Decentralized Localization for Dynamic and Sparse Robot Networks

Keith Y. K. Leung*, Timothy D. Barfoot†, Hugh H. T. Liu‡

University of Toronto Institute for Aerospace Studies
Toronto, Ontario, Canada, M3H 5T6
keith.leung@robotics.utias.utoronto.ca*, tim.barfoot@utoronto.ca†, liu@utias.utoronto.ca‡

***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 for information exchange between all robots are not guaranteed. This paper presents a decentralized state estimation algorithm guaranteed to work in dynamic networks without connectivity requirements. We show 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 the same can be performed by all other robots in the network. Our technique is validated through simulations.**

I. Introduction

Communication and the mutual exchange of information are key performance factors for many cooperative multi-robot 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 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 localization problem over a dynamic network, wherein connectivity of all robots is not guaranteed. Furthermore, the decentralized state estimation algorithm that we present is: (1) *equivalent* to a centralized state estimator whenever possible, (2) *scalable* to any number of robots, where it is not necessary for each robot to know the total number of robots in the team, (3) *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, that it will not compromise the others' ability to do so. The question of what information to keep or discard

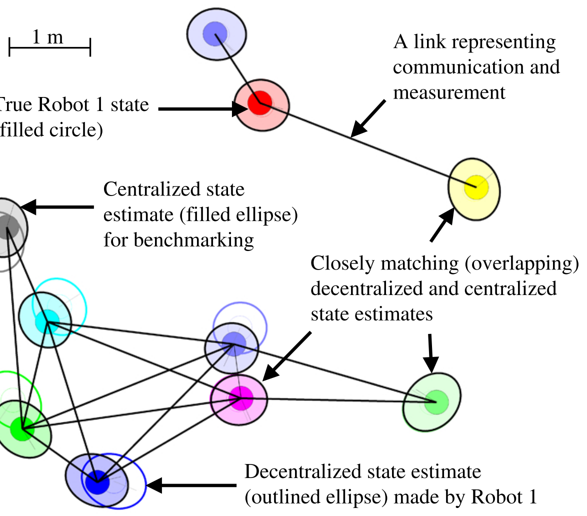
1 m
A link representing communication and measurement
True Robot 1 state (filled circle)
Centralized state estimate (filled ellipse) for benchmarking
Closely matching (overlapping) decentralized and centralized state estimates
Decentralized state estimate (outlined ellipse) made by Robot 1

Fig. 1. A 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.

at what time is answered 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 decentralized algorithm. The problem of distributed and decentralized state estimation has been studied by a number of researchers for linear stochastic systems without inputs. 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 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 *inter-nodal transformation matrix* was introduced and can be used to determine the minimal communication connections required between nodes to produce a state estimate. 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 neighbouring nodes, but it was shown only to work in an acyclic network. This work was extended by Utete and Durrant-Whyte [5] for an arbitrary network topology, wherein communication restrictions are applied so that the channel filter can be used. Later, this was simulated by Bourgault and Durrant-Whyte

[6] for a team of unmanned autonomous vehicles (UAV) and it was shown how the UAVs successfully coordinated with each other in a search mission.

For system models with inputs, Roumeliotis and Bekey [7] performed distributed multi-robot localization by decomposing the Kalman filter into a number of filters that can perform the prediction step of the Kalman filter locally on each robot. However, a fully connected network is still required to perform the update step. Howard [8] looked at performing multi-robot *simultaneous localization and mapping (SLAM)*, wherein each robot is unaware of each other's initial pose and begins state estimation in a decentralized manner. Maps are combined when robots encounter each other and the mapping process eventually becomes centralized. Madhavan et al. [9] studied how cooperative localization and mapping can be performed with a heterogeneous set of sensors and demonstrated this concept with field trials. Rekleitis et al. [10] examined how sensing paradigm and the number of robots affect localization performance with multiple robots.

