Decentralized Localization of Sparsely-Communicating Robot Networks: A Centralized-Equivalent Approach

Keith Y. K. Leung, *Student Member, IEEE*, Timothy D. Barfoot, and Hugh H. T. Liu, *Member, IEEE*

***Abstract*—Finite-range sensing and communication are factors in the connectivity of a dynamic mobile-robot network. State estimation becomes a difficult problem when communication connections allowing information exchange between all robots are not guaranteed. This paper presents a decentralized state-estimation algorithm guaranteed to work in dynamic robot networks without connectivity requirements. We prove that a robot only needs to consider its own knowledge of network topology in order to produce an estimate equivalent to the centralized state estimate whenever possible while ensuring that the same can be performed by all other robots in the network. We prove certain properties of our technique and then it is validated through simulations. We present a comprehensive set of results, indicating the performance benefit in different network connectivity settings, as well as the scalability of our approach.**

***Index Terms*—Autonomous agents, decentralized state estimation, finite sensing and communication, localization, networked robots.**

I. Introduction

A COOPERATIVE multirobot system is beneficial in many applications. Besides allowing for greater coverage in exploration and searching tasks, it also allows for the implementation of more complex strategies over a single robot. A greater number of robots can also provide a certain degree of redundancy to ensure the completion of tasks should a portion of the multirobot team become disabled.

Communication and the mutual exchange of information are key performance factors for many cooperative multirobot systems. Research in this area often assumes that robots can broadcast information to all other team members, or it assumes a static network configuration. However, limited communication range becomes a factor in larger workspaces or environments populated with structures that obstruct communication. Limited sensing and communication range, as well as network dynamics, pose an added layer of difficulty in both cooperative state estimation (localization) and cooperative planning. Recent work by

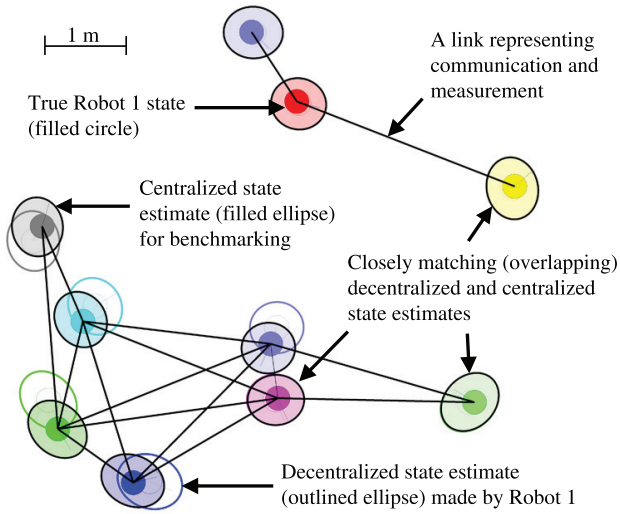


Fig. 1. Simulation of our decentralized state-estimation algorithm with a dynamic network of 10 robots. The decentralized state estimates shown are from the perspective of robot 1.

Burgard *et al.* [1] studied a method for cooperative exploration and partially looked at the performance of their proposed method under limited communication range but concluded that further study is required. The novel contribution in our paper is the study of the cooperative-localization problem over a dynamic network, wherein connectivity of all robots is not guaranteed, and each robot must localize all robots using odometry, relative measurements, and through the communication of this information. We present a decentralized state-estimation algorithm that is a) *equivalent* to a centralized state estimator whenever possible, b) *scalable* to any number of robots in the sense that it is not necessary to know the total number of robots in the team (but only if communication between robots is bidirectional and if the communication range limit is greater than or equal to the measurement range limit), and c) *general* in that many recursive filtering methods can be applied within our framework. The brute-force solution to obtaining a decentralized state estimate that is equivalent to a centralized estimate (whenever possible) would be to have each robot keep all past data and communicate this with all robots encountered. This is neither as scalable nor as efficient as recursive state-estimation techniques. One of the challenges of decentralized state estimation is to ensure that when a robot replaces information with a state estimate, it will not compromise the others' ability to do so. The question of what information to keep or discard at what time is answered

Manuscript received December 17, 2008; revised June 9, 2009. First published December 31, 2009; current version published February 9, 2010. This paper was recommended for publication by Associate Editor S. Roumeliotis and Editor W. K. Chung upon evaluation of the reviewers' comments. This paper was presented in part at the IEEE International Conference on Robotics and Automation, Kobe, Japan, 2009, with title "Decentralized Localization for Dynamic and Sparse Robot Networks." This work was supported by the Natural Sciences and Engineering Research Council of Canada.

The authors are with the University of Toronto Institute for Aerospace Studies, Toronto, ON M3H 5T6, Canada (e-mail: keith.leung@robotics.utias.utoronto.ca; tim.barfoot@utoronto.ca; liu@utias.utoronto.ca).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TRO.2009.2035741

by our proposed algorithm, and we show that this decision can be based on each robot's individual knowledge of the network topology, making the approach implementable as a recursive decentralized algorithm.

In the following section, we will review the literature on distributed and decentralized state estimation. In Section III, we will formulate our decentralized state-estimation problem. In developing a solution to the problem, we begin to examine how information flows within a dynamic network (see Section IV). We will also introduce several theorems regarding information flow, which are used in our proposed algorithm in Section V. This algorithm is tested through simulation, and we will present the results in Section VI. To provide an insight to the scalability of our approach, we will examine how the number of robots, communication range, and the size of the workspace affects localization performance.

II. Related Work

The problem of distributed and decentralized state estimation has been studied by various researchers in the past for uncontrolled linear stochastic systems, as well as for sensor networks. The work we present in this paper takes a more general approach and examines nonlinear systems with control inputs, but the following related research is worth mentioning. Berg and Durrant-Whyte [2], [3] looked at a computer network of arbitrary topology, in which each node in the network is attempting to estimate the partial or full state of the system. Their work showed how the information filter can be used to easily incorporate observation data from many nodes. The concept of the *internodal transformation matrix* was introduced for a node to incorporate relevant information from other nodes. This matrix can also be used to determine the minimal communication connections required between the nodes of the system to achieve optimal state estimation. Grime and Durrant-Whyte [4] examined the decentralized state-estimation problem in a network, wherein information can propagate by hopping between nodes. The *channel filter* was introduced to ensure that only new information is passed to neighboring nodes, but it was shown only to work in an acyclic network (with no loops). This work was extended by Utete and Durrant-Whyte [5] for an arbitrary network topology, wherein communication restrictions are applied, enabling the channel filter to work. Later, this work was simulated by Bourgault and Durrant-Whyte [6] for a team of unmanned autonomous vehicles (UAV) that formed a chain network for communication and state estimation. The simulation showed how the UAVs successfully coordinated with each other in a search mission.

More closely related to this paper is the following research on cooperative localization. Kurazume and Hirose [7] introduced one of the first methods for cooperative localization, where a robot team was split into two groups. One group of robots moved, while the other remained stationary, essentially serving as landmarks for the first group of robots for localization. This concept of using other robots in a team for localization has since been extended by various researchers. It was shown in [8] that given a limited set of relative measurements from a network of robots in a formation, localizing the robots (and determining the correct formation) is an NP-hard problem.

Roumeliotis and Bekey [9] performed distributed multirobot localization by decomposing the extended Kalman filter (EKF) into a number of filters that can perform the prediction step of the EKF locally on each robot. As the highlight, this work showed how the propagation of the covariance matrix can be factored using singular-value decomposition such that the factored terms can be computed by each robot individually using their own odometry data. The factored terms only need to be combined before a measurement update, which then requires full network connectivity (i.e., all robots need to communicate with each other for the EKF correction step). More recently, Nerurkar *et al.* [10] performed cooperative localization using a distributed *maximum a posteriori (MAP)* estimator.

Howard [11] looked at performing multirobot *simultaneous localization and mapping (SLAM)*, wherein each robot is unaware of each other's initial pose and begins state estimation in a decentralized manner. When robots encountered each other for the first time, their individual maps are combined into a common map using relative poses. The mapping process then continued as robots broadcasted new observations to each other. The notion of a virtual robot traveling backward in time was introduced to allow the incorporation of information gathered by a robot before the common map was merged.

Madhavan *et al.* [12] studied how cooperative localization and mapping can be performed using robots with heterogeneous sets of sensors. It was shown how localization performance can improve for robots with poor localization capabilities by using relative measurements to a robot with better localization capabilities [equipped with a differential global positioning system (DGPS)]. This work was demonstrated by trials wherein two robots cooperatively performed vision-based terrain mapping.

Rekleitis *et al.* [13] examined how sensing paradigm and the number of robots affect localization performance with multiple robots. The sensing paradigms included range-only, bearing-only, range-and-bearing, and full-pose sensing (range, bearing, and relative orientation). The results indicated that full-pose measurement provides slightly better results than range-and-bearing measurement, as well as range-only measurement. Increasing the number of robots also showed better localization results. However, bearing-only measurement performed poorly, regardless of the number of robots used.

Later, Roumeliotis and Rekleitis [14] analytically quantified the benefit of cooperative localization. It was shown how the number of robots can influence localization performance. Furthermore, it was discovered that there are diminishing returns in increasing the number of robots to reduce uncertainty.

Most recently, Trawny *et al.* [15] showed how the communication cost of cooperative localization can be significantly reduced by using quantized measurements. By quantizing measurements to 4 bits, results were practically the same compared to using real-valued measurements in a MAP estimator.

Static network connectivity, or the ability to broadcast information to all other robots in a multirobot team, is an important requirement for the works mentioned previously. In this paper,

we will address the challenge with performing state estimation over a time-varying network. The difficulty involved is the obstruction of data flow between robots and the unpredictable sequence in which data are received. Ferguson and How [16] examined various filters and network architectures to arrive at suboptimal (full and partial) state estimates. For the various methods and network architectures compared, the tradeoff between performance and computational requirement was shown. This paper will look at how every robot in the team can obtain the best possible (centralized) estimate under the aforementioned network properties.

In a dynamic network, the sequence in which information is communicated to a robot is often out of order, and this leads to the *out-of-sequence measurements (OOSM)* problem. Bar-Shalom [17], [18] examined some possible remedies for treating OOSM in state estimation using a Kalman filter (also known as the negative timestep problem) and applied this to target tracking. It was shown that for a missed measurement, the only way to incorporate it so as to produce a centralized state estimate is to sequentially reprocess all following measurements. This is an important concept to note for the rest of this paper. In situations where past measurements are no longer available, various methods were proposed to arrive at an approximate estimate. For the special case when the missed measurement is from a single timestep back, it was shown that the centralized state estimate can be recovered.

As network properties can cause difficulties in cooperative localization by preventing the exchange of information, robots may also experience the problem of overusing the same piece of information. This problem, known as *cyclic update*, occurs when one robot provides information to another robot to update its state estimate, which, in turn, is provided back to the first robot for updating its state estimate. The result of this is overconfident state estimates. Previously, we mentioned how the channel filter was applied in sensor networks for this reason. Howard *et al.* [19] introduced the *dependency tree* as a remedy to this problem for a multirobot system, but it is not guaranteed to work in all cases. We will show that the decentralized state-estimation method that we present in this paper does not suffer from cyclic updates. In contrast, we handle this by giving each piece of information a unique identifier and then limit the amount of information through the Markov property.

Also worth mentioning is the *consensus problem*, but note that the work presented in the current paper does not fit into the consensus-problem framework. For a network of agents, solving the consensus problem requires determining a *consensus algorithm*. This algorithm specifies how agents should interact with their neighboring agents and defines the actions to be used by each agent, with the goal of converging the states of all agents to some common value. In general, the study of the consensus problem looks at how coherent global behavior can be produced by local control laws or estimation methods. An example of this is distributed formation control for multivehicle systems [20]. Another example of an application that involves more complex local actions is distributed control for object clustering [21]. For distributed state estimation, Schizas *et al.* [22] looked at how consensus can be reached in wireless sensor networks. Recently, research on the consensus problem has extended to switching (dynamic) network topologies, but convergence can only be guaranteed under some network topology restrictions [23].

To the knowledge of the authors, there has been no previous work by other researchers on a localization method that is able to achieve our objective of providing a decentralized state estimate in a dynamic network (wherein connectivity is not guaranteed) that is equivalent to the centralized state estimate whenever possible. In [24], we introduced the preliminary concepts regarding information flow and the first version of our decentralized framework. Simulations were conducted for a single-network configuration. Compared with our past work, this paper adds significantly more insight into the decentralized localization problem by providing all the theorems and proofs that we have developed regarding information flow in a robot network. We also provide greater details for our decentralized state-estimation algorithm and the scalability of our approach. Most importantly, this paper provides simulated localization results for a broad range of network connectivity settings to give a comprehensive assessment on the performance of our approach.

