Approximate Distributed Kalman Filtering for Cooperative Multi-agent Localization

Prabir Barooah[1], Wm. Joshua Russell[2], and João P. Hespanha[2],⋆

[1] University of Florida, Gainesville, FL 32611, USA
pbarooah@ufl.edu
[2] University of California, Santa Barbara, CA 93106, USA
wjrussell@umail.ucsb.edu, hespanha@ece.ucsb.edu

Abstract. We consider the problem of estimating the locations of mobile agents by fusing the measurements of displacements of the agents as well as relative position measurements between pairs of agents. We propose an algorithm that computes an approximation of the centralized optimal (Kalman filter) estimates. The algorithm is distributed in the sense each agent can estimate its own position by communication only with nearby agents. The problem of distributed Kalman filtering for this application is reformulated as a parameter estimation problem. The graph structure underlying the reformulated problem makes it computable in a distributed manner using iterative methods of solving linear equations. With finite memory and limited number of iterations before new measurements are obtained, the algorithm produces an approximation of the Kalman filter estimates. As the memory of each agent and the number of iterations between each time step are increased, the approximation improves. Simulations are presented that show that even with small memory size and few iterations, the estimates are quite close to the centralized optimal. The error covariances of the location estimates produced by the proposed algorithm are significantly lower than what is possible if inter-agent relative position measurements are not available.

1   Introduction

Mobile autonomous agents such as unmanned ground robots and unmanned aerial vehicles that are equipped with on-board sensing, actuation, computation and communication capabilities hold great promise for applications such as surveillance, disaster relief, and scientific exploration. Irrespective of the application, their successful use generally requires that the agents be able to obtain accurate estimates of their positions. Although typically position information is provided by GPS, in many scenarios GPS may be available only intermittently, or sometimes not available at all. These situations include underwater operation,

presence of urban canyons, or hostile jamming. In such a situation, localization is typically performed by integrating measurements of displacements, which can be obtained from IMUs (inertial measurement units) or/and vision sensors [1–3]. Since these measurements are noisy, their integration over time lead to high rate of error growth [3–5].

When multiple agents operate cooperatively, it is possible to reduce localization errors if information on *relative positions* between pairs of agents are available. Such measurements can be obtained by vision-based sensors such as cameras and LIDARs, or RF sensors through AoA and TDoA measurements. These measurements, although noisy, furnish information about the agent's locations in addition to that provided by displacement measurements. The problem of fusing information on relative positions between mobile agents for estimating their locations is commonly known as *cooperative localization* in the robotics literature [6–8]. The typical approach is to use an extended Kalman filter, to fuse both odometry data and robot-to-robot relative distance measurements [8, 9].

However, computing the Kalman filter estimates requires that all the measurements (inter-agent as well as displacement measurements of all the agents) are made available to a central processor. Such a centralized approach requires routing data though an ad-hoc network of mobile agents, which is a difficult problem [10]. Therefore, a *distributed scheme* that allows agents to estimate their positions accurately through local computation and communication instead of relying on a central processor is preferable. In the sensor networks and control literature, the problem of distributed Kalman filtering has drawn quite a bit of attention [11–14]. The papers [12, 15] in particular present a distributed Kalman filtering techniques for multi-agent localization. The paper [16] addresses cooperative localization in mobile sensor networks with intermittent communication, in which an agent updates its prediction based on the agents it encounters, but does not use inter-agent relative position measurements.

In this paper we approach the problem of distributed Kalman filtering for cooperative localization from a novel angle, by reformulating the problem as a parameter estimation problem. The Kalman filter computes the LMMSE estimates of the states of a linear system given the measurements. It is a classical result that under infinite prior covariance, the LMMSE is the same as the BLUE (best linear unbiased estimator) [17]. For the problem at hand, the BLUE estimator has a convenient structure that can be described in terms of a graph consisting of nodes (agent locations) and edges (relative measurements). This structure can be exploited to distribute the computations by using parallel iterative methods of solving linear equations. The proposed method therefore is designed to compute the BLUE estimates using iterative techniques, and is based our earlier work on localization in static sensor networks [18, 19]. If the agents were to have infinite memory and could run infinitely many iterations before they move and change their positions, the estimates produced by the proposed algorithm are equal to the centralized BLUE estimates, and therefore the same as the Kalman filter

estimates. Due to memory and time constraints, the proposed algorithm uses only a small subset of all the past measurements and runs only one (or a few) iterations. This makes the method an approximation of the Kalman filter. Fewer the number of iterations and smaller the size of past measurements retained, the easier it is to implement the algorithm in a distributed setting. Simulations show the approximations are remarkably close to the centralized Kalman filter/BLUE estimates even when a very small amount of past data is used and only one iteration is executed between successive motion updates.

The rest of the paper is organized as follows. Section 2 describes the estimation problem precisely. Section 3 describes centralized Kalman filtering for cooperative localization and its reformulation as a problem of parameter estimation in measurement graphs. Section 4 describes the proposed algorithm, and simulations are presented in Section 5.