Static network connectivity is an important requirement for the works mentioned previously. The challenge with performing state estimation over a time-varying network is the obstruction of data flow between robots and the unpredictable sequence in which data is received. Ferguson and How [11] examined various other filters and network architectures to arrive at sub-optimal (full and partial) state estimates. For the various methods and network architectures compared, the trade off between performance and computational requirement was shown. Bar-Shalom et al. [12], [13] examined some possible remedies for *out-of-sequence measurements (OOSM)* for state estimation using a Kalman filter. It was shown that for a missed measurement, the only way to incorporate it to produce an optimal state estimate is to sequentially reprocess all following measurements. In situations where past measurements are no longer available, various methods were proposed to arrive at an approximate estimation. For the special case when the missed measurement is from a single timestep back, it was shown that the optimal state estimate can be achieved.

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*. In general, the study of the consensus problem looks at how coherent global behaviour can be produced by local control laws or estimation methods. An example of this is distributed formation control for multi-vehicle systems [14]. Another example of an application that involves more complex local actions is distributed control for object clustering [15]. For distributed state estimation, Schizas et al. [16] looked at how consensus can be reached in wireless sensor networks. Recently, research on the consensus problem has extended to dynamic network topologies, but convergence can only be guaranteed under some network topology restrictions [17].

In the following section we will formulate our state estimation problem. Section III examines information flow in a network and introduces theorems, which are used in our proposed algorithm in section IV. Simulation results of the algorithm are shown in section V.

II. Problem Formulation

In a multi-robot system, let N represent the set that contains the unique identification indices of all robots. The total number of robots corresponds to $|N|$, the cardinality of the set, and we assume that the identification indices 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:

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

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. We will assume that $r_{\mathsf{obs}} = r_{\mathsf{comm}}$ to simplify the explanation of our work but could be different if required. Let

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

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

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

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

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

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

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

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

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

represent the set of all measurements made at timestep k,

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

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

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

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*

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

when performing state estimation, as it limits memory and processing requirements. 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 that 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, but 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 $\mathsf{bel}(X_{i,k})$) in a decentralized manner.

III. Information Flow in a Dynamic Network

Under the assumption of sporadic communication and observations, it is essential to track the information available to each robot for making state estimations. 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 from the perspective of a particular robot.

A. The Global Perspective

Let $\mathcal{G}_{k_1:k_2}$ be a directed graph (di-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 will be referred to as $\mathcal{G}_{k_1:k_2}$, the *global information flow graph*. An example of this 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 that make 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

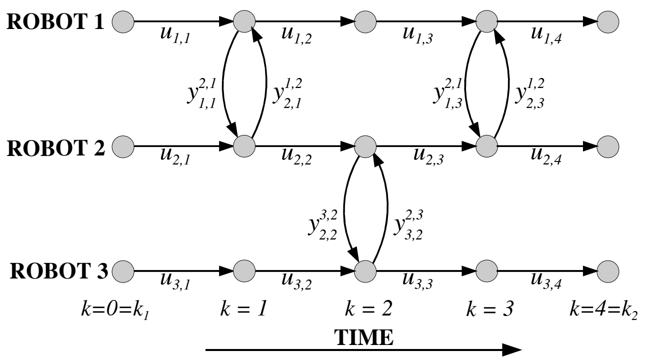

Fig. 2. An example global information flow graph $\mathcal{G}_{k_1:k_2}$ indicating state transition and communication links established between timesteps k_1 and k_2 for 3 robots.

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, $S_{i,0} = \{\mathsf{bel}(\mathbf{x}_{i,0})\}$. 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:

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

The above equations model how information flow 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 (due to the page constraint, the proof of existence for this can be found in [18]):

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 $S_{i,k_e} \supseteq S_{j,k_c}, (\forall i, j)$.

B. The 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. The graph topology from the point of view of an individual robot will differ as illustrated in Fig. 3.