III. Problem Formulation

In a multirobot system, let N represent the set that contains the unique identification indexes of all robots. The total number of robots corresponds to $|N|$, the cardinality of the set, and we assume that the identification indexes of the robots range from 1 to $|N|$. Furthermore, we define $N_{i,k}$ as the set of robots known to robot i at a specific timestep k. We assume a general system model for the robots

$$\mathbf{x}_{i,k} = \mathbf{g}\left(\mathbf{x}_{i,k-1}, \mathbf{u}_{i,k}, \epsilon_k\right)$$
$$\mathbf{y}_{i,k}^{j,i} = \mathbf{h}\left(\mathbf{x}_{i,k}, \mathbf{x}_{j,k}, \delta_k\right) \quad (\forall j)(d_k^{j,i} \leq r_{\mathsf{obs}})$$

where for timestep k, $\mathbf{x}_{i,k}$ represents the state (pose) of robot i, $\mathbf{u}_{i,k}$ represents the odometry information of robot i, $\mathbf{g}(\cdot)$ is the state-transition function (with process noise ϵ_k), $\mathbf{y}_{i,k}^{j,i}$ represents the measurement (e.g., range/bearing) of robot j with respect to robot i, $\mathbf{h}(\cdot)$ is the measurement function (with measurement noise, δ_k), $d_k^{j,i}$ is the distance between robot i and j, and r_{obs} is the measurement range limit. Robots within communication range r_{comm} of each other are able to exchange and relay information, which includes state estimates, odometry data, and measurement data. Let

$$X_k = \{\mathbf{x}_{i,k}\} \quad (\forall i \in N)$$

represent the set of all robot states at time step k, and let

$$X_{Q,k} = \{\mathbf{x}_{i,k}\} \quad (\forall i \in Q)(Q \subseteq N)$$

represent the set of states at timestep k for the robots in some subset Q of N. Similarly, let

$$U_k = \{\mathbf{u}_{i,k}\} \quad (\forall i \in N)$$

represent the set of odometry information from all robots at timestep k, and let

$$U_{Q,k} = \{\mathbf{u}_{i,k}\} \quad (\forall i \in Q)(Q \subseteq N)$$

represent the set of odometry data at timestep k for all robots in subset Q of N. Let

$$Y_k = \{\mathbf{y}_{i,k}^{j,i}\} \quad (\forall i,j)(d_k^{j,i} \leq r_{\mathsf{obs}})$$

represent the set of all measurements made at timestep k

$$Y_{i,k} = \{\mathbf{y}_{i,k}^{j,i}\} \quad (\forall j)(d_k^{j,i} \leq r_{\mathsf{obs}})$$

represent the set of all measurements made by robot i at timestep k, and let

$$Y_{Q,k} = \{\mathbf{y}_{i,k}^{j,i}\} \quad (\forall i,j \in Q)(d_k^{j,i} \leq r_{\mathsf{obs}})$$

represent the set of measurements made between robots in set Q.

Due to uncertainty in both state transition and measurements, the true state of the system cannot be found deterministically, but can only be estimated using odometry and measurement data. In general, the centralized *belief* is represented by a probability density function $p(\cdot)$ over all robot states X_k

$$\mathsf{bel}(X_k) := p\left(X_k | \mathsf{bel}(X_0), U_{1:k}, Y_{1:k}\right)$$

which is conditioned on the initial belief $\mathsf{bel}(X_0)$, past odometry data, and past range and bearing measurements. From a practical and computation point of view, it is helpful to apply the *Markov property* [25]

$$p\left(X_k | \mathsf{bel}(X_0), U_{1:k}, Y_{1:k}\right) = p\left(X_k | \mathsf{bel}(X_{k-1}), U_k, Y_k\right)$$

when performing state estimation. This property makes the belief over the current state of a system independent of all past states, and it limits memory and processing requirements by allowing state estimation to be performed recursively. This can be accomplished for a centralized state estimator using the *Bayes filter* [26]

$$\begin{aligned}\mathsf{bel}(X_k) &= p\left(X_k | \mathsf{bel}(X_0), U_{1:k}, Y_{1:k}\right) \\ &= \eta\, p\left(Y_k | X_k\right) \int p\left(X_k | X_{k-1}, U_k\right) \\ &\qquad p\left(X_{k-1} | \mathsf{bel}(X_0), U_{1:k-1}, Y_{1:k-1}\right) dX_{k-1}\end{aligned}$$

where η is a normalizing constant to ensure that the resulting posterior probability density function $\mathsf{bel}(X_k)$ preserves the axiom of total probability. However, in a decentralized framework wherein robots are not always in contact with each other, the Markov property can only be applied once a robot obtains sufficient information regarding other robots through communication. Furthermore, each robot must ensure that other robots will no longer require any of the past information it possesses (because it will be discarded when applying the Markov property). Hence, the key problem is to determine the necessary and sufficient conditions under which the Markov property can be applied in order to obtain an estimate equivalent to that obtainable by a centralized state estimator when robots are only occasionally exchanging information with each other. Accordingly, our objective is for each robot i to estimate the state of all known robots (i.e., find $\mathsf{bel}(X_k)$) in a decentralized manner.

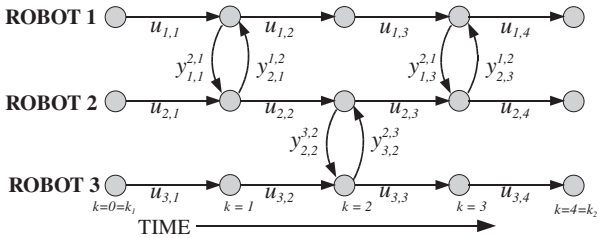


Fig. 2. Example global information flow graph $\mathcal{G}_{k_1:k_2}$ indicating state transition and communication links established between timesteps k_1 and k_2 for three robots.

IV. Information Flow in a Dynamic Network

In the case of sporadic communication and observations, it is essential to track the information available to each robot for making state estimations to ensure that 1) a robot does not apply the Markov property without receiving all information required to calculate the centralized state estimate, and 2) a robot does not reuse information and cause cyclic updates to occur. A graph is a convenient tool for representing network topology. We will first examine the robot network from the perspective of an outside observer having the ability to see all the interactions between robots. We will then look at the network locally (i.e., from the perspective of a particular robot). To simplify many of the following concepts in this paper, we will assume that $r_{\mathsf{comm}} = r_{\mathsf{obs}}$, but this is only for explanation purposes. We will revisit this assumption in a later section to show that the two do not need to be equivalent.

A. Global Perspective

Let $\mathcal{G}_{k_1:k_2}$ be a directed graph that shows the communication links established between robots from timestep k_1 to k_2. This graph can be used to show the flow and distribution of information, and we will refer to $\mathcal{G}_{k_1:k_2}$ as the *global information flow graph*. The number of nodes in $\mathcal{G}_{k_1:k_2}$ is the product of the number of robots and the number of timesteps represented by the graph. These nodes v can be systematically numbered using the time index k and the robot index i, such that

$$v(i,k) = (k - k_1)|N| + i, \quad (k_1 \leq k \leq k_2)(1 \leq i \leq |N|).$$

An example of an information flow graph is depicted in Fig. 2. In relation to the system model, an arc connecting two robots at a timestep represents a measurement between the two robots. This also represents a communication window (which allows the exchange of information possessed by both robots). Horizontal arcs represent state transitions. The presence of an arc connecting two nodes implies that all information at the originating node is also available at the destination node. Furthermore, odometry and measurement information that are labeled on the arcs making up the path in between are also available at the destination node.

As a robot traverses a workspace and occasionally observes and communicates with another robot, it will begin to accumulate information regarding the entire team. The specific data in its possession will depend on the evolving topology of the information flow graph. Let the *knowledge set* $S_{i,k}$ consist of all odometry and measurement data, as well as the previous state estimates known to robot i at time k. We will assume at the initial time that $S_{i,0} = \{\mathsf{bel}(\mathbf{x}_{i,0})\}$. A more in-depth discussion on this initial condition will follow in a later section. At each timestep, the knowledge set expands with the addition of new odometry data as well as measurement data if another robot is observed. Let $R_{i,k}$ represent the set of robots within distance r_{comm} of robot i at time k, and let $S^-_{i,k}$ represent the knowledge set after state transition and observations but before communication is established with any other robot

$$S^-_{i,k} = S_{i,k-1} \cup \{\mathbf{u}_{i,k}, Y_{i,k}\}. \tag{1}$$

When communication occurs between robots i and j, they will make their knowledge sets available to each other, and the knowledge set of both robots will become identical

$$S_{i,k} = S_{j,k} = S^-_{i,k} \cup S^-_{j,k}, \quad (\forall j \in R_{i,k}). \tag{2}$$

The previous equations model how information flows within the robot network at every timestep. With the progression of time, the knowledge set for each robot will continue to expand, causing the information storage requirement to also increase. If the Markov property is not exploited in an estimator, the amount of data in each knowledge set will increase over time without bound. In most centralized recursive-state estimators, we make use of the Markov property to reduce memory-storage requirements. In our decentralized state-estimation problem, this must be done with extreme care to ensure that all robots can also make the same state estimate. For this purpose, a *checkpoint* is defined as follows.

Definition 1: A *checkpoint* $C(k_c, k_e)$ is an event that occurs at the checkpoint time k_c that first comes into existence at k_e, in which the set of knowledge for each robot i contains for all j:

1) the previous state estimate of robot j at some timestep $k_{s,j} \leq k_c$;
2) all the odometry and measurement data of robot j from timestep $k_{s,j}$ to k_c.

Equivalently written using mathematics, a checkpoint occurs at timestep k_c when $S_{i,k_e} \supseteq S_{j,k_c} (\forall i, j)$.

Using Fig. 2 as an example, and assuming that each robot begins with $S_{i,0} = \{\mathsf{bel}(\mathbf{x}_{i,0})\}$, one of the checkpoints that can be found in this figure is $C(1,3)$. It can be verified that at $k = 3$, each robot has the previous state estimate of all robots (at $k = 0$). Also, each robot has the odometry and measurement data of all robots up to $k = 1$.

The importance of a checkpoint is that it allows us to apply the Markov property, thereby replacing an old state estimate, odometry, and measurement data up to k_c, with a new state estimate. To make use of a checkpoint, it is first necessary to prove its existence for a given information flow graph. According to the definition of a checkpoint, we need to show that the knowledge set of each robot at timestep k_e contains the state estimate

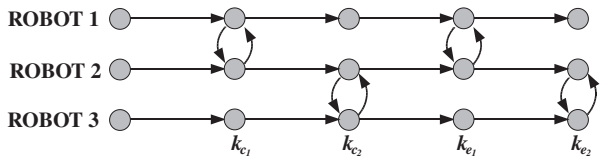


Fig. 3. Checkpoint $C(k_{c_2}, k_{e_2})$ cannot come into existence before an earlier occurring checkpoint $C(k_{c_1}, k_{e_1})$ comes into existence. Hence, k_{e_1} must be less than k_{e_2} since k_{c_1} is less than k_{c_2}.

for all robots at a timestep $k_{s,j}$, as well as the odometry and measurement data for all robots from timestep $k_{s,j}$ to k_c. Note that from (1) and (2), a property of knowledge sets is that

$$S_{i,k_1} \subseteq S_{i,k_2}, \quad (k_1 \leq k_2)$$

provided that the Markov property is not applied between k_1 and k_2. This property guarantees that all information is retained as the knowledge set (information) that a robot accumulates over time. By this argument, the knowledge set of a robot must always contain its previous state estimate, all its previous odometry data, as well as measurements. That is, if the Markov property is not applied

$$\begin{aligned} \mathsf{bel}(\mathbf{x}_{i,k_s}) \in S_{i,k_s} &\rightarrow \mathsf{bel}(\mathbf{x}_{i,k_s}) \in S_{i,k}, \quad (k \geq k_s) \\ \mathbf{u}_{i,k} \in S_{i,k} &\rightarrow \mathbf{u}_{i,k} \in S_{i,k_c}, \quad (k_c \geq k) \\ \mathbf{y}_{i,k} \in S_{i,k} &\rightarrow \mathbf{y}_{i,k} \in S_{i,k_c}, \quad (k_c \geq k). \end{aligned}$$

We now present a theorem regarding checkpoint existence.