2   Problem Description

Consider a group of n mobile agents that need to estimate their own positions with respect to a geostationary coordinate frame, whose origin is denoted by x_0. In the absence of GPS, we arbitrarily fix the initial position of one of the agents, say, the first agent, as the origin. Time is measured with a discrete index $k = 0, 1, 2, \ldots$ The noisy measurement of the displacement of agent j obtained by its on-board sensors during the k-th time interval is denoted by $u_j(k)$, so that

$$u_j(k) = x_j(k+1) - x_j(k) - w_j(k), \tag{1}$$

where $x_j(k)$ is the position of agent j at time k, and $\{w_j(k)\}$ is a zero-mean noise with the property that $E[w_j(k)w_i(\ell)] = 0$ unless $i = j, k = \ell$. We assume that certain pairs of agents, say i, j, can also obtain noisy measurements $y_{ij}(k)$ of their relative position

$$y_{ij}(k) = x_i(k) - x_j(k) + v_{ij}(k), \tag{2}$$

where $v_{ij}(k)$ is zero-mean measurement noise with $E[v_{ij}(k)v_{mn}(\ell)] = 0$ unless $(ij) = (mn), k = \ell$. We assume that the agents are equipped with compasses, so that all these measurements are expressed in a common Cartesian reference frame. The goal is to combine the agent-to-agent relative position measurements with agent displacement measurements to obtain estimates of their locations that are more accurate than what is possible from the agent displacement measurements alone. In addition, the computation should be distributed so that every agent can compute its own position estimate by communication with a small subset of the other agents, called *neighbors*. We assume that two agents that can obtain each others' relative position measurement at time index k can also exchange information through wireless communication during the interval from k to $k + 1$.

3   Kalman Filtering vs. BLU Estimation from Relative Measurements

The availability of the displacement measurements allow us to write the following process model for the j-th vehicle:

$$x_j(k+1) = x_j(k) + u_j(k) + w_j(k), \tag{3}$$

where $u_j(k)$ is now viewed as a known input. One can stack the states $x_j(k), j = 1, \ldots, n$ into a tall vector $\mathbf{x}(k)$ and write the system dynamics

$$\mathbf{x}(k+1) = \mathbf{x}_j(k) + \mathbf{u}(k) + \mathbf{w}(k), \quad \mathbf{y}(k) = C(k)\mathbf{x}(k) + \mathbf{v}(k)$$

where $C(k)$ is appropriately defined so that entries of $\mathbf{y}(k)$ are the inter-agent relative position measurements (2). When the control input and the measurements $\{\mathbf{u}(t)\}, \{\mathbf{y}(t)\}, t \in \{0, 1, \ldots, k\}$ are made available to a central processor, along with the initial conditions $\hat{\mathbf{x}}(0|-1)$ and $P(0|-1) = Cov(\hat{\mathbf{x}}(0|-1) - \mathbf{x}(0), \hat{\mathbf{x}}(0|-1) - \mathbf{x}(0))$, and the noise covariances $Q(t) := Cov(\mathbf{w}(t), \mathbf{w}(t)), R(t) := Cov(\mathbf{v}(t), \mathbf{v}(t))$, a Kalman filter can be used to compute estimate $\hat{\mathbf{x}}^{\text{Kalman}}(k|k) = E*(\hat{x}(k)|\{u\}, \{y\})$ of the state $\mathbf{x}(t)$, where $E*(X|Y)$ denotes the LMMSE estimate of a r.v. X in terms of the r.v. Y [20].

To distribute the computations of the Kalman filter, we reformulate the problem into an equivalent, deterministic parameter estimation problem. We associate the positions $\{x_0\} \cup \{x_j(t)\}, j \in \{1, \ldots, n\}, t \in \{0, 1, \ldots, k\}$ of the entities until time k with the *nodes* $\mathbf{V}(k)$ of a *measurement graph* $\mathbf{G}(k) = (\mathbf{V}(k), \mathbf{E}(k))$. The *edges* $\mathbf{E}(k)$ of the graph correspond to the relative position measurements between the nodes $\mathbf{V}(k)$. If an edge is between two nodes that correspond to subsequent positions of same agent j, then the edge corresponds to a measurement of the displacement of the agent between those two time instants. If , on the other hand, the edge is between two nodes that correspond to the positions of two distinct agents at a particular time instants, then the edge corresponds to the noisy measurement of the relative position between the agents. All measurements mentioned above are of the type

$$\zeta_e = x_u - x_v + \epsilon_e \tag{4}$$

where u and v are nodes of the measurement graph $\mathbf{G}(k)$ and $e = (u, v)$ is an edge (an ordered pair of nodes). In particular, an edge exists between a node pair u and v if and only if a relative measurement of the form (4) is available between the two nodes. Since a measurement of $x_u - x_v$ is different from that of $x_v - x_u$, the edges in the measurement graph are directed. The edge directions are arbitrary. Figure 1 shows an example of a measurement graph.