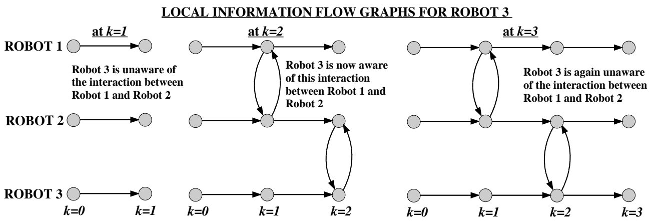
LOCAL INFORMATION FLOW GRAPHS FOR ROBOT 3
at k=1
at k=2
at k=3
ROBOT 1
ROBOT 2
ROBOT 3
Robot 3 is unaware of the interaction between Robot 1 and Robot 2
Robot 3 is now aware of this interaction between Robot 1 and Robot 2
Robot 3 is again unaware of the interaction between Robot 1 and Robot 2
k=0
k=1
k=0
k=1
k=2
k=0
k=1
k=2
k=3

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

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. For this, we present the definition of a *partial checkpoint* and present a proof of 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 $S_{i,k_e} \supseteq S_{j,k_c}$, $(\forall j)$.

Theorem 1: $C_p(k_{c,i}, k_{e,i})$ 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: $S_{i,k_{e,i}} \supseteq S_{j,k_{c,i}}, (\forall j) \Leftrightarrow S_{i,k_{e,i}} \supseteq (\mathbf{u}_{j,k_{c,i}} \text{ or } \mathsf{bel}(\mathbf{x}_{j,k_{c,i}})), (\forall j)$.

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

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

In the second line, we expand $S_{j,k_{c,i}}(\forall j)$ 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 $\mathbf{u}_{j,k_{c,i}}$ or $\mathsf{bel}(\mathbf{x}_{j,k_{c,i}})$ for all robots j will always be available. Now assuming that the knowledge set of each robot at $k_{e,i}$ contains $\mathbf{u}_{j,k_{c,i}}$ or $\mathsf{bel}(\mathbf{x}_{j,k_{c,i}})$:

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

Since odometry data and the belief at $k_{c,i}$ from all robots j is available, this implies on line 2 that $S_{j,k}$ must also be available for all robots j, where $k_{c,i} \leq k \leq k_{e,i}$. 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. ■

This theorem is important as it provides a practical method to detect checkpoints when our decentralized state estimation framework is implemented on a real system. A partial checkpoint can come into existence at different times for each robot depending on the evolving topology of the robot network. Also, a checkpoint exists for the entire system of robots when a partial checkpoint exists for all robots. We now present 2 important theorems regarding the use of partial checkpoints, the proofs of which can again be found in [18] due to page limitation.

Theorem 2: 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, $(\forall i)$.
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 a 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.

Theorem 3: Suppose that $(\forall i)$, robot i applies the Markov property when $C_p(k_{c,i}, k_{e,i})$ exists, (detected using Theorem 1). 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 at timestep k_c.
With the above theorem, we are certain that robots can apply the Markov property based on local knowledge, and 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. This important result will be used to develop our decentralized state estimation algorithm.

IV. Decentralized State Estimation Algorithm

Based on the theoretical development in the previous section, Algorithm 1 is designed to perform decentralized state estimation in a scalable manner that is guaranteed to work in a dynamic and sparse mobile robot network. The same algorithm is implemented on each robot and iterates every timestep. The algorithm requires as inputs: the current timestep, k, odometry data, $u_{i,k}$, 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, $S_{j,k}$ $(\forall j \in R_{i,k})$).

We assume that initially each robot only has a state estimate of itself in its knowledge set. Line 1 updates the knowledge set of robot i by implementing (1) and (2). Line 2 determines the set of robots known to i by looking for part beliefs $\mathsf{bel}^*(\mathbf{X}_{Q,k_s})$ 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 search for a partial checkpoint begins with the 'for' loop on line 3. Line 4 uses Theorem 1 to detect the existence of a checkpoint. If one is found, we use the