Theorem 1.1: $C(k_c, k_e)$ exists if and only if a path exists from $v(i, k_c)$ to $v(j, k_e)$ on $\mathcal{G}_{k_c, k_e} (\forall i, j)$.

Proof: We will first assume that $C(k_c, k_e)$ exists. This implies that all odometry measurements and state estimates from all robots at k_c are in the knowledge sets of all robots at k_e. Clearly, this is only possible if there exists an information flow path from $v(i, k_c)$ to $v(j, k_e)$ $(\forall i, j)$.

Now we will assume that a path exists from $v(i, k_c)$ to $v(j, k_e)$ $(\forall i, j)$. By the knowledge set update rules, all odometry measurements and state estimates from all robots at k_c will become incorporated into the knowledge sets of all robots at k_e. Therefore, $C(k_c, k_e)$ exists. ■

The necessary and sufficient conditions of Theorem 1.1 provide a method for verifying checkpoint existence (an indication of when the Markov property is applicable), but there exists a more practical verification for checkpoint existence for implementation purposes. Before showing this, we need to introduce a lemma. Referring to Fig. 3, the lemma states that in the interval between the occurrence and existence time of one checkpoint (i.e., k_{c_1} and k_{e_1}), we can be certain that another checkpoint will not come into existence. Hence, information in knowledge sets within this interval will not be replaced by state estimates since we know (for now) that the Markov property can only be applied at a checkpoint.

Lemma 1.1: Suppose $C(k_{c_1}, k_{e_1})$ and $C(k_{c_2}, k_{e_2})$ exist and that $k_{e_1} \neq k_{e_2}$. Then $k_{c_1} < k_{c_2}$ if and only if $k_{e_1} < k_{e_2}$.

Proof: We will use the information flow graph as an aid in this proof. Furthermore, the notation $v_1 \xrightarrow{\text{path}} v_2$ will be used

to denote that a path exists from node v_1 to node v_2. We will show that there is a violation in the existence of a checkpoint if the necessary and sufficient conditions are not followed. More formally, let us first assume that

$$
\begin{aligned}
& k_{c_1} < k_{c_2} \\
& \Rightarrow v(i,k_{c_1}) \xrightarrow{\text{path}} v(i,k_{c_2}) \qquad (\forall i) \\
& \Rightarrow v(i,k_{c_2}) \xrightarrow{\text{path}} v(j,k) \qquad (\forall i,\forall j)(k_{e_2} \le k) \\
& \Rightarrow \begin{cases} v(i,k_{c_1}) \xrightarrow{\text{path}} v(j,k) & (\forall i,\forall j)(k_{e_2} \le k), \text{if } (k_{e_1} \ge k_{e_2}) \\ v(i,k_{c_1}) \xrightarrow{\text{path}} v(j,k) & (\forall i,\forall j)(k_{e_1} \le k), \text{if } (k_{e_1} < k_{e_2}) \end{cases} \\
& \Rightarrow k_{e_1} < k_{e_2}.
\end{aligned}
$$

To explain in greater detail, line 2 is true simply from the fact that a node representing a robot at a given time is always connected to a node representing the same robot at a future time. Line 3 is merely restating the assumption that k_{e_2} is the earliest time at which checkpoint 2 exists. In line 4, given that $k_{e_1} \ge k_{e_2}$, this implies that the earliest time at which checkpoint 1 exists is k_{e_2} and not k_{e_1}. Only when we are on line 5, where $k_{e_1} < k_{e_2}$, can k_{e_1} be the earliest time at which checkpoint 1 exists. Therefore, it must be true that $k_{e_1} < k_{e_2}$. Now, we will assume that

$$
\begin{aligned}
& k_{e_1} < k_{e_2} \\
& \Rightarrow v(i,k_{c_1}) \xrightarrow{\text{path}} v(j,k) \qquad (\forall i,\forall j)(k_{e_1} \le k) \\
& \Rightarrow (k_{c_1} \ge k_{c_2}),\ v(i,k_{c_2}) \xrightarrow{\text{path}} v(i,k_{c_1}) \qquad (\forall i) \\
& \Rightarrow \begin{cases} v(i,k_{c_2}) \xrightarrow{\text{path}} v(j,k), & (\forall i,\forall j)(k_{e_1} \le k), \text{if } (k_{c_1} \ge k_{c_2}) \\ v(i,k_{c_2}) \xrightarrow{\text{path}} v(j,k), & (\forall i,\forall j)(k_{e_2} \le k), \text{if } (k_{c_1} < k_{c_2}) \end{cases} \\
& \Rightarrow k_{c_1} < k_{c_2}.
\end{aligned}
$$

On line 2, we are restating the assumption that k_{e_1} is the earliest time at which checkpoint 1 exists. On line 3, given that $k_{c_1} \ge k_{c_2}$, we know a node representing a robot is always connected to a node representing the same robot at a future time. This implies on line 4 that the earliest time at which checkpoint 2 comes into existence is k_{e_1} and is invalid. On the other hand, if we are given that $k_{c_1} < k_{c_2}$ on line 5, the earliest time at which checkpoint 2 comes into existence becomes k_{e_2}. Hence, it must be true that $k_{c_1} < k_{c_2}$. ■

Using the earlier lemma, we now present a theorem which gives a more practical method for verifying the existence of a checkpoint for implementation purposes. Basically, we will show that it is possible to know when a checkpoint exists by looking for a subset of odometry information in the knowledge set of each robot.

Theorem 1.2: $C(k_c,k_e)$ exists if and only if the knowledge set of each robot at k_e contains $\mathbf{u}_{j,k_c}$ or $\mathsf{bel}(\mathbf{x}_{j,k_c})(\forall j)$. Expressed mathematically, $S_{i,k_e} \supseteq S_{j,k_c}(\forall i,j) \Leftrightarrow S_{i,k_e} \supseteq (\mathbf{u}_{j,k_c} \text{ or } \mathsf{bel}(\mathbf{x}_{j,k_c}))(\forall i,\forall j)$.

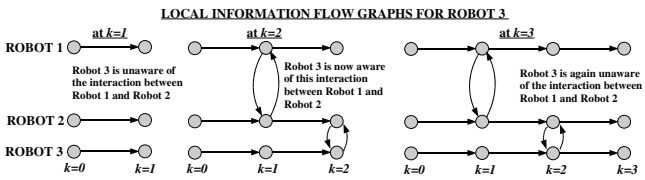


Fig. 4. Example showing the local information flow graph topology of a single robot (robot 3) as time progresses.

Proof: We will approach this proof by first assuming that $C(k_c,k_e)$ exists

$$
\begin{aligned}
& S_{i,k_e} \supseteq S_{j,k_c} \quad (\forall i,\forall j) \\
& \Rightarrow S_{i,k_e} \supseteq \begin{cases} \left\{\mathsf{bel}(\mathbf{x}_{j,k_{s,j}}), Y_{j,k_{s,j}+1:k_c}, \mathbf{u}_{j,k_{s,j}+1:k_c}\right\} \\ \quad (\forall i,\forall j) \text{ if } (k_{s,j} < k_c) \\ \left\{\mathsf{bel}(\mathbf{x}_{j,k_{s,j}})\right\} \quad (\forall i,\forall j) \text{ if } (k_{s,j} = k_c) \end{cases} \\
& \Rightarrow S_{i,k_e} \supseteq \mathbf{u}_{j,k_c} \text{ or } \mathsf{bel}(\mathbf{x}_{j,k_c}) \quad (\forall i,\forall j).
\end{aligned}
$$

In the second line, we expand $S_{j,k_c}(\forall j)$ to show the information that can be found in the knowledge sets depending on $k_{s,j}$, the time of the latest belief for robot j. In the last line, we show that either $\mathbf{u}_{j,k_c}$ or $\mathsf{bel}(\mathbf{x}_{j,k_c})$ for all robots j will always be available.

Now, assuming that the knowledge set of each robot at k_e contains $\mathbf{u}_{j,k_c}$ or $\mathsf{bel}(\mathbf{x}_{j,k_c})$

$$
\begin{aligned}
& S_{i,k_e} \supseteq \mathbf{u}_{j,k_c} \text{ or } \mathsf{bel}(\mathbf{x}_{j,k_c}) \qquad (\forall i,\forall j) \\
& \Rightarrow S_{i,k_e} \supseteq S_{j,k} \qquad (\forall i,\forall j)(k_c \le k \le k_e) \\
& \Rightarrow S_{i,k_e} \supseteq S_{j,k_c} \qquad (\forall i,\forall j).
\end{aligned}
$$

Since odometry data and the belief at k_c from all robots j are available, this implies on line 2 that $S_{j,k}$ must also be available for all robots j, where $k_c \le k \le k_e$. Furthermore, we know that the knowledge set of a robot will always contain its past knowledge set, provided the Markov property has not been applied. We are certain of this because Lemma 1.1 indicates that another checkpoint cannot come into existence between k_c and k_e. Therefore, the Markov property can never be applied within this timeframe. ■

B. Local Perspective

The information flow graph is a global graph in the sense that it represents the interactions of all robots as viewed by an outside observer. In relation to Fig. 2, the graph topology from the point of view of an individual robot will differ, as illustrated in Fig. 4.

This leads to the question of whether or not it is necessary for a robot to know the complete knowledge set of all other robots before determining that a checkpoint exists. It turns out that we do not. To show this, we present the definition of a *partial checkpoint* and a theorem for its existence.

Definition 2: A *partial checkpoint* $C_p(k_{c,i},k_{e,i})$ is an event that occurs for robot i at time $k_{c,i}$ that first comes into existence at $k_{e,i}$, in which the set of knowledge for robot i contains for all j:

1) the previous state estimate of robot j at some timestep $k_{s,j} \leq k_{c,i}$;
2) all the odometry and measurement data of robot j from timestep $k_{s,j}$ to $k_{c,i}$.

Equivalently written using mathematics, a partial checkpoint for robot i occurs at timestep $k_{c,i}$ when $S_{i,k_e} \supseteq S_{j,k_c}\ (\forall j)$.

We will only present here the theoretical statements regarding the existence of a partial checkpoint. The proofs of these are largely similar to the proofs of Theorem 1.1, Lemma 1.1, and Theorem 1.2, and are located in the Appendix.

Theorem 2.1: For robot i, $C_p(k_{c,i}, k_{e,i})$ exists if and only if a path exists from $v(j, k_{c,i})$ to $v(i, k_{e,i})$ on $\mathcal{G}_{k_c,k_e}\ (\forall j)$.

Lemma 2.1: Suppose $C_p(k_{c_1,i}, k_{e_1,i})$ and $C_p(k_{c_2,i}, k_{e_2,i})$ exist, and $k_{e_1,i} \neq k_{e_2,i}$. Then, $k_{c_1,i} < k_{c_2,i}$ if and only if $k_{e_1,i} < k_{e_2,i}$.

Theorem 2.2: $C_p(k_{c,i}, k_{e,i})$ exists if and only if the knowledge set of robot i at k_e contains $\mathbf{u}_{j,k_{c,i}}$ or $\mathsf{bel}(\mathbf{x}_{j,k_{c,i}})(\forall j)$. Expressed mathematically, $S_{i,k_{e,i}} \supseteq S_{j,k_{c,i}}(\forall j) \Leftrightarrow S_{i,k_{e,i}} \supseteq (\mathbf{u}_{j,k_{c,i}} \text{ or } \mathsf{bel}(\mathbf{x}_{j,k_{c,i}}))(\forall j)$.

Similarly to Theorem 1.2, this Theorem 2.2 is important as it provides a practical method to detect partial checkpoints when implementing our decentralized state-estimation framework. Note that a partial checkpoint can come into existence at different times for each robot, depending on the evolving topology of the robot network. We now present a lemma and two important theorems that relate partial checkpoints to checkpoints and show how the occurrence of these is affected when the Markov property is applied.

Lemma 3.1: $C(k_c, k_e)$ exists if and only if $C_p(k_{c,i}, k_{e,i})$ exists $(\forall i)$, with $(k_{c,i} = k_c)$, and $(k_{e,i} \leq k_e)$.

