Distributed Localization of Formations from Relative Range Measurements[*]

Giuseppe C. Calafiore[†], Luca Carlone[‡], and Mingzhu Wei[§]

Abstract

This paper considers the problem of distributed estimation of the position of autonomous agents in a networked formation, using noisy measurements of relative distances between agents. The underlying geometrical problem has been studied quite extensively in various fields, ranging from molecular biology to robotics, and it is known to lead to a hard non-convex optimization problem. Centralized algorithms however do exist that work reasonably well in finding local or global minimizers for this problem (e.g. semidefinite programming relaxations). Here, we explore a decentralized, or *distributed*, approach for localization from range measurements, and we propose a computational scheme based on a distributed gradient algorithm with Barzilai-Borwein stepsizes. The advantage of this distributed approach is that each agent may autonomously compute its position estimate, exchanging information only with its *neighbors*, without need of communicating with a central station and without needing complete knowledge of the network structure. The ideal decentralized scheme is proved to converge, under an hypothesis of network connectivity, to the same solution as its centralized counterpart. Performance of the actual implementation of this algorithm is analyzed via numerical simulations.

Keywords: Network localization, range measurements, distributed algorithms, consensus.

1 Introduction

Networked systems play an important role in many technological fields due to their capability of acquiring information on wide areas in a decentralized and autonomous way. Advances in wireless communication further increased the potentiality of multi agent networks in applications such as mobile robotics, target tracking, environmental monitoring and surveillance. Most of the mentioned applications require that each node has precise knowledge of its geometric position, since actions and observations are often location-dependent. For example, monitoring and surveillance become meaningless if it is not possible to associate to the observation the corresponding geographic information; formation control requires knowledge of agent positions; location aware routing benefits from the position information for optimizing the flow of information through the network, etc. Although agents' positions can be obtained through direct measures (e.g., GPS), in many cases this solution is not feasible due to technological constraints such as cost and power consumption. As a consequence, research effort has been recently oriented towards designing indirect methods for reconstructing the absolute node positions from relative and partial measurements. The problem of determining node positions is generically known as *network localization*: the absolute node positions (with respect to a local or global reference frame) need be estimated from partial relative measurements between nodes, that is, each node may measure the relative position from a set of *neighbor* nodes, and the global absolute positions of all nodes need be deduced from this information.

Localization problems for planar networks can be classified depending on the type of available relative measurements. A recent survey on both technological and algorithmic aspects can be

[*]This work was funded by MACP4LOG grant (RU/02/26) from Piemonte Region, Italy and PRIN 20087W5P2K grant from the Italian Ministry of University and Research.

[†]Dipartimento di Automatica e Informatica, Politecnico di Torino - Italy. `giuseppe.calafiore@polito.it`

[‡]CSPP, Laboratorio di Meccatronica, Politecnico di Torino - Italy. `luca.carlone@polito.it`

[§]Dipartimento di Automatica e Informatica, Politecnico di Torino - Italy. `mingzhu.wei@polito.it` found in [17]. If relative coordinates between nodes are measured (or, equivalently, relative angles and distances are measured) network localization can be formulated as a linear estimation problem, and efficiently solved using a Least-Squares approach; see, e.g., [1, 2] and the references therein. The case in which angle-only measurements are available to the nodes was pioneered by Stanfield [28] and further generalized in [9]. The latter is often referred to as *bearing localization* and can be attacked via maximum likelihood estimation as described in [17]. The most common setup is probably the one in which each node can measure only distances from a subset of other nodes in the formation. This case, which we shall name *range localization*, has been treated in various settings, see for instance [8] and [26]. Range localization naturally leads to a strongly NP-hard non-linear and non-convex optimization problem (see [25]), in which convergence to a global solution cannot in general be guaranteed. Convex relaxation methods based on semidefinite programming (SDP) have been recently proposed, who provably converge to the global solution, in the noiseless case, see, e.g., [4]. Although it achieves high accuracy in estimating sensor locations, the speed of the SDP approach is not satisfactory for practical applications with medium/high number of nodes. However, the SDP solution can also be used as a starting point for steepest descent based local optimization techniques that can further refine the SDP solution. The distributed gradient algorithm described in this paper can be thus also be employed for refinement of SDP-based solutions, see Section 5.3.

The actual reconstruction of a unique network configuration from range measurements is possible only under particular hypotheses on the topology of the networked formation (graph rigidity, see [8]). Moreover, localization in an absolute reference frame requires that a subset of the nodes (*anchor nodes* or *beacons*) already knows its exact location in the external reference frame. Otherwise, localization is possible only up to an arbitrary roto-translation. This latter setup is referred to as *anchor-free* localization; see, e.g., [22].

