l \in \mathcal{S}_{m,n}$. Therefore, they constitute an information transfer from the cooperative navigation part of CoSNAT to the distributed tracking part (cf. the directed edges entering the red dotted box in Fig. 3.2). The estimation of target state $\mathbf{x}_{m,n}$ is based on the belief $b^{(p)}(\mathbf{x}_{m,n})$ as given by (3.17), and thus on $\Phi_{m,n}^{(p)}(\mathbf{x}_{m,n})$. This improves on pure distributed tracking because probabilistic information about the states of the CAs $l \in \mathcal{S}_{m,n}$—provided by $\psi_{l\to m}^{(p-1)}(\mathbf{x}_{l,n})$—is taken into account. By contrast, pure distributed tracking uses the GLF $G_{m,n}(\mathbf{x}_{m,n})$ instead of $\Phi_{m,n}^{(p)}(\mathbf{x}_{m,n})$ (see (3.3)). According to (3.2), this presupposes that the CA states are known. In separate navigation and tracking, estimates of the CA states provided by cooperative navigation are used, rather than probabilistic information about the CA states as is done in CoSNAT. This probabilistic information transfer is visualized in Fig. 3.3 The improved accuracy of target state estimation achieved by our CoSNAT algorithm compared to separate navigation and tracking will be demonstrated in Section 3.5.1.

3.3.2 Distributed Calculation of the CA Beliefs

For distributed calculation of the CA belief $b^{(p)}(\mathbf{x}_{l,n})$, $l \in \mathcal{S}$, the following information is available at CA l: (i) equally weighted particles representing $\psi_{m\to l}^{(p-1)}(\mathbf{x}_{m,n})$ for all targets $m \in \mathcal{T}$ (which were calculated as described in Section 3.3.1 and 3.3.3); (ii) equally weighted particles representing $b^{(p-1)}(\mathbf{x}_{l',n})$ for all neighboring CAs $l' \in \mathcal{M}_{l,n}^{\mathcal{S}}$ (which were received from these CAs); and (iii) a PR of $b^{(P)}(\mathbf{x}_{l,n-1})$ (which was calculated at time $n-1$). Using this information and the measurements $\mathbf{y}_{l,k;n}$, $k \in \mathcal{M}_{l,n}$, a PR $\big\{\big(\mathbf{x}_{l,n}^{(j)}, w_{l,n}^{(j)}\big)\big\}_{j=1}^{J}$ of $b^{(p)}(\mathbf{x}_{l,n})$ can be calculated in a distributed manner by implementing (3.9), using NBP for mobile CAs as reviewed in Section 2.1.2 or the new low-complexity method presented in Section 2.4. Finally, resampling is performed to obtain equally weighted particles representing $b^{(p)}(\mathbf{x}_{l,n})$. This calculation of the CA beliefs improves on pure cooperative navigation as reviewed in Section 2.1 in that it uses the probabilistic information about the states of the targets $m \in \mathcal{M}_{l,n}^{\mathcal{T}}$ provided by the messages $\psi_{m\to l}^{(p)}(\mathbf{x}_{m,n})$ (cf. (3.16)). The improved accuracy of CA state estimation will be demonstrated in Section 3.5.

3.3.3 Distributed Calculation of the Extrinsic Informations

Since (3.15) is analogous to (3.9) and (3.16) to (3.10), particles for $\psi_{l\to m}^{(p)}(\mathbf{x}_{l,n})$ or $\psi_{m\to l}^{(p)}(\mathbf{x}_{m,n})$ can be calculated similarly as for the corresponding belief. However, the following shortcut reuses previous results. To obtain particles for $\psi_{m\to l}^{(p)}(\mathbf{x}_{m,n})$, we proceed as for $b^{(p)}(\mathbf{x}_{m,n})$ (see Section 3.3.1) but replace

$$\Phi_{m,n}^{(p)}(\mathbf{x}_{m,n}^{(j)}) \approx \exp\big(|\mathcal{S}|\,\zeta_{l,m;n}^{(j;C)}\big) \tag{3.22}$$

with

$$\exp\big(|\mathcal{S}|\,\zeta_{l,m;n}^{(j;C)} - \zeta_{l,m;n}^{(j;0)}\big) \qquad \text{for all } j = 1,\ldots,J.$$

(This is based on the relation $\psi_{m\to l}^{(p)}(\mathbf{x}_{m,n}) \propto b^{(p)}(\mathbf{x}_{m,n})/\phi_{l\to m}^{(p)}(\mathbf{x}_{m,n})$, cf. (3.10) and (3.16).) Here, $\zeta_{l,m;n}^{(j;C)}$ and $\zeta_{l,m;n}^{(j;0)}$ are already available locally from the calculation of $b^{(p)}(\mathbf{x}_{m,n})$.

3.3.4 Statement of the CoSNAT Algorithm

The proposed distributed CoSNAT algorithm is obtained by combining the operations discussed in Sections 3.3.1 through 3.3.3, as summarized in the following.

ALGORITHM 1: DISTRIBUTED COSNAT ALGORITHM

Initialization: The recursive algorithm described below is initialized at time $n=0$ with particles $\big\{\bar{\mathbf{x}}_{k,0}^{(j)}\big\}_{j=1}^{J}$ drawn from a prior PDF $f(\mathbf{x}_{k,0})$, for $k \in \{l\} \cup \mathcal{T}$.

Recursion at time n: At CA l, equally weighted particles $\big\{\bar{\mathbf{x}}_{k,n-1}^{(j)}\big\}_{j=1}^{J}$ representing the beliefs $b^{(P)}(\mathbf{x}_{k,n-1})$ with $k \in \{l\} \cup \mathcal{T}$ are available (these were calculated at time $n-1$). At time n, CA l performs the following operations.

Step 1—Prediction: From $\big\{\bar{\mathbf{x}}_{k,n-1}^{(j)}\big\}_{j=1}^{J}$, PRs $\big\{\mathbf{x}_{k,n}^{(j)}\big\}_{j=1}^{J}$ of the prediction messages $\phi_{\to n}(\mathbf{x}_{k,n})$, $k \in \{l\} \cup \mathcal{T}$ are calculated via message filtering (see Section 2.1.2), based on the state-transition model in (1.1). That is, $\mathbf{x}_{k,n}^{(j)} = g(\bar{\mathbf{x}}_{k,n-1}^{(j)}, \mathbf{u}_{k,n}^{(j)})$, where the particles $\big\{\mathbf{u}_{k,n}^{(j)}\big\}_{j=1}^{J}$ are drawn from $f(\mathbf{u}_{k,n})$.

Step 2—BP message passing: For each $k \in \{l\} \cup \mathcal{T}$, the belief is initialized as $b^{(0)}(\mathbf{x}_{k,n}) = \phi_{\to n}(\mathbf{x}_{k,n})$, in the sense that the PR of $\phi_{\to n}(\mathbf{x}_{k,n})$ is used as PR of $b^{(0)}(\mathbf{x}_{k,n})$. Then, for $p = 1, \ldots, P$:

a) A PR $\left\{\left(\mathbf{x}_{m,n}^{(j)}, w_{m,n}^{(j)}\right)\right\}_{j=1}^{J}$ of $b^{(p)}(\mathbf{x}_{m,n})$ in (3.10) is obtained via importance sampling with proposal distribution $\phi_{\to n}(\mathbf{x}_{m,n})$ (see Section 3.3.1). That is, using the particles $\left\{\mathbf{x}_{m,n}^{(j)}\right\}_{j=1}^{J}$ representing $\phi_{\to n}(\mathbf{x}_{m,n})$ (calculated in Step 1), weights $\left\{w_{m,n}^{(j)}\right\}_{j=1}^{J}$ are obtained by evaluating $\tilde{w}_{m,n}^{(j)} = \Phi(\mathbf{x}_{m,n}^{(j)})$ for all $j = 1, \ldots, J$ in a distributed manner as described in Section 3.3.1 and normalizing.

b) For each target $m \in \mathcal{M}_{l,n}^{\mathcal{T}}$, a PR of $\psi_{m\to l}^{(p)}(\mathbf{x}_{m,n})$ is calculated in a similar manner (see Section 3.3.3).

c) A PR $\left\{\left(\mathbf{x}_{l,n}^{(j)}, w_{l,n}^{(j)}\right)\right\}_{j=1}^{J}$ of $b^{(p)}(\mathbf{x}_{l,n})$ is calculated by implementing (3.9) as described in Section 3.3.2; this involves equally weighted particles of all $b^{(p-1)}(\mathbf{x}_{k,n})$, $k \in \mathcal{M}_{l,n}$.

d) For each $m \in \mathcal{M}_{l,n}^{\mathcal{T}}$, a PR of $\psi_{l\to m}^{(p)}(\mathbf{x}_{l,n})$ is calculated in a similar manner.

e) For all PRs calculated in Steps 2a–2d, resampling is performed to obtain equally weighted particles.

f) The equally weighted particles of $b^{(p)}(\mathbf{x}_{l,n})$ calculated in Step 2e are broadcast to all CAs l' for which $l \in \mathcal{M}_{l',n}^{\mathcal{S}}$, and equally weighted particles of $b^{(p)}(\mathbf{x}_{l_1,n})$ are received from each neighboring CA $l_1 \in \mathcal{M}_{l,n}^{\mathcal{S}}$. Thus, at this point, equally weighted particles $\left\{\bar{\mathbf{x}}_{k,n}^{(j)}\right\}_{j=1}^{J}$ of the following quantities are available at CA l: $b^{(p)}(\mathbf{x}_{k,n})$ for $k \in \{l\} \cup \mathcal{T} \cup \mathcal{M}_{l,n}^{\mathcal{S}}$; $\psi_{m\to l}^{(p)}(\mathbf{x}_{m,n})$ for $m \in \mathcal{M}_{l,n}^{\mathcal{T}}$; and $\psi_{l\to m}^{(p)}(\mathbf{x}_{l,n})$ for $m \in \mathcal{M}_{l,n}^{\mathcal{T}}$.

Step 3—Estimation: For $k \in \{l\} \cup \mathcal{T}$, an approximation of the global MMSE state estimate $\hat{\mathbf{x}}_{k,n}^{\mathrm{MMSE}}$ in (3.8) is computed from the PR $\left\{\left(\mathbf{x}_{k,n}^{(j)}, w_{k,n}^{(j)}\right)\right\}_{j=1}^{J}$ of $b^{(P)}(\mathbf{x}_{k,n})$ according to

$$\hat{\mathbf{x}}_{k,n} = \sum_{j=1}^{J} w_{k,n}^{(j)} \mathbf{x}_{k,n}^{(j)}, \quad k \in \{l\} \cup \mathcal{T}.$$

Alternatively, a PR-based approximation of the MAP estimator [Driessen and Boers, 2008] can be used.

The communication requirements of this algorithm will be analyzed in Section 3.4.3.

3.4 Variations and Implementation Aspects

Next, we discuss some variations and implementation aspects of the CoSNAT algorithm.

3.4.1 Local Distributed Tracking

The convergence of the consensus or gossip algorithms used to calculate (3.5) is slow if $|\mathcal{S}_{m,n}| \ll |\mathcal{S}|$, because then many initialization values $\beta_{l,m;n}^{(p,r)}(\mathbf{y}_{l,m;n})$ are zero. We therefore introduce a modification, termed *local distributed tracking* (LDT), in which $b^{(p)}(\mathbf{x}_{m,n})$ for a target $m \in \mathcal{T}$ is calculated via (3.10) only at CAs l that acquire a measurement of the target, i.e., $l \in \mathcal{S}_{m,n}$. The convergence is here faster due to the smaller "consensus network" ($l \in \mathcal{S}_{m,n}$ instead of $l \in \mathcal{S}$) and the fact that zero initialization values are avoided. LDT presupposes that the communication graph of the network formed by all CAs $l \in \mathcal{S}_{m,n}$ is connected. To ensure that CAs

$l' \in \mathcal{S}_{m,n+1} \backslash \mathcal{S}_{m,n}$ (i.e., with $l' \notin \mathcal{S}_{m,n}$ but $l' \in \mathcal{S}_{m,n+1}$) obtain the information needed to track target m at time $n+1$, each CA $l \in \mathcal{S}_{m,n}$ broadcasts $b^{(P)}(\mathbf{x}_{m,n})$ (calculated as described in Section 3.3.1) to its neighbors $l' \in \mathcal{C}_{l,n}$. Using $b^{(P)}(\mathbf{x}_{m,n})$, neighboring CAs $l' \in \mathcal{S}_{m,n+1} \setminus \mathcal{S}_{m,n}$ are then able to calculate $\phi_{\to n+1}(\mathbf{x}_{m,n+1})$ (see (2.6) and Section 2.1.2) and to track target m at time $n+1$ according to (3.10).

LDT has certain drawbacks. First, only CAs $l \in \mathcal{S}_{m,n}$ obtain an estimate of the state of target m. (Equivalently, each CA $l \in \mathcal{S}$ tracks only targets $m \in \mathcal{M}_{l,n}^{\mathcal{T}}$.) Second, the size of the consensus network, $|\mathcal{S}_{m,n}|$, has to be estimated at each time n. Third, in agent networks with few communication links, it is possible that a CA $l' \in \mathcal{S}_{m,n+1} \setminus \mathcal{S}_{m,n}$ cannot communicate with any CA $l \in \mathcal{S}_{m,n}$ at time n, i.e., $l \notin \mathcal{C}_{l',n}$. Then, CA l' does not obtain $b^{(P)}(\mathbf{x}_{m,n})$ and cannot track target m at time $n+1$, even though it acquired a corresponding measurement at time n. However, in many scenarios, the communication regions of the CAs are significantly larger than their measurement regions. The situation described above is then very unlikely.

3.4.2 Alternative Processing at $n = 1$

In the CoSNAT algorithm, a PR of $b^{(p)}(\mathbf{x}_{k,n})$ is calculated via importance sampling with $\phi_{\to n}(\mathbf{x}_{k,n})$ used as proposal distribution $q(\mathbf{x}_{k,n})$. At time $n = 1$, $\phi_{\to 1}(\mathbf{x}_{k,1}) = \int f(\mathbf{x}_{k,1}|\mathbf{x}_{k,0})\, f(\mathbf{x}_{k,0})\, d\mathbf{x}_{k,0}$. Often, insufficient prior information about the substate $\tilde{\mathbf{x}}_{k,0}$ actually involved in the measurement (1.2) is available, and thus a (partly) uninformative prior PDF $f(\mathbf{x}_{k,0})$ is used. (An example is given by agents with an uninformative position prior and measurements $\mathbf{y}_{l,k;n}$ that depend only on the positions of agent $k \in \mathcal{A}$ and CA $l \in \mathcal{S}$, such as, e.g., in (1.3).) This uninformative prior PDF implies that the proposal distribution $\phi_{\to 1}(\mathbf{x}_{k,1})$ is (partly) uninformative as well, and thus, if a moderate number of particles J is used, the generated particles will be widely spread and the estimation performance will be poor.

We therefore propose an alternative processing at time $n = 1$ for agents with an uniformative prior at time $n = 0$. This processing leads to accurate estimates at time $n = 1$ even if a moderate number of particles J is used; it can be employed for distributed tracking, dynamic cooperative navigation, and CoSNAT. Note that anchors do not use the alternative processing because their prior is perfectly informative.

In the following, we will describe the alternative processing for the calculation of a target belief $b(\mathbf{x}_{m,1})$, $m \in \mathcal{T}$; this scheme can be used for the calculation of a CA belief $b(\mathbf{x}_{l,1})$, $l \in \mathcal{S}$ with obvious modifications. For the prediction message at time $n = 1$, we adopt the model $\phi_{\to 1}(\mathbf{x}_{m,1}) = f(\tilde{\mathbf{x}}_{m,1})\, f(\breve{\mathbf{x}}_{m,1})$,

where $f(\tilde{\mathbf{x}}_{m,1})$ is an uninformative PDF of the position $\tilde{\mathbf{x}}_{m,1}$ (e.g., uniform on the entire localization region) and $f(\breve{\mathbf{x}}_{m,1})$ is an informative PDF of the complementary subvector $\breve{\mathbf{x}}_{m,1}$ of $\mathbf{x}_{m,1}$ (e.g., Gaussian with small variance). Now, instead of using $\phi_{\to 1}(\mathbf{x}_{m,1}) = f(\tilde{\mathbf{x}}_{m,1})\, f(\breve{\mathbf{x}}_{m,1})$ as proposal distribution, we use

$$q^{(p)}(\mathbf{x}_{m,1}) \;=\; \phi_{\hat{l}\to m}^{(p)}(\mathbf{x}_{m,1})\, f(\breve{\mathbf{x}}_{m,1})\,, \tag{3.23}$$

with some $\hat{l} \in \mathcal{S}_{m,1}$ (the choice of $\hat{l}$ will be discussed later). Here, $\phi^{(p)}_{\hat{l}\to m}(\mathbf{x}_{m,1})$ is a function of $\tilde{\mathbf{x}}_{m,1}$ but not of $\breve{\mathbf{x}}_{m,1}$; thus, with an abuse of notation, we will write $\phi^{(p)}_{l\to m}(\tilde{\mathbf{x}}_{m,1})$ instead of $\phi^{(p)}_{l\to m}(\mathbf{x}_{m,1})$. Replacing $f(\tilde{\mathbf{x}}_{m,1})$ by $\phi^{(p)}_{\hat{l}\to m}(\tilde{\mathbf{x}}_{m,1})$ is reasonable if $\phi^{(p)}_{\hat{l}\to m}(\tilde{\mathbf{x}}_{m,1})$ is more informative than $f(\tilde{\mathbf{x}}_{m,1})$. Particles $\{\mathbf{x}^{(j)}_{m,1}\}^J_{j=1}$ from $q^{(p)}(\mathbf{x}_{m,1})$ can be obtained by stacking particles $\{\tilde{\mathbf{x}}^{(j)}_{m,1}\}^J_{j=1}$ drawn from $\phi^{(p)}_{\hat{l}\to m}(\tilde{\mathbf{x}}_{m,1})$ and particles $\{\breve{\mathbf{x}}^{(j)}_{m,1}\}^J_{j=1}$ drawn from $f(\breve{\mathbf{x}}_{m,1})$. Typically, drawing particles from $f(\breve{\mathbf{x}}_{m,1})$ is trivial. Equally weighted particles representing $\phi^{(p)}_{\hat{l}\to m}(\tilde{\mathbf{x}}_{m,1})$ can be calculated from equally weighted particles representing $b^{(p)}(\mathbf{x}_{\hat{l},1})$ by using message filtering as described in Section 2.1.2.

A PR $\{(\mathbf{x}^{(j)}_{m,1}, w^{(j)}_{m,1})\}^J_{j=1}$ of $b^{(p)}(\mathbf{x}_{m,1})$, $m \in \mathcal{T}$ is obtained by means of importance sampling using the proposal distribution $q^{(p)}(\mathbf{x}_{m,1})$ in (3.23): particles $\{\mathbf{x}^{(j)}_{m,1}\}^J_{j=1}$ are drawn from $q^{(p)}(\mathbf{x}_{m,1})$, and weights $\{w^{(j)}_{m,1}\}^J_{j=1}$ are calculated according to (cf. (3.9) and (3.10))

$$\begin{aligned}
\tilde{w}^{(j)}_{m,1} &= \frac{\phi_{\to 1}(\mathbf{x}^{(j)}_{m,1}) \prod_{l\in\mathcal{S}_{m,1}} \phi^{(p)}_{l\to m}(\tilde{\mathbf{x}}^{(j)}_{m,1})}{q^{(p)}(\mathbf{x}^{(j)}_{m,1})} \\
&= \frac{f(\tilde{\mathbf{x}}^{(j)}_{m,1})\, f(\breve{\mathbf{x}}^{(j)}_{m,1}) \prod_{l\in\mathcal{S}_{m,1}} \phi^{(p)}_{l\to m}(\tilde{\mathbf{x}}^{(j)}_{m,1})}{\phi^{(p)}_{\hat{l}\to m}(\tilde{\mathbf{x}}^{(j)}_{m,1})\, f(\breve{\mathbf{x}}^{(j)}_{m,1})} \\
&= f(\tilde{\mathbf{x}}^{(j)}_{m,1}) \prod_{l\in\mathcal{S}_{m,1}\setminus\{\hat{l}\}} \phi^{(p)}_{l\to m}(\tilde{\mathbf{x}}^{(j)}_{m,1}),
\end{aligned}$$

followed by normalization.

To make $q^{(p)}(\mathbf{x}_{m,1})$ in (3.23) maximally informative, we choose

$$\hat{l} = \operatorname*{argmin}_{l\in\mathcal{S}_{m,1}} \tilde{\sigma}^2_l \,. \tag{3.24}$$

Here, $\tilde{\sigma}^2_l$ is the empirical variance of $b^{(p)}_l(\tilde{\mathbf{x}}_{l,1})$. For non-anchor CAs, $\tilde{\sigma}^2_l$ is calculated from equally weighted particles $\{\tilde{\mathbf{x}}^{(j)}_{l,1}\}^J_{j=1}$ representing $b^{(p)}_l(\tilde{\mathbf{x}}_{l,1})$ as

$$\tilde{\sigma}^2_l = \frac{1}{J}\sum_{j=1}^{J} \left\|\tilde{\mathbf{x}}^{(j)}_{l,1} - \tilde{\boldsymbol{\mu}}_l\right\|^2, \tag{3.25}$$

with

$$\tilde{\boldsymbol{\mu}}_l = \frac{1}{J}\sum_{j=1}^{J} \tilde{\mathbf{x}}^{(j)}_{l,1}. \tag{3.26}$$

For anchors, we set $\tilde{\sigma}^2_l = 0$.

A distributed implementation is obtained, by computing (3.24) using the min-consensus scheme [Olfati-Saber and Murray, 2004], which converges in I iterations [Olfati-Saber and Murray, 2004] (in the case of LDT, the number of iterations is the diameter of the CA network given by $\mathcal{S}_{m,1}$). This min-consensus scheme is also used to disseminate the optimum $\phi^{(p)}_{\hat{l}\to m}(\mathbf{x}_{m,1})$ within the network.

3.4.3 Communication Requirements and Delay

In the following discussion of the communication requirements of the proposed CoSNAT algorithm, we assume for simplicity that all states $\mathbf{x}_{k,n}$, $k \in \mathcal{A}$ have identical dimension L and all $\tilde{\mathbf{x}}_{k,n}$, $k \in \mathcal{A}$ (i.e., the substates actually involved in the measurements, cf. (1.2) and (1.3)) have identical dimension $\tilde{L}$. Furthermore, we denote by C the number of consensus or gossip iterations used for averaging, by P the number of message passing iterations, by J the number of particles, and by I the network diameter. Finally, $I_n^{\mathcal{T}}$ denotes the maximum of the diameters of the CA networks given by the $\mathcal{S}_{m,n}$, $m \in \mathcal{T}$. For an analysis of the delay introduced by the communication, we assume that broadcasting the belief of the CAs counts as one delay time slot, and broadcasting all values corresponding to one consensus iteration also counts as one delay time slot.

- For consensus-based calculation of the target beliefs $b^{(p)}(\mathbf{x}_{m,n})$, $m \in \mathcal{T}$ (see Section 3.3.1 and Step 2a in Algorithm 1), at each time n, CA $l \in \mathcal{S}$ broadcasts $N^{\mathrm{C}} \triangleq P(C+I)J|\mathcal{T}|$ real values to CAs $l' \in \mathcal{C}_{l,n}$. In the case of LDT (see Section 3.4.1), N^{C} is reduced to $N_{l,n}^{\mathrm{CR}} \triangleq P(C+I_n^{\mathcal{T}})J|\mathcal{M}_{l,n}^{\mathcal{T}}|$. Note that in the LDT case, C is smaller, and even much smaller for a large network. Consensus-based calculation of the target beliefs $b^{(p)}(\mathbf{x}_{m,n})$, $m \in \mathcal{T}$ contributes $P(C+I)$ time slots to the overall delay (or $P(C+I_n^{\mathcal{T}})$ time slots in the case of LDT), because the consensus coefficients of all targets can be broadcast in parallel.