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  $N_{i,k} \leftarrow \{Q\}$, $(\forall Q)(\mathsf{bel}^*(\mathbf{X}_{Q,k_s}) \in S_{i,k})(k_s \leq k)$
3  for $k_c \leftarrow k : 0$ do
4      if $U_{N_{i,k},k_c} \in S_{i,k}$ then
5          $\tilde{S}_{i,k_c} \leftarrow S_{i,k} - \left\{U_{N_{i,k},k_r}, Y_{N_{i,k},k_r}\right\}$ $(\forall k_r > k_c)$
6          $\mathsf{bel}^* \left(X_{N_{i,k},k_c}\right) \leftarrow p\left(X_{N_{i,k},k_c} | \tilde{S}_{i,k_c}\right)$
7          $S_{i,k} \leftarrow S_{i,k} \cup \mathsf{bel}^* \left(X_{N_{i,k},k_c}\right)$
             $S_{i,k} \leftarrow S_{i,k} -$
8              $\left\{U_{N_{i,k},k_r}, Y_{N_{i,k},k_r}, \mathsf{bel}^* \left(X_{N_{i,k},k_r}\right)\right\}$ $(\forall k_r \leq k_c)$
9          break
10     end
11 end
12 $\mathsf{bel}\left(X_{N_i,k}\right) \leftarrow p\left(X_{N_i,k} | S_{i,k}\right)$
13 return $\{\mathsf{bel}\left(X_{N_i,k}\right), S_{i,k}, N_{i,k}\}$
```

knowledge up to the checkpoint time (line 5) to obtain the state estimate on line 6 that it is equivalent to the centralized state estimate. This is entered into the knowledge set on line 7 and we proceed to discard information replaceable by bel^* on line 8. On line 9, we break out of the 'for' loop since a partial checkpoint has been found. Finally on line 12, we use all information in the knowledge set to produce the state estimate for the current timestep. (i.e., from k_c forwards, our estimate is temporary).

There are some important points to highlight about Algorithm 1. First, many recursive filtering method can be used on lines 6 and 12. Thus the algorithm is a very general framework that is widely applicable in any situation in which there is a need to perform decentralized state estimation in a dynamic network. Furthermore, 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, we assume that a 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.

In terms of scalability, it is unnecessary for a robot to know how many robots there are in the team initially since state estimates of robots are statistically independent before any encounters and are only correlated through measurements during encounters. When this occurs, estimates can be combined as follows (not explicitly shown in Algorithm 1):

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

Since we are exploiting the Markov property, computational memory usage is limited provided that robots do not wander away from the group permanently, which is a reasonable assumption if their task is to cooperatively localize.

V. Simulation

The theoretical development of a checkpoint already guarantees that a state estimate equivalent to the centralized estimate can be reached by all robots when a 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 are 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 is limited so 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 regardless of range limit for the centralized estimator so that estimates can actually be made for comparison.

A. Setup