If all relative measurements are gathered to some "central elaboration unit" which performs estimation over the whole network, the corresponding localization technique is said to be *centralized*. This is the approach that one implicitly assumes when writing and solving an optimization problem: all the data that is relevant for the problem description is available to the problem solver. However, such an assumption may be unrealistic in situations where data and information is actually stored locally at the agents, and where the computations can also be executed locally at the individual agents' level. Approaching such problems in a centralized way would require that each agent sends its information to the central elaboration unit, who solves the (usually large-scale) problem and transmits back the results to each agent. This may of course be highly undesirable due to intensive communication load among the central units and the agents. Moreover, since all the computation is performed by a single unit, for large networks the computational effort can be just too intensive. Also, the system is fragile, since failure in the central elaboration unit or in communication would compromise the functioning of the whole network. According to these considerations, *distributed* approaches are desirable for solving network localization. In a distributed setup each node communicates only with its *neighbors*, and performs local computations in order to obtain an estimate of its own position. As a consequence, the communication burden is equally spread over the network, the computation is decentralized and entrusted to each agent, improving both efficiency and robustness of the estimation process. First attempts to reduce the computational effort of localization by breaking the problem into smaller subproblems traces back to [12], in which a divide-and-conquer algorithm is proposed. Similar considerations are drawn in [18], where clustering is applied to the network in order to properly reconstruct network configuration. Recent approaches to the problem include [31] and [14]. The former faces the problem in the anchor-free setting, whereas the latter makes use of barycentric coordinates for localizing the nodes, under the hypothesis that non-anchor nodes lie in the convex hull of anchors. The distributed approach described in [14] guarantees global convergence, under the additional hypothesis that each node can increase his radius of communication until it gets contained in the convex hull of its neighbors. Reference [15] adds further convergence analysis on previous work of the authors, although the use of barycentric coordinates still requires assumptions on network topology.

In this paper, we consider the problem of distributed network localization with range only measurements, and we develop a distributed gradient algorithm that uses Barzilai-Borwein stepsizes, where the stepsizes are computed in the network in a decentralized way through consensus iterations. The idealized version of this algorithm has two time scales: per each outer gradient iteration, an (ideally infinite) series of inner communication iterations are performed among nodes in order to reach consensus on the stepsize parameter. In practice, consensus is reached at geometric rate, hence a finite and limited number of inner iterations suffices to compute the stepsize. Once the network topology is known, the required number of inner iterations can actually be fixed a priory, for given desired accuracy level. This technique is completely decentralized and it appears to be scalable for large networks. Moreover it presents advantages over the state-of-the-art techniques in terms of computational and communication requirements. As it is common to all gradient-based methods (either centralized or distributed), convergence of the algorithm can only be guaranteed for initial conditions that are close enough to the actual solution. For this reason, the proposed approach appears to be well suited as a refinement technique for centralized methods (such as SDP relaxations), or in dynamic situations where the nodes location is iteratively obtained as a small perturbation of a previously known location.

The rest of this paper is organized as follows. The problem setup is presented in Section 2. A centralized version of the gradient scheme is reviewed in Section 3. The main contribution of the paper is contained in Section 4, where the distributed range localization method is presented. Numerical experiments are reported in Section 5, and conclusions are drawn in Section 6.

Notation. I_n denotes the $n \times n$ identity matrix, $\mathbf{1}_n$ denotes a (column) vector of all ones of dimension n, $\mathbf{0}_n$ denotes a vector of all zeros of dimension n, $e_i \in \mathbb{R}^n$ denotes a vector with all zero entries, except for the i-th position, which is equal to one. We denote with $\lfloor x \rfloor$ the largest integer smaller than or equal to x. Subscripts with dimensions may be omitted when they can be easily inferred from context.

For a matrix X, X_{ij} denotes the element of X in row i and column j, and $X^\top$ denotes the transpose of X. $X > 0$ (resp. $X \geq 0$) denotes a positive (resp. non-negative) matrix, that is a matrix with all positive (resp. non-negative) entries. $\|X\|$ denotes the spectral (maximum singular value) norm of X, or the standard Euclidean norm, in case of vectors. For a square matrix $X \in \mathbb{R}^{n,n}$, we denote with $\sigma(X) = \{\lambda_1(X), \ldots, \lambda_n(X)\}$ the set of eigenvalues, or *spectrum*, of X, and with $\rho(X)$ the spectral radius: $\rho(X) \doteq \max_{i=1,\ldots,n} |\lambda_i(X)|$, where $\lambda_i(X)$, $i = 1, \ldots, n$, are the eigenvalues of X ordered with decreasing modulus, i.e. $\rho(X) = |\lambda_1(X)| \geq |\lambda_2(X)| \geq \cdots \geq |\lambda_n(X)|$.