- For calculation of the CA beliefs $b^{(p)}(\mathbf{x}_{l,n})$ (see Section 3.3.2 and Step 2e in Algorithm 1), at each time n, CA $l \in \mathcal{S}$ broadcasts $N^{\mathrm{NBP}} \triangleq PJ\tilde{L}$ real values to CAs l' with $l \in \mathcal{M}_{l',n}^{\mathcal{S}}$. The delay contribution is P time slots, because each CA has to broadcast its belief in each message passing iteration.

- If the alternative processing of Section 3.4.2 is used, then at time $n = 1$, CA $l \in \mathcal{S}$ broadcasts $N^{\mathrm{AP}} \triangleq PJ\tilde{L}I|\mathcal{T}|$ real values to CAs $l' \in \mathcal{C}_{l,n}$ (in addition to $N^{\mathrm{C(R)}}$ and N^{NBP}). In the case of LDT, N^{AP} is reduced to $N_l^{\mathrm{APR}} \triangleq PJ\tilde{L}I_n^{\mathcal{T}}|\mathcal{M}_{l,1}^{\mathcal{T}}|$. Since the dissemination of the proposal distribution for each target is consensus-based, using I consensus iterations in each of the P message passing iterations, the contribution to the overall delay at $n = 1$ is PI time slots (or $PI_n^{\mathcal{T}}$ time slots in the case of LDT).

- In the case of LDT, if at time n a target $m \in \mathcal{T}$ enters the measurement region of CA $l \in \mathcal{S}$, i.e., $l \in \mathcal{S}_{m,n} \backslash \mathcal{S}_{m,n-1}$, then $N^{\mathrm{LDT}} \triangleq JL$ real values are transmitted from one arbitrary CA $l' \in \mathcal{S}_{m,n-1} \cap \mathcal{C}_{l,n}$ to CA l. The contribution of this transmission to the overall delay is one time slot.

Therefore, at time $n \geq 1$, the total number of real values broadcast by CA $l \in \mathcal{S}$ during P message passing iterations is

$$N^{\mathrm{TOT}} = N^{\mathrm{NBP}} + N^{\mathrm{C}} = P(J\tilde{L} + CJ|\mathcal{T}|)\,.$$

The corresponding delay is $P(C+I+1)$ time slots. If the alternative processing is used, then at time $n=1$,

$$N_1^{\text{TOT}} = N^{\text{AP}} + N^{\text{NBP}} + N^{\text{C}} = P(J\tilde{L}I|\mathcal{T}| + J\tilde{L} + CJ|\mathcal{T}|)\,.$$

This results in a delay of $P(2I+C+1)$ time slots. In the case of LDT, at time $n \geq 1$, the number of real values broadcast by CA $l \in \mathcal{S}$ during P message passing iterations is

$$N_{l,n}^{\text{TOTR}} = N^{\text{NBP}} + N_{l,n}^{\text{CR}} = P(J\tilde{L} + CJ|\mathcal{M}_{l,n}^{\mathcal{T}}|)\,.$$

Here, the case underlying N^{LDT} was neglected because its occurrence strongly depends on the network topology and is typically very rare. If the alternative processing is used, then at time $n=1$,

$$N_{l,1}^{\text{TOTR}} = N_l^{\text{APR}} + N^{\text{NBP}} + N_{l,1}^{\text{CR}} = P(J\tilde{L}I|\mathcal{M}_{l,1}^{\mathcal{T}}| + J\tilde{L} + CJ|\mathcal{M}_{l,1}^{\mathcal{T}}|)\,.$$

If LDT is employed, the delay is $P(I_n^{\mathcal{T}} + C + 2)$ time slots. Note that with LDT, C is smaller than in the standard case. Also note that $N_{l,n}^{\text{TOTR}} \leq N^{\text{TOT}}$ since $|\mathcal{M}_{l,n}^{\mathcal{T}}| \leq |\mathcal{T}|$ for all $l \in \mathcal{S}$. In addition, N^{NBP}, N^{AP}, N_l^{APR}, and N^{LDT} can be reduced by transmitting the parameters of a suitable parametric representation for the beliefs. Typically, these parameters are obtained by clustering the particles representing the beliefs (see Section 2.2.1 and [Savic and Zazo, 2012]).

3.5 Simulation Results

We will study the performance and communication requirements of the proposed CoSNAT message passing algorithm in two dynamic scenarios and in a static scenario. Simulation source files and animated plots are available at http://www.nt.tuwien.ac.at/about-us/staff/florian-meyer/.

3.5.1 Dynamic Scenarios

We consider a network of $|\mathcal{S}|=12$ CAs and $|\mathcal{T}|=2$ targets as depicted in Fig. 3.4. Eight CAs are mobile and four are static anchors (i.e., CAs with perfect position information). Each CA has a communication range of 50 and attempts to localize itself (except for the anchors) and the two targets. The states of the mobile CAs and targets consist of position and velocity, i.e., $\mathbf{x}_{k,n} \triangleq (x_{1,k,n}\ x_{2,k,n}\ \dot{x}_{1,k,n}\ \dot{x}_{2,k,n})^{\text{T}}$. CAs $l \in \mathcal{S}$ acquire distance measurements according to (1.3), i.e., $y_{l,k;n} = \|\tilde{\mathbf{x}}_{l,n} - \tilde{\mathbf{x}}_{k,n}\| + v_{l,k;n}$, where $\tilde{\mathbf{x}}_{k,n} \triangleq (x_{1,k,n}\ x_{2,k,n})^{\text{T}}$ is the position of agent $k \in \mathcal{A}$ and the measurement noise $v_{l,k;n}$ is independent across l, k, and n and Gaussian with variance $\sigma_v^2 = 2$.

The states of the mobile CAs and targets evolve independently according to [Li and Jilkov, 2003]

$$\mathbf{x}_{k,n} = \mathbf{G}\mathbf{x}_{k,n-1} + \mathbf{W}\mathbf{u}_{k,n}, \quad n=1,2,\ldots,$$

where

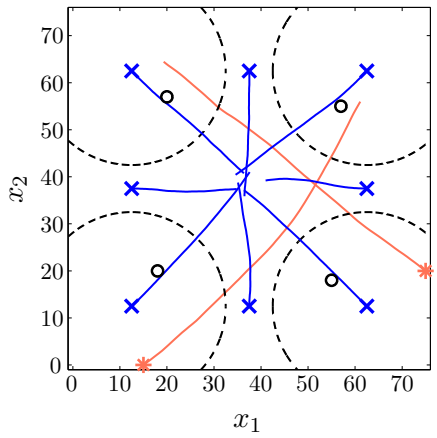


Figure 3.4: Network topology in the dynamic scenario, with initial agent positions and example realizations of the target and mobile CA trajectories. Initial mobile CA positions are indicated by crosses, anchor positions by circles, and initial target positions by stars. The dashed circles indicate the measurement regions of the four (mobile) CAs that are initially located near the corners.

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

The driving noise vectors $\mathbf{u}_{k,n} \in \mathbb{R}^2$ are Gaussian, i.e., $\mathbf{u}_{k,n} \sim \mathcal{N}(\mathbf{0}, \sigma_u^2 \mathbf{I})$, with variance $\sigma_u^2 = 5 \cdot 10^{-5}$ for the mobile CAs and $\sigma_u^2 = 5 \cdot 10^{-4}$ for the targets; furthermore, $\mathbf{u}_{k,n}$ and $\mathbf{u}_{k',n'}$ are independent unless $(k,n) = (k',n')$. Each mobile CA starts moving only when it is sufficiently localized in the sense that the empirical variance of the estimated position vector is below $5\sigma_v^2 = 10$; it then attempts to reach the center of the scene, $\tilde{\mathbf{x}}_\mathrm{c} = (37.5 \;\; 37.5)^\mathrm{T}$, in 75 time steps. The mobile CA trajectories are initialized using a Dirac-shaped prior located at $\boldsymbol{\mu}_{l,0} = \big(x_{1,l,0} \;\; x_{2,l,0} \;\; (\tilde{x}_{1,\mathrm{c}} - x_{1,l,0})/75 \;\; (\tilde{x}_{2,\mathrm{c}} - x_{2,l,0})/75\big)^\mathrm{T}$, where $x_{1,l,0}$ and $x_{2,l,0}$ are chosen as shown in Fig. 3.4. The two target trajectories are initialized using a Dirac-shaped prior located at $(15 \;\; 0 \;\; 0.8 \;\; 0.6)^\mathrm{T}$ and $(75 \;\; 20 \;\; -0.8 \;\; 0.6)^\mathrm{T}$ (see Fig. 3.4). Note that knowledge of these positions is not used in our simulations of the various algorithms.

We compare the proposed CoSNAT algorithm and its low-complexity variant according to Section 2.4—briefly termed "CoSNAT-1" and "CoSNAT-2," respectively—with that of a reference method that separately performs cooperative navigation by means of the NBP method of [Lien et al., 2012] and distributed tracking by means of a consensus-based distributed PF [Farahmand et al., 2011, Savic et al., 2014] as reviewed in Section 3.1; the latter uses the (mobile) CA position estimates provided by cooperative navigation. All three methods employ average consensus with Metropolis weights [Xiao and Boyd, 2003] for a distributed implementation. In all three methods, $P = 1$ message passing iteration and $J = 1000$ particles are used, and the alternative processing at time $n = 1$ described in Section 3.4.2 is employed. The average consensus uses $C = 6$ iterations. For our simulations of the algorithms, we used a position prior for targets and mobile CAs that is uniform on $[-200, 200] \times [-200, 200]$ and a Gaussian velocity

prior for the mobile CAs (after the mobile CA is sufficiently localized as described above) with mean $\big((\tilde{x}_{1,\mathrm{c}}-\hat{\tilde{x}}_{1,l,n'})/75 \;\; (\tilde{x}_{2,\mathrm{c}}-\hat{\tilde{x}}_{2,l,n'})/75\big)^{\mathrm{T}}$ and covariance matrix $\mathrm{diag}\{10^{-3}, 10^{-3}\}$. Here, $\hat{\tilde{\mathbf{x}}}_{l,n'}$ is the location estimate at the time n' at which mobile CA l is sufficiently localized for the first time. The velocity prior of the targets was Gaussian with mean $\boldsymbol{\mu}_{m,0}$ and covariance $\mathbf{C}_{m,0}$. Here, $\mathbf{C}_{m,0} = \mathrm{diag}\{0.001, 0.001\}$ represents the uncertainty in knowing the velocity $\dot{\mathbf{x}}_{m,0}$ of target m at time $n = 0$, and $\boldsymbol{\mu}_{m,0}$ is a random hyperparameter that was sampled for each simulation run from $\mathcal{N}(\dot{\mathbf{x}}_{m,0}, \mathbf{C}_{m,0})$.

We simulated two different dynamic scenarios. In dynamic scenario 1, the measurement range of those four mobile CAs that are initially located near the corners as shown in Fig. 3.4 (these CAs will be termed "corner CAs" in what follows) is limited as specified later whereas all the other CAs cover the entire field of size 75×75. In dynamic scenario 2, the measurement range of all CAs is limited to 20. We note that these scenarios cannot be tackled by SLAT algorithms [Taylor et al., 2006, Funiak et al., 2006, Savic and Wymeersch, 2013, Kantas et al., 2012, Teng et al., 2012] since they involve mobile CAs whereas SLAT assumes static CAs.

Fig. 3.5 shows the simulated ARMSE of self-localization and target localization for $n = 1, \ldots, 75$ in dynamic scenario 1 where the measurement range of the corner CAs is 20. The self-localization ARMSE was determined by averaging over all mobile CAs and over 100 simulation runs, and the target localization ARMSE by averaging over all mobile CAs, all targets, and 100 simulation runs. It is seen that CoSNAT-1 and CoSNAT-2 perform almost equally well, both with respect to self-localization and target tracking; thus, the significantly lower complexity of CoSNAT-2 is offset by only a small performance loss. Furthermore, the self-localization ARMSE of both CoSNAT algorithms is significantly smaller than that of the reference method. This is because with pure cooperative navigation, the corner CAs do not have enough partners for accurate self-localization, whereas with CoSNAT, they can use their measured distances to the targets to calculate the messages from the target nodes, $\phi^{(p)}_{m\to l}(\mathbf{x}_{m,n})$, which support self-localization. The target tracking ARMSEs of both CoSNAT algorithms and of the reference method are very similar at all times.

In Fig. 3.6, we show the self-localization and target localization ARMSEs averaged over time n versus the measurement range of the corner CAs. For small and large measurement range, CoSNAT performs similarly to the reference method but for different reasons: When the measurement range is smaller than 12.5, the targets appear in the measurement regions of the corner CAs only with a very small probability. Thus, at most times, the messages $\phi^{(p)}_{m\to l}(\mathbf{x}_{m,n})$ from the target nodes cannot be calculated. For measurement range larger than 25, the corner CAs measure three well-localized CAs at time $n = 1$, and thus they are also able to localize themselves using pure cooperative navigation. For measurement range between 15 and 25, CoSNAT significantly outperforms the reference method (cf. our discussion of Fig. 3.5). It is furthermore seen that up to a measurement range of 25, the target tracking ARMSE of the reference method increases with increasing measurement range. This is because in the reference method, only an estimate of the CA positions is used in distributed tracking. Therefore, the poorly localized corner CAs negatively affect distributed tracking, and this situation becomes

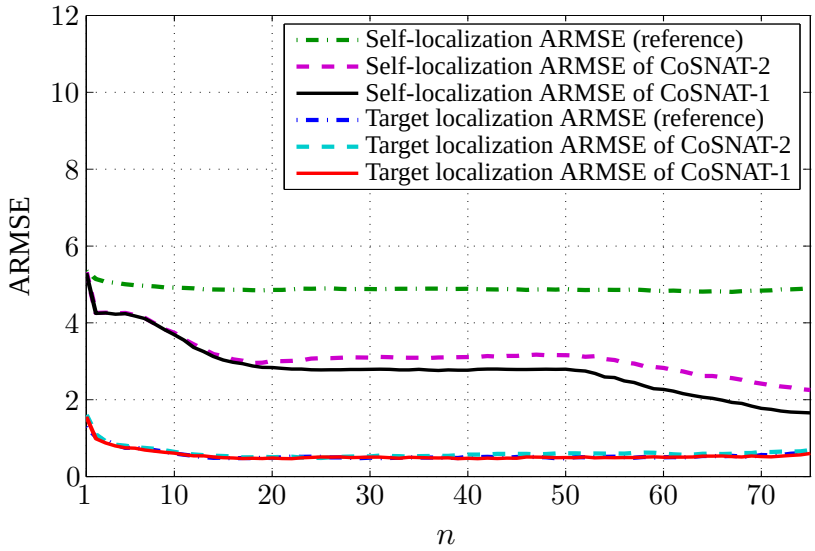


Figure 3.5: Self-localization and target localization ARMSEs versus time n (dynamic scenario 1).

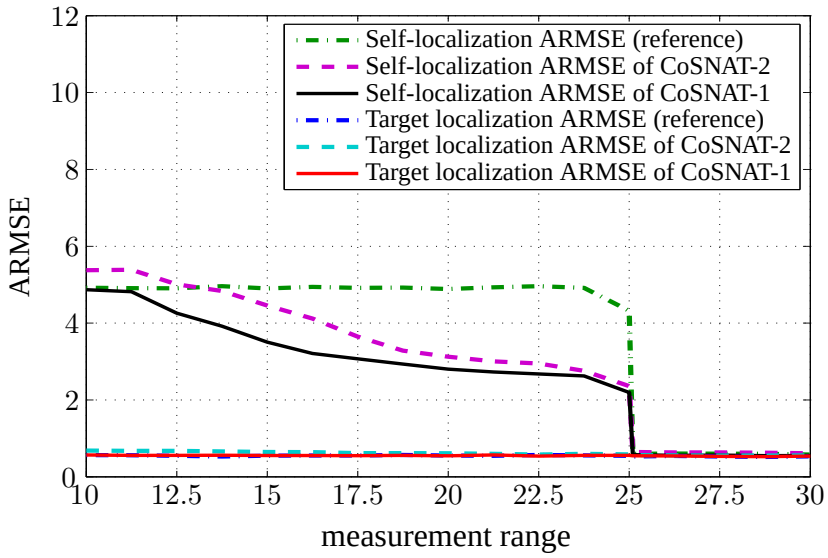


Figure 3.6: Self-localization and target localization ARMSEs versus measurement range of the corner CAs (dynamic scenario 1).

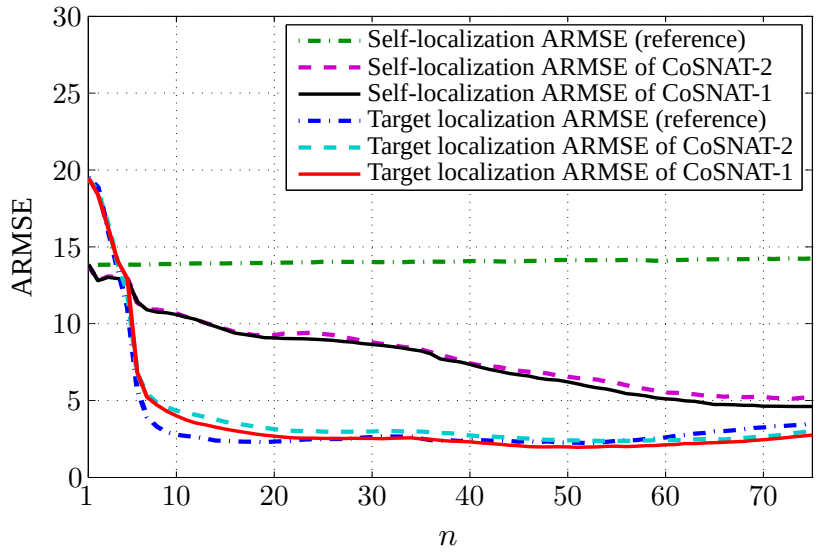


Figure 3.7: Self-localization and target localization ARMSEs versus time n (dynamic scenario 2).

more likely with increasing measurement range. By contrast, the target tracking ARMSE of CoSNAT stays constant for all measurement ranges. This is because in CoSNAT, the beliefs of the CA positions are used in distributed tracking, i.e., the actual uncertainty about the CA positions is taken into account. Thereby, the effect of poorly localized CAs on the tracking performance is considerably reduced.

Finally, Fig. 3.7 shows the self-localization and target localization ARMSEs for $n = 1, \ldots, 75$ in dynamic scenario 2 (i.e., the measurement range of all CAs is 20). It can be seen that with all three methods, the targets are roughly localized after a few initial time steps. However, with the reference method, due to the limited measurement range, not even a single CA can be localized. With both CoSNAT methods, once meaningful probabilistic information about the target positions is available, also the self-localization ARMSE decreases and most of the CAs can be localized after some time. This is possible since the CAs obtain additional information related to the measured targets. Which CAs are localized at what times depends on the target trajectories and varies between the simulation runs.

The quantities determining the communication requirements of CoSNAT-1 and CoSNAT-2 according to Section 3.4.3 are as follows. We have $N^{\mathrm{C}} = 18000$, $N^{\mathrm{NBP}} = 2000$, and $N^{\mathrm{AP}} = 12000$. For all CAs l, $N_{l,n}^{\mathrm{CR}} = 9000$ at times n where one target is measured, $N_{l,n}^{\mathrm{CR}} = N^{\mathrm{C}} = 18000$ at times n where two targets are measured, and $N_{l,n}^{\mathrm{CR}} = 0$ otherwise. Furthermore, $N_l^{\mathrm{APR}} = 6000$ for CAs that measure one target at time $n = 1$, $N_l^{\mathrm{APR}} = N^{\mathrm{AP}} = 12000$ for CAs that measure two targets at time $n = 1$, and $N_l^{\mathrm{APR}} = 0$ for all other CAs. The resulting average total communication requirement per CA and time step, averaged over all CAs, all times, and all simulation runs, is $\overline{N^{\mathrm{TOT}}} = 18107$ and $\overline{N^{\mathrm{TOTR}}} = 12916$ for dynamic scenario 1 and $\overline{N^{\mathrm{TOT}}} = 18107$ and $\overline{N^{\mathrm{TOTR}}} = 5440$ for dynamic scenario 2. At the times where a target enters the measurement region of a CA, additionally $N^{\mathrm{LDT}} = 4000$. According to Section 3.4.3, in dynamic scenario 1, the delay at time $n = 1$ is 13 time slots (14 in the LDT case), and at

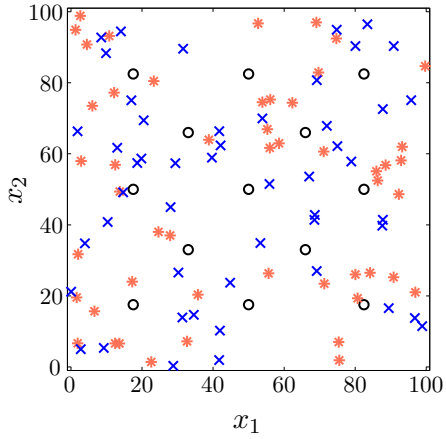


Figure 3.8: Network topology in the static scenario, with anchor positions and example realizations of non-anchor CA and target positions. Non-anchor CA positions are indicated by crosses, anchor positions by circles, and target positions by stars.

times $n = 2, \ldots, 75$, it is 10 time slots (11 in the LDT case). In dynamic scenario 2, the delay at time $n = 1$ is 13 time slots (12 in the LDT case), and at times $n = 2, \ldots, 75$, it is 10 time slots (9 in the LDT case). Note that in both dynamic scenarios, the reference method has the same communication requirements and causes the same delay as CoSNAT-1 and CoSNAT-2.

3.5.2 Static Scenario

Next, we consider a network of $|\mathcal{S}| = 63$ static CAs and $|\mathcal{T}| = 50$ static targets. 13 CAs are anchors located as depicted in Fig. 3.8. The 50 remaining CAs and the 50 targets are randomly (uniformly) placed in a field of size 100×100; a realization of the positions of the non-anchor CAs and targets is shown in Fig. 3.8. The states of the non-anchor CAs and of the targets are the positions, i.e., $\mathbf{x}_{k,n} = \tilde{\mathbf{x}}_{k,n} = (x_{1,k,n} \;\; x_{2,k,n})^{\mathrm{T}}$. Each CA performs distance measurements according to (1.3) with noise variance $\sigma_v^2 = 2$. All CAs have a measurement range of 22.5. The communication range of each CA is 50. The prior for the non-anchor CAs and for the targets is uniform on $[-200, 200] \times [-200, 200]$. In all three methods, $J = 1000$ particles are used. The average consensus scheme uses $C = 15$ iterations. Since all CAs and targets are static, we simulate only a single time step. This scenario is similar to that considered in [Wymeersch et al., 2009] for pure cooperative navigation, except that 50 of the CAs used in [Wymeersch et al., 2009] are replaced by targets and also anchor nodes perform measurements.

Fig. 3.9 shows the overall agent localization ARMSE (i.e., the average ARMSE of both CA self-localization and target localization) versus the message passing iteration index p. This error was determined by averaging over all agents and over 100 simulation runs. It is seen that CoSNAT-1 and CoSNAT-2 perform almost equally well, with a slightly faster convergence of CoSNAT-1, and significantly better than the reference method. Again, the better performance of CoSNAT is due to the fact that CAs that do not have enough partners for self-localization can use messages $\phi_{m \to l}^{(p)}(\mathbf{x}_{m,n})$ from well-localized target nodes to better localize themselves.

