Distributed localization and tracking with coordinated and uncoordinated motion models

Usman A. Khan, Soummya Kar, and José M. F. Moura

***Abstract*—This paper introduces the MDL algorithm for distributed localization in a network of mobile agents in $\mathbb{R}^m$, $m \geq 1$. The algorithm requires the dynamic agents with unknown locations to lie in the convex hull of at least $m+1$ mobile or static anchors that can track their positions over time perfectly. Under minimal assumptions on network connectivity and local triangulation at each agent, we show that the MDL algorithm leads to convergence of the estimated agent locations to the true locations under general deterministic motion models. In addition, MDL exhibits desirable tracking performance in random environments also and we explicitly characterize the steady state tracking error in such environments. A significant feature of the algorithm is its distributed nature, where each agent updates its current location estimate based on its neighbors' estimates and current local barycentric coordinates, computed using only local information. Simulations verify the effectiveness of the approach.**

I. Introduction

Localization is a fundamental problem in randomly deployed sensor networks. Usually the sensors have computational and communication limits due to power constraints. Hence, long distance communication and high order computation is infeasible. In [1], we presented a distributed localization algorithm, DILOC, in m-dimensional Euclidean space, $\mathbb{R}^m$, that only requires local communication. Furthermore, DILOC requires each sensor to compute a linear state update that is a function of the neighboring states. The state update is based on local barycentric coordinates that are computed using Cayley-Menger determinants, see [1] for details.

When the sensor network consists of mobile agents[1] e.g., robots, vehicles or cellphones, whose positions change as a function of time, simple DILOC updates are not applicable since the network configuration is not static. Hence, the state update (the optimal combining coefficients) changes with time as the neighborhood at each sensor changes. In the mobile case, the localization problem is not only to estimate the starting position of the agents, but also to track their motion.

All authors contributed equally to the paper. This work was partially supported by NSF under grants # ECS-0225449 and # CNS-0428404, and by ONR under grant # MURI-N000140710747.

UAK is with the Department of Electrical and Systems Engineering, University of Pennsylvania, 3401 Walnut St., Philadelphia, PA 19104. This work was completed when the first author was at Carnegie Mellon. SK and JMFM are with the Department of Electrical and Computer Engineering, Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh, PA 15213. {khanu@seas.upenn.edu}, {soummyak, moura}@ece.cmu.edu

[1]The term *agents*, *nodes* and *sensors* mean the same and are used interchangeably to denote the network elements with unknown locations.

In this paper, we present a distributed localization algorithm in m-dimensional Euclidean space, $\mathbb{R}^m$, that is application to mobile networks. In our setup, we assume that an arbitrary number of sensors with unknown locations lie in the convex hull of at leat $m+1$ anchors that precisely know their locations and motion (for instance, they may have a GPS unit). Our algorithm is distributed and requires only local distance information to compute the state updates at each time when the motion has taken place.

We present a broad motion model that captures several practical scenarios of coordinated and uncoordinated motion of mobile agents. As a motivation of coordinated motion, consider the anchors moving in a specified manner such that the underlying sensor network is guided in a desired direction. An example of uncoordinated motion is the motion of cell phones that move randomly is a given fixed region (or a cell). At each time step of the motion, the network configuration changes. Our algorithm is implemented on the same time scale as that of the motion. In this paper, we derive conditions under which our algorithm converges and establish minimal assumptions required for this setup.

Reference [2] discusses a tracking algorithm that tracks objects using wireless sensing devices sensing the objects. The wireless devices already know their locations and tracking is achieved by aggregating the sensed information. Another interesting tracking algorithm is provided in [3] that tracks humans on a tiled floor by using pressure sensors on the tiles and pressure and gait patterns of the humans. Our algorithm falls into token localization and tracking where the objects to be tracked possess a sensor (or an RFID tag) and helps the tracker. Reference [4] uses trilaterations to solve the localization/tracking problem that requires a large number of close by anchors to have a reasonable location estimate. Some other relevant references in this direction include [5], [6], [7], [8].

We now describe the rest of the paper. Section II covers background on DILOC. Section III presents problem formulation and our motion model. Section IV presents our algorithm, whereas Section V discusses the convergence proof. We then present simulations in Section VI and finally Section VII concludes the paper.

II. Background

In this section, we summarize the distributed localization algorithm, DILOC, in [1]. In $\mathbb{R}^m$ $(m \geq 1)$, let Ω be the set of M sensors with unknown locations and let κ be the set of $m+1$ anchors (that know their locations exactly) such that the sensors lie inside the convex hull of the anchors, i.e.,

$$\mathcal{C}(\Omega) \subset \mathcal{C}(\kappa), \tag{1}$$

where $\mathcal{C}(\cdot)$ denotes the convex hull formed by the elements of the set in its argument. Let Θ be the set of all of the nodes in the network, i.e., $\Theta = \Omega \cup \kappa$. We assume that for each sensor, $l \in \Omega$, there exists a triangulation set[2], $\Theta_l \subset \Theta$ containing exactly $m+1$ neighbors of l, such that sensor l lies in the convex hull of Θ_l.