The *Extended Kalman Filter (EKF)* algorithm [19] is used as the filtering method on lines 7 and 13 of Algorithm 1. Note that any other filtering method can be applied. The state of each robot includes position, x, y, and orientation, θ. A discrete-time unicycle model 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 contain independent zero-mean Gaussian noise. When a robot i observes another robot j within range r_{comm}, 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 [20] to random waypoints in the bounded workspace.

B. Results

We now present the simulation results for decentralized state estimation with 10 robots[1]. Due to this high number (30) of states, we elect to show the estimation performance for two states in a particular simulation run with $r_{\mathsf{comm}} = 3$. The decentralized state estimates that we refer to in the plots are made by robot 1. Fig. 4 shows the decentralized estimation errors for the x-position of robot 1. Fig. 5 shows the difference compared to the centralized state estimator. Note that for most timesteps, performance between the two estimators are equivalent. Minor deviations occur when an observation is made that is not communicated to robot 1 immediately. The variance of the estimation error is also plotted. Similar performance can be observed in the other robots' decentralized state estimate of themselves.

Fig. 6 shows the decentralized estimation errors for the x-position of robot 2 and Fig. 7 compares this to the centralized state estimate. Deviation in performance is visible when robot 2 loses the connection to robot 1, during which time robot 1 can only assume the last known velocity of robot 2. The spikes seen in Fig. 6 occur when the last known

[1]The video included with the conference proceedings provides a visualization of a simulation run (shortened due to the file size restriction), and Fig.1 should be used as a legend.

velocity of robot 2 is significantly different than the actual velocity. Similar performance is observed for any robot's decentralized state estimate of another robot.

Memory usage by each robot is limited since the Markov property is exploited at partial checkpoints. The amount of memory used by robot 1 in a simulation run is shown in Fig. 8, which reaches a maximum of only 0.11 MB. This is representative of memory usage by other robots. Increases in memory usage occur as time passes since the last partial checkpoint. The more pronounced increases are indications of one or more robots losing connection with robot 1, which is not reestablished until a later time when another partial checkpoint is detected. Memory usage will reduce when this occurs and the Markov property is applied. The rate of memory usage increase between partial checkpoints will in general depend on the number of robots in the system (i.e. the number of states), as well as the freqeuncy of odometry readings and measurements. Currently, if a robot is unable to detect partial checkpoints (i.e. a robot in the team has failed), the consequence is the continual increase in memory usage and we plan to address this scenario in our future work.