2 Problem setup

Let $\mathcal{V} = \{v_1, \ldots, v_n\}$ be a set of n nodes (representing sensors, agents, robots, vehicles, etc.), and let $\mathcal{P} = \{p_1, \ldots, p_n\}$ denote a corresponding set of positions on the Cartesian plane, where $p_i = [x_i \ y_i]^\top \in \mathbb{R}^2$ are the coordinates of the i-th node. We shall call $\mathcal{P}$ a *configuration* of nodes. Suppose that some pairs of nodes, say nodes (i, j), have the possibility of measuring the relative distance between them:

$$d_{ij}^2 = \|p_i - p_j\|^2 + \epsilon_{ij},$$

where ϵ_{ij} represents the residual error between the true (squared) distance and its measure d_{ij}^2. We denote with $\mathcal{E}$ the set of unordered node pairs (i, j) such that a distance measurement exists between i and j. Our objective is to determine a node configuration $\{p_1, \ldots, p_n\}$ that minimizes a Least-Squares goodness of fit criterion

$$f(p) = \frac{1}{2} \sum_{(i,j)\in\mathcal{E}} \left(\|p_i - p_j\|^2 - d_{ij}^2 \right)^2 . \tag{1}$$

When the global minimum of f is zero we say that exact matching is achieved, that is a configuration $\{p_1, \ldots, p_n\}$ is found that exactly matches the given distance measurements:

$$\|p_i - p_j\|^2 = d_{ij}^2, \quad \forall\, (i, j) \in \mathcal{E}.$$

Otherwise, no geometric node configuration can exactly match the given range data, and we say that approximate matching is achieved by the optimal configuration.

Actual recovery of the absolute geometric node positions from distance measurements is only possible if the node configuration is *generically globally rigid* (*ggr*), [8]. In this case, the objective function in (1) has a unique global minimum, if the positions of at least three nodes is known and fixed in advance (anchor nodes, or beacons), or it has several equivalent global minima corresponding to congruence transformations (roto-translations) of the configuration, if no anchors are specified. If the configuration is not *ggr*, instead, there exist many different geometric configurations (also called *flexes*) that match exactly or approximately the distance data and that correspond to equivalent global minima of the cost f. In this work, we focus on a *distributed* numerical technique to compute a local or global minimum of f, in the neighborhood of some given initial position estimate. In our approach we treat under the same framework both anchor-based and anchor-free localization problems. In particular, when anchor nodes are specified at fixed positions, we just set the respective node position variables to the given values, and eliminate these variables from the optimization.

In the next section, we review a centralized gradient based approach for locally solving the localization problem (1). Then, Section 4 contains the main contribution of this paper, providing a distributed version of the localization algorithm.

3 A centralized gradient-based localization algorithm

Let $p \doteq [p_1^\top \; p_2^\top \; \cdots \; p_n^\top]^\top$, where $p_i^\top = [x_i \; y_i]$, denote the vector of node positions. The minimization objective (1) is rewritten as

$$f(p) = \frac{1}{2} \sum_{(i,j)\in\mathcal{E}} g_{ij}^2(p), \quad g_{ij}(p) \doteq \|p_i - p_j\|^2 - d_{ij}^2, \tag{2}$$

and we let $p^{(0)}$ denote the vector of initial position estimates. This vector of initial guess can be available either as the result of a preliminary optimization phase (e.g., via SDP relaxation), or as the output of our position estimation algorithm itself at a previous time period (this may be the case, for instance, in the context of iterative dynamic tracking of a moving formation). We next describe a centralized iterative method to determine a local minimum of the cost function, starting from $p^{(0)}$. We remark that in a centralized methods the whole vector p must be stored and updated by some "central computing unit" that also has full knowledge of all the distance measurements.