The distribution localization algorithm, DILOC [1], at each sensor, $l \in \Omega$, is given by the convex linear combination:

$$\mathbf{x}_l(t+1) = \sum_{j\in\Omega\cap\Theta_l} p_{lj}\mathbf{x}_j(t) + \sum_{k\in\kappa\cap\Theta_l} b_{lk}\mathbf{u}_k, \qquad l \in \Omega, \tag{2}$$

where $\mathbf{x}_l = [x_{1,l}, \ldots, x_{m,l}]$ is the (location) coordinate row of the lth sensor (in $\mathbb{R}^m$), $\mathbf{u}_k = [u_{1,k}, \ldots, u_{m,k}]$ is the (location) coordinate row of the kth anchor, and p_{lj} and b_{lk} are the barycentric coordinates[3] [10], [11], of sensor l in terms of its neighbors in Θ_l.

Define the matrices:

$$\begin{aligned}
\mathbf{P} &= \{p_{lj}\}, && (3)\\
\mathbf{B} &= \{b_{lk}\}, && (4)\\
\mathbf{\Gamma} &= \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{B} & \mathbf{P} \end{bmatrix}, && (5)\\
\mathbf{X}(t) &= [\mathbf{x}_1(t)^T, \ldots, \mathbf{x}_M(t)^T]^T, && (6)\\
\mathbf{U}(t) &= \mathbf{U} = [\mathbf{u}_1^T, \ldots, \mathbf{u}_{m+1}^T]^T. && (7)
\end{aligned}$$

DILOC (2) can be written compactly as

$$\begin{bmatrix} \mathbf{U}(t+1) \\ \mathbf{X}(t+1) \end{bmatrix} = \mathbf{\Gamma} \begin{bmatrix} \mathbf{U}(t) \\ \mathbf{X}(t) \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{B} & \mathbf{P} \end{bmatrix} \begin{bmatrix} \mathbf{U}(t) \\ \mathbf{X}(t) \end{bmatrix}, \tag{8}$$

which gives the following expression for updating the sensor coordinates.

$$\begin{aligned}
\mathbf{X}(t+1) &= \mathbf{P}\mathbf{X}(t) + \mathbf{B}\mathbf{U}, && (9)\\
&= \mathbf{P}^{t+1}\mathbf{X}(0) + \sum_{k=0}^{t} \mathbf{P}^k \mathbf{B}\mathbf{U}. && (10)
\end{aligned}$$

Under appropriate assumptions on the connectivity of the underlying sensor network, we have

$$\rho(\mathbf{P}) < 1, \tag{11}$$

where ρ denotes the spectral radius, and DILOC converges to

$$\lim_{t\to\infty} \mathbf{X}(t+1) = (\mathbf{I} - \mathbf{P})^{-1}\mathbf{B}\mathbf{U}, \tag{12}$$

which are the exact sensor coordinates. The convergence proof and the required conditions are studied in great detail in [1].

[2]The identification of a triangulation set is an important step in the algorithm. A convex hull inclusion test is provided in [1] to test if a sensor lies in the convex hull of $m+1$ arbitrarily chosen sensors from its neighbors. Probabilistic bounds on the communication radius and the sensor deployment density are also provided in [1] to guarantee a successful triangulation with arbitrary high probability.

[3]The barycentric coordinates are computed using only the local distance information among the elements in the set, $\Theta_l \cup \{l\}$, by using the Cayley-Menger determinants [1], [9].

III. Problem formulation: Motion dynamics of mobile agents

We consider the following model for the motion dynamics of the agents in our network. Let $\mathbf{C}^*(t)$ denote the exact locations of the nodes at time t, i.e.,

$$\mathbf{C}^*(t) = \begin{bmatrix} \mathbf{U}^*(t) \\ \mathbf{X}^*(t) \end{bmatrix}. \tag{13}$$

The motion model, we consider, is as follows.

$$\mathbf{C}^*(t) = \mathbf{A}\mathbf{C}^*(t) + \mathbf{z}(t) + \mathbf{y}(t), \tag{14}$$

which can be partitioned into anchors and sensors as

$$\begin{aligned}
\begin{bmatrix} \mathbf{U}^*(t+1) \\ \mathbf{X}^*(t+1) \end{bmatrix} &= \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{A}_{ux} & \mathbf{A}_{xx} \end{bmatrix} \begin{bmatrix} \mathbf{U}^*(t) \\ \mathbf{X}^*(t) \end{bmatrix} \\
&+ \begin{bmatrix} \mathbf{z}_u(t) \\ \mathbf{z}_x(t) \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \mathbf{y}_x(t) \end{bmatrix}. && (15)
\end{aligned}$$

The $n \times n$ matrix[4] $\mathbf{A}$ relates the motion of a sensor to its neighbors such that the network may move in a coordinated fashion. The matrix $\mathbf{z}(t)$ is the deterministic drift added to the coordinates, whereas the matrix $\mathbf{y}(t)$ is the random drift with bounded norm. We state this explicitly as an assumption.