Proof: The underlying message in this lemma is that when all robots detect a partial checkpoint occurring for the same timestep, then a checkpoint will exist. The approach to this proof is to use the definitions of a checkpoint and a partial checkpoint. We will show that when the knowledge set of each robot satisfies the condition for partial checkpoint existence for timestep k_c, then we also satisfy the condition for checkpoint existence for timestep k_c.

First, assume

$$
\begin{aligned}
&C(k_c, k_e) \text{ exists} \\
&\Rightarrow S_{i,k_e} \supseteq S_{j,k_c} \qquad (\forall i, \forall j) \\
&\Rightarrow (\exists k_{e,i} \leq k_e), S_{i,k_{e,i}} \supseteq S_{j,k_c} \qquad (\forall i, \forall j) \\
&\Rightarrow (\exists k_{e,i} \leq k_e), S_{i,k_{e,i}} \supseteq S_{j,k_{c,i}} \qquad (\forall i, \forall j)(k_{c,i} = k_c) \\
&\Rightarrow C_p(k_{c,i}, k_{e,i}) \text{ exists} \qquad (\forall i)(k_{c,i} = k_c)(k_{e,i} \leq k_e).
\end{aligned}
$$

In line 2, we rewrite the expression of a checkpoint using its definition. Note that $S_{j,k_c}, (\forall j)$ is available to all robots at earliest timestep k_e, but it is possible for individual robots to obtain this at an earlier time $k_{e,i}$ as indicated on line 3. In line 4, k_c is simply replaced by $k_{c,i}$. Finally, we use the definition of a partial checkpoint to arrive at line 5. Now, we will assume that $C_p(k_{c,i}, k_{e,i})$ exists $(\forall i)$, with $(k_{c,i} = k_c)$ and $(k_{e,i} \leq k_e)$

$$
\begin{aligned}
&C_p(k_{c,i}, k_{e,i}) \text{ exists} \qquad (\forall i)(k_{c,i} = k_c)(k_{e,i} \leq k_e) \\
&\Rightarrow S_{i,k_{e,i}} \supseteq S_{j,k_{c,i}} \qquad (\forall i, \forall j)(k_{c,i} = k_c)(k_{e,i} \leq k_e) \\
&\Rightarrow S_{i,k_e} \supseteq S_{j,k_c} \quad (\forall i, \forall j) \\
&\Rightarrow C(k_c, k_e) \text{ exists.}
\end{aligned}
$$

The definition of a partial checkpoint is used in line 2. In line 3, we note that S_{i,k_e} is a superset of $S_{i,k_{e,i}}$ since $k_{e,i} \leq k_e$. Furthermore, we replace all $k_{c,i}$ with k_c and use the definition of a checkpoint to arrive at the last line. ■

Using this lemma, we can be sure that when partial checkpoints (that occur at the same time $k_{c,i}$ for all robots) exist, then a checkpoint also exists (with $k_c = k_{c,i}(\forall i)$).

Theorem 3.1: Suppose $C(k_c, k_e)$ exists, and robot m applies the Markov property when $C_p(k_c, k_{e,m})$ exists (i.e., at $k_{e,m}$). Then, $C_p(k_c, k_{e,i})$ continues to exist $(\forall i)$.

Proof: We approach this proof by examining the knowledge set of each robot and the changes caused by applying the Markov property. We then verify that partial checkpoints continue to exist for all robots.

When a checkpoint exists, and before the Markov property is applied by robot m, the knowledge sets of all robots i contain the belief at some previous time $k_{s,j}$, for all robots j, as well as odometry and measurements up to k_c

$$
\begin{aligned}
&C(k_c, k_e) \text{ exists} \\
&\Rightarrow C_p(k_c, k_{e,i}) \text{ exists} \qquad (\forall i) \\
&\Rightarrow S_{i,k_{e,i}} \supseteq S_{j,k_c} \qquad (\forall i, \forall j) \\
&\Rightarrow S_{i,k_{e,i}} \supseteq \begin{cases} \{\mathsf{bel}(\mathbf{x}_{j,k_{s,j}}), \mathbf{u}_{j,k_{s,j}+1:k_c}\ Y_{j,k_{s,j}+1:k_c}\} & \\ \qquad\qquad (\forall i, \forall j), \text{ if } (k_{s,j} < k_c) & \\ \{\mathsf{bel}(\mathbf{x}_{j,k_{s,j}})\} \qquad (\forall i, \forall j), \text{ if } (k_{s,j} = k_c). & \end{cases}
\end{aligned}
$$

It is of interest to know how the knowledge sets of robots that are receiving information from robot m will change once robot m applies the Markov property. Again using $\xrightarrow{\text{path}}$ to denote the existence of a path on the information flow graph, let

$$
Q = \left\{ \text{all robots } i | v(m, k_{e,m}) \xrightarrow{\text{path}} v(i, k_{e,i}) \right\}
$$

and let $\overline{Q} = N - Q$. Now, if we suppose that robot m has applied the Markov property at $k_{e,m}$, then $S_{m,k_{e,m}} \supseteq \mathsf{bel}(\mathbf{x}_j, k_c)(\forall j)$. Furthermore, all robots in Q will also obtain this belief in their knowledge set

$$
\begin{aligned}
S_{i,k_{e,i}} \supseteq &\begin{cases} \{\mathsf{bel}(\mathbf{x}_{j,k_c})\} \qquad (\forall i, \forall j \in Q) & \\ \{\mathsf{bel}(\mathbf{x}_{j,k_{s,j}}), \mathbf{u}_{j,k_{s,j}+1:k_c}, Y_{j,k_{s,j}+1:k_c}\} & \\ \qquad\qquad (\forall i, \forall j \in \overline{Q}), \text{ if } (k_{s,j} < k_c) & \\ \{\mathsf{bel}(\mathbf{x}_{j,k_{s,j}})\} \qquad (\forall i, \forall j \in \overline{Q}), \text{ if } (k_{s,j} = k_c) & \end{cases} \\
&\Rightarrow S_{i,k_{e,i}} \supseteq S_{j,k_c} \qquad (\forall i, \forall j) \\
&\Rightarrow C_p(k_c, k_{e,i}) \text{ exists} \qquad (\forall i).
\end{aligned}
$$

For robots in $\overline{Q}$, their knowledge sets will contain the same information as before the Markov property was applied by robot m. Regardless, each of the three cases for $S_{i,k_{e,i}}$ shown earlier allows us to conclude that by definition, a partial checkpoint exists for all robots i. Fig. 5 is an illustration of this theorem. ■

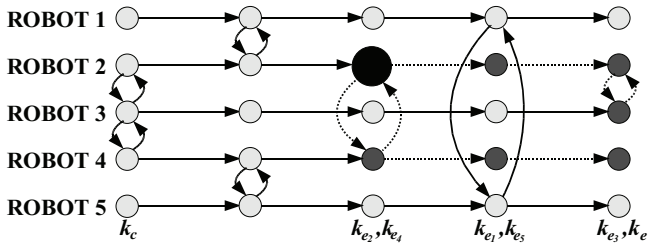


Fig. 5. Theorem 3.1. $C(k_c, k_e)$ exists, and robot 2 applies the Markov property at k_{e_2} when $C_p(k_c, k_{e_2})$ exists (black node). The set of robots in Q are the darker shaded nodes. The robots represented by these nodes will receive $\mathsf{bel}(X_{k_c})$, which is calculated by robot 2. Robots in $\overline{Q}$ are represented by the lighter shaded nodes. The dotted arcs indicate the information flow paths between these nodes and the robot that applied the Markov property.

The implication of this theorem is significant because we are now certain that a robot's decision to invoke the Markov property as soon as its partial checkpoint exists will have no effect on the other robots' abilities to obtain a partial checkpoint (that occurs for the same timestep k_c). Hence, all robots only need to consider their local knowledge when applying the Markov property. Note also the possibility for a robot to communicate a state estimate that it has calculated. This allows other robots to skip the redundant calculation for the same estimate. We will now show that the centralized state estimate is obtainable by all robots regardless of when each robot applies the Markov property.

Theorem 3.2: Suppose that $(\forall i)$, robot i applies the Markov property when $C_p(k_{c,i}, k_{e,i})$ exists (detected using Theorem 2.2). Then, $(\forall C(k_c, k_e)), S_{i,k_{e,i}} \supseteq \{\mathsf{bel}(X_{k_c})\}(\forall i)$, where $k_{e,i} \leq k_e$, and $\mathsf{bel}(X_{k_c})$ is the centralized state estimate.

Proof: Whenever there exists a checkpoint, we know according to Lemma 3.1 that partial checkpoints occurring at the same time also exist for all robots. We also know from Theorem 3.2 that regardless of when the Markov property is applied by each robot, partial checkpoints will always exists for all robots, and therefore, all robots are able to apply the Markov property in any order. Hence, the first line of the following mathematical statements is true before and after the Markov property is applied by all robots:

$$(\forall C(k_c, k_e)), C_p(k_c, k_{e,i}) \text{ exists} \qquad (\forall i)(k_{e,i} \leq k_e)$$
$$\Rightarrow (\forall C(k_c, k_e)), S_{i,k_{e,i}} \supseteq \{\mathsf{bel}(X_{k_c})\} \qquad (\forall i)(k_{e,i} \leq k_e).$$

When robot i applies the Markov property at $k_{e,i}$, the centralized state estimate will replace the equivalent information in its knowledge set (up to time k_c). When all robots have applied the Markov property, all robots will have the centralized state estimate. Furthermore, we are certain that this will occur at $k_e = \max_i k_{e,i}$. ■

With the previous theorem, not only are we certain that robots can apply the Markov property based on local knowledge without affecting the ability for others to do so, we are also certain now that all robots are able to obtain the centralized state estimate if each robot applies the Markov property whenever possible. Cyclic updates will never occur because 1) the knowledge set update rules ensure that there is no repeating data in a knowledge set, and 2) each robot will know whether an estimate it has is equivalent to the centralized estimate (which is never updated again). These important results will be used to develop our decentralized state-estimation algorithm.

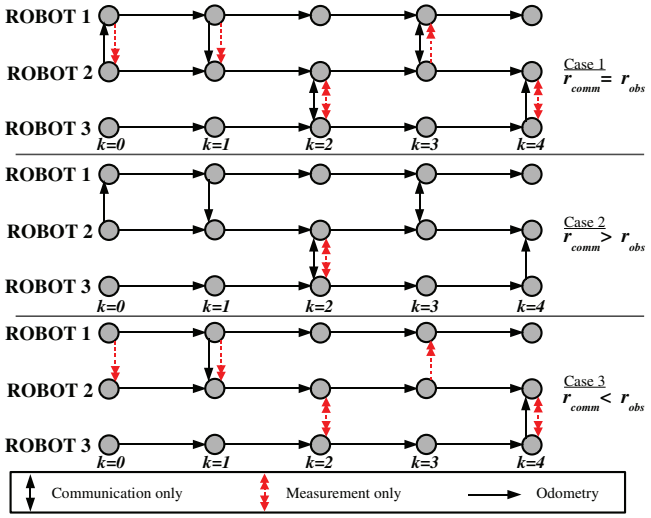


Fig. 6. Even when communication and measurement ranges are different or unidirectional, Theorems 2.1 or 2.2 allow us to correctly determine when each robot can apply the Markov property to obtain the centralized state estimate.

C. Communication and Measurement Ranges

Up to this point, we have let $r_{\mathsf{comm}} = r_{\mathsf{obs}}$, and implicitly assumed that communication and measurements are bidirectional. The applicability of the theorems presented remains the same regardless of whether communication range is different from measurement range and whether they are unidirectional or bidirectional. This is because using Theorem 2.1 or 2.2, we ensure that each robot will apply the Markov property if and only if it has all the information necessary to calculate the centralized estimate at the partial checkpoint occurrence time. Fig. 6 is an example illustrating this fact. In case 1, $r_{\mathsf{comm}} = r_{\mathsf{obs}}$ as we have previously assumed. Using the theorems that we have developed, robots 2 and 3 will both find $C_p(1, 2)$, then robots 1 and 2 will find $C_p(2, 3)$. Finally, robot 2 will find $C_p(3, 4)$. In all partial checkpoint instances, it can be verified that all robots have gathered the required information to calculate the centralized estimate, regardless of whether communication and measurements occur unidirectionally or bidirectionally. In case 2 ($r_{\mathsf{comm}} > r_{\mathsf{obs}}$), most of the measurements seen from the previous case do not occur. Still, all partial checkpoint instances are identical to case 1, and it can again be verified that the centralized estimates can be calculated at partial checkpoint occurrence times. In case 3 ($r_{\mathsf{comm}} < r_{\mathsf{obs}}$), most of the communication instances seen from case 1 do not occur, and it is not possible for robots 1 and 3 to calculate the centralized estimate at the timesteps shown. $C_p(1, 4)$ is the only partial checkpoint that occurs for robot 2, which is the same conclusion that Theorems 2.1 or 2.2 will provide when they are applied.