3.1   BLUE Estimation

We briefly review the BLUE (best linear unbiased estimator) from relative measurements for graphs that do not change with time. The BLUE is optimal (minimal variance) among all linear unbiased estimators. Consider a measurement
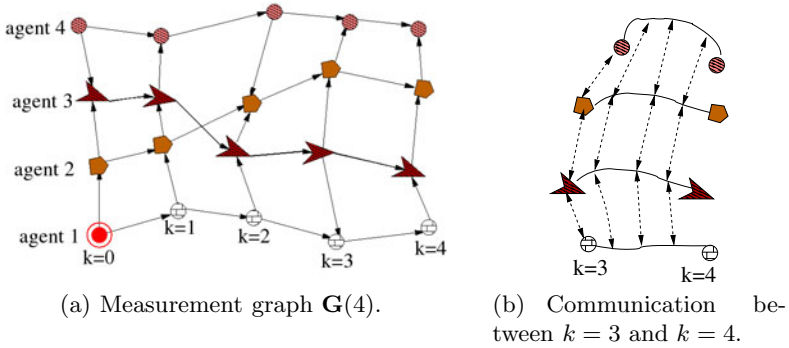
(a) Measurement graph $\mathbf{G}(4)$.

(b) Communication between $k = 3$ and $k = 4$.

Fig. 1. (a) An example of a measurement graph generated as a result of the motion of a group of four mobile agents. The graph shown here is $\mathbf{G}(4)$, i.e., the snapshot at the 4th time instant. The unknown variables at current time $k = 4$ are the positions $x_i(t)$, $i \in \{1, 2, \ldots, 4\}$, at the time instants $k \in \{0, 1, \ldots, 4\}$, except for the initial position of agent 1: $x_1(0)$, which is taken as the reference. (b) The communication during the time interval between $k = 3$ and $k = 4$. In the situations shown in the figure, 4 rounds of communication occur between a pair of agents in this time interval.

graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, where the nodes in $\mathbf{V}$ correspond to variables and edges in $\mathbf{E}$ correspond to relative measurement between node variables of the form (4). Let $V_r \subset \mathbf{V}$ denote the non-empty subset of nodes whose variables are known, which are called *reference variables* , and $n = |\mathbf{V} \setminus \mathbf{V}_r|$ be the number of unknown variables that are to be estimated. Let $\mathbf{x}$ be the vector obtained by stacking together the unknown variables. As described in [19], given a measurement graph with n unknown variables, the BLU estimate $\hat{\mathbf{x}}^*$ is given by the solution of a system of linear equations

$$\mathcal{L}\mathbf{x} = \mathbf{b}, \tag{5}$$

where $\mathcal{L}$ and $\mathbf{b}$ depend on the measurement graph $\mathbf{G}$, the measurement error covariance matrices $P_e, e \in \mathbf{E}$, the measurements $\zeta_e, e \in \mathbf{E}$ and the reference variables $x_r, r \in \mathbf{V}_r$. The matrix $\mathcal{L}$ is invertible (so that BLU estimate $\hat{\mathbf{x}}^*$ exists and is unique) if and only if for every node, there is an undirected path between the node and at least one reference node [21]. Under this condition, the covariance matrix of the estimation error $\Sigma := cov(\hat{\mathbf{x}}^*, \hat{\mathbf{x}}^*)$ is given by

$$\Sigma = \mathcal{L}^{-1}. \tag{6}$$

By splitting the matrix $\mathcal{L} = M - N$ where M is a block diagonal degree matrix and N is a generalized adjacency matrix of the network, the above system of equations can be written as $M\mathbf{x} = N\mathbf{x} + \mathbf{b}$, which leads to the following block-Jacobi iterative method for solving it:

$$\hat{\mathbf{x}}(k+1) = M^{-1}N\hat{\mathbf{x}}(k) + M^{-1}\mathbf{b}. \tag{7}$$

where M is block diagonal and N is sparse. In particular, only those entries on the i-th block row of N are non-zero that corresponds to i's neighbors in $\mathbf{G}$. This structure makes it possible to compute the updates in (7) in a distributed manner, so that each node only needs to communicate with its neighbors. The resulting algorithm can be shown to converge under certain assumptions on the measurement error covariance matrices, even when executed in an asynchronous manner, and in the presence of temporary communication faults. The reader is referred to [19, 21] for the details.

When the measurement graph is time-varying, if all the measurements corresponding to the edges in $\mathbf{G}(k)$ are available to a central processor at time k, the processor can compute the BLU estimates of all the node variables in the graph $\mathbf{G}(k)$ (which correspond to the present as well as past positions of the agents) by solving (5). The resulting estimate is denoted by $\hat{\mathbf{x}}^{\mathrm{BLUE}}(k)$.

The following result, which follows from standard results in estimation theory shows that under uninformative prior, the blue estimate is equivalent to the Kalman filter estimates.