Assumption M.2.

$$\|\mathbf{y}(t)\| \leq a \qquad \mathbb{P} \text{ a.s.} \tag{16}$$

Although, the motion of the anchors is independent of the sensors (because of the upper right zero block in $\mathbf{A}$), the sensors move in such a way that they remain in the convex hull of the anchors. To guarantee this we may assume that $\mathbf{A}$ is stochastic and the drift matrices are such that the sensors do not leave the convex hull of the anchors. Since the anchors know their exact locations at all time, the random drift in the anchors motions is zero, i.e., $\mathbf{y}_u(t) = \mathbf{0},\ \forall t$. We further assume that each sensor l, knows the l row of the matrices $\mathbf{A}$ and $\mathbf{z}$. The above is the general form of our motion model that we consider in this paper. Some special cases can be derived that we elaborate below.

Uncoordinated motion in a fixed region: Consider $\mathbf{z}_u(t) = 0,\ \forall t$ and $\mathbf{A}_{ux} = \mathbf{0}, \mathbf{A}_{xx} = \mathbf{I}$. In this scenario, the anchors remain fixed and the sensors move randomly inside their convex hull. This can be thought of as the motion of wireless objects that move randomly inside a given region (or a cell). The drift at each sensor, $\mathbf{z}_l(t) + \mathbf{y}_l(t)$, is such that each sensor l does not leave the convex hull of the anchors.

Coordinated motion driven by anchors: Consider another scenario where $\mathbf{z}_x(t) = 0,\ \forall t$, and $\mathbf{A}_{ux} \neq \mathbf{0}, \mathbf{A}_{xx} \neq \mathbf{0}$ are such that each column of $\mathbf{A}_{ux}$ contains at least one non-zero element and the resulting $\mathbf{A}$ is a stochastic matrix. This form of a motion model is driven by anchors and the conditions on

[4]The identity matrix on the upper left block of $\mathbf{A}$ emphasizes that the anchors move independently of each other and the sensors. The motion of the anchors is captured by the deterministic drift $\mathbf{z}_u(t)$. In some problems of interest, like formation control of mobile vehicles, the anchors may be coupled among each other and the identity block can be replaced by a matrix with a specific structure, like a stochastic matrix. In that case our analysis will hold with obvious modifications in the proofs.

$\mathbf{A}_{ux}, \mathbf{A}_{xx}$, guarantee that the sensors move in a coordinated manner driven by the anchors.

Given the motion dynamics in eq. (14), we would like to estimate the and track the location of each sensor in a distributed manner.

IV. Algorithm and Assumptions

Consider the motion model for mobile agents presented in Section III. In the general case, we have

$$\begin{bmatrix} \mathbf{U}^*(t+1) \\ \mathbf{X}^*(t+1) \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{A_{ux}} & \mathbf{A_{xx}} \end{bmatrix} \begin{bmatrix} \mathbf{U}^*(t) \\ \mathbf{X}^*(t) \end{bmatrix} + \begin{bmatrix} \mathbf{z_u}(t) \\ \mathbf{z_x}(t) \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \mathbf{y_x}(t) \end{bmatrix}. \tag{17}$$

We now present the following algorithm for distributed localization of mobile sensors:

Algorithm MDL:

$$\mathbf{C}(t+1) = \mathbf{\Gamma}_{t+1} \left(\mathbf{A}\mathbf{C}(t) + \begin{bmatrix} \mathbf{z_u}(t) \\ \mathbf{z_x}(t) \end{bmatrix} \right) \tag{18}$$

Here: $\mathbf{C}(t) = [\mathbf{U}^T(t)\ \mathbf{X}^T(t)]^T$ corresponds to the location estimates of the agents at time t. Note that under the assumptions of the motion model, we have

$$\mathbf{U}(t) = \mathbf{U}^*(t), \quad \forall t. \tag{19}$$

Also, the time-varying matrix Γ_{t+1} denotes the matrix of local barycentric coordinates computed at time $t+1$ based on distance measurements corresponding to the network configuration $\mathbf{C}^*(t+1)$.