V. Decentralized State-Estimation Algorithm

The theoretical development in the previous section provides the basis for developing our decentralized state-estimation algorithm. We will first discuss the topic of initial conditions, and show one of the scalable aspects of our method. This is followed by the detailed explanation of our decentralized state-estimation algorithm and a look into computational complexity.

A. Initial Conditions

For a system of robots, we have assumed that each robot initially only has a belief of its own state. Hence, each robot is only aware of its own existence, and a robot will only know of the existence of another robot at first contact (i.e., when communication or a measurement is made between the robots). Correlation between the estimates on the states of two robots is generated through relative measurements. When the estimates of the states of all robots are correlated, any odometry and measurement data for a single robot will not only influence the belief over the individual robot's state but the belief over all robot states as well. It is precisely this reason why there is the need for the notion of a checkpoint: so that all odometry and measurement data are accounted for to produce decentralized state estimates equivalent to those produced by a centralized state estimator.

Due to the independence of state estimates before an encounter, individual (or a group of) robots can be treated as independent subsystems. In other words, the system of all robots can be decoupled into smaller subsystems and into $|N|$ subsystems (of individual robots) at the initial timestep.

Suppose Q_1 and Q_2 are two sets of robots that have never encountered each other before, and let $\mathsf{bel}(X_{Q_1,k})$ and $\mathsf{bel}(X_{Q_2,k})$ represent the beliefs over the states of the two groups, respectively. Before accounting for any measurements between the two groups, statistical independence allows the state estimate for the combined system to be written as

$$\mathsf{bel}(X_{Q_1,k}, X_{Q_2,k}) = \mathsf{bel}(X_{Q_1,k})\mathsf{bel}(X_{Q_2,k}). \tag{3}$$

To implement this, we will always combine state estimates made at the same timestep if they are found within a knowledge set using (3). Measurements can then be processed to couple and update the states in the combined state estimate. This seemingly simple process contributes to the scalability aspect of our decentralized state-estimation algorithm, in the sense that it is unnecessary for each robot to initially know how many robots there are. However, this is only possible when communication is bidirectional and when the communication range limit is greater than or equal to the measurement range limit (i.e., the coupling of state estimates is detectable). Otherwise, each robot will initially need to know the total number of robots on the team.

B. Algorithm

Algorithm 1 is designed to perform decentralized state estimation in a scalable manner and is guaranteed to work in a dynamic mobile-robot network wherein connectivity between all robots is not guaranteed. The same algorithm is implemented on each robot and iterates every timestep. The required inputs are the current timestep k, odometry data $\mathbf{u}_{i,k}$, measurements $Y_{i,k}$, the latest knowledge set $S_{i,k-1}$, and the knowledge sets of all robots (to which information exchange is possible at the current timestep), $S_{j,k}$ $(\forall j \in R_{i,k})$. Fig. 7 is a graphical overview of the algorithm. Each robot first updates its knowledge set with the current odometry and measurements. If other robots are within

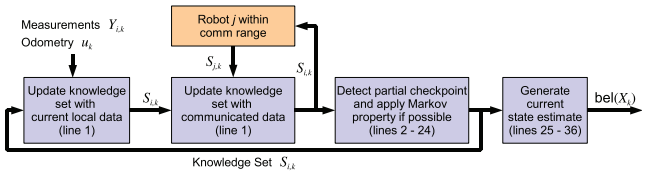


Fig. 7. Iteration of the decentralized state-estimation algorithm, showing how a knowledge set is updated at timestep k, as well as the calculation of the current belief. Line numbers correspond to those in Algorithm 1.

Algorithm 1: DecentralizedStateEst(k, $\mathbf{u}_{i,k}$, $Y_{i,k}$, $S_{i,k-1}$, $S_{j,k}$ $(\forall j \in R_{i,k})$)

1 $S_{i,k} \leftarrow S_{i,k-1} \cup \{\mathbf{u}_{i,k}\} \cup \{Y_{i,k}\} \cup \{S_{j,k}\}(\forall j \in R_{i,k})$
2 $N_{i,k} \leftarrow \{Q | (\forall \mathsf{bel}(\mathbf{X}_{Q,k_s}) \in S_{i,k}), (k_s \leq k)\}$
3 **repeat**
4 $\quad \mathsf{flag}_{\mathsf{repeat}} = \mathsf{false}$
$\quad \{k_{s_1}, Q_1\} \leftarrow$ find smallest k_{s_1} such that
5 $\quad\quad \mathsf{bel}^*(X_{Q_1}, k_{s_1}) \in S_{i,k}$
6 $\quad k_{s_2} \leftarrow k$
7 $\quad$ **if** $Q_1 \neq N_{i,k}$ **then**
$\quad\quad \{k_{s_2}, Q_2\} \leftarrow$ find smallest k_{s_2} such that
8 $\quad\quad\quad \mathsf{bel}^*(X_{Q_2}, k_{s_2}) \in S_{i,k}$ $(Q_1 \neq Q_2)$
9 $\quad$ **end**
10 $\quad$ **for** $k_c \leftarrow k_{s_2} : k_{s_1}$ **do**
11 $\quad\quad$ **if** $U_{Q_1,k_c} \in S_{i,k}$ ***or*** $k_c = k_{s_1}$ **then**
12 $\quad\quad\quad \tilde{S}_{i,k_c} \leftarrow S_{i,k} - \{U_{Q_1,k_r}, Y_{Q_1,k_r}\}$ $(\forall k_r > k_c)$
13 $\quad\quad\quad \mathsf{bel}^*(X_{Q_1,k_c}) \leftarrow p\left(X_{Q_1,k_c} | \tilde{S}_{i,k_c}\right)$
14 $\quad\quad\quad S_{i,k} \leftarrow S_{i,k} \cup \mathsf{bel}^*(X_{Q_1,k_c})$
$\quad\quad\quad S_{i,k} \leftarrow S_{i,k} -$
15 $\quad\quad\quad\quad \{U_{Q_1,k_r}, Y_{Q_1,k_r}, \mathsf{bel}^*(X_{Q_1,k_r})\}$ $(\forall k_r \leq k_c)$
16 $\quad\quad\quad$ break
17 $\quad\quad$ **end**
18 $\quad$ **end**
19 $\quad$ **if** $\{\mathsf{bel}^*(X_{Q_1,k_c}), \mathsf{bel}^*(X_{Q_2,k_c})\} \in S_{i,k}$ **then**
20 $\quad\quad \mathsf{bel}^*(X_{Q_3,k_c}) \leftarrow \mathsf{bel}^*(X_{Q_1,k_c})\,\mathsf{bel}^*(X_{Q_2,k_c})$
$\quad\quad S_{i,k} \leftarrow S_{i,k} \cup \mathsf{bel}^*(X_{Q_3,k_c}) -$
21 $\quad\quad\quad \{\mathsf{bel}^*(X_{Q_1,k_c}), \mathsf{bel}^*(X_{Q_2,k_c})\}$
22 $\quad\quad \mathsf{flag}_{\mathsf{repeat}} = \mathsf{true}$
23 $\quad$ **end**
24 **until** $\mathsf{flag}_{\mathsf{repeat}} = \mathsf{false}$
$\{k_{s_1}, Q_1\} \leftarrow$ find smallest k_{s_1} such that
25 $\quad \mathsf{bel}^*(X_{Q_1,k_{s_1}}) \in S_{i,k}$
26 **while** $Q_1 \neq N_{i,k}$ **do**
27 $\quad \{k_{s_2}, Q_2\} \leftarrow$ find smallest k_{s_2} such that
28 $\quad \mathsf{bel}(X_{Q_2}, k_{s_2}) \in S_{i,k}$ $(Q_1 \neq Q_2)(k_{s_1} \leq k_{s_2})$
29 $\quad \tilde{S}_{i,k_{s_2}} \leftarrow S_{i,k} - \{U_{Q_1,k_u}, Y_{Q_1,k_r}\}$ $(\forall k_u > k_{s_2})$
30 $\quad \mathsf{bel}\left(X_{Q_1,k_{s_2}}\right) \leftarrow p\left(X_{Q_1,k_{s_2}} | \tilde{S}_{i,k_{s_2}}\right)$
31 $\quad \mathsf{bel}\left(X_{Q_1,k_{s_2}}, X_{Q_2,k_{s_2}}\right) \leftarrow \mathsf{bel}\left(X_{Q_1,k_{s_2}}\right) \mathsf{bel}\left(X_{Q_2,k_{s_2}}\right)$
32 $\quad Q_1 \leftarrow Q_1 \cup Q_2$
33 $\quad k_{s_1} \leftarrow k_{s_2}$
34 **end**
35 $\mathsf{bel}(X_{N_i,k}) \leftarrow p(X_{N_i,k} | S_{i,k})$
36 **return** $\{\mathsf{bel}(X_{N_i,k}), S_{i,k}, N_{i,k}\}$

communication range, new information from other robots are appended to the knowledge set. Within this updated knowledge set, we apply Theorem 2.2 to detect partial checkpoints and apply the Markov property at the partial checkpoint time if possible. Finally, the current state estimate is generated. Essentially, each robot is running its own centralized state estimator on the available information from the partial checkpoint timestep to the current timestep. Note that in most cases, it is not possible to reuse the current state estimate at a later time to generate the centralized state estimate for the same timestep. This is due to the OOSM problem mentioned in Section II.

At the very first iteration of our algorithm, we assume that each robot initially only has a state estimate of itself in its knowledge set. On line 1, we update the knowledge set of robot i by implementing (1) and (2). Line 2 determines the set of all robots known to i by looking for part beliefs $\mathsf{bel}^*(X_{Q,k_s})$ in $S_{i,k}$, where Q represents a set of robots. Note that bel^* indicates a belief that is equivalent to the state estimate obtainable using a centralized state estimator. The loop beginning on line 3 repeats according to the flag variable set on line 4. The purpose of this loop is to systematically combine multiple beliefs for independent subgroups of robots in $S_{i,k}$. To do this, on line 5, we search for the earliest state estimate $\mathsf{bel}^*(X_{Q_1}, k_{s_1})$ in the knowledge set. If $Q_1 = N$, then we already have the belief over all known robots. Otherwise, we search for the next earliest estimate $\mathsf{bel}^*(X_{Q_2}, k_{s_2})$ on line 8. The intention here is to calculate $\mathsf{bel}^*(X_{Q_1}, k_{s_2})$ so that the beliefs over the two groups can be combined.

The search for a partial checkpoint (for system Q_1) begins with the "for" loop on line 10. If $Q_1 = N$, we will attempt to look for a partial checkpoint that is closest to the current timestep, and this is why k_{s_2} is initially set equal to k on line 6. Otherwise, we will attempt to find a partial checkpoint at k_{s_2}. If k_{s_1} and k_{s_2} are the same, the partial checkpoint search is skipped and we proceed directly to line 19 and combine the estimates found on lines 5 and 8. Line 11 uses Theorem 2.2 to detect the existence of a partial checkpoint. If one is found, we use the knowledge up to the partial checkpoint time determined on line 12 to obtain the state estimate on line 13. The new estimate is entered into the knowledge set on line 14, and we proceed to discard information replaceable by bel^* on line 15. On line 16, we break out of the partial checkpoint search since one has been found.

Line 19 checks if there are two estimates for the same timestep in the knowledge set. If a pair is found, we proceed to combine them according to (3) on line 20, and update the knowledge set on line 21. The repeat flag defined on line 4 is made true here because there may be beliefs for other subgroups of robots in the knowledge set that can be combined (i.e., in the next iteration of the line 3–24 loop, the newly combined belief on line 20 becomes the belief we will find on line 5, and we will attempt to find the next earliest belief for another subgroup on line 8). After searching for partial checkpoints, we turn our attention to determine the state estimate for the current timestep. Again, multiple estimates (for independent groups of robots at different timesteps) may still exist in the knowledge set. We again begin with the earliest state estimate in the knowledge set on line 25, and aim to produce a state estimate at the time of the next earliest estimate (line 27) in the knowledge set so that the two can be combined. This process repeats in the loop between lines 26 and 33 until we have a single state estimate over the states of all known robots. The current state estimate is then determined on line 35, which is based on the estimate at the last partial checkpoint and any information since then to the current time. Hence, it uses all the information available and is the best estimate that can be produced at the current time. In calculating this current estimate, we may not have all the information required to make an estimate that is equivalent to the centralized state estimator (i.e., a partial checkpoint does not exist). In this situation, we assume the last known velocity for robots from which we do not have odometry data, but this is only temporary.