3.1 A gradient-based method

The most basic iterative method for finding a local minimizer of $f(p)$ is the so called gradient algorithm. Let $p^{(\tau)}$ be the configuration computed by the algorithm at iteration τ, being $p^{(0)}$ the given initial configuration: at each iteration the solution is updated according to the rule

$$p^{(\tau+1)} = p^{(\tau)} - \alpha_\tau \nabla f(p^{(\tau)}), \tag{3}$$

where α_τ is the step length, which may be computed at each iteration via exact or approximate line search, and where

$$\nabla f(p) \;=\; \sum_{(i,j)\in\mathcal{E}} g_{ij}(p) \nabla g_{ij}(p) \tag{4}$$

where gradient ∇g_{ij} is a row vector of n blocks, with each block composed of two entries, thus $2n$ entries in total, and with the only non-zero terms corresponding to the blocks in position i and j:

$$\nabla g_{ij}(p) \;=\; 2[\mathbf{0}_2^\top \; \cdots \; \mathbf{0}_2^\top \; (p_i - p_j)^\top \; \mathbf{0}_2^\top \; \cdots \; \mathbf{0}_2^\top \; (p_j - p_i)^\top \; \mathbf{0}_2^\top \; \cdots \; \mathbf{0}_2^\top].$$

The gradient method is guaranteed to converge to a local minimizer whenever $\{p : f(p) \le f(p^{(0)})\}$ is bounded and the step lengths satisfy the Wolfe conditions, see, e.g., [19]. Although the rate of convergence of the gradient method can be poor, we are interested in this method here since it requires first-order only information (no Hessian need be computed), and it can be suitably adapted to a distributed implementation, as discussed in Section 4.2.

The Barzilai and Borwein scheme A critical part of the gradient algorithm is the computation of suitable stepsizes α_τ. Exact line search prescribes to compute the stepsize by solving the unidimensional optimization problem

$$\min_{\alpha} f(p^{(\tau)} - \alpha \nabla f(p^{(\tau)})).$$

Determining the optimal α can however be costly in terms of evaluations of objective and gradient. Moreover, an approach based on exact or approximate line search is not suitable for the decentralized implementation that we are seeking. Barzilai and Borwein (BB) in [3] proposed an alternative simple and effective technique for selection of the step size, which requires few storage and inexpensive computations. The BB approach prescribes to compute the step size according to the formula

$$\alpha_\tau = \frac{\|p^{(\tau)} - p^{(\tau-1)}\|^2}{(p^{(\tau)} - p^{(\tau-1)})^\top (\nabla f(p^{(\tau)}) - \nabla f(p^{(\tau-1)}))}, \tag{5}$$

hence no line searches or matrix computations are required to determine α_τ. The gradient algorithm with BB stepsizes is particularly well suited for large scale problems, and it is guaranteed to converge to the global optimum in the convex quadratic case, see [10, 23]. For the nonconvex situation - as it is the case for any algorithm - global convergence cannot in general be guaranteed. As we shall show in Section 4.2, the use of BB stepsizes permits a fully distributed implementation of the gradient method.

4 The distributed localization method

We shall use graph formalism to describe the distributed structure of the localization problem and of the corresponding algorithm. The next section introduces some useful concepts and definitions related to graphs.

4.1 Graph preliminaries

A *graph* $\mathcal{G}(\mathcal{V}, \mathcal{E})$ is induced by the set of nodes $\mathcal{V} = \{v_1, \ldots, v_n\}$ representing graph vertices, and by pairs of nodes (v_i, v_j) such that a relative distance measurement exists between v_i and v_j, representing graph edges in the edge set $\mathcal{E}$. Throughout this paper we shall assume that whenever a distance measurement exists between two nodes, then these two nodes can communicate bidirectionally. Therefore, graph $\mathcal{G}$ describes both the measurement as well as the communication structure of the node formation. For each $i \in \mathcal{V}$, the set of *neighbors* $\mathcal{N}_i$ of node i is defined as the set of nodes with which node i can exchange information, that is $\mathcal{N}_i = \{j \in \mathcal{V}, j \neq i : (i,j) \in \mathcal{E}\}$.

We recall that graph $\mathcal{G}$ is said to be *connected* if a path (i.e. a sequence of edges) exists between any pair of nodes of $\mathcal{G}$, see standard references on graph theory, such as [7, 11]. There is a natural way of associating a non-negative matrix A to a graph $\mathcal{G}$, by considering matrices whose (i,j) entry is positive whenever an edge exists between nodes (i,j) in the corresponding graph, and it is zero otherwise. Let us thus introduce the following set of non-negative matrices with positive diagonal entries:

$$\mathcal{M} \doteq \{A \in \mathbb{R}^{n,n} : A \geq 0,\ A_{ii} > 0,\ i = 1, \ldots, n\}.$$

We have the following definition.