Some Remarks on MDL: Before proceeding to the convergence analysis of MDL, we present some discussion on the above update rule. Recall the algorithm DILOC for distributed localization in a network of static agents (Section II), where the update is of the form:

$$\mathbf{C}(t+1) = \mathbf{\Gamma}\mathbf{C}(t). \tag{20}$$

In that case, the network configuration (i.e., the inter-sensor distances) remains constant over time, and the matrix $\mathbf{\Gamma}$ of local barycentric coordinates does not change over time. The exact location $\mathbf{C}^*$ is a fixed point of the linear operator $\mathbf{\Gamma}$, i.e.,

$$\mathbf{C}^* = \mathbf{\Gamma}\mathbf{C}^*. \tag{21}$$

Under the assumptions of DILOC (connectivity, triangulation etc., see [1] for details), the operator $\mathbf{\Gamma}$ has the desired contraction properties and the update rule in eqn. (20) converges to the unique fixed point of $\mathbf{\Gamma}$, i.e., we have

$$\lim_{t\to\infty} \mathbf{C}(t) = \mathbf{C}^* \tag{22}$$

In the mobile network case, if $\mathbf{\Gamma}_{t+1}$ is the matrix of local barycentric coordinates computed on the basis of the network configuration at time $t+1$, we have the following fixed point condition:

$$\mathbf{C}^*(t+1) = \mathbf{\Gamma}_{t+1} \left(\mathbf{A}\mathbf{C}^*(t) + \begin{bmatrix} \mathbf{z_u}(t) \\ \mathbf{z_x}(t)+ \end{bmatrix} \right). \tag{23}$$

For a moment, assume that the unknown random perturbation $\mathbf{y_x}(t)$ is absent. In this case, under appropriate uniform contractive properties of the linear operators $\{\mathbf{\Gamma}_t\}_{t\geq 0}$ (to be detailed in the assumptions provided later), it is reasonable to expect that the MDL update should converge to the exact coordinates as $t \to \infty$, i.e.,

$$\lim_{t\to\infty} \|\mathbf{C}(t) - \mathbf{C}^*(t)\| = 0 \tag{24}$$

This is the key intuition behind the MDL update rule. In the general case, when random perturbations are present and are unpredictable, the update takes the form of eqn. (18). In this case, we expect a steady state convergence error, i.e.,

$$\limsup_{t\to\infty} \|\mathbf{C}(t) - \mathbf{C}^*(t)\| \leq \mathbf{e}^* \tag{25}$$

where the steady-state error depends on the distribution of the random perturbation, as shown later.

We now present the key assumption on network connectivity and triangulations at time t, required for establishing desired convergence properties of the MDL algorithm:

Assumption M.1 For every t, define the matrix $\mathbf{H}_t$ by:

$$\mathbf{H}_t = \mathbf{P}_t \mathbf{A}_{xx}. \tag{26}$$

Here $\mathbf{P}_t$ is the block coming from the natural decomposition of the matrix $\mathbf{\Gamma}_t$ as:

$$\mathbf{\Gamma}_t = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{B}_t & \mathbf{P}_t \end{bmatrix} \tag{27}$$

We make the following assumption on network connectivity and triangulation:

There exists $0 < \varepsilon < 1$, such that,

$$\mathbb{P}\left(\|\mathbf{H}_t\| \leq \varepsilon, \quad \forall t\right) = 1 \tag{28}$$

(note that the matrices $\mathbf{H}_t$ are now random, because of the random perturbations $\mathbf{y}_x(t)$ that affect the network configuration and thus $\mathbf{H}_t$.)

Since, $\|\mathbf{A}_{xx}\| \leq 1$ (being a sub-block of a stochastic matrix, $\mathbf{A}$), a sufficient condition for **M.1** to hold is:

$$\mathbb{P}\left(\|\mathbf{P}_t\| \leq \varepsilon, \quad \forall t\right) = 1. \tag{29}$$

The fact, that **M.1** is reasonable, is demonstrated from the fact, that, in the static case, under minimal assumptions on network connectivity and triangulation [1], we have $\|\mathbf{P}\| \leq \varepsilon$ for some $\varepsilon < 1$. In the dynamic case, assuming the network is sufficiently dense or the motion is coordinated, the network structure does not change drastically over time and hence Assumption **M.1**, which is a uniformity condition on the relative network structure is reasonable.

V. Convergence Analysis of MDL

In this section we present the convergence analysis of the algorithm MDL with the general motion model discussed in Section III and Assumption **M.1**.