There are some important points to highlight about Algorithm 1. First, as mentioned already, it is unnecessary for a robot to initially know how many robots there are in the system. This is the scalable aspect of our algorithm. Second, since our algorithm is based on the information-flow graph, the sequence of communication between multiple robots or delays in communication can be handled naturally without any changes to our algorithm. This is one of the practical advantages with our framework. Third, any recursive-filtering method can be used on lines 13, 30, and 35. To give a few examples, the *EKF*, the *unscented Kalman filter*, and the *particle filter* could be used in our decentralized state-estimation framework. Thus, the algorithm is very general and is widely applicable in any situation in which there is a need to perform decentralized state estimation in a dynamic network. Furthermore, an equivalent to the centralized state estimate can always be calculated at the time of a partial checkpoint. Although the state estimate at the current time may be suboptimal due to missing information, the equivalent centralized state estimate is guaranteed to be obtainable later. For the moment, a robot assumes that another robot maintains its last known velocity until its odometry is known. We are working on incorporating motion planning information to produce more accurate motion predictions when robots are not connected. Finally, note that the cyclic update problem is never encountered.

C. Complexity

Since we are exploiting the Markov property, computation and memory usage are limited provided that all robots are able to detect partial checkpoints in the future, which is a reasonable assumption if their task is to cooperatively localize. The computational complexity of Algorithm 1, its memory usage, as well as communication bandwidth requirements will vary depending on the number of timesteps since the previous partial checkpoint $k - k_c$, the number of states that need to be estimated n (which is proportional to the number of robots $|N|$), and the filtering method selected for use within Algorithm 1. The frequency at which partial checkpoints occur depends on factors such as communication range as well as the size of the workspace. Fig. 8 illustrates the worst-case scenario for a four-robot team. The scenario shows that while relative measurements are made between all robots at all timesteps, information exchange does

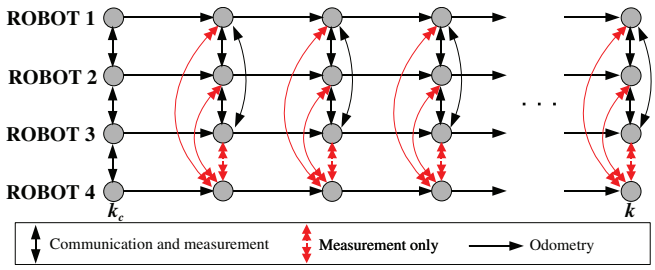


Fig. 8. Worst-case scenario in terms of memory usage, computation, and communication.

not occur with one of the robots (Robot 4). This may occur if Robot 4 or its communication hardware fails. Hence, new partial checkpoints will not come into existence for all the robots, and information will continue to accumulate in each robot's knowledge set. This will continue until communication is reestablished to Robot 4.

Assume that the EKF is used in the previous scenario. Since $n = c|N|$, where c is a constant representing the number of states per robot, we can simplify this complexity analysis by letting $c = 1$ without affecting the end result. In a centralized estimator (where all robots can exchange information at all timesteps), there are $(n^2 - n)$ measurements at each timestep since every robot makes a measurement of every other robot. Assuming that the dimension of each measurement is constant (i.e., all measurements always provide only range and bearing information), and we process measurements sequentially, the computational complexity of processing each measurement is $O(n^2)$ (from the Kalman gain calculation and the covariance update steps). With $(n^2 - n)$ measurements, the overall computational complexity is therefore $O(n^4)$ for the centralized estimator. Furthermore, storing the covariance matrix and measurements requires memory usage of $O(n^2)$.

Using the EKF in our decentralized estimator, memory usage will be of $O((k - k_c)n^2)$ for each robot in the worst-case scenario. This is because it is necessary to keep all information in the knowledge set that comes after k_c and up to the current timestep k. For the worst-case scenario shown, at every timestep, all robots except Robot 4 communicate with each other, passing on measurement and odometry information accumulated since the last partial checkpoint to $n - 2$ robots (remember that $n = |N|$). This makes the communication bandwidth requirement of $O((k - k_c)n^3)$ for each robot. The computational complexity of calculating the current state estimate is $O((k - k_c)n^4)$ for each robot, but only when a new partial checkpoint is discovered since the calculation must then be performed from the last partial checkpoint time. Otherwise, by knowing the state estimate from the previous timestep $(k - 1)$, computational complexity of calculating the current state estimate is $O(n^4)$. In general, the difference in computational complexity from the centralized approach (i.e., the $k - k_c$ factor) is the result of network connectivity (i.e., the cost of operating in a dynamic and sparse network).

In practice, performing state estimation at high frequency may cause difficulties in real-time implementations as the number of timesteps since the previous partial checkpoint $(k - k_c)$ increases at a high rate. A practical solution may be to aggregate odometry data between measurements to effectively increase the size of each timestep. It is important to note that in our simulations, we found that the worst-case scenario of Fig. 8 rarely occurs for a prolonged number of timesteps. Recall also from Theorem 3.1 that the ability for robots to pass on state estimates will reduce the need for other robots to perform the calculations for the same state estimate, hence reducing computational cost. It is worth mentioning that if robots keep track of the information they have already sent to other robots, redundancy in communication can be reduced (in exchange for increase in memory usage). Furthermore, robots consistently in contact with each other can form a subgroup and temporarily store their current state estimate to reduce the amount of computation required for the next timestep (assuming that no past information is added to their knowledge sets). We plan to address these extensions in our future work. We also plan to look at robot failure cases to decide when it no longer becomes feasible to maintain the centralized estimate.

VI. Simulations

In the theoretical development of a checkpoint, we showed that a state estimate equivalent to the centralized estimate can be reached by all robots when a checkpoint exists (or by a particular robot when a partial checkpoint exists). It is of interest to compare the performance of the proposed decentralized state estimation algorithm against a centralized state estimator. For this purpose, simulations were performed for a group of uniquely identifiable robots moving in a workspace in which each robot does not initially know the total number of robots in the team. The intention of the simulation is to have each robot estimate the states of all robots known to itself (shown in Fig. 1). Communication range and measurement range are set equal and are limited so that the robots are in a dynamic network that is not always fully connected. Note, the centralized state estimator would simply not work under these assumptions, but we allow robots the ability to always communicate with each other at all timesteps for the centralized estimator so that estimates can actually be made for comparison.

A. Setup

The *EKF* algorithm [27] is used as the filtering method on lines 13, 20, and 35 of Algorithm 1. Note that many other recursive filtering methods can be applied. The state of each robot includes position (x, y) and orientation θ. A discrete-time unicycle model [27] is used for state transition for each robot, where the inputs (odometry data) are translational and angular velocities (v, ω). The two inputs are assumed to be corrupted by independent zero-mean Gaussian noise. When robot i observes another robot j within range r_{obs}, it is able to measure the range $r^{j,i}$, as well as the bearing $\phi^{j,i}$, of robot j with respect to robot i. Each measurement component is assumed to contain independent zero-mean additive Gaussian noise. The robot starts with a random pose and an estimate of that pose in its knowledge set, and each robot will move using the same visual servoing control law [28] to random waypoints in the bounded workspace.

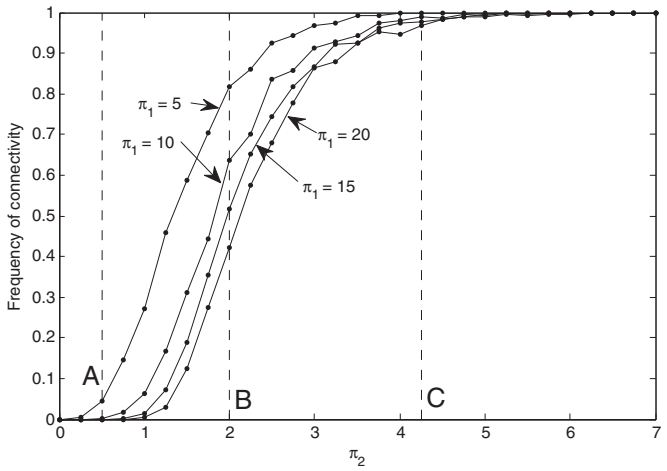


Fig. 9. Frequency of connectivity observed for different numbers of stationary robots.

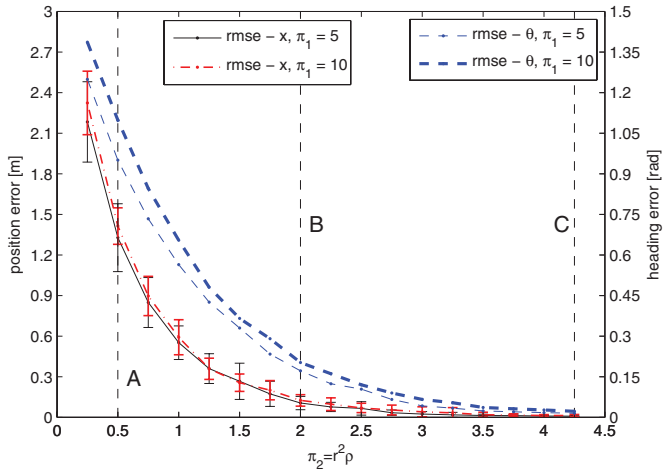


Fig. 10. Average error between decentralized and centralized state estimates.

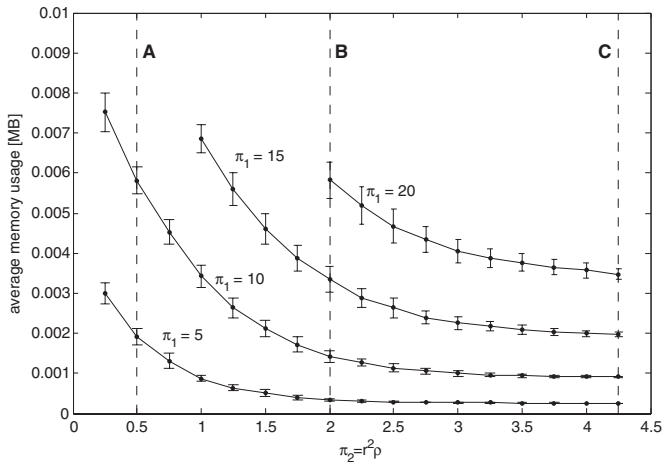


Fig. 11. Average memory usage of a robot for decentralized state estimation.