Lemma 1. *If $P^{-1}(0|-1) = 0$, then $\hat{x}^{\mathrm{Kalman}}(k|k) = \hat{\mathbf{x}}^{\mathrm{BLUE}}(k)$ for every k.*   □

For the problem at hand, we assume that no prior information on the agent positions are available at time 0 except for inter-agent position measurements $\mathbf{y}(0)$ and the position of one of the agents that is used as a reference. In that case the information form of the Kalman filter can be used to compute the LMMSE estimate of the agent positions [22]. The result above shows that under the assumption of no prior information, the BLUE estimates are identical to the Kalman filter estimates. Therefore, from now on we do not distinguish between $\hat{x}^{\mathrm{Kalman}}(k|k)$ and $\hat{\mathbf{x}}^{\mathrm{BLUE}}(k)$, and refer to them simply as the centralized optimal estimate $\hat{\mathbf{x}}^*(k)$.

In the next section we will utilize the BLUE formulation to devise distributed algorithms to compute the estimates.

4   A Distributed Algorithm for Dynamic Localization

In this section we present a distributed algorithm to obtain estimates of the positions mobile agents that are close to the centralized BLU estimator described in the previous section. By distributed we mean every agent should be able to estimate its own position, and all the information needed to carry out the computation should come from local sensing and communication with its neighboring agents.

4.1   Infinite Memory and Bandwidth

We first describe the algorithm by assuming that every agent can store and broadcast an unbounded amount of data. We will relax this assumption later.

For every agent j, let $\mathbf{V}_j(k)$ contain all the nodes that correspond to the positions of itself and the positions of the agents with whom j has had relative
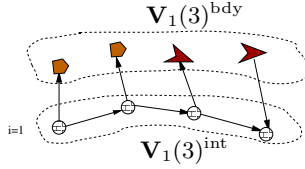
Fig. 2. The subgraph $\mathbf{G}_1(3)$ of agent 1 at time 3, for the measurement graph shown in Figure 1

measurements, up to and including time k. Let $\mathbf{E}_j(k)$ be the subset of edges in $\mathbf{G}(k)$ that are incident on nodes that correspond to j's current or past positions. By assumption, the relative measurements and their error covariances $\zeta_e, P_e, e \in \mathbf{E}_j(k)$ are available to agent j at time k. We now define the *local subgraph* of agent j at time k as $\mathbf{G}_j(k) = (\mathbf{V}_j(k), \mathbf{E}_j(k))$. Figure 2 shows the subgraph of agents 1 at time $k = 3$ corresponding to the measurement graph shown in Figure 1.

The nodes in a local subgraph $\mathbf{G}_j(k)$ are divided into two categories - the *internal nodes* $\mathbf{V}_j(k)^{\mathrm{int}}$ and the *boundary nodes* $\mathbf{V}_j(k)^{\mathrm{bdy}}$. The internal nodes are the nodes that correspond to the positions of the agent up and including time t. The boundary nodes consist of the nodes in the local subgraph that correspond to the positions of the neighboring agents. Thus, $\mathbf{V}_j(k) = \mathbf{V}_j(k)^{\mathrm{int}} \cup \mathbf{V}_j(k)^{\mathrm{bdy}}$ and $\mathbf{V}_j(k)^{\mathrm{int}} \cap \mathbf{V}_j(k)^{\mathrm{bdy}} = \emptyset$.

To explain the algorithm, imagine first that the agents have stopped moving at time t_{stop}. We have proposed a distributed algorithm in [19, 23] for localization of static agents that is based on the Jacobi iterative method of solving linear equations [24]. If agents stop moving, they can use the Jacobi algorithm to compute the optimal estimate of its entire position history in distributed manner. We describe the procedure briefly, which will serve as a stepping stone into developing the proposed algorithm.

Algorithm A.1: static agents. Let o be the global reference node. Consider the set of past positions of agent j until time t_{stop}, i.e., $\{x_v, v \in \mathbf{V}_j(t_{\mathrm{stop}})^{\mathrm{int}} \backslash \{o\}\}$. The vector of the node variables in this set is denoted by $\mathbf{x}_j(t_{\mathrm{stop}})$. The set of unknown node positions at time t_{stop} is $\cup_j \mathbf{x}_j(t_{\mathrm{stop}})$. Let $\hat{\mathbf{x}}_j^\tau(t_{\mathrm{stop}})$ be the estimate of $\mathbf{x}_j(t_{\mathrm{stop}})$ obtained by agent j at the end of the τth iteration. The estimates are obtained and improved using the following distributed algorithm, starting with an arbitrary initial condition:

At τ^{th} iteration, every agent j does the following.

1. It broadcasts the current estimate $\hat{\mathbf{x}}_j^\tau(t_{\mathrm{stop}})$ to all of its neighboring agents. Consequently, it also receives the current estimates $\hat{\mathbf{x}}_i^\tau(t_{\mathrm{stop}})$ from each of its neighboring agents, i.
2. It (agent j) assigns the boundary nodes $\mathbf{V}_j(t_{\mathrm{stop}})^{\mathrm{bdy}}$ as the reference nodes of its local subgraph $\mathbf{G}_j(t_{\mathrm{stop}})$ and sets the reference variables to be the estimates of those node variables that it has recently received from its neighbors. With this assignment of reference node variables and with the relative

measurements $\{\zeta_e, e \in \mathbf{E}_j(t_{\text{stop}})\}$, agent j then sets up the system of linear equations (5) for its local subgraph $\mathbf{G}_j(t_{\text{stop}})$, and solves these equations to obtain an updated estimate of $\hat{\mathbf{x}}_j^{\tau+1}(t_{\text{stop}})$ of its "internal" node variables. $\qquad \square$

The following result about the behavior of the estimates follows from the convergence property of the Jacobi algorithm (see [19, 23]).