Theorem 1: Consider the dynamic sensor motion model with $\mathbf{C}^*(t)$ denoting the sensor configuration at time t as given in eqn. (14). Let $\mathbf{C}(t)$ be the sensor state estimates generated by the distributed localization algorithm MDL (18). We then have

$$\mathbb{P}\left(\limsup_{t\to\infty}\|\mathbf{C}(t)-\mathbf{C}^*\|\leq\frac{a}{1-\varepsilon}\right)=1 \quad (30)$$

Proof: By construction of the local barycentric coordinates, it follows that for all t (recall eqn. (23)),

$$\begin{aligned}\mathbf{C}^*(t+1) &= \mathbf{\Gamma}_{t+1}\begin{bmatrix}\mathbf{I} & \mathbf{0}\\ \mathbf{A_{ux}} & \mathbf{A_{xx}}\end{bmatrix}\mathbf{C}(t)^* \\ &+ \mathbf{\Gamma}_{t+1}\begin{bmatrix}\mathbf{z_u}(t)\\ \mathbf{z_x}(t)\end{bmatrix}+\mathbf{\Gamma}_{t+1}\begin{bmatrix}\mathbf{0}\\ \mathbf{y_x}(t)\end{bmatrix}. \quad (31)\end{aligned}$$

Subtracting this from the MDL state update

$$\mathbf{C}(t+1)=\mathbf{\Gamma}_{t+1}\left(\mathbf{AC}(t)+\begin{bmatrix}\mathbf{z_u}(t)\\ \mathbf{z_x}(t)\end{bmatrix}\right) \quad (32)$$

we have

$$\mathbf{e}(t+1)=\mathbf{\Gamma}_{t+1}\begin{bmatrix}\mathbf{I} & \mathbf{0}\\ \mathbf{A_{ux}} & \mathbf{A_{xx}}\end{bmatrix}\mathbf{e}(t)-\mathbf{\Gamma}_{t+1}\begin{bmatrix}\mathbf{0}\\ \mathbf{y_x}(t)\end{bmatrix} \quad (33)$$

where:

$$\mathbf{e}(t)=\mathbf{C}(t)-\mathbf{C}^*(t) \quad (34)$$

is the matrix of location estimation errors at time t. Decomposing $\mathbf{e}(t)$ into sensors and anchors, we have

$$\mathbf{e}(t)=\begin{bmatrix}\mathbf{e_u}(t)\\ \mathbf{e_x}(t)\end{bmatrix}=\begin{bmatrix}\mathbf{0}\\ \mathbf{e_x}(t)\end{bmatrix} \quad (35)$$

The fact, that $\mathbf{e_u}(t)=0$, follows from $\mathbf{U}(t)=\mathbf{U}^*(t)$, as the anchors know their locations exactly at all time t.

Multiplying out the various terms in eqn. (33) we then have the following update rule for $\mathbf{e}(t)$:

$$\mathbf{e_u}(t)=\mathbf{0},\quad\forall t \quad (36)$$

and

$$\mathbf{e_x}(t+1)=\mathbf{H}_{t+1}\mathbf{e_x}(t)-\mathbf{P}_{t+1}\mathbf{y_x}(t),\quad\forall t \quad (37)$$

where $\mathbf{H}_t$ is defined in eqn. (26). Continuing the recursion for $\mathbf{e_x}(t)$, we have

$$\begin{aligned}\mathbf{e_x}(t) &= \left(\prod_{k=1}^{t}\mathbf{H}_k\right)\mathbf{e_x}(0) \\ &- \sum_{k=0}^{t-1}\left(\prod_{j=t-k+2}^{t}\mathbf{H}_j\right)\mathbf{P}_{t-k+1}\mathbf{y_x}(t-k) \quad (38)\end{aligned}$$

We thus have

$$\begin{aligned}\|\mathbf{e_x}(t)\| &\leq \|\mathbf{e_x}(0)\|\prod_{k=1}^{t}\|\mathbf{H}_k\| \quad (39)\\ &+ \sum_{k=0}^{t-1}\|\mathbf{P}_{t-k+1}\|\,\|\mathbf{y_x}(t-k)\|\left(\prod_{j=t-k+2}^{t}\|\mathbf{H}_j\|\right).\end{aligned}$$

Under Assumptions **M.1, M.2**, we have for all t

$$\|\mathbf{H}_t\|\leq\varepsilon,\quad\mathbb{P}\text{ a.s.} \quad (40)$$

$$\|\mathbf{y_x}(t)\|\leq a,\quad\mathbb{P}\text{ a.s.} \quad (41)$$

Also, by construction, the matrix $\mathbf{P}_t$ is substochastic for all t and hence

$$\|\mathbf{P}_t\|\leq 1,\quad\mathbb{P}\text{ a.s.} \quad (42)$$

The following then holds $\mathbb{P}$ a.s. from eqn. (39)

$$\|\mathbf{e_x}(t)\|\leq\varepsilon^t\|\mathbf{e_x}(0)\|+a\sum_{k=0}^{t-1}\varepsilon^k \quad (43)$$

Taking the limit as $t\to\infty$ and noting that $0<\varepsilon<1$, we have $\mathbb{P}$ a.s.

$$\limsup_{t\to\infty}\|\mathbf{e_x}(t)\|\leq\frac{a}{1-\varepsilon} \quad (44)$$

The result then follows from eqns. (36) and (44). ■

Remarks: The convergence results are established under minimal conditions on network connectivity and triangulation, embedded in the Assumption **M.1**, i.e.,

$$\mathbb{P}\left(\|\mathbf{H}_t\|\leq\varepsilon,\quad\forall t\right)=1, \quad (45)$$

for some $\varepsilon<1$. The problem of finding the right value of ε is also of interest and conveys significant information on the convergence rate of the algorithm. Such a characterization of ε depends on the specifics of the motion model, for example, the geometry of sensor deployment, the various model matrices and the distribution of the random drift. This is an interesting problem in its own right and we intend to investigate this in the future.