B. Results

We postulate that the number of robots $|N|$, the size of the workspace A (or robot density ρ), and communication range r_{comm} are all the major factors that will influence the connectivity of the robot network and localization performance. Therefore, it is necessary to test if and how simulation results are affected by varying these parameters. To reduce the dimensionality of this analysis, Buckingham's Pi theorem [29] is applied to generate the dimensionless variables

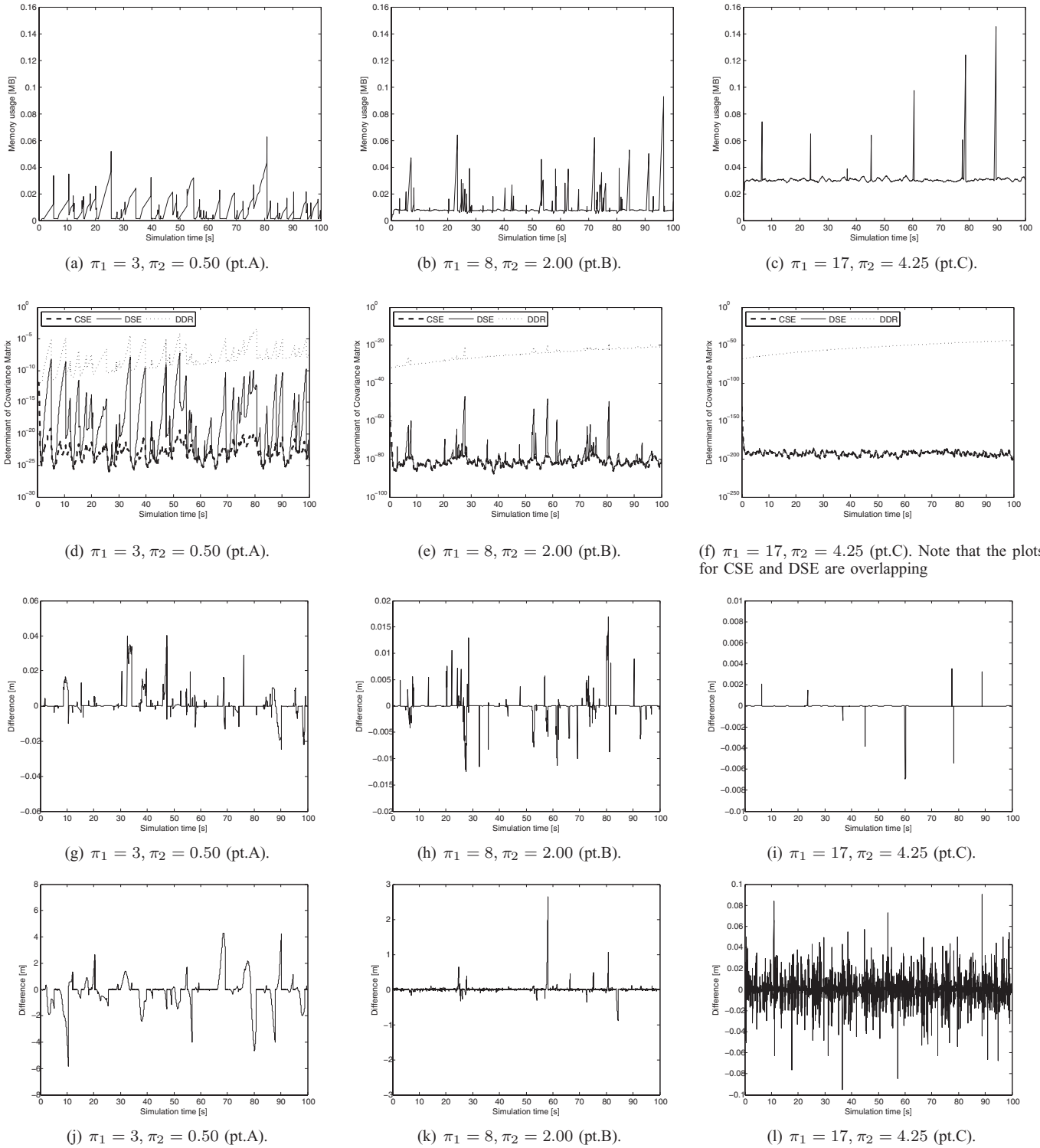
$$\pi_1 = |N|$$
$$\pi_2 = \frac{|N|}{A} r_{\text{comm}}^2 = \rho r_{\text{comm}}^2.$$

Since we have three variables ($|N|$, A, and r_{comm}), and the units of these contain only one fundamental quantity (distance), this enables us to use $3-1$ dimensionless parameters to analyze the effects of each of the three variables on localization performance.

There have been many studies on network connectivity in the past for random graphs. A network is *connected* at a given timestep if there exists a path between every pair of nodes. This is true on the information-flow graph if at timestep k a path exists from $v(i,k)$ to $v(j,k)$ $(\forall i,j)$. Furthermore, we define the *frequency of connectivity* as the percentage of time that a robot network is connected. This is shown in Fig. 9 for different numbers of (stationary) robots randomly populated in a workspace 1000 times to obtain an average at each data point. The trend observed also applies to moving robots [30], and we have also confirmed this through simulation.

The zero-to-one transition phenomenon observed in Fig. 9 is typical for *Bernoulli random graph models* wherein connections between robots are determined based on frequency (and not distance) [31], as well as the *fixed-radius random graph model* used for generating our robot network [32]. Furthermore, we note that as π_1 increases, the π_2 value at which phase transition occurs, as well as the steepness of the transition also increases to approach an asymptotic curve. This corresponds to the findings in [33].

This network connectivity phase transition is important in providing us with an indication of how our decentralized state-estimation algorithm performs. Low π_2 values can be interpreted as low robot density or short communication range. Under these conditions, robots are rarely in contact with each other and infrequently establish partial checkpoints. Conversely, high π_2 values (after the phase transition) correspond to high robot density or long communication range. Under these conditions, robots are frequently in contact with each other, establishing partial checkpoints. The range of π_2 values at which the phase transition occurs is an indication of when the implementation of our decentralized state estimate algorithm becomes advantageous. We will first present overall performance results collected over hundreds of simulation trials. Fig. 10 plots the root-mean-squared error between the decentralized state estimates produced by our algorithm and the centralized state estimates for x and y positions (which are almost overlapping in the figure, and thus, we chose not to show the plots for $\pi_1 = 15$ and $\pi_1 = 20$ for this reason), along with 2-standard-deviation error bars, as well as



(a) $\pi_1 = 3, \pi_2 = 0.50$ (pt.A). (b) $\pi_1 = 8, \pi_2 = 2.00$ (pt.B). (c) $\pi_1 = 17, \pi_2 = 4.25$ (pt.C).

(d) $\pi_1 = 3, \pi_2 = 0.50$ (pt.A). (e) $\pi_1 = 8, \pi_2 = 2.00$ (pt.B). (f) $\pi_1 = 17, \pi_2 = 4.25$ (pt.C). Note that the plots for CSE and DSE are overlapping

(g) $\pi_1 = 3, \pi_2 = 0.50$ (pt.A). (h) $\pi_1 = 8, \pi_2 = 2.00$ (pt.B). (i) $\pi_1 = 17, \pi_2 = 4.25$ (pt.C).

(j) $\pi_1 = 3, \pi_2 = 0.50$ (pt.A). (k) $\pi_1 = 8, \pi_2 = 2.00$ (pt.B). (l) $\pi_1 = 17, \pi_2 = 4.25$ (pt.C).

Fig. 12. Detailed simulation results for simulation trials at various connectivity settings. (a)–(c) Memory usage for robot 1. (d)–(f) Estimation uncertainty for the DSE, the CSE, and DDR. (g)–(i) Difference between the decentralized estimate from robot 1 and the centralized estimate of robot 1's own *x*-position. (j)–(l) Difference between the decentralized estimate from robot 1 and the centralized estimate for robot 2's *x*-position. Note the difference in scale for (g)–(l).

orientation θ. The averaged error at each data point is obtained over 50 simulation trials.

In Figs. 9 and 10, we have identified three specific π_2 values: Point A ($\pi_2 = 0.5$) corresponds with low frequency of connectivity, point B ($\pi_2 = 2.0$) is within the connectivity phase transition region, and point C ($\pi_2 = 4.25$) corresponds with high frequency of connectivity. Over the phase transition, the difference between the decentralized and centralized estimates has reduced drastically. This is where our decentralized state-estimation algorithm begins to show its merits. At point C, connectivity

has a high likelihood of being maintained and the difference in the decentralized and centralized estimates approaches zero. Note that although the performance between the two is almost identical, the centralized state estimator will only work if connectivity is guaranteed at all times.

Memory usage is limited since our decentralized state-estimation algorithm makes use of the Markov property. Actual memory usage will depend on how many robots there are in the network (i.e., the number of states to estimate) and the frequency of partial checkpoint occurrence. Fig. 11 is a plot of average memory usage with 2-standard-deviation error bars. Each plotted point represents the averaged results of over 50 simulation runs. At low frequency of connectivity settings, partial checkpoints occur infrequently and each robot must store all its accumulated data in the duration between partial checkpoints, leading to high average memory usage. As π_2 increases, the robot encounters occur more often and the frequency of partial checkpoint occurrence also increases, leading to reductions in memory usage. Note, as we increase the number of robots, memory usage also increases because the number of states that need to be estimated is increasing.

Now we will present detailed results corresponding to single simulation trials at points A, B, and C, beginning first with memory usage for a single robot (robot 1) in Fig. 12(a)–(c). In these figures, momentary increases in memory usage occur as a robot accumulates information between partial checkpoints and reduces when the Markov property is applied. The characteristics and the frequency of these fluctuations were observed to be dependent on connectivity settings. Note that at high frequency of connectivity, memory usage approaches that of a centralized estimator. Using these three graphs, we can also get a sense of computation and communication requirements. As memory usage increases, more information is stored in the knowledge set, which needs to be communicated and processed during state estimation. However, note that we rarely experience the worst-case scenario shown in Fig. 8, and even in the low-frequency-of-connectivity case, the timesteps between partial checkpoints are limited.

The difference between the overall uncertainty of robot 1's decentralized state estimate and that of the centralized state estimate can be observed in Fig. 12(d)–(f). The determinant of the estimation error covariance (which is proportional to the volume of the uncertainty ellipsoid) are shown in these plots for the centralized state estimator (CSE), our decentralized state estimator (DSE), and decentralized dead-reckoning (DDR), which only uses odometry data. As the frequency of connectivity increases, uncertainty for the decentralized state estimate approaches that of the centralized state estimate. Note that because there are no stationary landmarks, and since all robots are always in motion, the error covariance gradually inflates over time. In our tests, we have also simulated the Cramér–Rao lower bound for uncertainty by evaluating the Jacobians used in the EKF at the true states [34]. This theoretical lower bound for uncertainty is confirmed to be slightly lower than that of the centralized state estimator, providing evidence that both the centralized and decentralized estimators that have been implemented are not overconfident.

Next, we will show the difference between the mean of the decentralized state estimates, and that of the centralized state estimates. Since there are a large number of states being estimated (by each robot), we elected to only show a portion of these. For each trial, we will show the difference for robot 1's own x-position estimates [see Fig. 12(g)–(i)] and that of robot 2 [see Fig. 12(j)–(l)]. Similar results are observed for all other robots' poses. It is evident that as the frequency of connectivity increases, the difference between the decentralized and centralized state estimates decreases. Furthermore, the difference for a robot's estimate for itself is always closer to the centralized estimate compared to its estimate of another robot. This is because a robot is always aware of its own odometry and measurements but not necessarily that of another robot, depending on the evolving network topology.

It can be seen how the detailed results presented earlier follow the overall trend in localization performance and memory usage observed in Figs. 10 and 11. Once again, remember that centralized state estimation is not possible when full connectivity is not guaranteed between robots at each timestep, and the decentralized state estimator presented here is proven to allow the equivalent centralized state estimate to be reached under a dynamic network where robots sporadically communicate. At no time does the network need to be fully connected.

VII. Conclusion

An algorithm was presented that allows state estimation to be performed in a dynamic robot network, in which full connectivity is not assumed. We defined checkpoints and partial checkpoints, which are events during which a state estimate equivalent to the centralized estimate can be made by the decentralized state estimator, implying that the estimate is based on all past information. We have also introduced theorems to allow checkpoints and partial checkpoints to be detected in a practical manner. The proposed method is scalable in the sense that the number of robots in the network does not need to be known initially, but only when communication is bidirectional, and when the communication range limit is greater than the measurement range limit. Furthermore, we have shown through simulations that memory usage (although large compared to the centralized estimator) is limited by exploiting the Markov property at partial checkpoints. In our simulations, we also looked at how various factors captured by dimensionless variables affect performance of our decentralized state estimator in comparison to the centralized estimator, and how this relates to network connectivity. Results show that the performance of our decentralized state estimator begins to approach that of the centralized state estimator when a phase transition occurs in the frequency of network connectivity. It must be stressed again that a centralized state estimator will not be able to produce an estimate unless we assume full network connectivity at all times.

The natural extension of this research (already in progress) is to look at decentralized SLAM under the same assumptions on the network (i.e., include landmarks) and to conduct experiments with real robots. Furthermore, we are extending our algorithm to accommodate the situation in which a robot that is previously

part of the network fails or leaves the group permanently. Finally, we would like to incorporate decentralized planning and achieve a method of active SLAM for a dynamic network of mobile robots.

APPENDIX

Proof of Theorem 2.1: We will first assume that $C_p(k_{c,i}, k_{e,i})$ exists. This implies that all odometry measurements and state estimates from all robots at $k_{c,i}$ are in the knowledge sets of robots i at $k_{e,i}$. Clearly, this is only possible if there exists an information flow path between $v(i, k_c)$ to $v(j, k_e)$ $(\forall j)$.