Definition 1 *For $A \in \mathcal{M}$, we say that the matrix/graph pair $(A, \mathcal{G}(\mathcal{V}, \mathcal{E}))$ is* compatible *if $A_{ij} > 0 \Leftrightarrow (i,j) \in \mathcal{E}$.*

The notion of connectedness of a graph is related to the notion of *primitiveness* of a matrix compatible with that graph. A square matrix $A \geq 0$ is said to be *primitive* if there exist an integer $m \geq 1$ such that $A^m > 0$. The least integer m such that $A^m > 0$ is called the *index of primitivity* of A. If A is primitive then $\rho(A)$ is an algebraically simple eigenvalue of A and the eigenspace associated with this eigenvalue is one dimensional. The following theorem can be readily established (using for instance Theorem 6.2.24, Theorem 8.5.2 and Lemma 8.5.5 of [13]).

Theorem 1 *Let $A \in \mathcal{M}$ such that $(A, \mathcal{G}(\mathcal{V}, \mathcal{E}))$ is a compatible pair. Then A is primitive if and only if $\mathcal{G}$ is connected.*

Consider a subset of $\mathcal{M}$ composed of matrices in which the sum over each row is equal to one (such matrices are usually called (row) *stochastic*):

$$\mathcal{M}_{\mathrm{s}} \doteq \{A \in \mathcal{M} : \, A\mathbf{1} = \mathbf{1}\}.$$

For $A \in \mathcal{M}_{\mathrm{s}}$ we have that 1 is an eigenvalue of A. Observe that the spectral radius of a matrix is no larger than any norm of the matrix (see Theorem 5.6.9 of [13]), hence by taking the ℓ_∞-induced norm we have that for $A \in \mathcal{M}_{\mathrm{s}}$ it holds that

$$\rho(A) \leq \|A\|_\infty = \max_{i=1,\ldots,n} \sum_{j=1}^{n} |a_{ij}| = 1.$$

Since 1 is an eigenvalue of A it therefore must be $\rho(A) = 1$. It also follows that if $A \in \mathcal{M}_{\mathrm{s}}$ is primitive then A has a unique (i.e. an algebraically simple) eigenvalue at 1, hence all other eigenvalues have modulus strictly smaller than 1 and the fixed-point subspace

$$\mathcal{I}(A) \doteq \{x \in \mathbb{R}^n : \, Ax = x\}$$

is one dimensional. Now consider the subset of $\mathcal{M}_{\mathrm{s}}$ formed by symmetric matrices:

$$\mathcal{M}_{\mathrm{ss}} \doteq \{A \in \mathcal{M}_{\mathrm{s}} : \, A = A^\top\}$$

(a matrix $A \in \mathcal{M}_{\mathrm{ss}}$ is symmetric, non-negative with strictly positive diagonal elements and doubly stochastic, that is $A\mathbf{1} = \mathbf{1}$, $\mathbf{1}^\top A = \mathbf{1}^\top$). For $A \in \mathcal{M}_{\mathrm{ss}}$, let $\lambda_1, \ldots, \lambda_n$ be the (real) eigenvalues ordered with non-increasing modulus, then we clearly have that $\rho(A) = \lambda_1(A) \equiv \|A\| = 1$, and A can be written in diagonally factored form as

$$A = \frac{1}{n}\mathbf{1}\mathbf{1}^\top + Z, \quad Z = VDV^\top, \quad D = \mathrm{diag}(\lambda_2, \ldots, \lambda_n), \tag{6}$$

where $V \in \mathbb{R}^{n,n-1}$ is such that

$$V^\top V = I_{n-1}, \quad VV^\top = I_n - \frac{1}{n}\mathbf{1}\mathbf{1}^\top, \quad \mathbf{1}^\top V = 0. \tag{7}$$

The following key lemma relates the convergence rate of iterations $z_{k+1} = Az_k$ to the second-largest modulus eigenvalue of matrix A. This result is quite standard in consensus literature (see, e.g., Section II.D in [20]), but we restate and prove it here in our specific context, for the purpose of clarity.