Proposition 1. *The estimates of all the node variables* $\mathbf{x}(t_{\text{stop}})$ *(i.e., all agents' past positions up to time* t_{stop}*) converge to their centralized optimal estimates:* $\hat{\mathbf{x}}_j^{\tau}(t_{\text{stop}}) \to \hat{\mathbf{x}}_j^*(t_{\text{stop}})$ *as* $\tau \to \infty$. $\qquad \square$

As a result, if agents stop moving, by communicating with its neighbors and updating sufficiently many times, an agent can obtain an estimate of its entire position history that is arbitrarily close to the optimal estimates. Note that the description above implicitly assumes that the iterations are executed synchronously, i.e., there is a common iteration counter τ among all the agents. However, the result in Proposition 1 holds even if communication and computation is performed in an asynchronous way, where every agent has its own iteration counter τ^i. This follows from standard results in asynchronous iterations [23].

Now we are ready to describe the algorithm for localization of mobile agents with finite memory and finite bandwidth.

Estimation with mobile agents: Algo A.2. In the description below, T_{mem} is a fixed positive integer that denotes the "size" of a subgraph of $\mathbf{G}(k)$ every agent maintains in its local memory at time k. The parameter T_{mem} is fixed ahead of time and provided to all agents; its value is determined by the memory in each agent's local processor. The maximum number of iterations carried out by an agent j during the interval between times k and $k+1$, which is denoted by $\tau^{\max}$, depends on the maximum number of communication rounds between j and its neighbors during this interval.

Let $\mathbf{G}(k)^{T_{\text{mem}}} = (\mathbf{V}(k)^{T_{\text{mem}}}, \mathbf{E}(k)^{T_{\text{mem}}})$ be the subgraph containing nodes, $\mathbf{V}(t)^{T_{\text{mem}}}$, that correspond to all agent positions from time $\max(k - T_{\text{mem}}, 0)$ until time k and containing edges, $\mathbf{E}(k)^{T_{\text{mem}}}$, corresponding to relative measurements between to nodes in $\mathbf{G}(k)^{T_{\text{mem}}}$. More simply, $\mathbf{G}(k)^{T_{\text{mem}}}$ is the subgraph containing all nodes and edges corresponding to positions of agents and relative measurements at the current and previous T_{mem} time instants. In this case, $\hat{\mathbf{x}}_j^{\tau}(k)$ is a vector of the estimates of the positions of agent j from time instant $\max(k - T_{\text{mem}}, 0)$ to time k, obtained in the τth iteration.

1. If GPS is not available to every agent at $k = 0$, one agent's initial position serves as the global reference. Every other agent starts with the initial estimate that is obtained by adding the relative measurements on a path from itself to the agent whose initial position is taken as the global reference. For example, when agent 1 is the global reference and relative position measurements are available between agents with successively increasing indices, we have $\hat{x}_j(0) := y_{j,j-1}(0) + y_{j-1,j-2}(0) + \cdots + y_{2,1}(0) + x_1(0)$. We assume that

these measurements are transmitted to the agents initially before they start moving.
2. During the time interval between time indices k and $k+1$, each agent j updates the estimate of $\mathbf{x}_j(t)$ in the following way.
   - initialization: $\hat{x}_j^{(0)}(k) = \hat{x}_j^{(\tau_{\max})}(k-1) + u_j(k-1)$.
   - collect inter-agent measurements, i.e., obtain $\zeta_{v,w}$ for $v = i(k)$ and $w \in \mathcal{N}_j(t)$.
   - iterative update: node j now iteratively updates its position by the algorithm described in the previous section. Specifically, it runs the algorithm A.1 for the subgraph $\mathbf{G}_j(k)^{T_{\mathrm{mem}}}$.   □

(a) $T_{\mathrm{mem}} = 1$ (b) $T_{\mathrm{mem}} = 3$
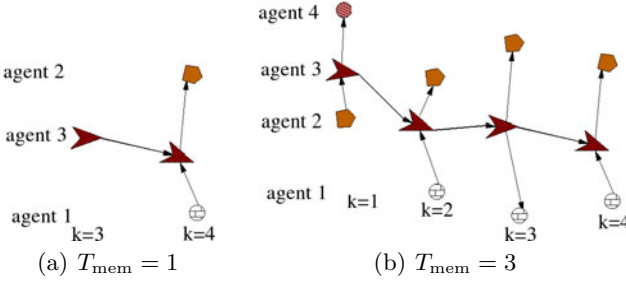
Fig. 3. Truncated subgraphs, $\mathbf{G}_3(4)$ of agent 3 at time 4 for the measurement graph shown in Figure 1