In this scenario, assuming $P = 3$, we have $N^{\mathrm{NBP}} = 6000$ and $N^{\mathrm{C}} = 2.70 \cdot 10^6$. The

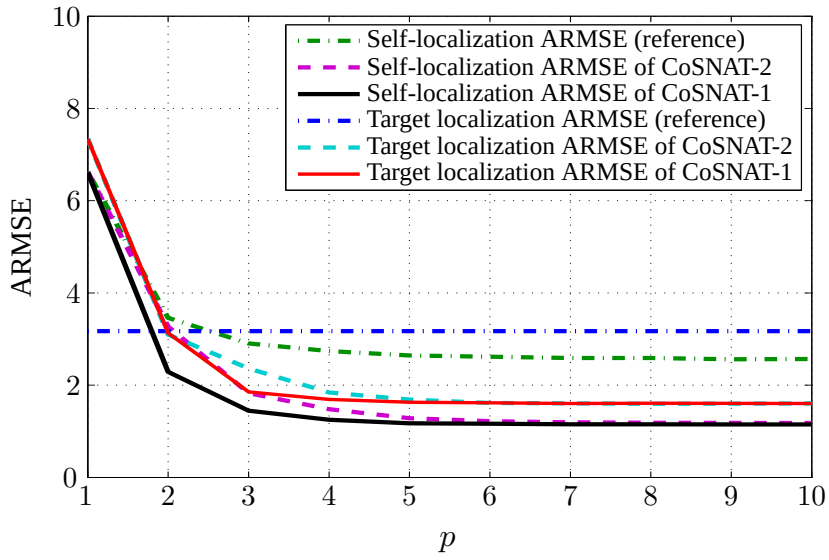


Figure 3.9: Agent localization ARMSE versus message passing iteration index p (static scenario).

remaining quantities determining the communication requirements depend on the positions of the non-anchor CAs and the targets, which are randomly chosen for each simulation run. The average quantities—averaged over all CAs, all times, and all simulation runs—were obtained as $\overline{N^{\mathrm{CR}}} = 6.91 \cdot 10^5$, $\overline{N^{\mathrm{AP}}} = 9.00 \cdot 10^5$, and $\overline{N^{\mathrm{APR}}} = 2.59 \cdot 10^5$. The average total communication requirements were obtained as $\overline{N^{\mathrm{TOT}}} = 3.61 \cdot 10^6$ and $\overline{N^{\mathrm{TOTR}}} = 9.56 \cdot 10^5$. Note that $\overline{N^{\mathrm{TOTR}}}$ is significantly smaller than $\overline{N^{\mathrm{TOT}}}$, due to the relatively large network size. For the reference method, we obtained $\overline{N_{\mathrm{ref}}^{\mathrm{TOT}}} = 1.21 \cdot 10^6$ and $\overline{N_{\mathrm{ref}}^{\mathrm{TOTR}}} = 3.23 \cdot 10^5$. The delay is 57 time slots (54 in the case of LDT).

Chapter 4

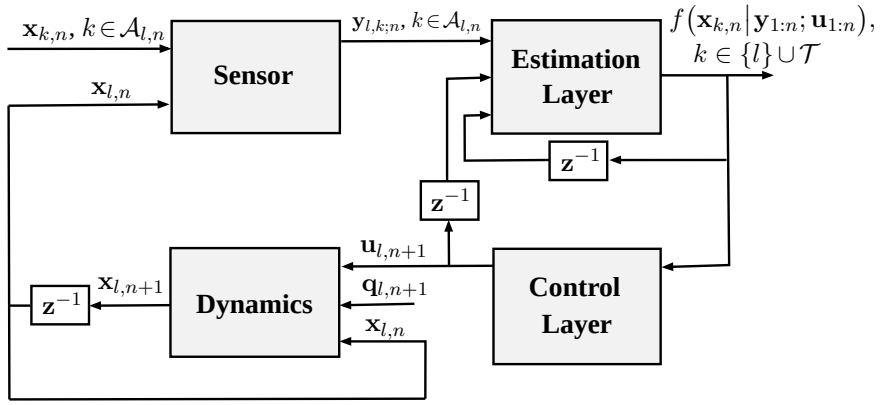
Information-Seeking Control for Navigation and Tracking

To further increase the accuracy and robustness of cooperative navigation and distributed tracking, we now extend the CoSNAT estimation algorithm developed in Chapter 3 to include cooperative information-seeking control. The aim of the proposed controller is to move the CAs in such a way that their measurements are maximally informative about the states to be estimated. Similar to the estimation algorithms developed in Chapters 2 and 3, the proposed controller is suited to nonlinear and non-Gaussian navigation and tracking problems, requires only communication with neighboring CAs, and is able to cope with a changing network topology. The global (holistic) objective function used for control is the negative joint posterior entropy of all states in the network at the next time step conditioned on all measurements at the next time step. An approximate maximization of this objective function is performed jointly by all CAs via a gradient ascent, which reduces to the particle-based evaluation of a local gradient at each CA. This evaluation is based on a probabilistic information transfer from the estimation layer to the control layer. We show through simulations that the proposed algorithm for combined CoSNAT estimation and control leads to intelligent behavior of the CAs and excellent navigation/tracking performance.

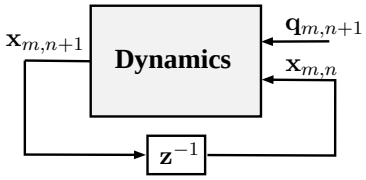
Contrary to Chapters 2 and 3, where the control variables of the CAs were surpressed, we will now use the full-blown state evolution model that includes the control variables $\mathbf{u}_{l,n}$. That is, the state evolution model for CAs $l \in \mathcal{S}$ is $\mathbf{x}_{l,n} = g_l(\mathbf{x}_{l,n-1}, \mathbf{u}_{l,n}, \mathbf{q}_{l,n})$ (cf, (1.1)), with corresponding state-transition PDF $f(\mathbf{x}_{l,n}|\mathbf{x}_{l,n-1}; \mathbf{u}_{l,n})$.

4.1 The Combined Estimation and Control Problem

In Chapters 2 and 3 we presented estimation algorithms for navigation and tracking in agent networks. Now, estimation is extended by information-seeking control of the agents to obtain higher navigation and tracking accuracy. A block diagram of the overall "signal processing system" for this setting is shown in Fig. 4.1 for a CA (including the estimation and control layers



(a)



(b)

Figure 4.1: Block diagram of the overall "signal processing system" for (a) a CA $l \in \mathcal{S}$ and (b) a target $m \in \mathcal{T}$.

of the proposed method) and for a target. Note that in this chapter, to simplify the notation, we switch to a different topology model. More specifically, we assume that CAs acquire measurements with respect to each other if and only if they are able to communicate. The methods proposed in this section can be generalized in a straightforward manner to scenarios where the measurement neighborhoods are subsets of the communication neighborhoods as in the previous sections. The communication and measurement topology of the network is described by the neighborhood sets $\mathcal{C}_{l,n}$, $\mathcal{T}_{l,n}$, and $\mathcal{A}_{l,n}$ as follows. CA l is able to receive data via a communication link from CA l' if $l' \in \mathcal{C}_{l,n} \subseteq \mathcal{S} \backslash \{l\}$. In addition, CA l is able to acquire a measurement $\mathbf{y}_{l,l';n}$ relative to CA l' if $l' \in \mathcal{C}_{l,n}$. This relation is symmetric, i.e., $l' \in \mathcal{C}_{l,n}$ implies $l \in \mathcal{C}_{l',n}$. Furthermore, CA $l \in \mathcal{S}$ acquires a measurement $\mathbf{y}_{l,m;n}$ relative to target m if $m \in \mathcal{T}_{l,n} \subseteq \mathcal{T}$. The targets are noncooperative in that they do not communicate and do not acquire any measurements.

The operations performed in the dynamics and sensor block are described by the state transition model in (1.1) and measurement model in (1.2), respectively. Algorithms for the *estimation layer* were discussed in Chapters 2 and 3. An information-seeking control algorithm for the *control layer* will be developed of this chapter. The twofold goal at each time n can be stated as follows:

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

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 in the *estimation layer* and the *control layer*, as shown in Fig. 4.1(a). In the estimation layer described in Chapters 2 and 3, CA l computes an approximation of the marginal posterior PDFs of the states $\mathbf{x}_{k,n}$, $k \in \{l\} \cup \mathcal{T}$ given all the past and present measurements and control vectors in the entire network. In the control layer presented in what follows, CA l uses these marginal posteriors and the statistical model to determine an optimal control variable $\mathbf{u}_{l,n+1}$; in both layers, the CAs communicate with neighboring CAs.

4.2 Objective Function and Controller

According to our definition at the beginning of Section 1.5.3, the vector comprising all the measurements in the network at the next time $n+1$ is $\mathbf{y}_{n+1} = \big[\mathbf{y}_{l,k;n+1}\big]_{l\in\mathcal{S}, k\in\mathcal{A}_{l,n+1}}$. However, in this definition of $\mathbf{y}_{n+1}$, we now formally replace $\mathcal{A}_{l,n+1}$ by $\mathcal{A}_{l,n}$ since at the current time n, the sets $\mathcal{A}_{l,n+1}$ are not yet known. Thus, with an abuse of notation, $\mathbf{y}_{n+1}$ is redefined as

$$\mathbf{y}_{n+1} \triangleq \big[\mathbf{y}_{l,k;n+1}\big]_{l\in\mathcal{S}, k\in\mathcal{A}_{l,n}}. \tag{4.1}$$

In the proposed control approach, each CA $l \in \mathcal{S}$ determines its next control variable $\mathbf{u}_{l,n+1}$ 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 [Cover and Thomas, 2006, 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:

$$\begin{aligned} -h\big(\mathbf{x}_{n+1}\big|\mathbf{y}_{n+1};\mathbf{y}_{1:n},\mathbf{u}_{1:n+1}\big) = \iint & f\big(\mathbf{x}_{n+1},\mathbf{y}_{n+1}\big|\mathbf{y}_{1:n};\mathbf{u}_{1:n+1}\big) \\ & \times \log f\big(\mathbf{x}_{n+1}\big|\mathbf{y}_{n+1},\mathbf{y}_{1:n};\mathbf{u}_{1:n+1}\big)\mathrm{d}\mathbf{x}_{n+1}\,\mathrm{d}\mathbf{y}_{n+1}, \end{aligned} \tag{4.2}$$

where $\log$ denotes the natural logarithm. Note that $h\big(\mathbf{x}_{n+1}\big|\mathbf{y}_{n+1};\mathbf{y}_{1:n},\mathbf{u}_{1:n+1}\big)$ depends on the *random vectors* $\mathbf{x}_{n+1}$ and $\mathbf{y}_{n+1}$, i.e., on their joint distribution but not on their values. Our notation indicates this fact by using a sans serif font for $\mathbf{x}_{n+1}$ and $\mathbf{y}_{n+1}$ in $h\big(\mathbf{x}_{n+1}\big|\mathbf{y}_{n+1};\mathbf{y}_{1:n},\mathbf{u}_{1:n+1}\big)$.

According to expression (4.2), $-h\big(\mathbf{x}_{n+1}\big|\mathbf{y}_{n+1};\mathbf{y}_{1:n},\mathbf{u}_{1:n+1}\big)$ is a function of the control vector $\mathbf{u}_{n+1}$. This function will be denoted as $D_h\big(\mathbf{u}_{n+1}\big)$, i.e.,

$$D_h\big(\mathbf{u}_{n+1}\big) \triangleq -h\big(\mathbf{x}_{n+1}\big|\mathbf{y}_{n+1};\mathbf{y}_{1:n},\mathbf{u}_{1:n+1}\big), \tag{4.3}$$

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), $\mathbf{x}_{n+1}$, and *all* the next measurements, $\mathbf{y}_{n+1}$.

Instead of a full-blown maximization of $D_h\big(\mathbf{u}_{n+1}\big)$, we perform one step of a gradient ascent [Fletcher, 1987] at each time n. Thus, $\mathbf{u}_{n+1}$ is determined as

$$\hat{\mathbf{u}}_{n+1} \;=\; \mathbf{u}_{n+1}^{(\mathrm{r})} + c_{n+1}\, \nabla D_h\big(\mathbf{u}_{n+1}\big)\big|_{\mathbf{u}_{n+1}=\mathbf{u}_{n+1}^{(\mathrm{r})}}\,, \tag{4.4}$$

where $\mathbf{u}_{n+1}^{(\mathrm{r})}$ is a reference vector and $c_{n+1} > 0$ is a step size. The choice of $\mathbf{u}_{n+1}^{(\mathrm{r})}$ depends on the manner in which the local control vectors $\mathbf{u}_{l,n}$ (which are subvectors of $\mathbf{u}_n$) enter into the state evolution functions $g_l\big(\mathbf{x}_{l,n-1}, \mathbf{u}_{l,n}, \mathbf{q}_{l,n}\big)$ in (1.1); two common choices are $\mathbf{u}_{n+1}^{(\mathrm{r})} = \mathbf{u}_n$ and $\mathbf{u}_{n+1}^{(\mathrm{r})} = \mathbf{0}$ (cf. Section 4.6.1).

Since $\mathbf{u}_{n+1} = \big[\mathbf{u}_{l,n+1}\big]_{l\in\mathcal{S}}$, we have

$$\nabla D_h(\mathbf{u}_{n+1}) = \Big[\frac{\partial D_h(\mathbf{u}_{n+1})}{\partial \mathbf{u}_{l,n+1}}\Big]_{l\in\mathcal{S}},$$

and thus the gradient ascent (4.4) with respect to $\mathbf{u}_{n+1}$ is equivalent to local gradient ascents at the individual CAs l with respect to the local control vectors $\mathbf{u}_{l,n+1}$. At CA l, the local gradient ascent is performed as

$$\hat{\mathbf{u}}_{l,n+1} \;=\; \mathbf{u}_{l,n+1}^{(\mathrm{r})} + c_{l,n+1}\, \frac{\partial D_h\big(\mathbf{u}_{n+1}\big)}{\partial \mathbf{u}_{l,n+1}}\bigg|_{\mathbf{u}_{n+1}=\mathbf{u}_{n+1}^{(\mathrm{r})}}\,, \tag{4.5}$$

where $\mathbf{u}_{l,n+1}^{(\mathrm{r})}$ is the part of $\mathbf{u}_{n+1}^{(\mathrm{r})}$ that corresponds to CA l (we have $\mathbf{u}_{n+1}^{(\mathrm{r})} = \big[\mathbf{u}_{l,n+1}^{(\mathrm{r})}\big]_{l\in\mathcal{S}}$). The local step sizes $c_{l,n+1}$ are constrained by the condition $\mathbf{u}_{l,n+1} \in \mathcal{U}_l$ for given domains $\mathcal{U}_l$. In practice, this condition can be easily satisfied by an appropriate scaling of the $c_{l,n+1}$. Note that, as in [Julian et al., 2012], 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 $\mathcal{U}_l$ 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. Because the objective function changes from one time step to another, the local ascent described by (4.5) generally does not converge; this is similar to existing information-seeking control algorithms [Hoffmann and Tomlin, 2010, Julian et al., 2012]. Indeed, the goal of the proposed control algorithm is to make available informative measurements to the estimation layer; because of the dynamic scenario, this does not correspond to a fixed optimum.

4.3 Expansion of the Objective Function

A central contribution of this work is a distributed particle-based technique for calculating the gradients $\frac{\partial D_h(\mathbf{u}_{n+1})}{\partial \mathbf{u}_{l,n+1}}\Big|_{\mathbf{u}_{n+1}=\mathbf{u}_{n+1}^{(\mathrm{r})}}$ in (4.5).

As a starting point for developing this technique, we next derive an expansion of the objective function $D_h\big(\mathbf{u}_{n+1}\big)$. We will use the following simplified notation. We do not indicate the conditioning on $\mathbf{y}_{1:n}$ and $\mathbf{u}_{1:n}$ because at time $n+1$, $\mathbf{y}_{1:n}$ has already been observed and $\mathbf{u}_{1:n}$ has already been determined; hence both are fixed. Also, we suppress the time index n and

designate variables at time $n+1$ by the superscript "+". For example, we write $h(\mathbf{x}^+|\mathbf{y}^+;\mathbf{u}^+)$ instead of $h\big(\mathbf{x}_{n+1}\big|\mathbf{y}_{n+1};\mathbf{y}_{1:n},\mathbf{u}_{1:n+1}\big)$.

For calculating the gradient, following [Hoffmann and Tomlin, 2010] and [Julian et al., 2012], we disregard the unknown driving noise $\mathbf{q}_l$ in (1.1) for $k=l\in\mathcal{S}$ by formally replacing it with its expectation, $\bar{\mathbf{q}}_l \triangleq \int \mathbf{q}_l\, f(\mathbf{q}_l)\,\mathrm{d}\mathbf{q}_l$. We can then rewrite (1.1) for $k=l\in\mathcal{S}$ (with n replaced by $n+1$) as

$$\mathbf{x}_l^+ \,=\, g_l(\mathbf{x}_l,\mathbf{u}_l^+,\bar{\mathbf{q}}_l^+) \,=\, \tilde{g}_l(\mathbf{x}_l,\mathbf{u}_l^+)\,,\quad l\in\mathcal{S}\,. \tag{4.6}$$

As shown in Appendix A, the conditional differential entropy in (4.3) can be expressed as

$$h(\mathbf{x}^+|\mathbf{y}^+;\mathbf{u}^+) \,=\, h(\mathbf{x}_\mathcal{S},\mathbf{x}_\mathcal{T}^+|\mathbf{y}^+;\mathbf{u}^+) \,+\, \sum_{l\in\mathcal{S}} G_l(\mathbf{u}_l^+)\,, \tag{4.7}$$

where $\mathbf{x}_\mathcal{S}\triangleq\big[\mathbf{x}_l\big]_{l\in\mathcal{S}}$, $\mathbf{x}_\mathcal{T}^+\triangleq\big[\mathbf{x}_m^+\big]_{m\in\mathcal{T}}$, and

$$G_l(\mathbf{u}_l^+) \,\triangleq\, \int f(\mathbf{x}_l)\,\log|J_{\tilde{g}_l}(\mathbf{x}_l;\mathbf{u}_l^+)|\,\mathrm{d}\mathbf{x}_l \qquad \text{with}\;\; J_{\tilde{g}_l}(\mathbf{x}_l;\mathbf{u}_l^+) \,\triangleq\, \det\frac{\partial\tilde{g}_l(\mathbf{x}_l,\mathbf{u}_l^+)}{\partial\mathbf{x}_l}\,. \tag{4.8}$$

The first term on the right-hand side of (4.7) can be decomposed as [Cover and Thomas, 2006, Chapter 8]

$$h(\mathbf{x}_\mathcal{S},\mathbf{x}_\mathcal{T}^+|\mathbf{y}^+;\mathbf{u}^+) \,=\, h(\mathbf{x}_\mathcal{S},\mathbf{x}_\mathcal{T}^+) - I(\mathbf{x}_\mathcal{S},\mathbf{x}_\mathcal{T}^+;\mathbf{y}^+;\mathbf{u}^+)\,. \tag{4.9}$$

Here, $I(\mathbf{x}_\mathcal{S},\mathbf{x}_\mathcal{T}^+;\mathbf{y}^+;\mathbf{u}^+)$ denotes the two-variable mutual information between $(\mathbf{x}_\mathcal{S},\mathbf{x}_\mathcal{T}^+)$ and $\mathbf{y}^+$ (with $\mathbf{u}^+$ being a deterministic parameter, i.e., not a third random variable), which is given by [Cover and Thomas, 2006, Chapter 8]

$$I(\mathbf{x}_\mathcal{S},\mathbf{x}_\mathcal{T}^+;\mathbf{y}^+;\mathbf{u}^+) = \iiint f(\mathbf{x}_\mathcal{S},\mathbf{x}_\mathcal{T}^+,\mathbf{y}^+;\mathbf{u}^+)\,\log\frac{f(\mathbf{x}_\mathcal{S},\mathbf{x}_\mathcal{T}^+,\mathbf{y}^+;\mathbf{u}^+)}{f(\mathbf{x}_\mathcal{S},\mathbf{x}_\mathcal{T}^+)\,f(\mathbf{y}^+;\mathbf{u}^+)}\mathrm{d}\mathbf{x}_\mathcal{S}\,\mathrm{d}\mathbf{x}_\mathcal{T}^+\,\mathrm{d}\mathbf{y}^+. \tag{4.10}$$

Note that $h(\mathbf{x}_\mathcal{S},\mathbf{x}_\mathcal{T}^+)$ in (4.9) does not depend on $\mathbf{u}^+$, since neither the CA states $\mathbf{x}_\mathcal{S}$ nor the future target states $\mathbf{x}_\mathcal{T}^+$ are controlled by the future control variable $\mathbf{u}^+$. We explicitly express the dependence of $I(\mathbf{x}_\mathcal{S},\mathbf{x}_\mathcal{T}^+;\mathbf{y}^+;\mathbf{u}^+)$ on $\mathbf{u}^+$ by defining the function

$$D_I(\mathbf{u}^+) \,\triangleq\, I(\mathbf{x}_\mathcal{S},\mathbf{x}_\mathcal{T}^+;\mathbf{y}^+;\mathbf{u}^+)\,. \tag{4.11}$$

Combining (4.3), (4.7), (4.9), and (4.11) then yields the following expansion of the objective function:

$$D_h(\mathbf{u}^+) \,=\, -h(\mathbf{x}_\mathcal{S},\mathbf{x}_\mathcal{T}^+) \,+\, D_I(\mathbf{u}^+) \,-\, G_l(\mathbf{u}_l^+) - \sum_{l'\in\mathcal{S}\setminus\{l\}} G_{l'}(\mathbf{u}_{l'}^+)\,. \tag{4.12}$$

This entails the following expansion of the gradient in (4.5):

$$\frac{\partial D_h(\mathbf{u}^+)}{\partial \mathbf{u}_l^+} = \frac{\partial D_I(\mathbf{u}^+)}{\partial \mathbf{u}_l^+} - \frac{\partial G_l(\mathbf{u}_l^+)}{\partial \mathbf{u}_l^+}\,. \tag{4.13}$$

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

4.4 Calculation of the Gradients

4.4.1 Gradient of $D_I(\mathbf{u}^+)$

The mutual information in (4.10) can be rewritten as

$$D_I(\mathbf{u}^+) = \iiint f(\mathbf{y}^+|\mathbf{x}_\mathcal{S}, \mathbf{x}_\mathcal{T}^+; \mathbf{u}^+)\, f(\mathbf{x}_\mathcal{S}, \mathbf{x}_\mathcal{T}^+) \log \frac{f(\mathbf{y}^+|\mathbf{x}_\mathcal{S}, \mathbf{x}_\mathcal{T}^+; \mathbf{u}^+)}{f(\mathbf{y}^+; \mathbf{u}^+)}\, \mathrm{d}\mathbf{x}_\mathcal{S}\, \mathrm{d}\mathbf{x}_\mathcal{T}^+\, \mathrm{d}\mathbf{y}^+.$$

Invoking [Julian et al., 2012, Th. 1], we obtain

$$\frac{\partial D_I(\mathbf{u}^+)}{\partial \mathbf{u}_l^+} = \iiint \frac{\partial f(\mathbf{y}^+|\mathbf{x}_\mathcal{S}, \mathbf{x}_\mathcal{T}^+; \mathbf{u}^+)}{\partial \mathbf{u}_l^+}\, f(\mathbf{x}_\mathcal{S}, \mathbf{x}_\mathcal{T}^+) \log \frac{f(\mathbf{y}^+|\mathbf{x}_\mathcal{S}, \mathbf{x}_\mathcal{T}^+; \mathbf{u}^+)}{f(\mathbf{y}^+; \mathbf{u}^+)}\, \mathrm{d}\mathbf{x}_\mathcal{S}\, \mathrm{d}\mathbf{x}_\mathcal{T}^+\, \mathrm{d}\mathbf{y}^+. \tag{4.14}$$

Next, we will develop a Monte Carlo approximation of $\frac{D_I(\mathbf{u}^+)}{\partial \mathbf{u}_l^+}\Big|_{\mathbf{u}^+=\mathbf{u}^{+(\mathrm{r})}}$ that uses importance sampling and is based on a factorization of the likelihood function $f(\mathbf{y}^+|\mathbf{x}_\mathcal{S}, \mathbf{x}_\mathcal{T}^+; \mathbf{u}^+)$. Subsequently, we will address the distributed computation of this approximation.