In the absence of the random drift (but with non-zero deterministic drift), the assumption **M.1** on network connectivity and local triangulation at all times can be relaxed and we may work with the much weaker assumption of successful local triangulation infinitely often (i.o.), i.e.,

$$\mathbb{P}\left(\|\mathbf{H}_t\|\leq\varepsilon,\quad\text{i.o. } t\right)=1 \quad (46)$$

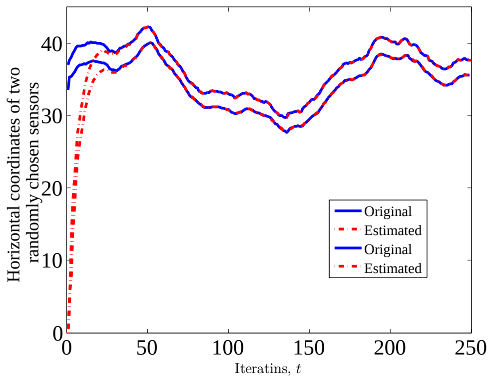
for some $\varepsilon<1$.

VI. SIMULATIONS

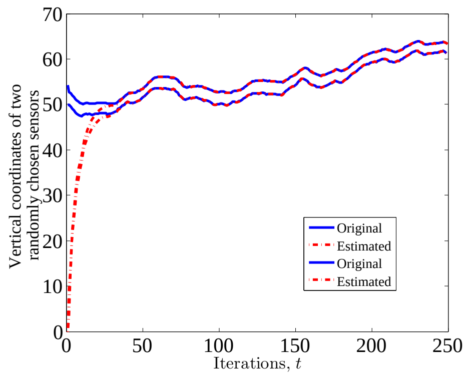
In this section, we present numerical simulations for the MDL algorithm. Consider a network of $n=50$ nodes in $\mathbb{R}^2$ (plane), where we have $m+1=3$ anchors and $M=47$ sensors. The sensors lie in the convex hull of the anchors. We assume a coordinated motion model on the sensors. In particular, we choose $\mathbf{A}_{ux}$ and $\mathbf{A}_{xx}$ such that $\mathbf{A}$ is a stochastic matrix and there is no drift on the sensors' motion, i.e., $\mathbf{z}_x=\mathbf{0}$. The anchors move with a known drift and the sensors move as a convex combination of their neighbors' movement. This guarantees that the sensors remain in the convex hull.

Fig. 2(a) shows the motion of the anchors and two randomly chosen sensors. Fig. 1(a) and fig. 1(b) show the horizontal and vertical coordinates of two randomly chosen sensors as solid lines. The estimated coordinates with the MDL algorithm are plotted as dashed lines. The initial condition of the algorithm are set to zero. Notice that MDL catches up with the motion and then tracks the motion of the sensors. Fig. 2(b) shows the normalized mean squared error, i.e.,

$$\text{MSE}_t=\frac{1}{M}\sum_{l=1}^{M}\sum_{i=1}^{m}\left(\mathbf{x}_{l,i}(t)-\mathbf{x}^*_{l,i}\right)^2, \quad (47)$$

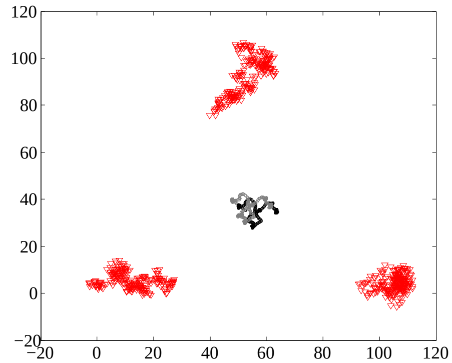


(a)

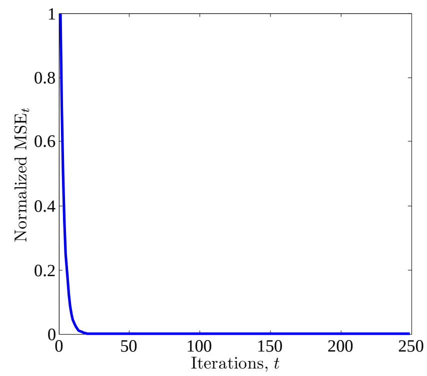


(b)

Fig. 1. Coordinated motion with known drift: (a) The horizontal coordinates of two randomly chosen sensors and the MDL estimates. (b) The vertical coordinates of two randomly chosen sensors and the MDL estimates.