Different performance characteristics will result when the number of robots and the communication range are changed. For 10 robots, Fig. 9 shows how the average error between the decentralized and centralized estimates for x-position and orientation θ is reduced as communication range limit r_{comm} increases. The average error plot for y-position is almost identical and therefore not plotted. Similarily, Fig. 10 shows how memory usage decreases with increasing communication range limit. Each data point in the above plots represents data averaged over 50 simulation trials. The trends observed in the above figures occur because as communication range increases, the frequency at which robots detect partial checkpoints also increases. A more detailed study of these trends is presented in [18].

It is important to remember that centralized state estimation is not possible when full connectivity is not guaranteed between robots, and the decentralized state estimator presented 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 in our algorithm.

VI. Conclusions

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 defines when a state estimate equivalent to the centralized estimate can be made by the decentralized state estimator, and implies that the estimate is based on all past information. The proposed method is scalable in that the number of robots in the network does not need to be known initially. 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. Simulations performed with 10 robots also showed that the error

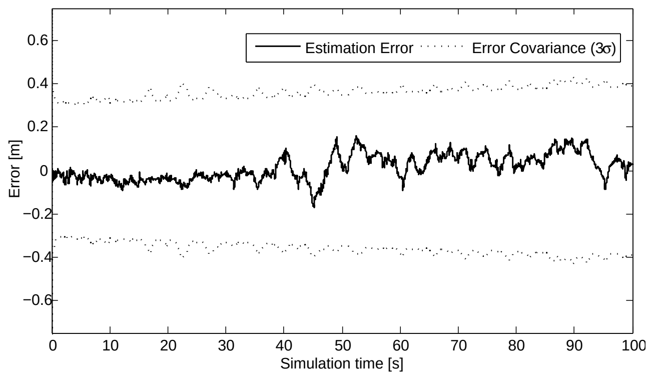

Fig. 4. Error in the decentralized state estimate made by robot 1 for the x-position of robot 1

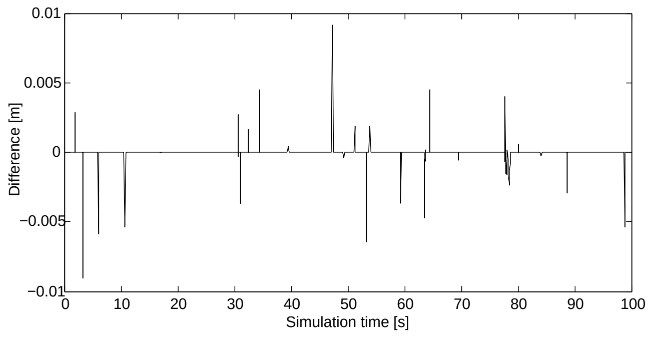

Fig. 5. Difference between the centralized and robot 1's decentralized state estimates for the x-position of robot 1

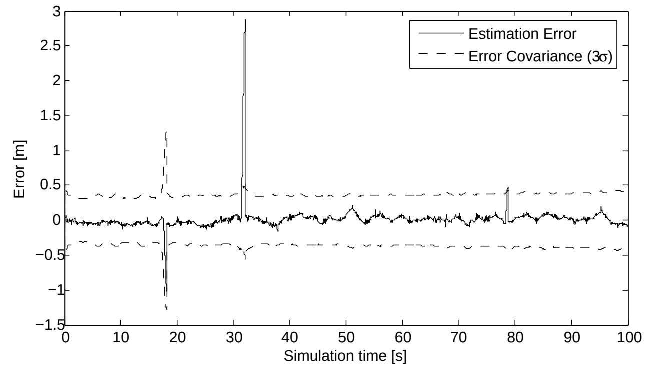

Fig. 6. Error in the decentralized state estimate made by robot 1 for the x-position of robot 2

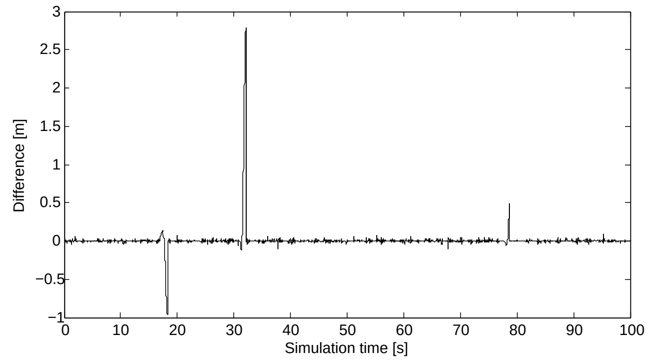

Fig. 7. Difference between the centralized and robot 1's decentralized state estimates for the x-position of robot 2

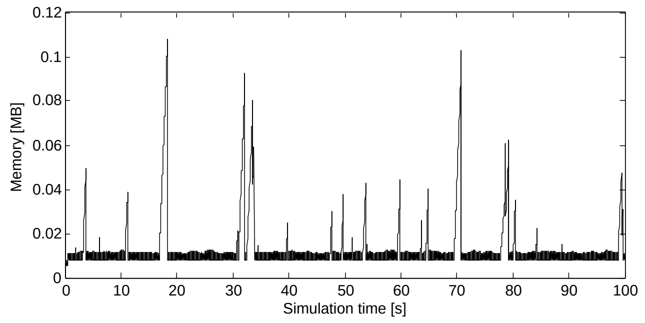

Fig. 8. Memory usage for robot 1

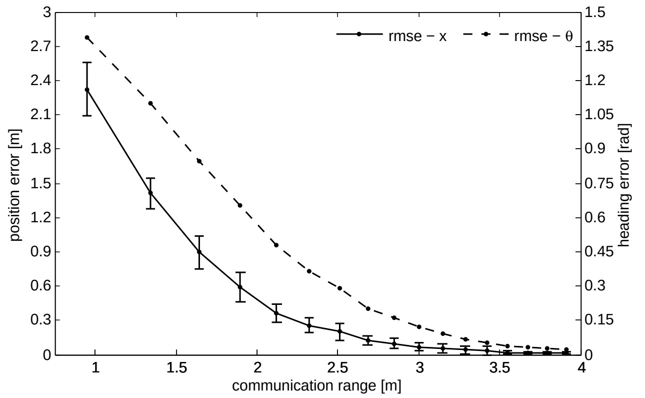

Fig. 9. Average difference between the centralized and decentralized estimates for a 10-robot system with various communication range limits. 2-standard-deviation error bars are shown for the x-position error plot.

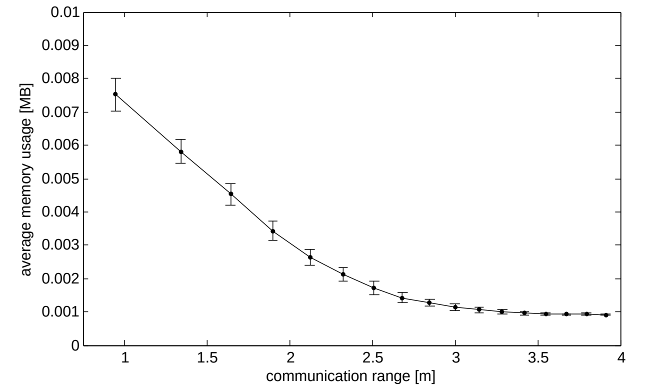

Fig. 10. Average memory usage for a 10-robot system with various communication range limits. 2-standard-devation error bars are shown.

performance of our decentralized state estimator is almost as good as the centralized state estimator. We acknowledge that the performance is dependent on the number of robots and the size of the workspace and studies on this are presented in [18]. Nevertheless, it is important to note 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. Also of interest is to show how communication delay can be accommodated by our decentralized state estimation algorithm (as is). Finally, we would like to incorporate decentralized planning and achieve a method of active SLAM for a dynamic network of mobile robots.

ACKNOWLEDGMENT

This research is supported by Natural Sciences and Engineering Research Council (NSERC) of Canada.

REFERENCES

[1] W. Burgard, M. Moors, C. Stachniss, and F. Schneider, “Coordinated multi-robot exploration,” *IEEE Transactions on Robotics*, vol. 21, no. 3, pp. 376–386, 2005.