Factorization of the Likelihood Function for CA l

The likelihood function $f(\mathbf{y}^+|\mathbf{x}_\mathcal{S}, \mathbf{x}_\mathcal{T}^+; \mathbf{u}^+)$ involved in (4.14) can be written as

$$f(\mathbf{y}^+|\mathbf{x}_\mathcal{S}, \mathbf{x}_\mathcal{T}^+; \mathbf{u}^+) = \prod_{l\in\mathcal{S}} \prod_{l'\in\mathcal{C}_l} f(\mathbf{y}_{l,l'}^+|\mathbf{x}_l, \mathbf{x}_{l'}; \mathbf{u}_l^+, \mathbf{u}_{l'}^+) \prod_{m\in\mathcal{T}_l} f(\mathbf{y}_{l,m}^+|\mathbf{x}_l, \mathbf{x}_m^+; \mathbf{u}_l^+)\,. \tag{4.15}$$

Using (4.6), the local likelihood functions involved in (4.15) are expressed as

$$\begin{aligned} f(\mathbf{y}_{l,l'}^+|\mathbf{x}_l, \mathbf{x}_{l'}; \mathbf{u}_l^+, \mathbf{u}_{l'}^+) &= f(\mathbf{y}_{l,l'}^+|\mathbf{x}_l^+, \mathbf{x}_{l'}^+)\big|_{\mathbf{x}_l^+=\tilde{g}_l(\mathbf{x}_l,\mathbf{u}_l^+),\, \mathbf{x}_{l'}^+=\tilde{g}_{l'}(\mathbf{x}_{l'},\mathbf{u}_{l'}^+)}\,,\ l\in\mathcal{S},\ l'\in\mathcal{C}_l \\ f(\mathbf{y}_{l,m}^+|\mathbf{x}_l, \mathbf{x}_m^+; \mathbf{u}_l^+) &= f(\mathbf{y}_{l,m}^+|\mathbf{x}_l^+, \mathbf{x}_m^+)\big|_{\mathbf{x}_l^+=\tilde{g}_l(\mathbf{x}_l,\mathbf{u}_l^+)}\,,\ l\in\mathcal{S},\ m\in\mathcal{T}_l\,. \end{aligned}$$

We can rewrite (4.15) for an arbitrary $l \in \mathcal{S}$ as

$$f(\mathbf{y}^+|\mathbf{x}_\mathcal{S}, \mathbf{x}_\mathcal{T}^+; \mathbf{u}^+) \;=\; \alpha_l(\mathbf{y}^+, \mathbf{x}_\mathcal{S}, \mathbf{x}_\mathcal{T}^+, \mathbf{u}^+)\, \beta_l(\mathbf{y}^+, \mathbf{x}_\mathcal{S}, \mathbf{x}_\mathcal{T}^+, \mathbf{u}^+)\,, \tag{4.16}$$

with

$$\alpha_l(\mathbf{y}^+, \mathbf{x}_\mathcal{S}, \mathbf{x}_\mathcal{T}^+, \mathbf{u}^+) \triangleq \prod_{l' \in \mathcal{C}_l} f(\mathbf{y}_{l,l'}^+|\mathbf{x}_l, \mathbf{x}_{l'}; \mathbf{u}_l^+, \mathbf{u}_{l'}^+) \prod_{m \in \mathcal{T}_l} f(\mathbf{y}_{l,m}^+|\mathbf{x}_l, \mathbf{x}_m^+; \mathbf{u}_l^+) \tag{4.17}$$

$$\beta_l(\mathbf{y}^+, \mathbf{x}_\mathcal{S}, \mathbf{x}_\mathcal{T}^+, \mathbf{u}^+) \triangleq \prod_{l_1 \in \mathcal{S}\setminus\{l\}} \prod_{l_1' \in \mathcal{C}_{l_1}} f(\mathbf{y}_{l_1,l_1'}^+|\mathbf{x}_{l_1}, \mathbf{x}_{l_1'}; \mathbf{u}_{l_1}^+, \mathbf{u}_{l_1'}^+) \prod_{m' \in \mathcal{T}_{l_1}} f(\mathbf{y}_{l_1,m'}^+|\mathbf{x}_{l_1}, \mathbf{x}_{m'}^+; \mathbf{u}_{l_1}^+)\,. \tag{4.18}$$

Here, $\alpha_l(\mathbf{y}^+, \mathbf{x}_\mathcal{S}, \mathbf{x}_\mathcal{T}^+, \mathbf{u}^+)$ depends on the local control vector $\mathbf{u}_l^+$ whereas $\beta_l(\mathbf{y}^+, \mathbf{x}_\mathcal{S}, \mathbf{x}_\mathcal{T}^+, \mathbf{u}^+)$ does not.

Monte Carlo Approximation

In Appendix B, it is shown that a Monte Carlo (i.e., particle-based) approximation of (4.14) evaluated at $\mathbf{u}^+ = \mathbf{u}^{+(\mathrm{r})}$ is given by[1]

$$\begin{aligned}\left.\frac{\partial D_I(\mathbf{u}^+)}{\partial \mathbf{u}_l^+}\right|_{\mathbf{u}^+=\mathbf{u}^{+(\mathrm{r})}} \approx\; & \frac{1}{JJ'} \sum_{j=1}^{J} \sum_{j'=1}^{J'} \frac{1}{\alpha_l\big(\mathbf{y}^{+(j,j')}, \mathbf{x}_\mathcal{S}^{(j)}, \mathbf{x}_\mathcal{T}^{+(j)}, \mathbf{u}^{+(\mathrm{r})}\big)} \\ & \times \left.\frac{\partial \alpha_l\big(\mathbf{y}^{+(j,j')}, \mathbf{x}_\mathcal{S}^{(j)}, \mathbf{x}_\mathcal{T}^{+(j)}, \mathbf{u}^+\big)}{\partial \mathbf{u}_l^+}\right|_{\mathbf{u}^+=\mathbf{u}^{+(\mathrm{r})}} \\ & \times \log \frac{\alpha_l\big(\mathbf{y}^{+(j,j')}, \mathbf{x}_\mathcal{S}^{(j)}, \mathbf{x}_\mathcal{T}^{+(j)}, \mathbf{u}^{+(\mathrm{r})}\big)\, \beta_l\big(\mathbf{y}^{+(j,j')}, \mathbf{x}_\mathcal{S}^{(j)}, \mathbf{x}_\mathcal{T}^{+(j)}, \mathbf{u}^{+(\mathrm{r})}\big)}{f\big(\mathbf{y}^{+(j,j')}; \mathbf{u}^+ = \mathbf{u}^{+(\mathrm{r})}\big)}\,,\end{aligned} \tag{4.19}$$

with

$$\begin{aligned} f\big(\mathbf{y}^{+(j,j')}; \mathbf{u}^+ = \mathbf{u}^{+(\mathrm{r})}\big) \approx\; & \frac{1}{J} \sum_{j''=1}^{J} \alpha_l\big(\mathbf{y}^{+(j,j')}, \mathbf{x}_\mathcal{S}^{(j'')}, \mathbf{x}_\mathcal{T}^{+(j'')}, \mathbf{u}^{+(\mathrm{r})}\big) \times \\ & \times\, \beta_l\big(\mathbf{y}^{+(j,j')}, \mathbf{x}_\mathcal{S}^{(j'')}, \mathbf{x}_\mathcal{T}^{+(j'')}, \mathbf{u}^{+(\mathrm{r})}\big)\,. \end{aligned} \tag{4.20}$$

Here, $\mathbf{y}^{+(j,j')}$, $\mathbf{x}_\mathcal{S}^{(j)}$, and $\mathbf{x}_\mathcal{T}^{+(j)}$ are particles of $\mathbf{y}^+$, $\mathbf{x}_\mathcal{S}$, and $\mathbf{x}_\mathcal{T}^+$, respectively that are drawn from the importance density [Doucet et al., 2001]

$$q(\mathbf{y}^+, \mathbf{x}_\mathcal{S}, \mathbf{x}_\mathcal{T}^+; \mathbf{u}^+ = \mathbf{u}^{+(\mathrm{r})}) \triangleq f(\mathbf{x}_\mathcal{S}, \mathbf{x}_\mathcal{T}^+)\, f(\mathbf{y}^+|\mathbf{x}_\mathcal{S}, \mathbf{x}_\mathcal{T}^+; \mathbf{u}^+ = \mathbf{u}^{+(\mathrm{r})})$$

via the following two-stage procedure. First, particles $\big\{\big(\mathbf{x}_\mathcal{S}^{(j)}, \mathbf{x}_\mathcal{T}^{+(j)}\big)\big\}_{j=1}^{J}$ are drawn from

$$f(\mathbf{x}_\mathcal{S}, \mathbf{x}_\mathcal{T}^+) \;=\; \prod_{l' \in \mathcal{S}} f(\mathbf{x}_{l'}) \prod_{m \in \mathcal{T}} f(\mathbf{x}_m^+)\,. \tag{4.21}$$

[1] With an abuse of notation, the superscript (j) now indicates the jth particle, whereas previously, in our full-blown notation, the subscript n indicated the nth time step.

(This factorization expresses the conditional statistical independence of the $\mathbf{x}_{l'}$, $l' \in \mathcal{S}$ and the $\mathbf{x}_m^+$, $m \in \mathcal{T}$ given $\mathbf{y}^{(1:n)}$. This is a common approximation, which was introduced in [Wymeersch et al., 2009] and is also used in the estimation layer presented in Chapters 2 and 3.) Then, for each particle $\big(\mathbf{x}_{\mathcal{S}}^{(j)}, \mathbf{x}_{\mathcal{T}}^{+(j)}\big)$, particles $\big\{\mathbf{y}^{+(j,j')}\big\}_{j'=1}^{J'}$ are drawn from the conditional PDF (cf. (4.16))

$$f\big(\mathbf{y}^+ \big| \mathbf{x}_{\mathcal{S}}^{(j)}, \mathbf{x}_{\mathcal{T}}^{+(j)}; \mathbf{u}^+ = \mathbf{u}^{+(\mathrm{r})}\big) = \alpha_l\big(\mathbf{y}^+, \mathbf{x}_{\mathcal{S}}^{(j)}, \mathbf{x}_{\mathcal{T}}^{+(j)}, \mathbf{u}^{+(\mathrm{r})}\big)\, \beta_l\big(\mathbf{y}^+, \mathbf{x}_{\mathcal{S}}^{(j)}, \mathbf{x}_{\mathcal{T}}^{+(j)}, \mathbf{u}^{+(\mathrm{r})}\big). \quad (4.22)$$

The distributed calculation of these particles will be discussed in Sections 4.4.1 and 4.4.1 and in Appendices C.1 and C.2. Finally, we note that using (4.17), one easily obtains a (rather unwieldy) expression of the derivative $\frac{\partial \alpha_l(\mathbf{y}^+, \mathbf{x}_{\mathcal{S}}, \mathbf{x}_{\mathcal{T}}^+, \mathbf{u}^+)}{\partial \mathbf{u}_l^+}$ occurring in (4.19). This expression involves the factors in (4.17) and the derivatives $\frac{\partial \tilde{g}_l(\mathbf{x}_l, \mathbf{u}_l^+)}{\partial \mathbf{u}_l^+}$, $\frac{\partial f(\mathbf{y}_{l,l'}^+ | \mathbf{x}_l^+, \mathbf{x}_{l'}^+)}{\partial \mathbf{x}_l^+}$ for $l' \in \mathcal{C}_l$, and $\frac{\partial f(\mathbf{y}_{l,m}^+ | \mathbf{x}_l^+, \mathbf{x}_m^+)}{\partial \mathbf{x}_l^+}$ for $m \in \mathcal{T}_l$. In the next two subsections, we present two alternative schemes for a *distributed*, particle-based computation of $\frac{\partial D_I(\mathbf{u}^+)}{\partial \mathbf{u}_l^+}\Big|_{\mathbf{u}^+ = \mathbf{u}^{+(\mathrm{r})}}$ according to (4.19) and (4.20). Both schemes are distributed in that they require only local communication, i.e., each CA $l \in \mathcal{S}$ transmits data only to its neighbors $l' \in \mathcal{C}_l$.

Flooding-Based Processing

We first discuss a distributed scheme where each CA $l \in \mathcal{S}$ performs an exact ("quasi-centralized") calculation of (4.19) and (4.20). As a result of the estimation layer, particles $\big\{\mathbf{x}_k^{(j)}\big\}_{j=1}^{J} \sim f(\mathbf{x}_k)$, $k \in \{l\} \cup \mathcal{T}$ are available at CA l (see Section 3.3.4, noting that $f(\mathbf{x}_k)$ was denoted $f(\mathbf{x}_k|\mathbf{y})$ there). A flooding algorithm [Lim and Kim, 2000] is now used to make available to each CA l the reference vectors $\mathbf{u}_{l'}^{+(\mathrm{r})}$ and the particles $\big\{\mathbf{x}_{l'}^{(j)}\big\}_{j=1}^{J} \sim f(\mathbf{x}_{l'})$ of all the other CAs $l' \in \mathcal{S} \backslash \{l\}$. In addition, CA l locally calculates predictive marginal posteriors for all target states via the following prediction step (which is (1.1) with n replaced by $n+1$):

$$f(\mathbf{x}_m^+) = \int f(\mathbf{x}_m^+ | \mathbf{x}_m)\, f(\mathbf{x}_m)\, \mathrm{d}\mathbf{x}_m\,, \quad m \in \mathcal{T}. \quad (4.23)$$

A particle-based implementation of (4.23) using the particles $\big\{\mathbf{x}_m^{(j)}\big\}_{j=1}^{J}$, $m \in \mathcal{T}$ produced by the estimation layer (which are available at CA l) and yielding particles $\big\{\mathbf{x}_m^{+(j)}\big\}_{j=1}^{J} \sim f(\mathbf{x}_m^+)$, $m \in \mathcal{T}$ is described in Chapters 2 and 3. Because all states $\mathbf{x}_k$ are conditionally independent given $\mathbf{y}^{(1:n)}$ (see (4.21)), particles $\big\{\big(\mathbf{x}_{\mathcal{S}}^{(j)}, \mathbf{x}_{\mathcal{T}}^{+(j)}\big)\big\}_{j=1}^{J} \sim f\big(\mathbf{x}_{\mathcal{S}}, \mathbf{x}_{\mathcal{T}}^+\big)$ can now be obtained by simple stacking operations. More specifically, the jth particle of $\mathbf{x}_{\mathcal{S}}$ is obtained by stacking the jth particles of the vectors $\mathbf{x}_{l'}$ for $l' \in \mathcal{S}$, i.e., $\mathbf{x}_{\mathcal{S}}^{(j)} = \big[\mathbf{x}_{l'}^{(j)}\big]_{l' \in \mathcal{S}}$, and similarly $\mathbf{x}_{\mathcal{T}}^{+(j)} = \big[\mathbf{x}_m^{+(j)}\big]_{m \in \mathcal{T}}$. Finally, for each $\big(\mathbf{x}_{\mathcal{S}}^{(j)}, \mathbf{x}_{\mathcal{T}}^{+(j)}\big)$, CA l computes particles

$$\big\{\mathbf{y}^{+(j,j')}\big\}_{j'=1}^{J'} \sim f\big(\mathbf{y}^+ \big| \mathbf{x}_{\mathcal{S}}^{(j)}, \mathbf{x}_{\mathcal{T}}^{+(j)}; \mathbf{u}^+ = \mathbf{u}^{+(\mathrm{r})}\big) \quad (4.24)$$

as described in Appendix C.1.

Using the particles $\big(\mathbf{x}_{\mathcal{S}}^{(j)}, \mathbf{x}_{\mathcal{T}}^{+(j)}\big)$ and $\mathbf{y}^{+(j,j')}$, $j = 1, \ldots, J$, $j' = 1, \ldots, J'$, as well as the reference vectors $\mathbf{u}_{l'}^{+(\mathrm{r})}$, $l' \in \mathcal{S}$, the gradient $\frac{\partial D_I(\mathbf{u}^+)}{\partial \mathbf{u}_l^+}\Big|_{\mathbf{u}^+ = \mathbf{u}^{+(\mathrm{r})}}$ can be computed locally at CA l according to (4.19) and (4.20). Note, however, that this flooding-based scheme presupposes that each CA l knows the state evolution models (1.1) and the measurement models (1.2) of all the other CAs $l' \in \mathcal{S} \backslash \{l\}$.

The communication cost of the flooding-based scheme, in terms of the number of real values transmitted by each CA, is $(JM + M_u)W \approx JMW$. Here, M and M_u are the dimensions of the $\mathbf{x}_l$ and the $\mathbf{u}_l$, respectively, and W depends on the network size and topology and is bounded as $1 \leq W \leq |\mathcal{S}|$. Thus, the number of transmissions scales linearly with J and does not depend on J'. In large networks, flooding protocols tend to require a large memory and book-keeping overhead and introduce a significant delay [Xiao et al., 2005]. If the network formed by the CAs is *fully* connected, i.e., $\mathcal{S} = \{l\} \cup \mathcal{C}_l$, then all the particles $\big\{\big(\mathbf{x}_{\mathcal{S}}^{(j)}, \mathbf{x}_{\mathcal{T}}^{+(j)}\big)\big\}_{j=1}^{J}$ can be obtained without flooding: CA l simply broadcasts its reference vector $\mathbf{u}_l^{+(\mathrm{r})}$ and its particles $\big\{\mathbf{x}_l^{(j)}\big\}_{j=1}^{J} \sim f(\mathbf{x}_l)$ to all the other CAs in the network and receives their reference vectors and particles. Here, the number of real values transmitted by each CA is only $JM + M_u$.

Finally, the computational complexity of the flooding-based scheme–i.e., evaluation of (4.19) and (4.20), with J and J' fixed—scales linearly with the number of agents in the network. Because the computational complexity and the communication cost increase with the network size, the flooding-based distributed processing scheme is primarily suited to small networks.

Consensus-Based Processing

A truly distributed computation of (4.19) and (4.20) that avoids the use of flooding protocol 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 particles $\big\{\mathbf{x}_l^{(j)}\big\}_{j=1}^{J} \sim f(\mathbf{x}_l)$ calculated in the estimation layer to all neighboring CAs $l' \in \mathcal{C}_l$, and it receives particles $\big\{\mathbf{x}_{l'}^{(j)}\big\}_{j=1}^{J} \sim f(\mathbf{x}_{l'})$ from all neighbors $l' \in \mathcal{C}_l$. In addition, CA l locally calculates particles $\big\{\mathbf{x}_m^{+(j)}\big\}_{j=1}^{J} \sim f(\mathbf{x}_m^+)$ for all $m \in \mathcal{T}_l$ via the prediction step (4.23) (with $\mathcal{T}$ replaced by $\mathcal{T}_l$), using the particle-based implementation described in Chapters 2 and 3. Thus, after the stacking operations $\mathbf{x}_{\mathcal{C}_l}^{(j)} = \big[\mathbf{x}_{l'}^{(j)}\big]_{l' \in \mathcal{C}_l}$ and $\mathbf{x}_{\mathcal{T}_l}^{+(j)} = \big[\mathbf{x}_m^{+(j)}\big]_{m \in \mathcal{T}_l}$, particles $\big\{\big(\mathbf{x}_l^{(j)}, \mathbf{x}_{\mathcal{C}_l}^{(j)}, \mathbf{x}_{\mathcal{T}_l}^{+(j)}\big)\big\}_{j=1}^{J} \sim f(\mathbf{x}_l, \mathbf{x}_{\mathcal{C}_l}, \mathbf{x}_{\mathcal{T}_l}^+)$ are available at CA l.

The key question for a distributed computation of (4.19) and (4.20) now is as to whether the quantities $\alpha_l\big(\mathbf{y}^{+(j,j')}, \mathbf{x}_{\mathcal{S}}^{(j'')}, \mathbf{x}_{\mathcal{T}}^{+(j'')}, \mathbf{u}^{+(r)}\big)$ and $\beta_l\big(\mathbf{y}^{+(j,j')}, \mathbf{x}_{\mathcal{S}}^{(j'')}, \mathbf{x}_{\mathcal{T}}^{+(j'')}, \mathbf{u}^{+(r)}\big)$ are locally available at CA l. The factors of $\alpha_l(\mathbf{y}^+, \mathbf{x}_{\mathcal{S}}, \mathbf{x}_{\mathcal{T}}^+, \mathbf{u}^+)$ (see (4.17)) 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 (4.17) that $\alpha_l(\mathbf{y}^+, \mathbf{x}_{\mathcal{S}}, \mathbf{x}_{\mathcal{T}}^+, \mathbf{u}^+) = f\big(\tilde{\mathbf{y}}_l^+ \big| \mathbf{x}_l, \mathbf{x}_{\mathcal{C}_l}, \mathbf{x}_{\mathcal{T}_l}^+; \mathbf{u}_{\mathcal{C}_l}^+\big)$, where $\mathbf{u}_{\mathcal{C}_l}^+ \triangleq \big[\mathbf{u}_{l'}^+\big]_{l' \in \mathcal{C}_l}$ and

$$\tilde{\mathbf{y}}_l^+ \triangleq \big[\mathbf{y}_{l,k}^+\big]_{k \in \mathcal{A}_l} = \big[d_l(\mathbf{x}_l^+, \mathbf{x}_k^+, \mathbf{v}_{l,k}^+)\big]_{k \in \mathcal{A}_l}. \tag{4.25}$$

Note that $\tilde{\mathbf{y}}_l^+$ comprises the measurements acquired by CA l at the next time (except that

$\mathcal{A}_l^+$ is replaced with $\mathcal{A}_l$ since it is not known at the current time n). Thus, we conclude that $\alpha_l(\mathbf{y}^+, \mathbf{x}_\mathcal{S}, \mathbf{x}_\mathcal{T}^+, \mathbf{u}^+)$ is available at CA l. On the other hand, the factors of $\beta_l(\mathbf{y}^+, \mathbf{x}_\mathcal{S}, \mathbf{x}_\mathcal{T}^+, \mathbf{u}^+)$ (see (4.18)) 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, $\beta_l(\mathbf{y}^+, \mathbf{x}_\mathcal{S}, \mathbf{x}_\mathcal{T}^+, \mathbf{u}^+)$ is *not* available at CA l.