The algorithm continues as long as the agents continue to move. Figure 3 shows an example of the local subgraphs used by an agent (agent 3 in Figure 1) for two cases, $T_{\mathrm{mem}} = 1$ and $T_{\mathrm{mem}} = 3$. Note that the proposed algorithm is particularly simple when $T_{\mathrm{mem}} = 1$, since in that case the iterative update is the solution to the following equation:

$$S_i(k-1)\hat{x}_j^\tau(k) = W_j^{-1}(k-1)\Big(\hat{x}_j^{\tau_{\max}}(k-1) + u_j(k-1)\Big) +$$
$$\sum_{i \in \mathcal{N}_j(k)} V_{j,i}^{-1}(k)\Big(\hat{x}_i^{\tau-1}(k) + y_{j,i}(k)\Big),$$

where $W_j(k) := cov(w_j(k), w_j^T(k))$ and $V_{j,i}(k) := cov(v_{ij}(k), v_{ij}^T(k))$ are the error covariances in the displacement measurement $u_j(k-1)$ and inter-agent relative measurement $y_{ij}(k)$, respectively, and $S_i(k) := W_j^{-1}(k) + \sum_{i \in \mathcal{N}_j(k)} V_{i,j}^{-1}(k)$. When all the measurement error covariances are equal, the update is simply:

$$\hat{x}_j^\tau(k) = \frac{1}{|N_j(k) + 1|}\Big(\hat{x}_j^{\tau_{\max}}(k-1) + u_j(k-1) + \sum_{i \in \mathcal{N}_j(k)} (\hat{x}_i^{\tau-1}(k) + y_{j,i}(k))\Big)$$

When $T_{\mathrm{mem}} = \infty$, the algorithm is simply the Jacobi iterations to compute the BLUE estimates of all the node variables in the graph $\mathbf{G}(k)$, i.e., the past and
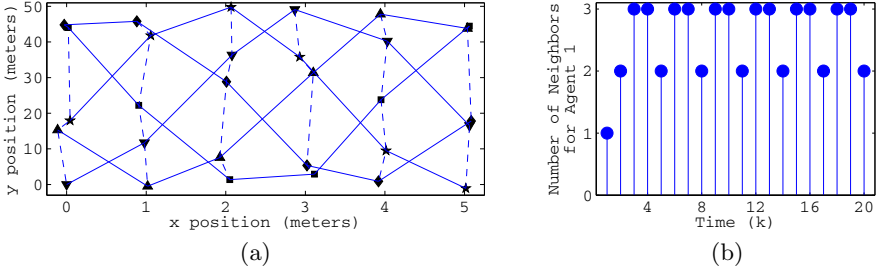
Fig. 4. A snapshot of the measurement graph $\mathbf{G}(k)$ at time $k = 5$ created by the motion of 5 mobile agents, for which the simulations reported here are conducted

present positions of the agents. In this case, Proposition 1 guarantees that the estimates computed converge to the BLUE estimates as $\tau^{\mathrm{max}} \to \infty$. When the algorithm is implemented with small values of T_{mem}, after a certain number of time steps, measurements from times earlier than T_{mem} steps into the past are no longer directly used. Past measurements are still used indirectly, since they affect the values of the reference variables used by the agents for their local subgraphs. With finite T_{mem}, the estimates are no longer guaranteed to reach their centralized optimal. A further reduction in accuracy comes from the fact that in practice τ^{max} may not be large enough to get close to convergence even in the truncated local subgraphs. The algorithm therefore produces an approximation of the centralized optimal estimates; the approximation becomes more accurate as τ^{max} and T_{mem} increases.

Communication and computation cost. The amount of data an agent needs to store and broadcast increases as the "size" of the truncated local subgraph that the agent keeps in local memory increases, and therefore, on T_{mem}. When the neighbors of an agent do not change with time, the number of nodes in its local truncated subgraph of an agent at any given time is $T_{\mathrm{mem}} + N_{\mathrm{nbr}}T_{\mathrm{mem}}$, where N_{nbr} is the number of neighbors of the agent. In this case, the number of edges in the truncated local subgraph is at most $T_{\mathrm{mem}} + T_{\mathrm{mem}}N_{\mathrm{nbr}}$ (the first term is the number of odometry measurements and the second term is the number of relative measurements between the agent and its neighboring agents that appear as edges in the subgraph). Therefore, an agent needs a local memory large enough to store $\ell[2T_{\mathrm{mem}}(1 + N_{\mathrm{nbr}}) + T_{\mathrm{mem}}N_{\mathrm{nbr}}]$ floating-point numbers, where $\ell = 2$ or 3 depending on whether positions are 2 or 3 dimensional vectors. An agent has to broadcast the current estimates of its interior node variables, i.e., ℓT_{mem} numbers, at every iteration. Thus, the communication, computation and memory requirements of the algorithm are quite low for small values of T_{mem}. We assume that the agents can obtain the error covariances of the measurements on the edges that are incident on themselves, so these need not be transmitted.