(a)



(b)

Fig. 2. Coordinated motion with known drift: (a) The motion of the anchors (shown as nablas) and two randomly chosen sensors (out of $N = 50$) shown as gray and black. (b) The normalized mean squared error.

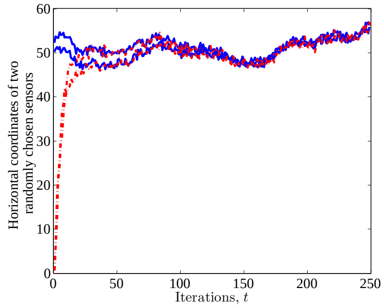
in the estimated coordinates.

In the next experiment, we consider the same model as before but introduce a random drift to the sensors motion. In particular, we choose a uniform random variable on the interval $[-2\ 2]$ as random drift in each coordinate at each sensor. For a network of $N = 50$ nodes, we show the resulting estimates as dashed lines against the original coordinates as solid lines in Fig. 3(a) and Fig. 3(b) for two randomly chosen sensors. Fig. 4(a) shows the motion of the anchors and these sensors. We then plot the normalized mean squared error in Fig. 4(b). Notice that the estimates catch up the exact sensor coordinates but there is an error due to the random drift. We use log scale to show the steady state error.

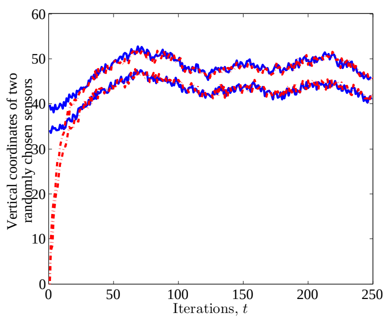
Finally, we present a simulation for uncoordinated motion in a fixed region. We choose $\mathbf{A} = \mathbf{I}$ and choose zero known drift, i.e., $\mathbf{z} = \mathbf{0}$. We choose $\mathbf{y}_x$ such that each of its element is a uniform random variable on the interval $[-2\ 2]$. Fig. 6(a) shows the motion for two randomly chosen sensors (notice that the anchors are fixed and do not move). For a network of $N = 50$ nodes, we show the resulting estimates as dashed lines against the original coordinates as solid lines in Fig. 5(a) and Fig. 5(b) for two randomly chosen sensors. We then plot the normalized mean squared error in Fig. 6(b). Notice that the estimates catch up the exact sensor coordinates but there is an error due to the random drift. We use log scale to show the steady state error.

VII. Conclusions

We present a distributed algorithm for localization and tracking of wireless devices with unknown locations in m-dimensional Euclidean spaces. Our algorithm requires the devices to be tracked to lie in the convex hull of at least $m+1$ anchors that know their exact locations. We present a general model to capture the motion dynamics and discuss realistic practical cases where such motion can occur as special cases of this general model. Under minimal assumptions on network connectivity and deployment, we show that algorithm converges to exact locations in the mobile case when there is no random drift in the motion. Under random drift, we bound the steady state error that arises due to this randomness.

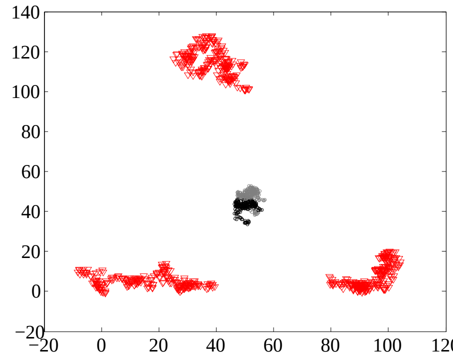


(a)

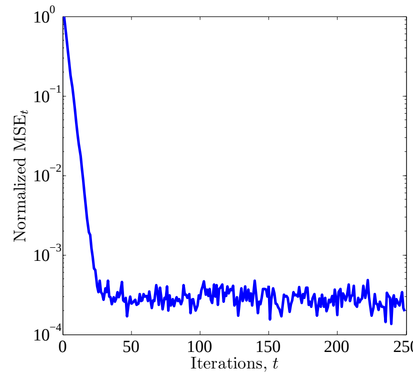


(b)

Fig. 3. Coordinated motion with random drift: (a) The horizontal coordinates of two randomly chosen sensors and the MDL estimates. (b) The vertical coordinates of two randomly chosen sensors and the MDL estimates.



(a)



(b)

Fig. 4. Coordinated motion with random drift: (a) The motion of the anchors (shown as nablas) and two randomly chosen sensors (out of $N = 50$) shown as gray and black. (b) The log-normalized mean squared error.

References

[1] Usman A. Khan, Soummya Kar, and José M. F. Moura, "Distributed sensor localization in random environments using minimalnumber of anchor nodes," *IEEE Transactions on Signal Processing*, vol. 57, no. 5, pp. 2000–2016, May 2009.