We will now present a distributed computation of $\beta_l\big(\mathbf{y}^{+(j,j')}, \mathbf{x}_\mathcal{S}^{(j'')}, \mathbf{x}_\mathcal{T}^{+(j'')}, \mathbf{u}^{+(\mathrm{r})}\big)$. Using (4.18), one can show that

$$\beta_l\big(\mathbf{y}^{+(j,j')}, \mathbf{x}_\mathcal{S}^{(j'')}, \mathbf{x}_\mathcal{T}^{+(j'')}, \mathbf{u}^{+(\mathrm{r})}\big) \;=\; \exp\big(|\mathcal{S}|\, F_{j,j',j''} - F_{j,j',j''}^{(l)}\big)\,, \tag{4.26}$$

with

$$F_{j,j',j''}^{(l)} \;\triangleq\; \log f\big(\tilde{\mathbf{y}}_l^{+(j,j')} \big| \mathbf{x}_l^{(j'')}, \mathbf{x}_{\mathcal{C}_l}^{(j'')}, \mathbf{x}_{\mathcal{T}_l}^{+(j'')}; \mathbf{u}_{\mathcal{C}_l}^{+} = \mathbf{u}_{\mathcal{C}_l}^{+(\mathrm{r})}\big) \tag{4.27}$$

and

$$F_{j,j',j''} \;\triangleq\; \frac{1}{|\mathcal{S}|} \sum_{l \in \mathcal{C}} F_{j,j',j''}^{(l)}\,, \tag{4.28}$$

for $j = 1, \ldots, J$, $j' = 1, \ldots, J'$, and $j'' = 1, \ldots, J$. To compute $F_{j,j',j''}^{(l)}$ in (4.27), CA l only needs particles

$$\big\{\tilde{\mathbf{y}}_l^{+(j,j')}\big\}_{j'=1}^{J'} \sim f\big(\tilde{\mathbf{y}}_l^{+} \big| \mathbf{x}_l^{(j)}, \mathbf{x}_{\mathcal{C}_l}^{(j)}, \mathbf{x}_{\mathcal{T}_l}^{+(j)}; \mathbf{u}_{\mathcal{C}_l}^{+} = \mathbf{u}_{\mathcal{C}_l}^{+(\mathrm{r})}\big)$$

and the reference vectors $\mathbf{u}_{l'}^{+(\mathrm{r})}$ for $l' \in \mathcal{C}_l$. The computation of the particles $\big\{\tilde{\mathbf{y}}_l^{+(j,j')}\big\}_{j'=1}^{J'}$ is described in Appendix C.2. The $\mathbf{u}_{l'}^{+(\mathrm{r})}$ can be obtained at CA l through communication with the neighbor CAs $l' \in \mathcal{C}_l$.

Once the $F_{j,j',j''}^{(l)}$ have been calculated at CA l, their averages $F_{j,j',j''}$ in (4.28) can be computed in a distributed way by using J^2J' parallel instances of an average consensus or gossip scheme [Olfati-Saber et al., 2007, Dimakis et al., 2010]. These schemes are iterative; they are are initialized at each CA l with $F_{j,j',j''}^{(l)}$. They are robust to communication link failures [Olfati-Saber et al., 2007, Dimakis et al., 2010] and use only communication between neighboring CAs (i.e., each CA $l \in \mathcal{S}$ transmits data to each neighbor $l' \in \mathcal{C}_l$). After convergence of the consensus or gossip scheme, $F_{j,j',j''}$ and, hence, $\beta_l\big(\mathbf{y}^{+(j,j')}, \mathbf{x}_\mathcal{S}^{(j'')}, \mathbf{x}_\mathcal{T}^{+(j'')}, \mathbf{u}^{+(\mathrm{r})}\big)$ for all j, j', j'' is available at each CA l.

At this point, CA l has available all the information required to evaluate $\alpha_l\big(\mathbf{y}^{+(j,j')}, \mathbf{x}_\mathcal{S}^{(j'')}, \mathbf{x}_\mathcal{T}^{+(j'')}, \mathbf{u}^{+(\mathrm{r})}\big)$ and $\frac{\partial \alpha_l(\mathbf{y}^{+(j,j')}, \mathbf{x}_\mathcal{S}^{(j)}, \mathbf{x}_\mathcal{T}^{+(j)}, \mathbf{u}^+)}{\partial \mathbf{u}_l^+}\Big|_{\mathbf{u}^+ = \mathbf{u}^{+(\mathrm{r})}}$ and an approximation of $\beta_l\big(\mathbf{y}^{+(j,j')}, \mathbf{x}_\mathcal{S}^{(j'')}, \mathbf{x}_\mathcal{T}^{+(j'')}, \mathbf{u}^{+(\mathrm{r})}\big)$ 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 (4.19) and (4.20). In the course of the overall distributed computation, CA l transmits $J^2J'|\mathcal{C}_l|R + JM + M_u \approx J^2J'|\mathcal{C}_l|R$ real values, where R is the number of iterations used for one instance of the consensus or gossip scheme. Asymptotically, for $R \to \infty$, this distributed computation of $\frac{\partial D_I(\mathbf{u}^+)}{\partial \mathbf{u}_l^+}\Big|_{\mathbf{u}^+ = \mathbf{u}^{+(\mathrm{r})}}$ converges to the exact centralized result given by (4.19) and (4.20). The speed of convergence depends on the topology and size of the network [Olfati-Saber et al., 2007, Dimakis et al., 2010]. 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 from 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 4.4.1 unless the network is large and R is small.

Finally, the computational complexity of the consensus-based processing—i.e., evaluation of (4.19) and (4.20), with J, J', and R fixed—is constant in the number of agents in the network.

4.4.2 Gradient of $G_l(\mathbf{u}_l^+)$

Next, we consider the second gradient in the expansion (4.13), i.e., using (4.8),

$$\frac{\partial G_l(\mathbf{u}_l^+)}{\partial \mathbf{u}_l^+}\bigg|_{\mathbf{u}_l^+=\mathbf{u}_l^{+(\mathrm{r})}} = \frac{\partial}{\partial \mathbf{u}_l^+}\left(\int f(\mathbf{x}_l)\,\log|J_{\tilde{g}_l}(\mathbf{x}_l;\mathbf{u}_l^+)|\,\mathrm{d}\mathbf{x}_l\right)\bigg|_{\mathbf{u}_l^+=\mathbf{u}_l^{+(\mathrm{r})}}.$$

We assume that for each value of $\mathbf{x}_l$, $J_{\tilde{g}_l}(\mathbf{x}_l;\mathbf{u}_l^+)$ is differentiable with respect to $\mathbf{u}_l^+$ at $\mathbf{u}_l^{+(\mathrm{r})}$ and nonzero for all $\mathbf{u}_l^+$ in some (arbitrarily small) neighborhood of $\mathbf{u}_l^{+(\mathrm{r})}$; note that this implies that $J_{\tilde{g}_l}(\mathbf{x}_l;\mathbf{u}_l^{+(\mathrm{r})}) \neq 0$ and that also $|J_{\tilde{g}_l}(\mathbf{x}_l;\mathbf{u}_l^+)|$ is differentiable with respect to $\mathbf{u}_l^+$ at $\mathbf{u}_l^{+(\mathrm{r})}$. We then have

$$\begin{aligned}\frac{\partial G_l(\mathbf{u}_l^+)}{\partial \mathbf{u}_l^+}\bigg|_{\mathbf{u}_l^+=\mathbf{u}_l^{+(\mathrm{r})}} &= \int f(\mathbf{x}_l)\,\frac{\partial \log|J_{\tilde{g}_l}(\mathbf{x}_l;\mathbf{u}_l^+)|}{\partial \mathbf{u}_l^+}\bigg|_{\mathbf{u}_l^+=\mathbf{u}_l^{+(\mathrm{r})}}\mathrm{d}\mathbf{x}_l \\ &= \int f(\mathbf{x}_l)\,\frac{1}{|J_{\tilde{g}_l}(\mathbf{x}_l;\mathbf{u}_l^{+(\mathrm{r})})|}\,\frac{\partial |J_{\tilde{g}_l}(\mathbf{x}_l;\mathbf{u}_l^+)|}{\partial \mathbf{u}_l^+}\bigg|_{\mathbf{u}_l^+=\mathbf{u}_l^{+(\mathrm{r})}}\mathrm{d}\mathbf{x}_l\,. \end{aligned} \tag{4.29}$$

Based on the particles $\{\mathbf{x}_l^{(j)}\}_{j=1}^J \sim f(\mathbf{x}_l)$, which were calculated in the estimation layer, a Monte Carlo approximation of (4.29) is obtained as

$$\frac{\partial G_l(\mathbf{u}_l^+)}{\partial \mathbf{u}_l^+}\bigg|_{\mathbf{u}_l^+=\mathbf{u}_l^{+(\mathrm{r})}} \approx \frac{1}{J}\sum_{j=1}^{J}\frac{1}{\big|J_{\tilde{g}_l}\big(\mathbf{x}_l^{(j)};\mathbf{u}_l^{+(\mathrm{r})}\big)\big|}\,\frac{\partial\big|J_{\tilde{g}_l}\big(\mathbf{x}_l^{(j)};\mathbf{u}_l^+\big)\big|}{\partial \mathbf{u}_l^+}\bigg|_{\mathbf{u}_l^+=\mathbf{u}_l^{+(\mathrm{r})}}. \tag{4.30}$$

For many practically relevant state evolution models (4.6), the computation of $\frac{\partial G_l(\mathbf{u}_l^+)}{\partial \mathbf{u}_l^+}\Big|_{\mathbf{u}_l^+=\mathbf{u}_l^{+(\mathrm{r})}}$ can be avoided altogether or $\frac{\partial G_l(\mathbf{u}_l^+)}{\partial \mathbf{u}_l^+}\Big|_{\mathbf{u}_l^+=\mathbf{u}_l^{+(\mathrm{r})}}$ can be calculated in closed form, without a particle-based approximation. Some examples are considered in the following.

1. $J_{\tilde{g}_l}(\mathbf{x}_l;\mathbf{u}_l^+)$ *does not depend on* $\mathbf{u}_l^+$: In this case, $\frac{\partial G_l(\mathbf{u}_l^+)}{\partial \mathbf{u}_l^+} = \mathbf{0}$. An important example is the "linear additive" state evolution model (cf. (4.6))

$$\tilde{g}_l(\mathbf{x}_l,\mathbf{u}_l^+) = \mathbf{A}\mathbf{x}_l + \zeta(\mathbf{u}_l^+) \tag{4.31}$$

with an arbitrary matrix $\mathbf{A}$ and function $\zeta(\cdot)$ of suitable dimensions. Here, we obtain $J_{\tilde{g}_l}(\mathbf{x}_l;\mathbf{u}_l^+) = \det\mathbf{A}$ and thus $\frac{\partial G_l(\mathbf{u}_l^+)}{\partial \mathbf{u}_l^+} = \mathbf{0}$. A second important example is the *odometry*

motion model [Thrun et al., 2005, Section 5.3]. Here, the local state $\mathbf{x}_l$ is the pose of a robot, which consists of the 2D position $(x_{l,1}, x_{l,2})$ and the orientation θ_l, and the control vector $\mathbf{u}_l$ consists of the translational velocity ν_l and the rotational velocity ω_l. The state evolution model is given by

$$\tilde{g}_l(\mathbf{x}_l, \mathbf{u}_l^+) = \begin{bmatrix} x_{l,1} + \nu_l^+ \cos(\theta_l + \omega_l^+) \\ x_{l,2} + \nu_l^+ \sin(\theta_l + \omega_l^+) \\ \theta_l + \omega_l^+ \end{bmatrix}.$$

Here, $J_{\tilde{g}_l}(\mathbf{x}_l; \mathbf{u}_l^+) = 1$ and thus $\frac{\partial G_l(\mathbf{u}_l^+)}{\partial \mathbf{u}_l^+} = \mathbf{0}$.

2. $J_{\tilde{g}_l}(\mathbf{x}_l; \mathbf{u}_l^+)$ *does not depend on* $\mathbf{x}_l$: If $J_{\tilde{g}_l}(\mathbf{x}_l; \mathbf{u}_l^+) = J_{\tilde{g}_l}(\mathbf{u}_l^+)$, then (4.8) simplifies to $G_l(\mathbf{u}_l^+) = \log |J_{\tilde{g}_l}(\mathbf{u}_l^+)|$. Thus, we have

$$\frac{\partial G_l(\mathbf{u}_l^+)}{\partial \mathbf{u}_l^+} = \frac{1}{|J_{\tilde{g}_l}(\mathbf{u}_l^+)|} \frac{\partial |J_{\tilde{g}_l}(\mathbf{u}_l^+)|}{\partial \mathbf{u}_l^+},$$

which can be calculated in closed form.

4.4.3 Summary of the Control Layer

The operations performed in the control layer are summarized in Fig. 4.2 for the flooding-based method discussed in Section 4.4.1 and in Fig. 4.3 for the consensus-based discussed in Section 4.4.1.

4.5 Special Cases

Next, we discuss two special cases: cooperative navigation (i.e., there are no targets) and distributed tracking (i.e., the CA positions are known).

4.5.1 Cooperative Navigation with Information-Seeking Control

Here, we assume that there are no targets, and thus the task considered is cooperative navigation. The estimation layer for this task is described in Chapter 2.

Control Layer

Since there are no targets, the component $D_I(\mathbf{u}^+) = I(\mathbf{x}_\mathcal{S}, \mathbf{x}_\mathcal{T}^+; \mathbf{y}^+; \mathbf{u}^+)$ of the objective function in (4.12) simplifies to $D_I(\mathbf{u}^+) = I(\mathbf{x}_\mathcal{S}; \mathbf{y}^+; \mathbf{u}^+)$. The expression of the gradient of $D_I(\mathbf{u}^+)$ in (4.19) and (4.20) here simplifies as well $\alpha_l(\mathbf{y}^+, \mathbf{x}_\mathcal{S}, \mathbf{x}_\mathcal{T}^+, \mathbf{u}^+) = \alpha_l(\mathbf{y}^+, \mathbf{x}_\mathcal{S}, \mathbf{u}^+)$ and $\beta_l(\mathbf{y}^+, \mathbf{x}_\mathcal{S}, \mathbf{x}_\mathcal{T}^+, \mathbf{u}^+) = \beta_l(\mathbf{y}^+, \mathbf{x}_\mathcal{S}, \mathbf{u}^+)$ (according to (4.17) and (4.18), since $\mathcal{T} = \emptyset$); furthermore, sampling from $f(\mathbf{x}_\mathcal{S}, \mathbf{x}_\mathcal{T}^+)$ (see Sections 4.4.1 and 4.4.1) reduces to sampling from $f(\mathbf{x}_\mathcal{S})$.

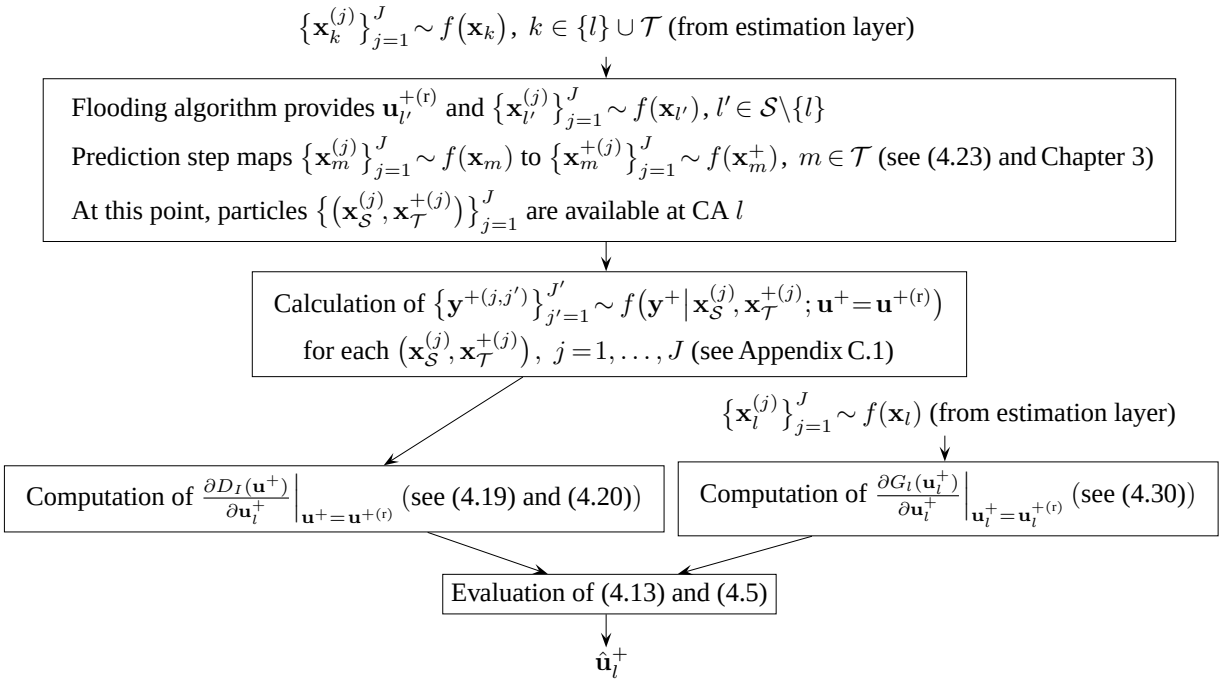


Figure 4.2: Flow chart of the flooding-based implementation of the control layer at CA l (see Section 4.4.1).

4.5.2 Distributed Target Tracking with Information-Seeking Control

Next, we discuss the case where the positions of the CAs are perfectly known, and thus our task is only the distributed tracking. The estimation layer for this task is reviewed in Section 3.1.

Control Layer

Since there are no unknown CA states, the objective function in (4.12) simplifies in that $D_I(\mathbf{u}^+) = I(\mathbf{x}_{\mathcal{T}}^+; \mathbf{y}^+; \mathbf{u}^+)$ and $G_l(\mathbf{u}_l^+) = 0$. The expression of the gradient of $D_I(\mathbf{u}^+)$ in (4.19) and (4.20) simplifies because $\alpha_l(\mathbf{y}^+, \mathbf{x}_{\mathcal{S}}, \mathbf{x}_{\mathcal{T}}^+, \mathbf{u}^+) = \alpha_l(\mathbf{y}^+, \mathbf{x}_{\mathcal{T}}^+, \mathbf{u}^+)$ and $\beta_l(\mathbf{y}^+, \mathbf{x}_{\mathcal{S}}, \mathbf{x}_{\mathcal{T}}^+, \mathbf{u}^+) = \beta_l(\mathbf{y}^+, \mathbf{x}_{\mathcal{T}}^+, \mathbf{u}^+)$; furthermore, sampling from $f(\mathbf{x}_{\mathcal{S}}, \mathbf{x}_{\mathcal{T}}^+)$ reduces to sampling from $f(\mathbf{x}_{\mathcal{T}}^+)$.

This special case was previously considered in [Julian et al., 2012]. More specifically, [Julian et al., 2012] studied estimation of one static global state and proposed a distributed, gradient-based, information-seeking controller and a particle-based implementation. Our work extends [Julian et al., 2012] to cooperative navigation and to tracking of a time-varying.

4.6 Simulation Results

We demonstrate the performance of the proposed method for three different scenarios. In Section 4.6.2, we study the behavior of the controller by considering noncooperative navigation of four mobile CAs based on distance measurements relative to an anchor. In Section 4.6.3, we consider cooperative navigation of three mobile CAs. Finally, in Section 4.6.4, two mobile CAs

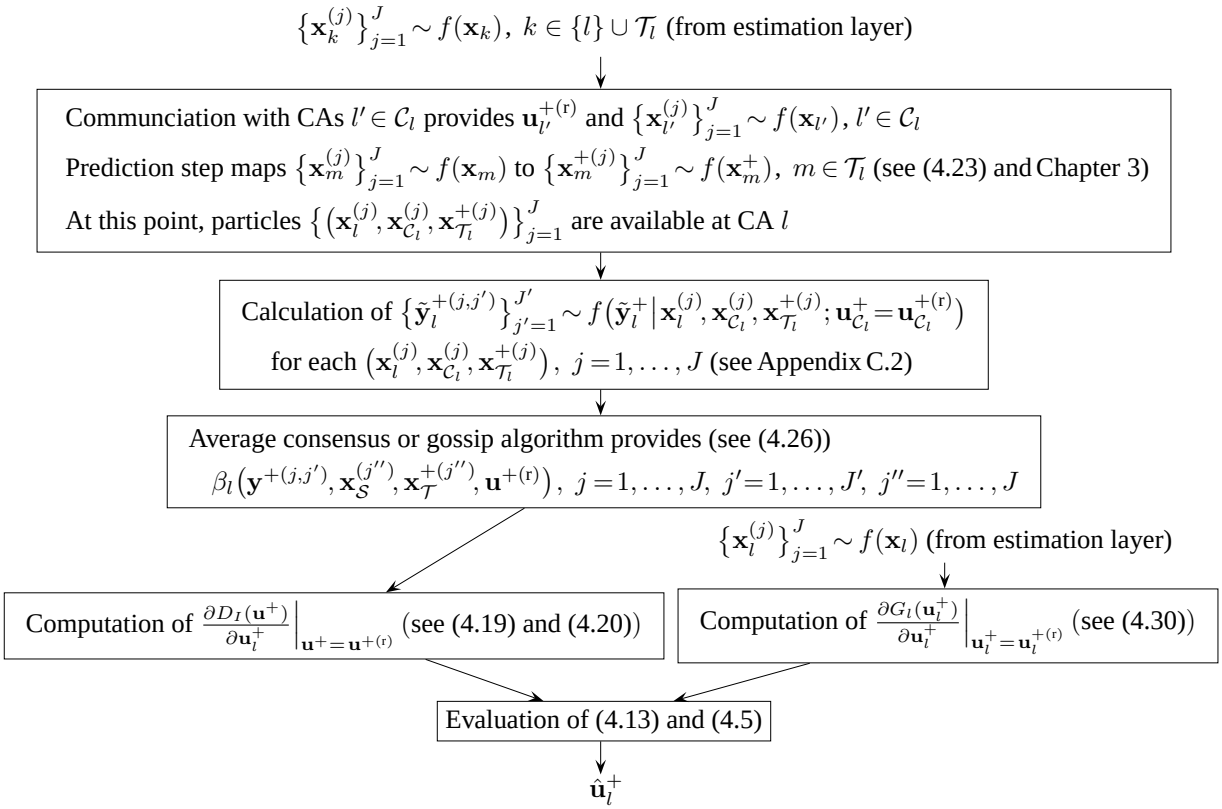


Figure 4.3: Flow chart of the consensus-based implementation of the control layer at CA l (see Section 4.4.1).

perform cooperative simultaneous navigation 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/.

4.6.1 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., $\mathbf{x}_{l,n} \triangleq \left[x_{l,n,1}, x_{l,n,2}\right]^{\mathrm{T}}$ in a global reference frame. 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 [Li and Jilkov, 2003]

$$\mathbf{x}_{l,n} = \mathbf{x}_{l,n-1} + \mathbf{u}_{l,n} + \mathbf{q}_{l,n}, \quad n = 1, 2, \ldots . \tag{4.32}$$

Here, $\mathbf{q}_{l,n} \in \mathbb{R}^2$ is zero-mean Gaussian with independent and identically distributed entries, i.e., $\mathbf{q}_{l,n} \sim \mathcal{N}(\mathbf{0}, \sigma_q^2 \mathbf{I})$ with $\sigma_q^2 = 10^{-3}$, and $\mathbf{q}_{l,n}$ and $\mathbf{q}_{l',n'}$ are independent unless $(l,n) = (l',n')$. The domain $\mathcal{U}_l$ of the control vector $\mathbf{u}_{l,n}$ is defined by the norm constraint $\left\|\mathbf{u}_{l,n}\right\| \leq u_l^{\max}$. For the interpretation of $\mathbf{u}_{l,n}$ within (4.32), it is assumed that the CAs know the orientation of the global reference frame. 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 (1.3), i.e., $y_{l,k;n} = \big\|\mathbf{x}_{l,n} - \mathbf{x}_{k,n}\big\| + v_{l,k;n}$, where the measurement noise $v_{l,k;n}$ is independent across l, k, and n and Gaussian with variance

$$\sigma^2_{l,k;n} = \begin{cases} \sigma_0^2\,, & \big\|\mathbf{x}_{l,n} - \mathbf{x}_{k,n}\big\| \le d_0 \\ \sigma_0^2\Big[\Big(\frac{\|\mathbf{x}_{l,n} - \mathbf{x}_{k,n}\|}{d_0} - 1\Big)^{\kappa} + 1\Big]\,, & \big\|\mathbf{x}_{l,n} - \mathbf{x}_{k,n}\big\| > d_0\,. \end{cases} \tag{4.33}$$

That is, the measurement noise variance stays constant up to some distance d_0 and then increases polynomially with some exponent κ. This is a simple model for time-of-arrival distance measurements [Garcia et al., 2014]. We set $\sigma_0^2 = 50$ and $\kappa = 2$ and, if not stated otherwise, $d_0 = 50$.