Lemma 1 *Let $A \in \mathcal{M}_{\mathrm{ss}}$ such that $(A, \mathcal{G}(\mathcal{V}, \mathcal{E}))$ is a compatible pair. If $\mathcal{G}$ is connected, then for any $z \in \mathbb{R}^n$ and any integer $k \geq 0$ it holds that*

$$\|A^k z - \mathbf{1}\zeta\| \leq |\lambda_2(A)|^k \cdot \|z\|, \quad |\lambda_2(A)| < 1,$$

where ζ is the average of the entries in vector z:

$$\zeta = \frac{1}{n}\sum_{i=1}^{n} z_i.$$

Proof. Since $A \in \mathcal{M}_{\rm ss}$ implies $A \in \mathcal{M}$, by Theorem 1 we have that $\mathcal{G}$ connected implies that A is primitive, hence $\lambda_1(A) = 1$ and all other eigenvalues $\lambda_2, \dots, \lambda_n$ have modulus strictly smaller than one. Then, since A is symmetric, we can factorize it according to (6), and using (7) it can be easily verified that

$$A^k = \frac{1}{n}\mathbf{1}\mathbf{1}^\top + VD^kV^\top.$$

Now note that $\frac{1}{n}\mathbf{1}\mathbf{1}^\top z = \mathbf{1}\zeta$, whence

$$\begin{aligned} \|A^k z - \frac{1}{n}\mathbf{1}\mathbf{1}^\top z\| &= \|A^k z - \mathbf{1}\zeta\| = \|VD^kV^\top z\| \\ &\leq \|VD^kV^\top\| \cdot \|z\| \leq \|D^k\| \cdot \|z\| = |\lambda_2|^k \cdot \|z\|, \end{aligned}$$

which concludes the proof. □

4.2 Distributed gradient algorithm with Barzilai-Borwein stepsizes

We here describe a distributed implementation of the update step (3) with BB stepsizes. The approach is divided into two phases: in phase 1 the nodes perform a distributed consensus algorithm in order to converge to the value (5) of the stepsize α_τ; in phase 2 each node locally updates its position according to the gradient step (3). The algorithm thus works on two time scales: an outer iterations scale involving the gradient updates, and an inner iterations scale involving the consensus fusion among nodes. This inner phase requires, ideally, an infinite number of consensus iterations in order for nodes to converge to the true value of the stepsize parameter. However, convergence occurs at geometric rate, hence a reasonable finite number of iterations suffices in practice for the nodes to approximately agree on the stepsize value. This issue is further discussed in Remark 1 below. Numerical experiments, using also very coarse stepsize computation (i.e. very few consensus iterations per outer step) support the idea that the algorithm is quite resilient to the choice of the stepsize parameter.

4.2.1 Distributed computation of stepsize

Suppose that each node i knows its own estimated position at the current iteration τ and at the previous one $\tau - 1$, along with the local block of gradient $\nabla_i f(p)$ (see eq. (13) below) at the same iterations, with the convention that $p^{(-1)} = 0$, $\nabla_i f(p^{(-1)}) = 0$. At "time" $\tau(0)$ each node i, $i = 1, \dots, n$, initializes two scalar values:

$$\begin{aligned} \rho_i(\tau(0)) &= \|p_i^{(\tau)} - p_i^{(\tau-1)})\|^2 && (8) \\ \psi_i(\tau(0)) &= (p_i^{(\tau)} - p_i^{(\tau-1)}))^\top(\nabla_i f(p^{(\tau)}) - \nabla_i f(p^{(\tau-1)})), && (9) \end{aligned}$$

and then starts a series of consensus iterations, exchanging data with its neighbors:

$$\begin{aligned} \rho_i(\tau(t+1)) &= W_{ii}\rho_i(\tau(t)) + \sum_{j \in \mathcal{N}_i} W_{ij}\rho_j(\tau(t)) && (10) \\ \psi_i(\tau(t+1)) &= W_{ii}\psi_i(\tau(t)) + \sum_{j \in \mathcal{N}_i} W_{ij}\psi_j(\tau(t)), \quad t = 0, 1, \dots, && (11) \end{aligned}$$

where $W \in \mathcal{M}_{\rm ss}$ is a symmetric doubly stochastic matrix, compatible with graph $\mathcal{G}$. The following proposition holds.