5 Simulations

We illustrate the algorithm's performance by numerical simulations. All simulation results are shown for the case $T_{mem} = 1$. Five agents move in a zig-zag trajectory; the resulting measurement graph is shown in Figure 4(a). A time trace of the number of neighbors of agent 1 is shown in Figure 4(b). The initial position of agent 1 (bottom left node in Figure 4) is taken as the reference. Every measurement of $x_u - x_v$ is obtained from noisy measurements of the distance $\|x_u - x_v\|$ and the angle between x_u and x_v. The distance and angle measurements are corrupted with additive Gaussian noise, with $\sigma_r = 0.05m$ and $\sigma_\theta = 5^o$. The measurement error covariances are estimated from the range and bearing measurements and the parameters σ_r, σ_θ (as explained in [19]), which makes the covariances of the errors on relative position measurements on distinct edges distinct.
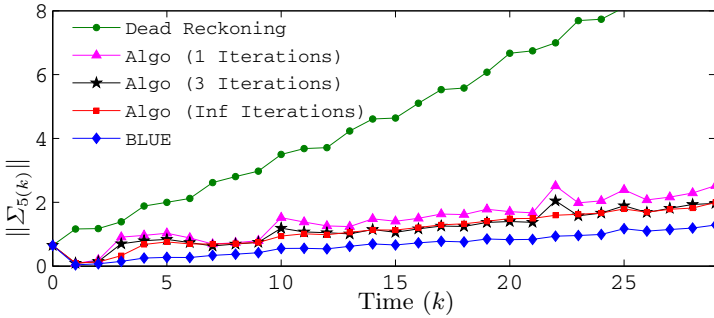

Fig. 5. Covariance of the estimate of the current position of agent 5 (of Figure 4) as a function of time. Agent 5 is the one farthest from agent 1, whose initial position being the reference node. Dead reckoning provides an estimate of positions by summing optometry data. Estimates from algorithm 2 are shown for $\tau^{max} = 1, 3$, and ∞. BLUE refers to the centralized optimal.
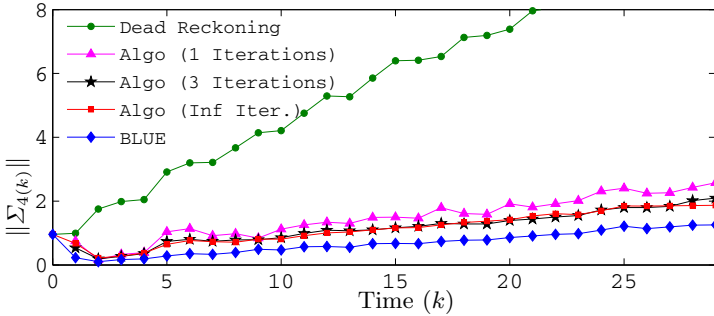

Fig. 6. Covariance of the estimate of the current position of agent 4 (of Figure 4) as a function of time

Covariances of agent position estimates produced by the proposed algorithm are estimated from 1000 Monte-Carlo runs. Figure 5 shows the covariance of the estimate of $x_5(t)$, the position of agent 5, as a function of t. Agent 5 is the one farthest away from agent 1. The figure shows that the location estimates produced by the proposed algorithm are much more accurate than those produced by integrating the displacement measurements alone (dead reckoning). It is seen from the plot that the estimation error covariance of the algorithm (even with $T_{\mathrm{mem}} = 1$ and $\tau^{\mathrm{max}} = 1$) is close to that of the centralized optimal estimator (BLUE). Comparison among the plots for $\tau^{\mathrm{max}} = 1, 3$ and ∞ shows that, as expected, the estimation accuracy improves with increasing number of iterations between every time step. However, it is also seen that the improvement levels off quickly. In fact, even with the minimal possible number of iterations $\tau^{\mathrm{max}} = 1$, the estimation accuracy is quite close to the best possible (with $\tau^{\mathrm{max}} = \infty$). This property of the algorithm enhances its applicability since good estimates are obtained with little delay. Figure 6 plots these variables for the second agent.

6   Conclusion

We presented a distributed algorithm for mobile agents to estimate their positions by fusing their own displacement measurements with inter-agent relative position measurements. The algorithm is distributed in the sense each agent can estimate its own position by communication only with nearby agents. The algorithm computes an approximation of the centralized optimal (Kalman filter) estimates. The problem of distributed Kalman filtering for this application is reformulated as a problem of computing the BLUE estimates. The graph structure of the BLUE estimation problem, which makes it computable using iterative methods of solving linear equations, makes distributing the computations possible. With finite memory and limited number of iterations before new measurements are obtained, the algorithm produces an approximation of the Kalman filter estimates. As the memory of each agent and the number of iterations between each time step are increased, the approximation improves. Simulations show, however, that even with small memory size and a single iteration, the estimates are quite close to the centralized optimal. Simulations further show that the error covariances of the state estimates that the proposed distributed algorithm yield are significantly lower than what is possible by dead reckoning.

There are several aspects of the proposed algorithm that need further investigation. Although numerical simulations show that the estimates produced by the algorithm are close to the centralized optimal estimates, a precise characterization of the difference is lacking. In particular, it will be useful to understand the affect of the parameters T_{mem} and τ^{max} on the performance of the algorithm. Moreover, the evolution error covariance will depend on the number of agents and the measurement graph, which is determined by agents' motion. The relationship between the covariance and agent motion is a subject of future research.