In the estimation layer, we use $J = 3600$ particles. (Choosing J below 3000 was observed in some rare cases to lead to a convergence to the wrong estimate.) A resampling step is performed to avoid weight degeneracy [Douc and Cappe, 2005]. Resampling transforms weighted particles $\big\{\big(\mathbf{x}^{(j)}_{k,n}, w^{(j)}_{k,n}\big)\big\}_{j=1}^{J}$ representing the belief $b\big(\mathbf{x}_{k,n}\big)$ into nonweighted particles $\big\{\mathbf{x}^{(j)}_{k,n}\big\}_{j=1}^{J}$. We use a somewhat nonorthodox type of resampling that helps move particles to positions with high probability mass, thereby reducing the number of particles needed. More specifically, at every Lth time step n, we particle from a kernel approximation of the belief; at all other time steps, we perform standard systematic resampling [Douc and Cappe, 2005]. The kernel approximation of the belief $b\big(\mathbf{x}_{k,n}\big)$ is obtained as [Silverman and Green, 1986]

$$\tilde{b}\big(\mathbf{x}_{k,n}\big) = \sum_{j=1}^{J} w^{(j)}_{k,n}\, K\big(\mathbf{x}_{k,n} - \mathbf{x}^{(j)}_{k,n}\big)\,,$$

with the Gaussian kernel

$$K(\mathbf{x}) = (2\pi\sigma_K^2)^{-1} \exp\big(-\|\mathbf{x}\|^2/(2\sigma_K^2)\big). \tag{4.34}$$

Here, the variance σ_K^2 is chosen as

$$\sigma_K^2 = \begin{cases} J^{1/3}\,\mathrm{tr}\big(\mathbf{C}_{k,n}\big)/2, & \text{if } \mathrm{tr}\big(\mathbf{C}_{k,n}\big) < 2\sigma_0^2 \\ \sigma_K^2 = \sigma_0^2, & \text{otherwise,} \end{cases}$$

where $\mathrm{tr}\big(\mathbf{C}_{k,n}\big)$ denotes the trace of the weighted particle covariance matrix defined as

$$\mathbf{C}_{k,n} = \sum_{j=1}^{J} w^{(j)}_{k,n}\mathbf{x}^{(j)}_{k,n}\mathbf{x}^{(j)\mathrm{T}}_{k,n} - \boldsymbol{\mu}_{k,n}\boldsymbol{\mu}^{\mathrm{T}}_{k,n} \quad \text{where } \boldsymbol{\mu}_{k,n} = \sum_{j=1}^{J} w^{(j)}_{k,n}\mathbf{x}^{(j)}_{k,n}\,.$$

This case distinction in the choice of σ_K^2 is used since $\sigma_K^2 = J^{1/3}\,\mathrm{tr}\big(\mathbf{C}_{k,n}\big)/2$ is only accurate for a unimodal distribution [Silverman and Green, 1986] whereas $\sigma_K^2 = \sigma_0^2$ is suitable for annularly shaped distributions (here, the width of the annulus is determined by σ_0^2 [Ihler et al., 2005]). We choose $L = 40$ if $\mathrm{tr}\big(\mathbf{C}_{k,n}\big) < 80$, $L = 20$ if $80 \le \mathrm{tr}\big(\mathbf{C}_{k,n}\big) < 1000$, and $L = 10$ if

$\mathrm{tr}\big(\mathbf{C}_{k,n}\big) \geq 1000$; this choice led to good results in our simulation setting.

We employ a censoring scheme [Lien et al., 2012] 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 $\mathrm{tr}\big(\mathbf{C}_{l,n}\big) < 10$ are used as navigation 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., $\mathrm{tr}\big(\mathbf{C}_{l,n}\big) \geq 10$), its objective function is $\tilde{D}_h\big(\mathbf{u}_{n+1}\big) \triangleq -h\big(\mathbf{x}_{l,n+1}\big|\mathsf{y}_{1,n+1}; y_{1,1:n)}, \mathbf{u}_{l,1:n+1}\big)$, i.e., the negative differential entropy of only the own state conditioned on only the own measurement relative to the anchor CA, $\mathsf{y}_{1,n+1}$.

The gradient ascent in the controller (see (4.5)) uses the reference points $\mathbf{u}_{l,n}^{(\mathrm{r})} = \mathbf{0}$, which are consistent with the state evolution model (4.32), and step sizes $c_{l,n}$ chosen such that $\big\|\mathbf{u}_{l,n}\big\| = u_l^{\max}$. Thus, the controller moves every mobile CA $l \in \mathcal{S}$ with maximum nominal speed (determined by $u_l^{\max}$) in the direction of maximum local increase of the objective function. The number of particles used in the control layer is $JJ' = 60000$, with $J = 1200$ and $J' = 50$. A reduction of J' to $J' = 1$ was observed to result in more jagged CA trajectories and a slightly slower reduction of the estimation error over time.

4.6.2 Noncooperative Navigation

In order to study the behavior of the controller, we consider four mobile CAs $l = 2, 3, 4, 5$ that perform navigation 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 $[0, 0]^{\mathrm{T}}$, 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 $[100, 0]^{\mathrm{T}}$ and move with identical nominal speed determined by $u_l^{\max} = 1$. The objective function for the control of CAs 2, 3, and 4 is $\tilde{D}_h\big(\mathbf{u}_{l,n+1}\big) \triangleq -h\big(\mathbf{x}_{l,n+1}\big|\mathsf{y}_{1,n+1)}; y_{1,1:n}, \mathbf{u}_{l,1:n+1}\big)$. 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 $u_l^{\max} = 1$. Fig. 4.4 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. 4.5 shows the position ARMSEs of the four mobile CAs. These ARMSEs 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

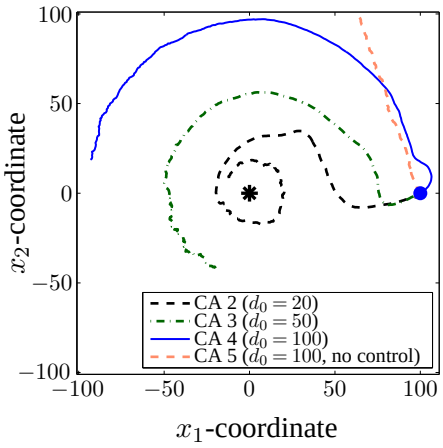


Figure 4.4: Example trajectories for noncooperative navigation with information-seeking control. The initial CA position and the anchor position are indicated by a bullet and a star, respectively.

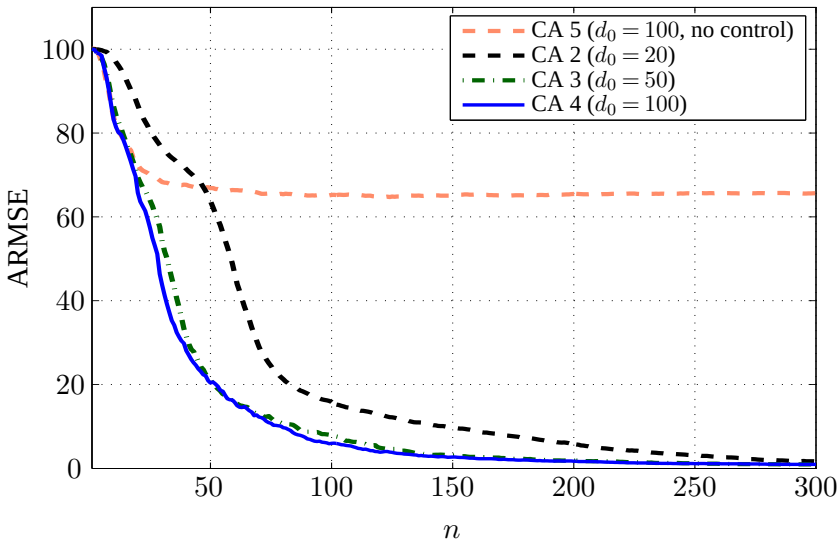


Figure 4.5: Position ARMSE for noncooperative navigation with information-seeking control.

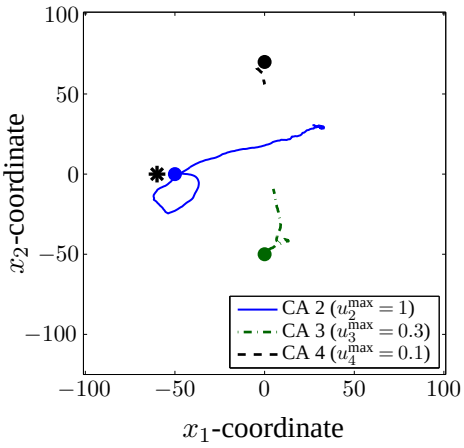


Figure 4.6: Example trajectories for cooperative navigation with information-seeking control (C–C scheme). The initial CA positions and the anchor position are indicated by bullets and a star, respectively.

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 (4.33)). 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.

4.6.3 Cooperative Navigation

Next, we study the proposed method for cooperative navigation with information-seeking control (abbreviated as C–C). There are three mobile CAs $l = 2, 3, 4$ with different start points ($[-50, 0]^{\mathrm{T}}$, $[0, -50]^{\mathrm{T}}$, and $[0, 70]^{\mathrm{T}}$ for $l = 2$, 3, and 4, respectively) and different nominal speeds ($u_l^{\max} = 1$, 0.3, and 0.1 for $l = 2$, 3, and 4, respectively). The mobile CAs measure their distances to a static anchor $l = 1$ located at $[-60, 0]^{\mathrm{T}}$ and to each other, using $d_0 = 50$. Example trajectories are shown in Fig. 4.6. For comparison, we also consider noncooperative navigation with information-seeking control as studied in Section 4.6.2 (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 $u_l^{\max}$.

Fig. 4.7 shows the position ARMSEs of the three schemes, which were determined by averaging over the three mobile CAs and over 300 simulation runs. It is seen that the ARMSEs of the two reference schemes N–C and C–N decrease only very slowly whereas, after about 70 time steps, the ARMSE 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

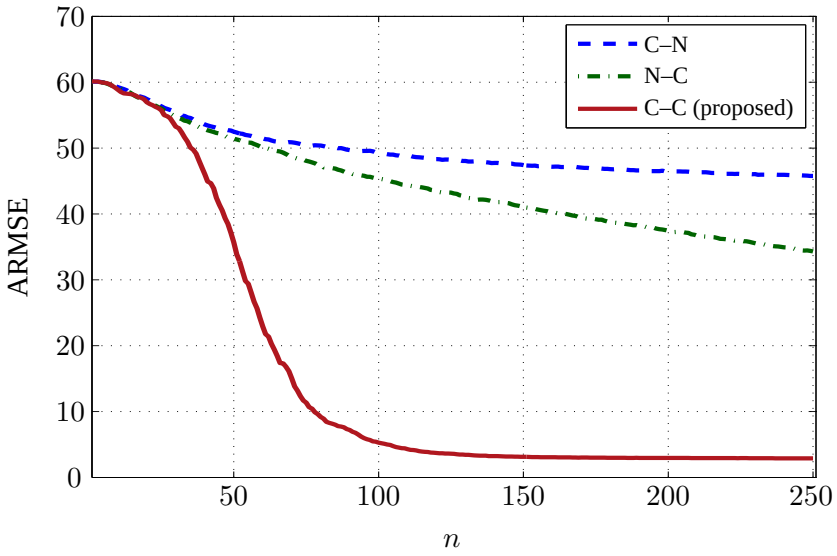


Figure 4.7: Navigation ARMSE of three different estimation/control methods.

such a way that it supports the navigation of the two other CAs. In fact, as shown by Fig. 4.6, 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.

4.6.4 CoSNAT

Finally, we consider CoSNAT. Two mobile CAs $l = 2, 3$ starting at position $[20, 20]^{\mathrm{T}}$ and $[-10, -10]^{\mathrm{T}}$, respectively and with nominal speed determined by $u_l^{\max} = 1$ cooperatively localize and track themselves and a mobile target. There is also a static anchor $l = 1$ at position $[-50, 0]^{\mathrm{T}}$. The target state $\mathbf{x}_{m,n} = \mathbf{x}_{4,n}$ consists of position and velocity, i.e., $\mathbf{x}_{4,n} \triangleq \left[x_{4,n,1}, x_{4,n,2}, \dot{x}_{4,n,1}, \dot{x}_{4,n,2}\right]^{\mathrm{T}}$. The target state evolves according to [Li and Jilkov, 2003]

$$\mathbf{x}_{4,n} = \mathbf{G}\mathbf{x}_{4,n-1} + \mathbf{W}\mathbf{q}_{4,n}, \qquad n = 1, 2, \ldots,$$

where

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

and $\mathbf{q}_{4,n} \in \mathbb{R}^2$ is zero-mean Gaussian with independent and identically distributed entries, i.e., $\mathbf{q}_{4,n} \sim \mathcal{N}(\mathbf{0}, \tilde{\sigma}_q^2 \mathbf{I})$ with $\tilde{\sigma}_q^2 = 10^{-5}$, and with $\mathbf{q}_{4,n}$ and $\mathbf{q}_{4,n'}$ independent unless $n = n'$. The target trajectory is initialized with position $\left[x_{4,0,1}, x_{4,0,2}\right]^{\mathrm{T}} = [50, 0]^{\mathrm{T}}$ and velocity $\left[\dot{x}_{4,0,1}, \dot{x}_{4,0,2}\right]^{\mathrm{T}} = [0.05, 0.05]^{\mathrm{T}}$. 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 $[0, 0]^{\mathrm{T}}$ and covariance matrix $\mathrm{diag}\{10^{-1}, 10^{-1}\}$. Fig. 4.8 shows an example of CA and target trajec-

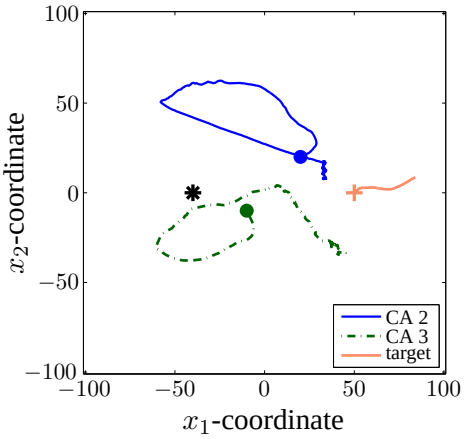


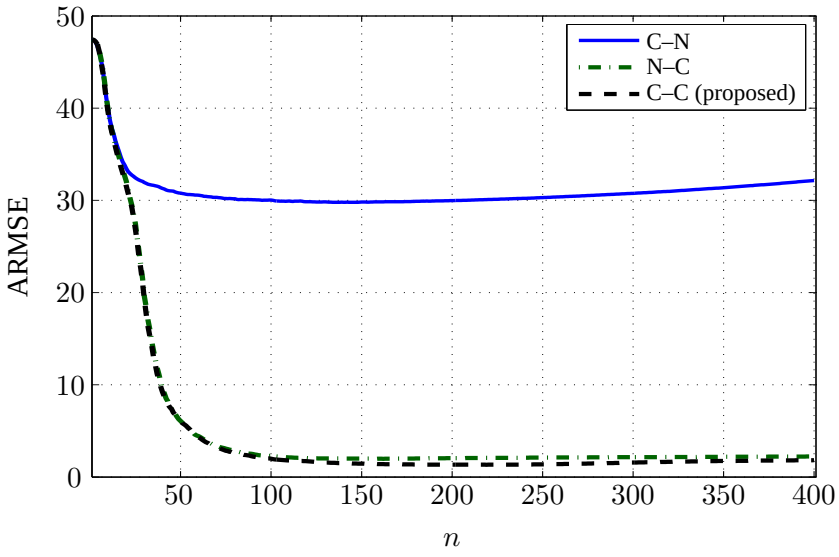
Figure 4.8: Example trajectories for CoSNAT with information-seeking control (C–C scheme). The initial CA positions are indicated by bullets, the initial target position by a cross, and the anchor position by a star.

tories obtained with the proposed method of cooperative navigation 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.

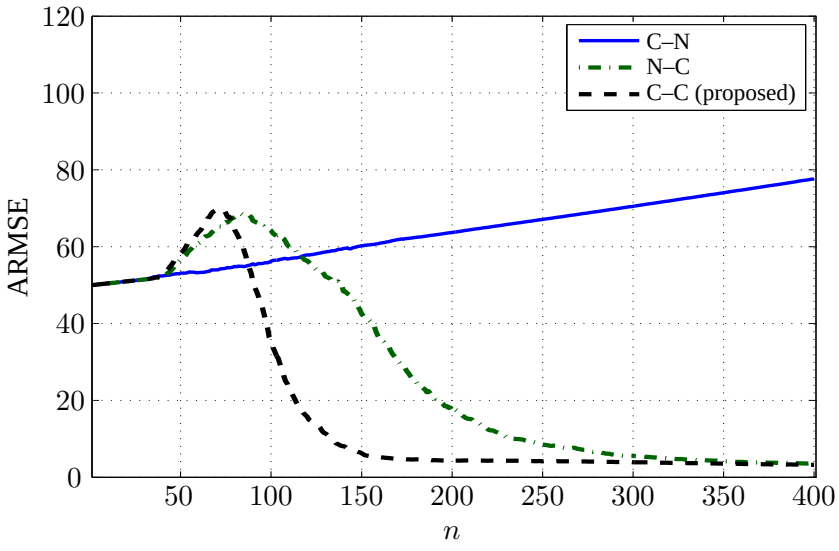
As before, we compare our C–C method with two reference methods, namely, noncooperative navigation with information-seeking control (N–C) and cooperative navigation with fixed, randomly chosen directions of movement (C–N). Fig. 4.9 shows the navigation ARMSEs and tracking ARMSEs of the three schemes, which were determined by averaging over the two CAs and over 100 simulation runs. The following observations can be made:

- The navigation performance of C–N is very poor: after an initial decrease, the ARMSE 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 navigation ARMSEs 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 tracking ARMSEs 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.



(a)



(b)

Figure 4.9: Performance of three different methods for simultaneous navigation and tracking: (a) Navigation ARMSE, (b) Tracking ARMSE.

This leads to a target position estimate of $[0, 0]^{\mathrm{T}}$ and in turn (since the target is initially located at $[50, 0]^{\mathrm{T}}$) to an initial tracking ARMSE of 50 at time $n = 1$. During the first 40 time steps, the ARMSE slowly increases since the target is slowly moving to the upper right corner. The ARMSE 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 ARMSE first increases and then decreases. The ARMSE of C–C 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 tracking ARMSE 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. 4.8.) The resulting target position estimate is located at that center. Thus, it is more distant from the true target position than the estimate $[0, 0]^{\mathrm{T}}$ 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 tracking ARMSE.

Chapter 5

Conclusion

This thesis proposed decentralized and scalable algorithms for *cooperative simultaneous navigation and tracking* (CoSNAT) in mobile agent networks. The development of such algorithms is challenging due to the potentially high mobility of the agents, the high accuracy and robustness required in many applications, and the decentralized structure of the network. A major advantage of the proposed algorithms for estimation and control is their generality. Our development was based on general state evolution and measurement models, an information-theoretic objective function for control, and sample-based representations of the involved probability distributions. These characteristics make the proposed algorithms suitable for arbitrary nonlinear and non-Gaussian system models. Numerical simulations of our CoSNAT estimation and control algorithms demonstrated intelligent behavior of the cooperative agents and high estimation accuracy.

5.1 Summary of Contributions

The conventional implementation of nonparametric belief propagation (NBP) for cooperative navigation suffers from high computation and communication costs. Therefore, for the case of two-dimensional (2D) position information and distance measurements between agents, we proposed a new particle-based BP scheme using parametric message representations and a new technique for message multiplication. In this scheme, computation and communication costs are significantly reduced.

In addition, we introduced a *dimension-augmented* reformulation of BP for cooperative navigation. This reformulation allows the systematic and straightforward application of an arbitrary sequential Bayesian estimator (or Bayesian filter) to BP-based cooperative navigation. We used this principle to develop a new nonparametric (particle-based) implementation of BP with reduced complexity. The main advantage of the proposed NBP algorithm is that its complexity scales only linearly—rather than quadratically as in conventional NBP [Ihler et al., 2005, Lien et al., 2012]—with the number of particles.

We also used the dimension-augmented BP scheme to develop the *sigma point BP* (SPBP) message passing algorithm for cooperative navigation. SPBP extends the sigma point filter—

also known as unscented Kalman filter—to Bayesian inference on general (loopy) factor graphs, such as they arise in cooperative navigation. Messages and marginal posterior probability density functions (PDFs) are represented by mean vectors and covariance matrices, which are calculated using sigma points and the unscented transformation. Thereby, SPBP avoids the typically high communication cost of conventional NBP. This fact makes SPBP especially well suited to cooperative navigation and certain other decentralized inference problems in wireless agent networks. Our analysis and simulation results demonstrated significant advantages of SPBP and of the new NBP scheme over conventional NBP regarding performance, complexity, and communication requirements.

Next, we extended the proposed NBP scheme to include noncooperative targets. This provided a framework and methodology for CoSNAT and, for the first time, enabled a consistent combination of cooperative navigation and distributed tracking for multiple mobile agents and targets. Starting from a factor graph formulation of the CoSNAT problem, we developed a particle-based, distributed BP algorithm for CoSNAT that employs a consensus scheme for a distributed calculation of the product of the target messages. More generally, the proposed integration of consensus in particle-based BP solves the problem of accommodating noncooperative agent nodes in distributed BP implementations, and thus it may be useful also for other distributed inference problems. The main advantage of the proposed CoSNAT framework and BP methodology over separate CSL and DTT and, also, over simultaneous localization and tracking (SLAT), is a probabilistic information transfer between the two parts of the factor graph corresponding to cooperative navigation and distributed tracking. This information transfer allows cooperative navigation to support distributed tracking and vice versa. Our simulation results demonstrated that this "turbo-like" principle [Wymeersch, 2007] can result in significant improvements in both navigation and tracking performance compared to state-of-the-art algorithms.

Finally, we extended our CoSNAT scheme by an information-seeking controller. The goal of this controller is to move the cooperative agents such that the information about the states to be estimated that is jointly carried by all the measurements in the agent network is maximized. This is achieved in a distributed, cooperative way 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 intelligent agent behavior and, in turn, to significantly improved estimation performance. For example, in a cooperative localization scenario with only one anchor present, mobile agents can localize themselves after a short time with an accuracy that is higher than that of the distance measurements.

5.2 Future Research

The development of distributed cooperative navigation and distributed tracking algorithms for agent networks is an active research area. In the following, we mention some possible extensions

of our work and some open questions and problems that establish interesting directions for future research.