Proposition 1 *If graph $\mathcal{G}$ is connected, then*

$$\lim_{t \to \infty} \frac{\rho_i(\tau(t))}{\psi_i(\tau(t))} = \alpha_\tau, \quad \textit{for all } i = 1, \dots, n.$$

Proof. First notice that the numerator of α_τ in (5) is given by $\sum_{i=1}^n \rho_i(\tau(0))$, where $\rho_i(\tau(0))$ is defined in (8), and that the denominator of α_τ is given by $\sum_{i=1}^n \psi_i(\tau(0))$, where $\psi_i(\tau(0))$ is defined in (9). We next show that iterations (10), (11) converge at geometric rate to the average of the $\rho_i(\tau(0))$ values and to the average of the $\psi_i(\tau(0))$ values, respectively, hence the ratio $\rho_i(\tau(t))/\psi_i(\tau(t))$ converge to α_τ, as claimed. To prove this fact, notice that iterations (10) can be written in compact vector form as follows:

$$\rho(\tau(t+1)) = W\rho(\tau(t)),$$

where $\rho(\tau) = [\rho_1(\tau) \cdots \rho_n(\tau)]^\top$. Given the initial vector $\rho(\tau(0))$, the above recursion generates at generic iteration $t = 1, 2, \ldots$ the vector

$$\rho(\tau(t)) = W^t \rho(\tau(0)).$$

Similarly, (11) generate at iteration t the vector

$$\psi(\tau(t)) = W^t \psi(\tau(0)).$$

Since $W \in \mathcal{M}_{\rm ss}$ is compatible with connected graph $\mathcal{G}$, by Lemma 1 we have that

$$\|\rho(\tau(t)) - \frac{1}{n}\sum_{j=1}^n \rho_j(\tau(0))\mathbf{1}\| \leq \gamma^t \|\rho(\tau(0))\|,$$

Where $\gamma < 1$ is the modulus of the second-largest modulus eigenvalue of W. Recalling the vector norm inequality $\|x\|_\infty \leq \|x\|$, we also have that

$$\|\rho(\tau(t)) - \frac{1}{n}\sum_{j=1}^n \rho_j(\tau(0))\mathbf{1}\|_\infty \leq \gamma^t \|\rho(\tau(0))\|,$$

that is

$$|\rho_i(\tau(t)) - \frac{1}{n}\sum_{j=1}^n \rho_j(\tau(0))| \leq \gamma^t \|\rho(\tau(0))\|, \quad \forall i = 1, \ldots, n,$$

and, similarly, we obtain that

$$|\psi_i(\tau(t)) - \frac{1}{n}\sum_{j=1}^n \psi_j(\tau(0))| \leq \gamma^t \|\psi(\tau(0))\|, \quad \forall i = 1, \ldots, n,$$

which shows that, as t tends to infinity, $\rho_i(\tau(t))$, $\psi_i(\tau(t))$ converge at geometric rate to the averages of the entries in vectors $\rho(\tau(0))$, $\psi(\tau(0))$, respectively. From this it follows that

$$\lim_{t\to\infty} \frac{\rho_i(\tau(t))}{\psi_i(\tau(t))} = \frac{\sum_{j=1}^n \rho_j(\tau(0))}{\sum_{j=1}^n \psi_j(\tau(0))} = \alpha_\tau,$$

thus proving the claim. □

Remark 1 From Proposition 1 it follows that, at inner iteration t, each node can locally construct the ratio

$$\alpha_i(\tau(t)) \doteq \frac{\rho_i(\tau(t))}{\psi_i(\tau(t))} \tag{12}$$

and use it as an approximation of the step size α_τ. The numerator and denominator of $\alpha_i(\tau(t))$ converge geometrically to the average of the entries in vectors $\rho(\tau(0))$, $\psi(\tau(0))$, respectively, and the convergence rate is dictated by γ, the modulus of the second-largest modulus eigenvalue of matrix W. We next show that, for t larger than some finite $\bar{t}$, also the convergence of $\alpha_i(\tau(t))$ to α_τ is at least geometric, and still dictated by γ. It follows that, whenever the network topology is known and γ can be computed a priori, then fixing a maximum number T of inner iterations induces a predictable error in the estimation of the stepsize parameter. This error can be made as small as desired by a suitable choice of (finite) T. Indeed, inspecting the proof of Proposition 1, and recalling Lemma 1, we see that $\rho_i(\tau(t)) = \bar{\rho} + e_i^\top V D^t V^\top \rho(\tau(0))$, $\psi_i(\tau(t)) = \bar{\psi} + e_i^\top V D^t V^\top \psi(\tau(0))$, where e_i is the i-th unit vector, and we defined $\bar{\rho} = \frac{1}{n}\sum_{j=1}^n \rho_j(\tau(0))$, $\bar{\psi} = \frac{1}{n}\sum_{j=1}^n \psi_j(\tau(0))$. Recalling that $\alpha_\tau = \bar{\rho}/\bar{\psi}$, we next construct the relative convergence error. Let $\eta > 0$ be a small constant, and let $\bar{t}$ be the smallest integer larger than $(\log\eta - \log\|\psi(\tau(0))\|)/\log\gamma^{-1}$. Then,