Next, we assume that a path exists between $v(i, k_c)$ to $v(j, k_e)$ $(\forall j)$. By the knowledge set update rules, all odometry measurements and state estimates from all robots at $k_{c,i}$ will become incorporated into the knowledge sets of robot i at $k_{e,i}$. Therefore, $C_p(k_{c,i}, k_{e,i})$ exists. ∎

Proof of Lemma 2.1: First, let us assume that $k_{c_1} < k_{c_2}$

$$
\begin{aligned}
&k_{c_1,i} < k_{c_2,i} \\
&\Rightarrow v(j, k_{c_1,i}) \xrightarrow{\text{path}} v(j, k_{c_2,i}) \qquad (\forall j) \\
&\Rightarrow v(j, k_{c_2,i}) \xrightarrow{\text{path}} v(i, k) \qquad (\forall j)(k_{e_2,i} \le k) \\
&\Rightarrow \begin{cases} v(j, k_{c_1,i}) \xrightarrow{\text{path}} v(i, k) & (\forall j)(k_{e_2,i} \le k) \text{ if } (k_{e_1,i} \ge k_{e_2,i}) \\ v(j, k_{c_1,i}) \xrightarrow{\text{path}} v(i, k) & (\forall j)(k_{e_1,i} \le k) \text{ if } (k_{e_1,i} < k_{e_2,i}) \end{cases} \\
&\Rightarrow k_{e_1,i} < k_{e_2,i}.
\end{aligned}
$$

Now, we will assume that $k_{e_1,i} < k_{e_2,i}$

$$
\begin{aligned}
&k_{e_1,i} < k_{e_2,i} \\
&\Rightarrow v(j, k_{c_1,i}) \xrightarrow{\text{path}} v(i, k) \qquad (\forall j)(k_{e_1,i} \le k) \\
&\Rightarrow (k_{c_1,i} \ge k_{c_2,i})\, v(j, k_{c_2,i}) \xrightarrow{\text{path}} v(i, k_{c_1,i}) \qquad (\forall j) \\
&\Rightarrow \begin{cases} v(j, k_{c_1,i}) \xrightarrow{\text{path}} v(i, k) & (\forall j)(k_{e_1,i} \le k) \text{ if } (k_{c_1,i} \ge k_{c_2,i}) \\ v(j, k_{c_1,i}) \xrightarrow{\text{path}} v(i, k) & (\forall j)(k_{e_2,i} \le k) \text{ if } (k_{c_1,i} < k_{c_2,i}) \end{cases} \\
&\Rightarrow k_{c_1,i} < k_{c_2,i}.
\end{aligned}
$$

∎

Proof of Theorem 2.2: Assume that $C_p(k_{c,i}, k_{e,i})$ exists

$$
\begin{aligned}
&S_{i,k_{e,i}} \supseteq S_{j,k_{c,i}} \qquad (\forall j) \\
&\Rightarrow S_{i,k_{e,i}} \supseteq \begin{cases} \{\mathsf{bel}(x_{j,k_{s,j}}), Y_{j,k_{s,j}+1:k_{c,i}}\, \mathbf{u}_{j,k_{s,j}+1:k_{c,i}}\} \\ \quad (\forall j), \text{ if } (k_{s,j} < k_{c,i}) \\ \{\mathsf{bel}(x_{j,k_{s,j}})\} \qquad (\forall j), \text{ if } (k_{s,j} = k_{c,i}) \end{cases} \\
&\Rightarrow S_{i,k_{e,i}} \supseteq \mathbf{u}_{j,k_{c,i}} \text{or } \mathsf{bel}(\mathbf{x}_{j,k_{c,i}}) \qquad (\forall j).
\end{aligned}
$$

Now, assuming that the knowledge set of each robot at $k_{e,i}$ contains $\mathbf{u}_{j,k_{c,i}}$ or $\mathsf{bel}(\mathbf{x}_{j,k_{c,i}})$

$$
\begin{aligned}
&S_{i,k_{e,i}} \supseteq \mathbf{u}_{j,k_{c,i}} \text{or } \mathsf{bel}(\mathbf{x}_{j,k_{c,i}}) \qquad (\forall j) \\
&\Rightarrow S_{i,k_{e,i}} \supseteq S_{j,k} \qquad (\forall j)(k_{c,i} \le k \le k_{e,i}) \\
&\Rightarrow S_{i,k_{e,i}} \supseteq S_{j,k_{c,i}} \qquad (\forall j).
\end{aligned}
$$

∎

REFERENCES

[1] W. Burgard, M. Moors, C. Stachniss, and F. Schneider, "Coordinated multi-robot exploration," *IEEE Trans. Robot*, vol. 21, no. 3, pp. 376–386, Jun. 2005.

[2] T. M. Berg and H. F. Durrant-Whyte, "Distributed and decentralized estimation," in *Proc. Singapore Int. Conf. Intell. Control Instrum.*, 1992, vol. 2, pp. 1118–1123.

[3] T. M. Berg and H. F. Durrant-Whyte, "General decentralized Kalman filters," in *Proc. Amer. Control Conf.*, 1994, vol. 2, pp. 2273–2274.

[4] S. Grime and H. F. Durrant-Whyte, "Data fusion in decentralized sensor networks," *Control Eng. Pract.*, vol. 2, no. 5, pp. 849–863, 1994.

[5] S. Utete and H. F. Durrant-Whyte, "Reliability in decentralised data fusion networks," in *Proc. IEEE Int. Conf. MFI*, 1994, pp. 215–221.

[6] F. Bourgault and H. F. Durrant-Whyte, "Communication in general decentralized filters and the coordinated search strategy," presented at the Int. Conf. Inf. Fusion Conf., Stockholm, Sweden, 2004.

[7] R. Kurazume and S. Hirose, "An experimental study of a cooperative positioning system," *Auton. Robot*, vol. 8, no. 1, pp. 43–52, 2000.

[8] Y. Dieudonne, O. Labbani-Igbida, and F. Petit, "On the solvability of the localization problem in robot networks," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2008, pp. 480–485.

[9] S. I. Roumeliotis and G. A. Bekey, "Distributed multirobot localization," *IEEE Trans. Robot. Autom.*, vol. 18, no. 5, pp. 781–795, Oct. 2002.

[10] E. D. Nerurkar, S. I. Roumeliotis, and A. Martinelli, "Distributed maximum a posteriori estimation for multi-robot cooperative localization," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2009, pp. 1402–1409.

[11] A. Howard, "Multi-robot simultaneous localization and mapping using particle filters," *Int. J. Robot. Res.*, vol. 25, no. 12, pp. 1243–1256, 2006.

[12] R. Madhavan, K. Fregene, and L. E. Parker, "Distributed cooperative outdoor multirobot localization and mapping," *Auton. Robot*, vol. 17, no. 1, pp. 23–39, 2004.

[13] I. M. Rekleitis, G. Dudek, and E. Milios, "Multi-robot cooperative localization: A study of trade-offs between efficiency and accuracy," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2002, pp. 2690–2695.

[14] S. I. Roumeliotis and I. M. Rekleitis, "Propagation of uncertainty in cooperative multirobot localization: Analysis and experimental results," *Auton. Robot*, vol. 17, no. 1, pp. 41–54, 2004.

[15] N. Trawny, S. I. Roumeliotis, and G. B. Giannakis, "Cooperative multi-robot localization under communication constraints," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2009, pp. 4394–4400.

[16] P. Ferguson and J. How, "Decentralized estimation algorithms for formation flying spacecraft," in *Proc. AIAA Conf. Guid. Navigat. Control*, 2003, pp. 1–12.

[17] Y. Bar-Shalom, "Update with out-of-sequence measurements in tracking: Exact solution," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 38, no. 3, pp. 769–777, Jul. 2002.

[18] Y. Bar-Shalom, H. Chen, and M. Mallick, "One-step solution for the multistep out-of-sequence-measurement problem in tracking," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 40, no. 1, pp. 27–37, Jan. 2004.

[19] A. Howard, M. J. Mataric, and G. S. Sukhatme, "Putting the 'i' in 'team': An ego-centric approach to cooperative localization," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2003, pp. 1–7.

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

[21] T. D. Barfoot and G. M. T. D'Eleuterio, "Evolving distributed control for an object clustering task," *Complex Syst.*, vol. 15, no. 3, pp. 183–201, 2005.

[22] I. D. Schizas, A. Ribeiro, S. I. Roumeliotis, and G. B. Giannakis, "Consensus in ad hoc wsns with noisy links—part I: Distributed estimation of deterministic signals," *IEEE Trans. Signal Process*, vol. 56, no. 1, pp. 350–364, Jan. 2008.

[23] L. Moreau, "Stability of multiagent systems with time-dependent communication links," *IEEE Trans. Autom. Control*, vol. 50, no. 2, pp. 169–182, Feb. 2005.

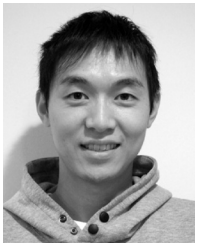
[24] K. Y. K. Leung, T. D. Barfoot, and H. H. T. Liu, "Decentralized localization for dynamic and sparse robot networks," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2009, pp. 1–7.

[25] Z. Brzeźniak and T. Zastawniak, *Basic Stochastic Processes: A Course Through Exercise*. New York: Springer-Verlag, 1999.

[26] A. H. Jazwinsky, *Stochastic Processes and Filtering Theor*. New York: Academic, 1970.

[27] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotic*. Cambridge, MA: MIT Press, 2005.

[28] R. Siegwart and Nourbakhsh, *Introduction to Autonomous Mobile Robot*. Cambridge, MA: MIT Press, 2004.
[29] E. Buckingham, "On physically similar systems; illustrations of the use of dimensional equations," *Phys. Rev.*, vol. 4, no. 4, pp. 345–376, 1914.
[30] M. Sanchez, P. Manzoni, and Z. J. Haas, "Determination of critical transmission range in ad-hoc networks," in *Proc. Multiaccess, Mobility Teletraffic Wireless Commun.*, 1999, pp. 1–11.
[31] B. Bollobas, *Random Graph*. New York: Academic, 1985.
[32] B. Krishnamachari, S. B. Wicker, and R. Bejar, "Phase transition phenomena in wireless ad hoc networks," in *Proc. IEEE Global Telecommun. Conf.*, 2001, pp. 2921–2925.
[33] J. Frank and C. U. Martel, "Phase transitions in the properties of random graphs," in *Proc. Principles Pract. Constraint Program.*, 1995, pp. 62–69.
[34] Z. Jiang, S. Zhang, and L. Xie, "Cramer–Rao lower bound analysis for mobile robot navigation," in *Proc. Int. Conf. Intell. Sens., Sens. Netw. Inf. Process.*, 2005, pp. 229–234.



Keith Y. K. Leung (S'08) received the B.A.Sc. and M.A.Sc. degrees in mechanical engineering (mechatronics option) from the University of Waterloo, Waterloo, ON, Canada, in 2005 and 2007, respectively. He is currently working toward the Ph.D. degree with the University of Toronto Institute for Aerospace Studies, Toronto, ON.

He is a Research Assistant with the Autonomous Space Robotics Laboratory and the Flight Systems Control Laboratory, Toronto.

Timothy D. Barfoot received the B.A.Sc. degree in engineering science (aerospace option) in 1997 from the University of Toronto, Toronto, ON, Canada, and the Ph.D. degree in aerospace engineering in 2002 from the University of Toronto Institute for Aerospace Studies (UTIAS), Toronto.

He is currently an Assistant Professor with UTIAS, where he leads the Autonomous Space Robotics Laboratory. Before joining UTIAS, he worked at MDA Space Missions in the Controls and Analysis Group on applications of mobile robotics to space exploration and underground mining.

Dr. Barfoot is a Professional Engineer in the province, Ontario and the Canada Research Chair (Tier II) in Autonomous Space Robotics.

Hugh H. T. Liu (M'00) received the B.Sc. degree from Shanghai Jiao Tong University, Shanghai, China, the M.Sc. degree from Beijing University of Aerospace and Aeronautics, Beijing, China, and the Ph.D. degree from the University of Toronto, Toronto, ON, Canada.

He is currently an Associate Professor with the University of Toronto Institute for Aerospace Studies (UTIAS), where he leads the Flight Systems and Control Laboratory. He is also the Associate Director of Graduate Studies with UTIAS.

Dr. Liu is a Registered Professional Engineer in the province of Ontario and is a member of the Canadian Aeronautics and Space Institute. He is also a member of the American Institute of Aeronautics and Astronautics Guidance, Navigation, and Control Technical Committee. He is currently an Associate Editor of the Conference Editorial Board of the IEEE Control Systems Society.