[2] A. Arora, S. Bapat P. Dutta, V. Kulathumani, H. Zhang, V. Naik, V. Mittal, H. Cao, M. Gouda, Y. Choi, T. Herman, S. Kulkarni, U. A. M. Nesterenko, A. Vora, and M. Miyashita, "Line in the sand: A wireless sensor network for target detection, classification, and tracking," in *Report OSU-CISRC-12/03-TR71, Ohio State University*, 2003.

[3] R. J. Orr and G. D. Abowd, "The smart floor: a mechanism for natural user identification and tracking," in *CHI 00 extended abstracts on Human factors in computing systems*, New York, NY, 2000, pp. 275–276.

[4] A. Chakraborty N. B. Priyantha and H. Balakrishnan, "The cricket location-support system," in *Proceedings of the 6th annual international conference on Mobile computing and networking*, New York, NY, 2000, pp. 32–43.

[5] P. Bahl and V. N. Padmanabhan, "Radar: An in-building rf-based user location and tracking system," in *INFOCOM '00*, 2000, p. 775784.

[6] D. Moore, J. Leonard, D. Rus, and S. Teller, "Robust distributed network localization with noisy range measurementsh," in *Proceedings of the Second ACM Conference on Embedded Networked Sensor Systems*, Baltimore, MD, Nov. 3-5 2004, pp. 50–61.

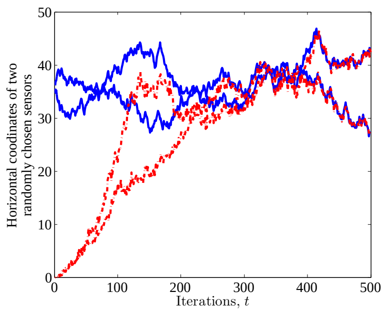
[7] A. Smith, H. Balakrishnan, M. Goraczko, and N. Priyantha, "Tracking moving devices with the cricket location system," in *Proceedings of the 2nd international conference on Mobile systems, applications, and services*, New York, NY, 2004, p. 190202.

[8] K. Lorincz and M. Welsh, "Motetrack: A robust, decentralized approach to rf-based location tracking," in *Proceedings of the International Workshop on Location and Context- Awareness*, May 2005.

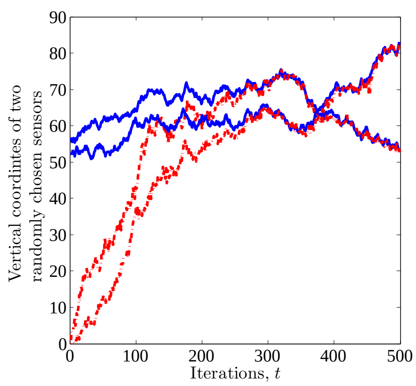
[9] M. J. Sippl and H. A. Scheraga, "Cayley–Menger coordinates," *Proceedings of the National Academy of Sciences of U.S.A.*, vol. 83, no. 8, pp. 2283–2287, Apr. 1986.

[10] G. Springer, *Introduction to Riemann Surfaces*, Addison-Wesley, Reading, MA, 1957.

[11] J. Hocking and G. Young, *Topology*, Addison-Wesley, MA, 1961.

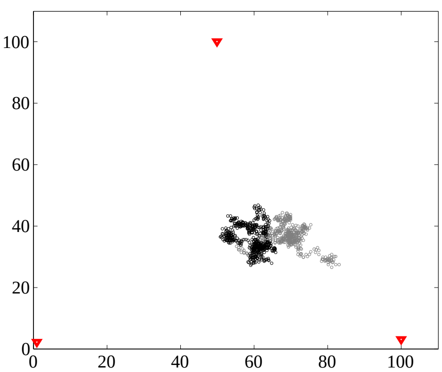


(a)

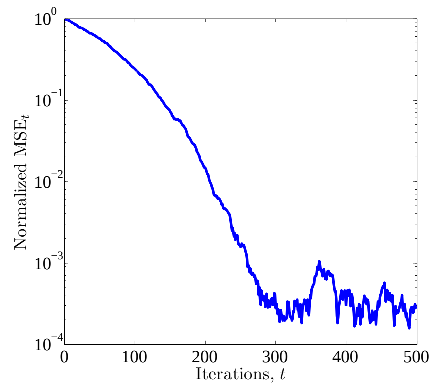


(b)

Fig. 5. Uncoordinated motion in a fixed region with random drift: (a) The horizontal coordinates of two randomly chosen sensors and the MDL estimates. (b) The vertical coordinates of two randomly chosen sensors and the MDL estimates.



(a)



(b)

Fig. 6. Uncoordinated motion in a fixed region with random drift: (a) The motion of two randomly chosen sensors (out of $N = 50$) shown as gray and black. (b) The log-normalized mean squared error.