- The dimension-augmented reformulation of BP introduced in Chapter 2 can be combined with other existing sequential Bayesian estimation algorithms (besides the particle filter and the sigma point filter considered in Chapter 2) to obtain new methods for navigation and tracking in networks. In addition, this approach may also be suitable for other inference problems.

- The proposed CoSNAT framework and methodology can be extended to accommodate additional tasks (i.e., in addition to cooperative navigation and distributed tracking) that involve local states of cooperative agents and/or global states of noncooperative agents. Examples of such tasks include distributed synchronization [Etzlinger et al., 2014, Meyer et al., 2013a, Etzlinger et al., 2013] and cooperative mapping [Dedeoglu and Sukhatme, 2000].

- An important goal is an extension of the CoSNAT framework and methodology to scenarios involving an unknown number of targets [Mahler, 2007] and measurements with data-association uncertainty [Bar-Shalom et al., 2009].

- The proposed information-seeking controller is myopic, i.e., its objective function involves the agent states only one time step ahead. A controller with a receding horizon [Mayne et al., 2000] is expected to improve the performance, especially in scenarios with multiple time-varying global states.

- Finally, the complexity and communication cost of the proposed estimation-and-control method can be reduced by introducing variational approximations [Mazuelas et al., 2013] and using cubature points [Arasaratnam and Haykin, 2009] instead of random samples.

Appendices

Appendix A

Proof of Equation (4.7)

We will use the following transformation rule for differential entropy [Deville and Deville, 2012, 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 $J_g(\mathbf{a}) = \det \frac{\partial g(\mathbf{a})}{\partial \mathbf{a}}$,

$$h(\mathbf{b}) = h(\mathbf{a}) + e(\mathbf{a}), \quad \text{with } e(\mathbf{a}) \triangleq \int f(\mathbf{a}) \log |J_g(\mathbf{a})| \, \mathrm{d}\mathbf{a}. \tag{A.1}$$

The conditional differential entropy $h(\mathbf{x}^+|\mathbf{y}^+;\mathbf{u}^+)$ can be expanded as [Cover and Thomas, 2006, Chapter 8]

$$h(\mathbf{x}^+|\mathbf{y}^+;\mathbf{u}^+) = h(\mathbf{x}^+,\mathbf{y}^+;\mathbf{u}^+) - h(\mathbf{y}^+;\mathbf{u}^+). \tag{A.2}$$

The vector $\mathbf{x}^+$ consists of $\mathbf{x}_l^+$ and $\mathbf{x}_{\mathcal{A}\setminus\{l\}}^+ \triangleq \left[\mathbf{x}_k^+\right]_{k\in\mathcal{A}\setminus\{l\}}$, and there is $\mathbf{x}_l^+ = \tilde{g}_l(\mathbf{x}_l,\mathbf{u}_l^+)$ (see (4.6)). Thus, the first term on the right-hand side of (A.2) can be expressed as

$$h(\mathbf{x}^+,\mathbf{y}^+;\mathbf{u}^+) = h\big(\tilde{g}_l(\mathbf{x}_l,\mathbf{u}_l^+),\mathbf{x}_{\mathcal{A}\setminus\{l\}}^+,\mathbf{y}^+;\mathbf{u}^+\big).$$

Applying the transformation rule (A.1) to the "extended state evolution mapping"

$$\tilde{g}_l^* : \left[\mathbf{x}_l^{\mathrm{T}},\mathbf{x}_{\mathcal{A}\setminus\{l\}}^{+\mathrm{T}},\mathbf{y}^{+\mathrm{T}}\right]^{\mathrm{T}} \mapsto \left[\big(\tilde{g}_l(\mathbf{x}_l,\mathbf{u}_l^+)\big)^{\mathrm{T}},\mathbf{x}_{\mathcal{A}\setminus\{l\}}^{+\mathrm{T}},\mathbf{y}^{+\mathrm{T}}\right]^{\mathrm{T}},$$

we then obtain

$$h(\mathbf{x}^+,\mathbf{y}^+;\mathbf{u}^+) = h\big(\mathbf{x}_l,\mathbf{x}_{\mathcal{A}\setminus\{l\}}^+,\mathbf{y}^+;\mathbf{u}^+\big) + e\big(\mathbf{x}_l,\mathbf{x}_{\mathcal{A}\setminus\{l\}}^+,\mathbf{y}^+;\mathbf{u}_l^+\big), \tag{A.3}$$

where

$$e\big(\mathbf{x}_l,\mathbf{x}_{\mathcal{A}\setminus\{l\}}^+,\mathbf{y}^+;\mathbf{u}_l^+\big) \triangleq \iiint f\big(\mathbf{x}_l,\mathbf{x}_{\mathcal{A}\setminus\{l\}}^+,\mathbf{y}^+\big) \log \big|J_{\tilde{g}_l^*}\big(\mathbf{x}_l,\mathbf{x}_{\mathcal{A}\setminus\{l\}}^+,\mathbf{y}^+;\mathbf{u}_l^+\big)\big| \mathrm{d}\mathbf{x}_l \, \mathrm{d}\mathbf{x}_{\mathcal{A}\setminus\{l\}}^+ \, \mathrm{d}\mathbf{y}^+.$$

Here, $J_{\tilde{g}_l^*}\big(\mathbf{x}_l,\mathbf{x}_{\mathcal{A}\setminus\{l\}}^+,\mathbf{y}^+;\mathbf{u}_l^+\big)$ is the Jacobian determinant of $\tilde{g}_l^*\big(\mathbf{x}_l,\mathbf{x}_{\mathcal{A}\setminus\{l\}}^+,\mathbf{y}^+;\mathbf{u}_l^+\big)$. It is easily seen that

$$J_{\tilde{g}_l^*}\big(\mathbf{x}_l,\mathbf{x}_{\mathcal{A}\setminus\{l\}}^+,\mathbf{y}^+;\mathbf{u}^+\big) = J_{\tilde{g}_l}(\mathbf{x}_l;\mathbf{u}^+),$$

and thus we obtain further

$$\begin{aligned} e\big(\mathbf{x}_l, \mathbf{x}^+_{\mathcal{A}\setminus\{l\}}, \mathbf{y}^+; \mathbf{u}^+_l\big) &= \iint \left[\int f\big(\mathbf{x}_l, \mathbf{x}^+_{\mathcal{A}\setminus\{l\}}, \mathbf{y}^+\big) \mathrm{d}\mathbf{x}^+_{\mathcal{A}\setminus\{l\}}\, \mathrm{d}\mathbf{y}^+ \right] \log |J_{\tilde{g}_l}(\mathbf{x}_l; \mathbf{u}^+_l)|\, \mathrm{d}\mathbf{x}_l \\ &= \int f(\mathbf{x}_l) \log |J_{\tilde{g}_l}(\mathbf{x}_l; \mathbf{u}^+_l)|\, \mathrm{d}\mathbf{x}_l \\ &= G_l(\mathbf{u}^+_l)\,. \end{aligned} \tag{A.4}$$

Inserting (A.4) into (A.3) and the resulting expression of $h(\mathbf{x}^+, \mathbf{y}^+; \mathbf{u}^+)$ into (A.2) gives

$$h(\mathbf{x}^+|\mathbf{y}^+; \mathbf{u}^+) \;=\; h\big(\mathbf{x}_l, \mathbf{x}^+_{\mathcal{A}\setminus\{l\}}, \mathbf{y}^+; \mathbf{u}^+\big) + G_l(\mathbf{u}^+_l) - h(\mathbf{y}^+; \mathbf{u}^+)\,. \tag{A.5}$$

Next, we repeat this transformation procedure but apply it to the term $h\big(\mathbf{x}_l, \mathbf{x}^+_{\mathcal{A}\setminus\{l\}}, \mathbf{y}^+; \mathbf{u}^+\big)$ in (A.5) instead of $h(\mathbf{x}^+, \mathbf{y}^+; \mathbf{u}^+)$. Consider an arbitrary $l' \in \mathcal{C}\setminus\{l\}$, and note that $\mathbf{x}^+_{\mathcal{A}\setminus\{l\}}$ consists of $\mathbf{x}^+_{l'}$ and $\mathbf{x}^+_{\mathcal{A}\setminus\{l,l'\}} \triangleq \big[\mathbf{x}^+_k\big]_{k\in\mathcal{A}\setminus\{l,l'\}}$, where $\mathbf{x}^+_{l'} = \tilde{g}_{l'}(\mathbf{x}_{l'}, \mathbf{u}^+_{l'})$ according to (4.6). Proceeding as above and inserting the resulting expression of $h\big(\mathbf{x}_l, \mathbf{x}^+_{\mathcal{A}\setminus\{l\}}, \mathbf{y}^+; \mathbf{u}^+\big)$ into (A.5) yields

$$h(\mathbf{x}^+|\mathbf{y}^+; \mathbf{u}^+) \;=\; h\big(\mathbf{x}_l, \mathbf{x}_{l'}, \mathbf{x}^+_{\mathcal{A}\setminus\{l,l'\}}, \mathbf{y}^+; \mathbf{u}^+\big) + G_{l'}(\mathbf{u}^+_{l'}) + G_l(\mathbf{u}^+_l) - h(\mathbf{y}^+; \mathbf{u}^+)\,.$$

We continue this procedure in a recursive fashion, splitting off CA state vectors from $\mathbf{x}^+_{\mathcal{A}\setminus\{l,l'\}}$ until only the target states (contained in $\mathbf{x}^+_{\mathcal{T}}$) are left, and applying the transformation rule at each recursion. In the end, we obtain

$$h(\mathbf{x}^+|\mathbf{y}^+; \mathbf{u}^+) \;=\; h(\mathbf{x}_{\mathcal{C}}, \mathbf{x}^+_{\mathcal{T}}, \mathbf{y}^+; \mathbf{u}^+) \;+ \sum_{l\in\mathcal{C}} G_l(\mathbf{u}^+_l) - h(\mathbf{y}^+; \mathbf{u}^+)\,.$$

Finally, Equation (4.7) is obtained by noting that

$$h(\mathbf{x}_{\mathcal{C}}, \mathbf{x}^+_{\mathcal{T}}, \mathbf{y}^+; \mathbf{u}^+) = h(\mathbf{x}_{\mathcal{C}}, \mathbf{x}^+_{\mathcal{T}}|\mathbf{y}^+; \mathbf{u}^+) + h(\mathbf{y}^+; \mathbf{u}^+).$$

Appendix B

Derivation of (4.19) and (4.20)

B.1 Derivation of (4.19)

Using (4.16) in (4.14) and recalling that $\beta_l(\mathbf{y}^+, \mathbf{x}_\mathcal{C}, \mathbf{x}_\mathcal{T}^+, \mathbf{u}^+)$ does not depend on $\mathbf{u}_l^+$ (see (4.18)) yields

$$\frac{\partial D_I(\mathbf{u}^+)}{\partial \mathbf{u}_l^+} = \iiint f(\mathbf{x}_\mathcal{C}, \mathbf{x}_\mathcal{T}^+)\, \beta_l(\mathbf{y}^+, \mathbf{x}_\mathcal{C}, \mathbf{x}_\mathcal{T}^+, \mathbf{u}^+) \frac{\partial \alpha_l(\mathbf{y}^+, \mathbf{x}_\mathcal{C}, \mathbf{x}_\mathcal{T}^+, \mathbf{u}^+)}{\partial \mathbf{u}_l^+} \times \log \frac{\alpha_l(\mathbf{y}^+, \mathbf{x}_\mathcal{C}, \mathbf{x}_\mathcal{T}^+, \mathbf{u}^+)\, \beta_l(\mathbf{y}^+, \mathbf{x}_\mathcal{C}, \mathbf{x}_\mathcal{T}^+, \mathbf{u}^+)}{f(\mathbf{y}^+; \mathbf{u}^+)} \mathrm{d}\mathbf{x}_\mathcal{C}\, \mathrm{d}\mathbf{x}_\mathcal{T}^+\, \mathrm{d}\mathbf{y}^+. \tag{B.1}$$

Setting $\mathbf{u}^+ = \mathbf{u}_\mathrm{r}^+$, and multiplying and dividing the integrand in (B.1) by $\alpha_l(\mathbf{y}^+, \mathbf{x}_\mathcal{C}, \mathbf{x}_\mathcal{T}^+, \mathbf{u}_\mathrm{r}^+)$, we obtain further

$$\left.\frac{\partial D_I(\mathbf{u}^+)}{\partial \mathbf{u}_l^+}\right|_{\mathbf{u}^+ = \mathbf{u}_\mathrm{r}^+} = \iiint q(\mathbf{y}^+, \mathbf{x}_\mathcal{C}, \mathbf{x}_\mathcal{T}^+; \mathbf{u}^+ = \mathbf{u}_\mathrm{r}^+)\, \frac{1}{\alpha_l(\mathbf{y}^+, \mathbf{x}_\mathcal{C}, \mathbf{x}_\mathcal{T}^+, \mathbf{u}_\mathrm{r}^+)} \times \left.\frac{\partial \alpha_l(\mathbf{y}^+, \mathbf{x}_\mathcal{C}, \mathbf{x}_\mathcal{T}^+, \mathbf{u}^+)}{\partial \mathbf{u}_l^+}\right|_{\mathbf{u}^+ = \mathbf{u}_\mathrm{r}^+} \times \log \frac{\alpha_l(\mathbf{y}^+, \mathbf{x}_\mathcal{C}, \mathbf{x}_\mathcal{T}^+, \mathbf{u}_\mathrm{r}^+)\, \beta_l(\mathbf{y}^+, \mathbf{x}_\mathcal{C}, \mathbf{x}_\mathcal{T}^+, \mathbf{u}_\mathrm{r}^+)}{f(\mathbf{y}^+; \mathbf{u}^+ = \mathbf{u}_\mathrm{r}^+)} \mathrm{d}\mathbf{x}_\mathcal{C}\, \mathrm{d}\mathbf{x}_\mathcal{T}^+\, \mathrm{d}\mathbf{y}^+, \tag{B.2}$$

where

$$q(\mathbf{y}^+, \mathbf{x}_\mathcal{C}, \mathbf{x}_\mathcal{T}^+; \mathbf{u}^+ = \mathbf{u}_\mathrm{r}^+) \triangleq f(\mathbf{x}_\mathcal{C}, \mathbf{x}_\mathcal{T}^+)\, \alpha_l(\mathbf{y}^+, \mathbf{x}_\mathcal{C}, \mathbf{x}_\mathcal{T}^+, \mathbf{u}_\mathrm{r}^+)\, \beta_l(\mathbf{y}^+, \mathbf{x}_\mathcal{C}, \mathbf{x}_\mathcal{T}^+, \mathbf{u}_\mathrm{r}^+)\,.$$

Then, (4.19) is recognized to be a Monte Carlo approximation of (B.2) that is obtained by performing importance sampling [Doucet et al., 2001] using $q(\mathbf{y}^+, \mathbf{x}_\mathcal{C}, \mathbf{x}_\mathcal{T}^+; \mathbf{u}^+ = \mathbf{u}_\mathrm{r}^+)$ as importance density, i.e., the particles $\mathbf{y}^{+(j,j')}$, $\mathbf{x}_\mathcal{C}^{(j)}$, and $\mathbf{x}_\mathcal{T}^{+(j)}$ occurring in (4.19) are drawn from $q(\mathbf{y}^+, \mathbf{x}_\mathcal{C}, \mathbf{x}_\mathcal{T}^+; \mathbf{u}^+ = \mathbf{u}_\mathrm{r}^+)$. Using (4.16), this importance density can be expressed as

$$q(\mathbf{y}^+, \mathbf{x}_\mathcal{C}, \mathbf{x}_\mathcal{T}^+; \mathbf{u}^+ = \mathbf{u}_\mathrm{r}^+) = f(\mathbf{x}_\mathcal{C}, \mathbf{x}_\mathcal{T}^+) f(\mathbf{y}^+ | \mathbf{x}_\mathcal{C}, \mathbf{x}_\mathcal{T}^+; \mathbf{u}^+ = \mathbf{u}_\mathrm{r}^+) = f(\mathbf{x}_\mathcal{C}, \mathbf{x}_\mathcal{T}^+, \mathbf{y}^+; \mathbf{u}^+ = \mathbf{u}_\mathrm{r}^+)\,.$$

The second expression, $f(\mathbf{x}_{\mathcal{C}}, \mathbf{x}_{\mathcal{T}}^{+}) f(\mathbf{y}^{+} | \mathbf{x}_{\mathcal{C}}, \mathbf{x}_{\mathcal{T}}^{+}; \mathbf{u}^{+} = \mathbf{u}_{\mathrm{r}}^{+})$, underlies the two-stage sampling procedure described in Section 4.4.1.

B.2 Derivation of (4.20)

We have

$$f(\mathbf{y}^{+}; \mathbf{u}^{+} = \mathbf{u}_{\mathrm{r}}^{+}) = \iint f(\mathbf{y}^{+} | \mathbf{x}_{\mathcal{C}}, \mathbf{x}_{\mathcal{T}}^{+}; \mathbf{u}^{+} = \mathbf{u}_{\mathrm{r}}^{+}) f(\mathbf{x}_{\mathcal{C}}, \mathbf{x}_{\mathcal{T}}^{+}) \, \mathrm{d}\mathbf{x}_{\mathcal{C}} \, \mathrm{d}\mathbf{x}_{\mathcal{T}}^{+} . \tag{B.3}$$

Using particles $\left\{ \left(\mathbf{x}_{\mathcal{C}}^{(j)}, \mathbf{x}_{\mathcal{T}}^{+(j)} \right) \right\}_{j=1}^{J} \sim f(\mathbf{x}_{\mathcal{C}}, \mathbf{x}_{\mathcal{T}}^{+})$ (see Section 4.4.1), a Monte Carlo approximation of (B.3) is obtained as

$$f(\mathbf{y}^{+}; \mathbf{u}^{+} = \mathbf{u}_{\mathrm{r}}^{+}) \approx \frac{1}{J} \sum_{j=1}^{J} f\left(\mathbf{y}^{+} \middle| \mathbf{x}_{\mathcal{C}}^{(j)}, \mathbf{x}_{\mathcal{T}}^{+(j)}; \mathbf{u}^{+} = \mathbf{u}_{\mathrm{r}}^{+} \right) .$$

Evaluating this for $\mathbf{y}^{+} = \mathbf{y}^{+(j,j')}$ (again see Section 4.4.1) and inserting (4.16) yields (4.20).

Appendix C

Drawing Particles from Likelihood Functions

C.1 Flooding-Based Approach

We consider the flooding-based setting of Section 4.4.1, i.e., we present the drawing of particles from $f\big(\mathbf{y}^+\big|\mathbf{x}_{\mathcal{C}}^{(j)},\mathbf{x}_{\mathcal{T}}^{+(j)};\mathbf{u}^+=\mathbf{u}_{\mathrm{r}}^+\big)$. Note that particles $\big\{\mathbf{x}_{l'}^{(j)}\big\}_{j=1}^{J}\sim f(\mathbf{x}_{l'})$, $l'\in\mathcal{C}$ and $\big\{\mathbf{x}_{m}^{+(j)}\big\}_{j=1}^{J}\sim f(\mathbf{x}_{m}^{+})$, $m\in\mathcal{T}$ 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 (4.1) and (1.2), the composite measurement vector $\mathbf{y}^+$ can be written as

$$\mathbf{y}^+ \;=\; \big[d_l(\mathbf{x}_l^+,\mathbf{x}_k^+,\mathbf{v}_{l,k}^+)\big]_{l\in\mathcal{C},k\in\mathcal{A}_l}. \tag{C.1}$$

First, CA l obtains particles

$$\big\{\mathbf{x}_{l'}^{+(j)}\big\}_{j=1}^{J}\sim\tilde{f}(\mathbf{x}_{l'}^+)\;\triangleq\;\int\delta\big(\mathbf{x}_{l'}^+-\tilde{g}_{l'}(\mathbf{x}_{l'},\mathbf{u}_{\mathrm{r},l'}^+)\big)\,f(\mathbf{x}_{l'})\,\mathrm{d}\mathbf{x}_{l'} \tag{C.2}$$

for all $l'\in\mathcal{C}$ by evaluating $\tilde{g}_{l'}(\mathbf{x}_{l'},\mathbf{u}_{l'}^+)$ in (4.6) at $\mathbf{x}_{l'}=\mathbf{x}_{l'}^{(j)}$ and $\mathbf{u}_{l'}^+=\mathbf{u}_{\mathrm{r},l'}^+$, i.e.,

$$\mathbf{x}_{l'}^{+(j)}\;=\;\tilde{g}_{l'}\big(\mathbf{x}_{l'}^{(j)},\mathbf{u}_{\mathrm{r},l'}^+\big)\,,\quad j=1,\ldots,J. \tag{C.3}$$

Thus, at this point, particles $\big\{\mathbf{x}_{k}^{+(j)}\big\}_{j=1}^{J}$ for all $k\in\mathcal{A}$ are available at CA l. Next, for each $j\in\{1,\ldots,J\}$, CA l draws particles $\big\{\mathbf{v}_{l,k}^{+(j,j')}\big\}_{j'=1}^{J'}\sim f(\mathbf{v}_{l,k}^+)$ for $l\in\mathcal{C}$ and $k\in\mathcal{A}_l$. Finally, CA l obtains particles

$$\big\{\mathbf{y}^{+(j,j')}\big\}_{j'=1}^{J'}\sim f\big(\mathbf{y}^+\big|\mathbf{x}_{\mathcal{C}}^{(j)},\mathbf{x}_{\mathcal{T}}^{+(j)};\mathbf{u}^+=\mathbf{u}_{\mathrm{r}}^+\big) \tag{C.4}$$

by evaluating (C.1) at $\mathbf{x}_l^+=\mathbf{x}_l^{+(j)}$, $\mathbf{x}_k^+=\mathbf{x}_k^{+(j)}$, and $\mathbf{v}_{l,k}^+=\mathbf{v}_{l,k}^{+(j,j')}$, i.e.,

$$\mathbf{y}^{+(j,j')}\;=\;\big[d_l\big(\mathbf{x}_l^{+(j)},\mathbf{x}_k^{+(j)},\mathbf{v}_{l,k}^{+(j,j')}\big)\big]_{l\in\mathcal{C},k\in\mathcal{A}_l},\quad j'=1,\ldots,J'.$$

C.2 Consensus-Based Approach

In the following we describe the drawing of particles from $f\big(\tilde{\mathbf{y}}_l^+ \big| \mathbf{x}_l^{(j)}, \mathbf{x}_{\mathcal{C}_l}^{(j)}, \mathbf{x}_{\mathcal{T}_l}^{+(j)}; \mathbf{u}_{\mathcal{C}_l}^+ = \mathbf{u}_{\mathrm{r},\mathcal{C}_l}^+\big)$ as considered in the consensus-based setting of Section 4.4.1. Note that particles $\big\{\mathbf{x}_{l'}^{(j)}\big\}_{j=1}^J \sim f(\mathbf{x}_{l'})$, $l' \in \{l\} \cup \mathcal{C}_l$ and $\big\{\mathbf{x}_m^{+(j)}\big\}_{j=1}^J \sim f(\mathbf{x}_m^+)$, $m \in \mathcal{T}_l$ are available at CA l. We recall from (4.25) that $\tilde{\mathbf{y}}_l^+ = \big[d_l(\mathbf{x}_l^+, \mathbf{x}_k^+, \mathbf{v}_{l,k}^+)\big]_{k\in\mathcal{A}_l}$. Based on the analogy of this expression to (C.1), the desired particles

$$\big\{\tilde{\mathbf{y}}_l^{+(j,j')}\big\}_{j'=1}^{J'} \sim f\big(\tilde{\mathbf{y}}_l^+ \big| \mathbf{x}_l^{(j)}, \mathbf{x}_{\mathcal{C}_l}^{(j)}, \mathbf{x}_{\mathcal{T}_l}^{+(j)}; \mathbf{u}_{\mathcal{C}_l}^+ = \mathbf{u}_{\mathrm{r},\mathcal{C}_l}^+\big) \tag{C.5}$$

can be calculated by carrying out the steps of Appendix C.1 with obvious modifications—in particular, $\mathbf{y}^+$ is replaced by $\tilde{\mathbf{y}}_l^+$, $\mathcal{C}$ by $\{l\} \cup \mathcal{C}_l$, and $\mathcal{T}$ by $\mathcal{T}_l$. More specifically, CA l obtains particles $\big\{\mathbf{x}_{l'}^{+(j)}\big\}_{j=1}^J$ for $l' \in \{l\} \cup \mathcal{C}_l$ according to (C.3) and, for each $j \in \{1,\dots,J\}$, draws particles $\big\{\mathbf{v}_{l,k}^{+(j,j')}\big\}_{j'=1}^{J'} \sim f(\mathbf{v}_{l',k}^+)$ for $k \in \mathcal{A}_l$. Then, for each $j \in \{1,\dots,J\}$, it obtains particles $\big\{\tilde{\mathbf{y}}_l^{+(j,j')}\big\}_{j'=1}^{J'}$ by evaluating (4.25) at $\mathbf{x}_l^+ = \mathbf{x}_l^{+(j)}$, $\mathbf{x}_k^+ = \mathbf{x}_k^{+(j)}$, and $\mathbf{v}_{l,k}^+ = \mathbf{v}_{l,k}^{+(j,j')}$, i.e.,

$$\tilde{\mathbf{y}}^{+(j,j')} = \big[d_l\big(\mathbf{x}_l^{+(j)}, \mathbf{x}_k^{+(j)}, \mathbf{v}_{l,k}^{+(j,j')}\big)\big]_{k\in\mathcal{A}_l}, \quad j' = 1,\dots,J'.$$

Bibliography

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

[Anderson and Moore, 2005] Anderson, B. D. O. and Moore, J. B. (2005). *Optimal Filtering*. Dover Publications, Mineola, NY.