[2] T. Berg and H. Durrant-Whyte, “Distributed and decentralized estimation,” in *Proc. of the Singapore Int’l Conference on Intelligent Control and Instrumentation*, vol. 2, 1992, pp. 1118–1123.

[3] ——, “General decentralized kalman filters,” in *American Control Conference*, vol. 2, 1994, pp. 2273–2274.

[4] S. Grime and H. Durrant-Whyte, “Data fusion in decentralized sensor networks,” *Control Engineering Practice*, vol. 2, no. 5, pp. 849–863, 1994.

[5] S. Utete and H. Durrant-Whyte, “Reliability in decentralised data fusion networks,” in *Proc. of IEEE Int’ll Conference on Multisensor Fusion and Integration for Intelligent Systems*, 1994, pp. 215–221.

[6] F. Bourgault and H. Durrant-Whyte, “Communication in general decentralized filters and the coordinated search strategy,” in *Proceedings of the 7th International Conference on Information Fusion*, 2004.

[7] S. Roumeliotis and G. Bekey, “Distributed multirobot localization,” *Robotics and Automation, IEEE Trans. on*, vol. 18, no. 5, pp. 781–795, 2002.

[8] A. Howard, “Multi-robot simultaneous localization and mapping using particle filters,” *International Journal of Robotics Research*, vol. 25, no. 12, pp. 1243–1256, 2006.

[9] R. Madhavan, K. Fregene, and L. Parker, “Distributed cooperative outdoor multirobot localization and mapping,” *Autonomous Robots*, vol. 17, no. 1, pp. 23–39, 2004.

[10] I. Rekleitis, G. Dudek, and E. Milios, “Multi-robot cooperative localization: A study of trade-offs between efficiency and accuracy,” in *Proceedings of the IEEE/RSJ IROS*, 2002.

[11] P. Ferguson and J. How, “Decentralized estimation algorithms for formation flying spacecraft,” in *Proceedings of the AIAA Guidance, Navigation, and Control Conf.*, 2003.

[12] Y. Bar-Shalom, “Update with out-of-sequence measurements in tracking: exact solution,” *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 38, no. 3, pp. 769–777, 2002.

[13] Y. Bar-Shalom, H. Chen, and M. Mallick, “One-step solution for the multistep out-of-sequence-measurement problem in tracking,” *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 40, no. 1, pp. 27–37, 2004.

[14] R. Olfati-Saber, J. Fax, and R. Murray, “Consensus and cooperation in networked multi-agent systems,” *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215–233, 2007.

[15] T. Barfoot and G. D’Eleuterio, “Evolving distributed control for an object clustering task,” *Complex Systems*, vol. 15, no. 3, pp. 183–201, 2005.

[16] I. Schizas, A. Ribeiro, S. Roumeliotis, and G. Giannakis, “Consensus in ad hoc wsns with noisy links - part i: Distributed estimation of deterministic signals,” *Signal Processing, IEEE Transactions on*, vol. 56, no. 1, pp. 350–364, 2008.

[17] L. Moreau, “Stability of multiagent systems with time-dependent communication links,” *Automatic Control, IEEE Transactions on*, vol. 50, no. 2, pp. 169–182, 2005.

[18] K. Y. K. Leung, T. D. Barfoot, and H. H. T. Liu, “Decentralized localization for general robot networks,” *(submitted to) IEEE Transaction on Robotics (available upon request)*.

[19] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. The MIT Press, 2005.

[20] R. Siegwart and Nourbakhsh, *Introduction to Autonomous Mobile Robots*. MIT Press, 2004.