References

[1] Borenstein, J., Everett, H.R., Feng, L., Wehe, D.: Mobile robot positioning: Sensors and techniques. Journal of Robotic Systems, Special Issue on Mobile Robots 14(4), 231–249 (1997)

[2] Nistér, D., Naroditsky, O., Bergen, J.R.: Visual odometry. In: Conference on Computer Vision and Pattern Recognition (CVPR 2004), pp. 652–659 (2004)

[3] Olson, C.F., Matthies, L.H., Schoppers, M., Maimone, M.W.: Rover navigation using stereo ego-motion. Robotics and Autonomous Systems 43(4), 215–229 (2003)

[4] Makadia, A., Daniilidis, K.: Correspondenceless ego-motion estimation using an imu. In: IEEE International Conference on Robotics and Automation, pp. 3534–3539 (2005)

[5] Oskiper, T., Zhu, Z., Samarasekera, S., Kumar, R.: Visual odometry system using multiple stereo cameras and inertial measurement unit. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2007), June 17-22, pp. 1–8 (2007)

[6] Kurazume, R., Nagata, S., Hirose, S.: Cooperative positioning with multiple robots. In: The IEEE International Conference in Robotics and Automation, pp. 1250–1257 (1994)

[7] Rekleitis, I.M., Dudek, G., Milios, E.E.: Multi-robot cooperative localization: a study of trade-offs between efficiency and accuracy. In: The IEEE/RSJ International Conference on Intelligent Robots and System, vol. 3, pp. 2690–2695 (2002)

[8] Mourikis, A.I., Roumeliotis, S.I.: Performance analysis of multirobot cooperative localization. IEEE Transactions on Robotics 22(4), 666–681 (2006)

[9] Roumeliotis, S.I., Bekey, G.A.: Distributed multirobot localization. IEEE Transactions on Robotics and Automation (5), 781–795 (2002)

[10] Mueller, S., Tsang, R.P., Ghosal, D.: Multipath routing in mobile ad hoc networks: Issues and challenges. In: Calzarossa, M.C., Gelenbe, E. (eds.) MASCOTS 2003. LNCS, vol. 2965, pp. 209–234. Springer, Heidelberg (2004)

[11] Spanos, D.P., Olfati-Saber, R., Murray, R.M.: Approximate distributed kalman filtering in sensor networks with quantifiable performance. In: 4th International Symposium on Information Processing in Sensor Networks (IPSN 2005) (2005)

[12] Alriksson, P., Rantzer, A.: Distributed kalman filtering using weighted averaging. In: 17th International Symposium on Mathematical Theory of Networks and Systems (MTNS) (2006)

[13] Olfati-Saber, R.: Distributed kalman filtering for sensor networks. In: 46th IEEE Conference on Decision and Control (December 2007)

[14] Carli, R., Chiuso, A., Schenato, L., Zampieri, S.: Distributed kalman filtering using consensus strategies. In: 46th IEEE Conference on Decision and Control (December 2007)

[15] Alriksson, P., Rantzer, A.: Experimental evaluation of a distributed kalman filter algorithm. In: 46th IEEE Conference on Decision and Control (December 2007)

[16] Zhang, P., Martonosi, M.: LOCALE: collaborative localization estimation for sparse mobile sensor networks. In: International Conference on Information Processing in Sensor Networks (IPSN), pp. 195–206 (2008)

[17] Mendel, J.M.: Lessons in Estimation Theory for Signal Processing, Communications and Control. Prentice-Hall, Englewood Cliffs (1995)

[18] Barooah, P., da Silva, N.M., Hespanha, J.P.: Distributed optimal estimation from relative measurements for localization and time synchronization. In: Gibbons, P.B., Abdelzaher, T., Aspnes, J., Rao, R. (eds.) DCOSS 2006. LNCS, vol. 4026, pp. 266–281. Springer, Heidelberg (2006)

[19] Barooah, P., Hespanha, J.P.: Estimation from relative measurements: Algorithms and scaling laws. IEEE Control Systems Magazine 27(4), 57–74 (2007)
[20] Rhodes, I.: A tutorial introduction to estimation and filtering. IEEE Transactions on Automatic Control 16, 688–706 (1971)
[21] Barooah, P.: Estimation and Control with Relative Measurements: Algorithms and Scaling Laws. PhD thesis, University of California, Santa Barbara (July 2007)
[22] Anderson, B.D.O., Moore, J.B.: Optimal Filtering. Dover, Mineola (2005)
[23] Barooah, P., Hespanha, J.P.: Distributed optimal estimation from relative measurements. In: Proceedings of the 3rd International Conference on Intelligent Sensing and Information Processing (ICISIP), December 2005, pp. 226–231 (2005)
[24] Golub, G.H., van Loan, C.F.: Matrix Computations, 3rd edn. Johns Hopkins University Press, Baltimore (1996)