[Arasaratnam and Haykin, 2009] Arasaratnam, I. and Haykin, S. (2009). Cubature Kalman filters. *IEEE Trans. Autom. Control*, 54(6):1254–1269.

[Arulampalam et al., 2002] Arulampalam, M. S., Maskell, S., Gordon, N., and Clapp, T. (2002). A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Trans. Signal Process.*, 50:174–188.

[Atanasov et al., 2015] Atanasov, N. A., Ny, J. L., and Pappas, G. J. (2015). Distributed algorithms for stochastic source seeking with mobile robot networks. *J. Dyn. Sys., Meas., Control*, 137(3).

[Bar-Shalom et al., 2009] Bar-Shalom, Y., Daum, F., and Huang, J. (2009). The probabilistic data association filter. *IEEE Control Syst. Mag.*, 29(6):82–100.

[Bar-Shalom et al., 2002] Bar-Shalom, Y., Kirubarajan, T., and Li, X.-R. (2002). *Estimation with Applications to Tracking and Navigation*. John Wiley & Sons, Inc., New York, NY.

[Briers et al., 2005] Briers, M., Doucet, A., and Singh, S. S. (2005). Sequential auxiliary particle belief propagation. In *Proc. IEEE Fusion-08*, Budapest, Hungary.

[Bullo et al., 2009] Bullo, F., Cortés, J., and Martínez, S. (2009). *Distributed Control of Robotic Networks: A Mathematical Approach to Motion Coordination Algorithms*. Princeton University Press, Princeton, NJ.

[Burgard et al., 1997] Burgard, W., Fox, D., and Thrun, S. (1997). Active mobile robot localization by entropy minimization. In *Proc. EUROBOT '97*, pages 155–162, Brescia, Italy.

[Caceres et al., 2011] Caceres, M. A., Penna, F., Wymeersch, H., and Garello, R. (2011). Hybrid cooperative positioning based on distributed belief propagation. *IEEE J. Sel. Areas Commun.*, 29(10):1948–1958.

[Corke et al., 2010] Corke, P., Wark, T., Jurdak, R., Hu, W., Valencia, P., and Moore, D. (2010). Environmental wireless sensor networks. *Proc. IEEE*, 98(11):1903–1917.

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

[Dardari et al., 2012] Dardari, D., Falletti, E., and Luise, M. (2012). *Satellite and Terrestrial Radio Positioning Techniques*. Academic Press, Oxford, UK.

[Dario et al., 2005] Dario, I. A., Akyildiz, I. F., Pompili, D., and Melodia, T. (2005). Underwater acoustic sensor networks: Research challenges. *Ad Hoc Networks*, 3:257–279.

[Das and Wymeersch, 2012] Das, K. and Wymeersch, H. (2012). Censoring for Bayesian cooperative positioning in dense wireless networks. *IEEE J. Sel. Areas Commun.*, 30(9):1835–1842.

[Daum and Huang, 2003] Daum, F. and Huang, J. (2003). Curse of dimensionality and particle filters. In *Proc. IEEE AC '02*, pages 1979–1993, Big Sky, MT.

[Dedeoglu and Sukhatme, 2000] Dedeoglu, G. and Sukhatme, G. S. (2000). Landmark-based matching algorithm for cooperative mapping by autonomous robots. In *Distributed Autonomous Robotic Systems 4*, pages 251–260, New York, NY. Springer.

[Deville and Deville, 2012] Deville, Y. and Deville, A. (2012). Exact and approximate quantum independent component analysis for qubit uncoupling. In *Proc. LVA/ICA '12*, pages 58–65, Tel Aviv, Israel.

[Dimakis et al., 2010] Dimakis, A. G., Kar, S., Moura, J. M. F., Rabbat, M. G., and Scaglione, A. (2010). Gossip algorithms for distributed signal processing. *Proc. IEEE*, 98(11):1847–1864.

[Douc and Cappe, 2005] Douc, R. and Cappe, O. (2005). Comparison of resampling schemes for particle filtering. In *Proc. ISPA-05*, pages 64–69, Zagreb, Croatia.

[Doucet et al., 2001] Doucet, A., de Freitas, N., and Gordon, N. (2001). *Sequential Monte Carlo Methods in Practice*. Springer, New York, NY.

[Driessen and Boers, 2008] Driessen, H. and Boers, Y. (2008). Map estimation in particle filter tracking. In *Proc. IET 2008*, pages 41–45, Birmingham, UK.

[Durrant-Whyte and Bailey, 2006] Durrant-Whyte, H. and Bailey, T. (2006). Simultaneous localization and mapping: Part I. *IEEE Trans. Robot. Autom.*, 13(2):99–110.

[Etzlinger et al., 2013] Etzlinger, B., Meyer, F., Springer, A., Hlawatsch, F., and Wymeersch, H. (2013). Cooperative simultaneous localization and synchronization: A distributed hybrid message passing algorithm. In *Proc. Asilomar Conf. Sig., Syst., Comput.*, pages 1978–1982, Pacific Grove, CA.

[Etzlinger et al., 2014] Etzlinger, B., Wymeersch, H., and Springer, A. (2014). Cooperative synchronization in wireless networks. *IEEE Trans. Signal Process.*, 62(11):2837–2849.

[Farahmand et al., 2011] Farahmand, S., Roumeliotis, S. I., and Giannakis, G. B. (2011). Set-membership constrained particle filter: Distributed adaptation for sensor networks. *IEEE Trans. Signal Process.*, 59(9):4122–4138.

[Fletcher, 1987] Fletcher, R. (1987). *Practical Methods of Optimization*. Wiley, New York, NY.

[Funiak et al., 2006] Funiak, S., Guestrin, C., Paskin, M., and Sukthankar, R. (2006). Distributed localization of networked cameras. In *Proc. IPSN-06*, pages 34–42, Nashville, TN.

[Gan et al., 2007] Gan, G., Ma, C., and Wu, J. (2007). *Data Clustering: Theory, Algorithms, and Applications*. SIAM, Philadelphia, PA.

[Garcia et al., 2014] Garcia, G. E., Muppirisetty, L. S., Schiller, E. M., and Wymeersch, H. (2014). On the trade-off between accuracy and delay in cooperative UWB localization: Performance bounds and scaling laws. *IEEE Trans. Wireless Commun.*, 13(8):4574–4585.

[Gordon et al., 1993] Gordon, N. J., Salmond, D. J., and Smith, A. F. M. (1993). Novel approach to nonlinear/non-gaussian bayesian state estimation. *IEE Proc.-F Radar and Signal Process.*, 140(2):107–113.

[Grocholsky, 2002] Grocholsky, B. (2002). *Information-Theoretic Control of Multiple Sensor Platforms*. PhD thesis, University of Sydney, Sydney, Australia.

[Groves, 2008] Groves, P. D. (2008). *Principles of GNSS, inertial, and multisensor integrated navigation systems*. Artech House, London, UK.

[Haykin, 2001] Haykin, S. (2001). Kalman filters. In Haykin, S., editor, *Kalman Filtering and Neural Networks*, chapter 1, pages 1–21. Wiley, New York, NY.

[Hlinka et al., 2013] Hlinka, O., Hlawatsch, F., and Djuric, P. M. (2013). Distributed particle filtering in agent networks: A survey, classification, and comparison. *IEEE Signal Process. Mag.*, 30(1):61–81.

[Hlinka et al., 2014] Hlinka, O., Hlawatsch, F., and Djuric, P. M. (2014). Consensus-based distributed particle filtering with distributed proposal adaptation. *IEEE Trans. Signal Process.*, 62(12):3029–3041.

[Hlinka et al., 2012] Hlinka, O., Slučiak, O., Hlawatsch, F., Djurić, P. M., and Rupp, M. (2012). Likelihood consensus and its application to distributed particle filtering. *IEEE Trans. Signal Process.*, 60(8):4334–4349.

[Hoffmann and Tomlin, 2010] Hoffmann, G. M. and Tomlin, C. J. (2010). Mobile sensor network control using mutual information methods and particle filters. *IEEE Trans. Autom. Control*, 55(1):32–47.

[Ihler et al., 2005] Ihler, A. T., Fisher, J. W., Moses, R. L., and Willsky, A. S. (2005). Nonparametric belief propagation for self-localization of sensor networks. *IEEE J. Sel. Areas Commun.*, 23(4):809–819.

[Jordan, 1999] Jordan, M. I., editor (1999). *Learning in Graphical Models*. MIT Press, Cambridge, MA.

[Julian et al., 2012] Julian, B. J., Angermann, M., Schwager, M., and Rus, D. (2012). Distributed robotic sensor networks: An information-theoretic approach. *Int. J. Robot. Res.*, 31(10):1134–1154.

[Julier and Uhlmann, 1997] Julier, S. J. and Uhlmann, J. K. (1997). A new extension of the Kalman filter to nonlinear systems. In *Proc. AeroSense-97*, pages 182–193, Orlando, FL.

[Kantas et al., 2012] Kantas, N., Singh, S., and Doucet, A. (2012). Distributed maximum likelihood for simultaneous self-localization and tracking in sensor networks. *IEEE Trans. Signal Process.*, 60(10):5038–5047.

[Kaplan and Hegarty, 2005] Kaplan, E. D. and Hegarty, C. J. (2005). *Understanding GPS - Principles and Applications*. Artech House, Norwood, MA.

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

[Kim and Kumar, 2012] Kim, K.-D. and Kumar, P. R. (2012). Cyber physical systems: A perspective at the centennial. *Proc. IEEE*, 100:1287–1308.

[Ko et al., 2010] Ko, J., Lu, C., Srivastava, M. B., Stankovic, J. A., Terzis, A., and Welsh, M. (2010). Wireless sensor networks for healthcare. *Proc. IEEE*, 98(11):1947–1960.

[Kschischang et al., 2001] Kschischang, F. R., Frey, B. J., and Loeliger, H.-A. (2001). Factor graphs and the sum-product algorithm. *IEEE Trans. Inf. Theory*, 47(2):498–519.

[Li et al., 2002] Li, D., Wong, K. D., Hu, Y. H., and Sayeed, A. M. (2002). Detection, classification, and tracking of targets. *IEEE Signal Process. Mag.*, 19(2):17–29.

[Li and Jia, 2012] Li, W. and Jia, Y. (2012). Consensus-based distributed multiple model UKF for jump markov nonlinear systems. *IEEE Trans. Autom. Control*, 57(1):227–233.

[Li and Jilkov, 2003] Li, X. R. and Jilkov, V. P. (2003). Survey of maneuvering target tracking. Part I: Dynamic models. *IEEE Trans. Aerosp. Electron. Syst.*, 39:1333–1364.

[Lien et al., 2012] Lien, J., Ferner, J., Srichavengsup, W., Wymeersch, H., and Win, M. Z. (2012). A comparison of parametric and sample-based message representation in cooperative localization. *Int. J. Navig. Observ.*

[Lim and Kim, 2000] Lim, H. and Kim, C. (2000). Multicast tree construction and flooding in wireless ad hoc networks. In *Proc. MSWIM '00*, pages 61–68, New York, NY.

[Liu et al., 2007] Liu, J., Chu, M., and Reich, J. (2007). Multitarget tracking in distributed sensor networks. *IEEE Signal Process. Mag.*, 24(3):36–46.

[Loeliger, 2004] Loeliger, H.-A. (2004). An introduction to factor graphs. *IEEE Signal Process. Mag.*, 21(1):28–41.

[Mahler, 2007] Mahler, R. P. S. (2007). *Statistical Multisource-Multitarget Information Fusion*. Artech House, Norwood, MA.

[Mayne et al., 2000] Mayne, D. Q., Rawlings, J. B., Rao, C. V., and Scokaert, P. O. M. (2000). Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814.

[Mazuelas et al., 2009] Mazuelas, S., Bahillo, A., Lorenzo, R. M., Fernandez, P., Lago, F. A., Garcia, E., Blas, J., and Abril, E. J. (2009). Robust indoor positioning provided by real-time RSSI values in unmodified WLAN networks. *IEEE J. Sel. Topics Signal Process.*, 3(5):821–831.

[Mazuelas et al., 2013] Mazuelas, S., Shen, Y., and Win, M. Z. (2013). Belief condensation filtering. *IEEE Trans. Signal Process.*, 61(18):4403–4415.

[Meissner, 2014] Meissner, P. (2014). *Multipath-Assisted Indoor Positioning*. PhD thesis, Graz University of Technology, Graz, Austria.

[Meyer et al., 2013a] Meyer, F., Etzlinger, B., Hlawatsch, F., and Springer, A. (2013a). A distributed particle-based belief propagation algorithm for cooperative simultaneous localization and synchronization. In *Proc. Asilomar Conf. Sig., Syst., Comput.*, pages 527–531, Pacific Grove, CA.

[Meyer et al., 2013b] Meyer, F., Hlawatsch, F., and Wymeersch, H. (2013b). Cooperative simultaneous localization and tracking (CoSLAT) with reduced complexity and communication. In *Proc. IEEE ICASSP-13*, pages 4484–4488, Vancouver, Canada.

[Meyer et al., 2014a] Meyer, F., Hlinka, O., and Hlawatsch, F. (2014a). Sigma point belief propagation. *Signal Process. Lett.*, 21(2):145–149.

[Meyer et al., 2014b] Meyer, F., Hlinka, O., Wymeersch, H., Riegler, E., and Hlawatsch, F. (2014b). Cooperative simultaneous localization and tracking in mobile agent networks. submitted. Available online: http://arxiv.org/abs/1403.1824.

[Meyer et al., 2012] Meyer, F., Riegler, E., Hlinka, O., and Hlawatsch, F. (2012). Simultaneous distributed sensor self-localization and target tracking using belief propagation and likelihood consensus. In *Proc. 46th Asilomar Conf. Sig., Syst., Comp.*, pages 1212–1216, Pacific Grove, CA.

[Meyer et al., 2014c] Meyer, F., Wymeersch, H., Fröhle, M., and Hlawatsch, F. (2014c). Distributed estimation with information-seeking control in agent networks. submitted. Available online: http://arxiv.org/abs/1408.3732.

[Meyer et al., 2015] Meyer, F., Wymeersch, H., and Hlawatsch, F. (2015). Cooperative localization with information-seeking control. In *Proc. IEEE ICASSP-15*, Brisbane, Australia. to appear.

[Mohammadi and Asif, 2011] Mohammadi, A. and Asif, A. (2011). Consensus-based distributed unscented particle filter. In *Proc. IEEE SSP '11*, pages 237–240, Nice, France.

[Morbidi and Mariottini, 2013] Morbidi, F. and Mariottini, G. L. (2013). Active target tracking and cooperative localization for teams of aerial vehicles. *IEEE Trans. Control Syst. Technol.*, 21(5):1694–1707.

[Nayak and Stojmenović, 2010] Nayak, A. and Stojmenović, I. (2010). *Wireless Sensor and Actuator Networks: Algorithms and Protocols for Scalable Coordination and Data Communication*. Wiley, Hoboken, NJ.

[Olfati-Saber et al., 2007] Olfati-Saber, R., Fax, J. A., and Murray, R. M. (2007). Consensus and cooperation in networked multi-agent systems. *Proc. IEEE*, 95(1):215–233.

[Olfati-Saber and Murray, 2004] Olfati-Saber, R. and Murray, R. M. (2004). Consensus problems in networks of agents with switching topology and time-delays. *IEEE Trans. Autom. Control*, 49:1520–1533.

[Pedersen et al., 2011] Pedersen, C., Pedersen, T., and Fleury, B. H. (2011). A variational message passing algorithm for sensor self-localization in wireless networks. In *Proc. IEEE ISIT-11*, pages 2158–2162, Saint Petersburg, Russia.

[Pham et al., 2009] Pham, T.-D., Ngo, H. Q., Le, V.-D., Lee, S., and Lee, Y.-K. (2009). Broadcast gossip based distributed hypothesis testing in wireless sensor networks. In *Proc. ATC-09*, pages 84–87, Haiphong, Vietnam.

[Press et al., 1992] Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. (1992). *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, New York, NY.

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

[Ross, 2014] Ross, P. E. (2014). Open-source drones for fun and profit. *IEEE Spectr.*, 51(3):54–59.

[Ryan et al., 2007] Ryan, A. D., Durrant-Whyte, H., and Hedrick, J. K. (2007). Information-theoretic sensor motion control for distributed estimation. In *Proc. IMECE '07*, Seattle, WA.

[Sathyan and Hedley, 2013] Sathyan, T. and Hedley, M. (2013). Fast and accurate cooperative tracking in wireless networks. *IEEE Trans. Mobile Comput.*, 12(9):1801–1813.

[Savic and Wymeersch, 2013] Savic, V. and Wymeersch, H. (2013). Simultaneous localization and tracking via real-time nonparametric belief propagation. In *Proc. IEEE ICASSP-13*, pages 5180–5184, Vancouver, Canada.

[Savic et al., 2014] Savic, V., Wymeersch, H., and Zazo, S. (2014). Belief consensus algorithms for fast distributed target tracking in wireless sensor networks. *Signal Processing*, 95:149–160.

[Savic and Zazo, 2012] Savic, V. and Zazo, S. (2012). Reducing communication overhead for cooperative localization using nonparametric belief propagation. *IEEE Commun. Lett.*, 1(4):308–311.

[Schwager et al., 2011] Schwager, M., Dames, P., Rus, D., and Kumar, V. (2011). A multi-robot control policy for information gathering in the presence of unknown hazards. In *Proc. ISRR-11*, Flagstaff, AZ.

[Shen et al., 2012] Shen, Y., Mazuelas, S., and Win, M. Z. (2012). Network navigation: Theory and interpretation. *IEEE J. Sel. Areas Commun.*, 30(9):1823–1834.

[Shima and Rasmussen, 2009] Shima, T. and Rasmussen, S. (2009). *UAV Cooperative Decision and Control: Challenges and Practical Approaches*. SIAM, Philadelphia, PA.

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

[Skolnik, 2008] Skolnik, M. (2008). *Radar Handbook*. McGraw-Hill Education.

[Stilwell and Bishop, 2000] Stilwell, D. J. and Bishop, B. E. (2000). Platoons of underwater vehicles. *IEEE Control Syst. Mag.*, 20(6):45–52.

[Taj and Cavallaro, 2011] Taj, M. and Cavallaro, A. (2011). Distributed and decentralized multicamera tracking. *IEEE Signal Process. Mag.*, 28(3):46–58.

[Taylor et al., 2006] Taylor, C., Rahimi, A., Bachrach, J., Shrobe, H., and Grue, A. (2006). Simultaneous localization, calibration, and tracking in an ad hoc sensor network. In *Proc. IPSN-06*, pages 27–33, Nashville, TN.

[Teng et al., 2012] Teng, J., Snoussi, H., Richard, C., and Zhou, R. (2012). Distributed variational filtering for simultaneous sensor localization and target tracking in wireless sensor networks. *IEEE Trans. Veh. Technol.*, 61(5):2305–2318.

[Thrun et al., 2005] Thrun, S., Burgard, W., and Fox, D. (2005). *Probabilistic Robotics*. MIT Press, Cambridge, MA.

[Wainwright and Jordan, 2008] Wainwright, M. J. and Jordan, M. I. (2008). Graphical models, exponential families, and variational inference. *Found. Trends Mach. Learn.*, 1.

[Waite, 2001] Waite, A. D. (2001). *Sonar for Practising Engineers*. Wiley, Chichester, UK.

[Wan and van der Merwe, 2001] Wan, E. A. and van der Merwe, R. (2001). The unscented Kalman filter. In Haykin, S., editor, *Kalman Filtering and Neural Networks*, chapter 7, pages 221–280. Wiley, New York, NY.

[Weiss and Freeman, 2001] Weiss, Y. and Freeman, W. T. (2001). Correctness of belief propagation in Gaussian graphical models of arbitrary topology. *Neural Computation*, 13(10):2173–2200.

[Wu et al., 2011] Wu, Y.-C., Chaudhari, Q. M., and Serpedin, E. (2011). Clock synchronization of wireless sensor networks. *IEEE Signal Process. Mag.*, 28(1).

[Wymeersch, 2007] Wymeersch, H. (2007). *Iterative Receiver Design*. Cambridge University Press, New York, NY.

[Wymeersch et al., 2009] Wymeersch, H., Lien, J., and Win, M. Z. (2009). Cooperative localization in wireless networks. *Proc. IEEE*, 97(2):427–450.

[Wymeersch et al., 2012] Wymeersch, H., Penna, F., and Savic, V. (2012). Uniformly reweighted belief propagation for estimation and detection in wireless networks. *IEEE Trans. Wireless Comm.*, 11(4):1587–1595.

[Xiao and Boyd, 2003] Xiao, L. and Boyd, S. (2003). Fast linear iterations for distributed averaging. *Systems and Control Letters*, 53:65–78.

[Xiao et al., 2005] Xiao, L., Boyd, S., and Lall, S. (2005). A scheme for robust distributed sensor fusion based on average consensus. In *Proc. IPSN '05*, pages 63–70, Los Angeles, CA.

[Zachery et al., 2011] Zachery, R. A., Sastry, S. S., and Kumar, V. (2011). Special issue on swarming in natural and engineered systems [scanning the issue]. *Proc. of the IEEE*, 99(9):1466–1469.

[Zhao and Nehorai, 2007] Zhao, T. and Nehorai, A. (2007). Distributed sequential Bayesian estimation of a diffusive source in wireless sensor networks. *IEEE Trans. Signal Process.*, 55(4):1511–1524.