$$\begin{aligned}
\varepsilon_i(t) &= \frac{|\alpha_i(\tau(t)) - \alpha_\tau|}{|\alpha_\tau|} = \frac{|e_i^\top V D^t V^\top (\alpha_\tau^{-1}\rho(\tau(0)) - \psi(\tau(0)))|}{|\bar{\psi} + e_i^\top V D^t V^\top \psi(\tau(0))|} \\
&\leq \frac{\|e_i\| \cdot \|V D^t V^\top\| \cdot \|\alpha_\tau^{-1}\rho(\tau(0)) - \psi(\tau(0))\|}{|\bar{\psi}| - \|e_i\| \cdot \|V D^t V^\top\| \cdot \|\psi(\tau(0))\|} \\
&\leq \frac{\gamma^t \|\alpha_\tau^{-1}\rho(\tau(0)) - \psi(\tau(0))\|}{|\bar{\psi}| - \gamma^t\|\psi(\tau(0))\|} \\
&\leq \text{const.}\gamma^t, \quad \text{for } t \geq \bar{t},
\end{aligned}$$

where const. $= \frac{\|\alpha_\tau^{-1}\rho(\tau(0)) - \psi(\tau(0))\|}{|\bar{\psi}| - \eta}$, which shows that the modulus of relative error decreases at least at geometric rate. A practical implementation of the algorithm works by fixing a priori a sufficiently large T, and running iterations (10), (11) up to $t \leq T$. An analysis on the effect of the choice of T on the algorithm performance is presented via numerical simulations in Section 5.2.

Choice of weights Note that, in order to being able to update its estimate, node i needs to know its current neighbors' weights W_{ij}. These coefficients may be imposed a-priori on the nodes (but this would require a centralized a-priori knowledge of the network topology) or, more practically, negotiated autonomously on-line by the nodes. A standard and effective rule for building the weights autonomously is the so-called Metropolis rule, see [29, 30], which prescribes weights as follows:

$$W_{ij} = \begin{cases} \dfrac{1}{\max(|\mathcal{N}_i|, |\mathcal{N}_j|)} & \text{if } (i,j) \in \mathcal{E},\ i \neq j \\ 1 - \displaystyle\sum_{j \in \mathcal{N}_i \setminus i} W_{ij} & \text{if } i = j \\ 0 & \text{otherwise,} \end{cases}$$

where $|\mathcal{N}_i|$ denotes the cardinality of $\mathcal{N}_i$. This choice of weights is indeed well suited for distributed implementation, since each node only needs to know the number of its neighbors and exchange information with them. With these weights, iterations (10), (11) are based on local-only information, which means that each node can execute the update without need of knowing the global structure of the network, or even the number n of nodes composing the network.

4.2.2 Distributed gradient update

Let $\nabla_i f(p)$ denote the i-th 1×2 block in the gradient $\nabla f(p)$ in (4). Given the specific structure of $\nabla g_{ij}(p)$, which is nonzero only for the blocks in position i and j, that are equal to $(p_i - p_j)^\top$ and $(p_j - p_i)^\top$, respectively, it is immediate to verify that

$$\nabla_i f(p) = \sum_{j \in \mathcal{N}_i} (p_i - p_j)^\top g_{ij}(p), \tag{13}$$

where $\mathcal{N}_i \doteq \{j : (i,j) \in \mathcal{E}\}$ is the set of *neighbors* of node i in graph $\mathcal{G}$. Observe next from (2) that $g_{ij}(p)$ is actually a function of $p_i - p_j$ only, which represents the current mismatch between $\|p_i - p_j\|^2$ and d_{ij}^2. It follows that the portion of gradient $\nabla_i f(p)$ can be computed individually by node i by simply querying the neighbors for their current estimated positions.

Assuming that the step size α_τ is known at each node, each node would be able to locally update its estimated position according to the distributed gradient rule:

$$\begin{aligned} p_i^{(\tau+1)} &= p_i^{(\tau)} - \alpha_\tau \nabla_i f(p^{(\tau)}), \quad i = 1, \ldots, n. \\ &= p_i^{(\tau)} - \alpha_\tau \sum_{j \in \mathcal{N}_i} (p_i - p_j)^\top g_{ij}(p), \quad i = 1, \ldots, n. \end{aligned} \tag{14}$$

In practice, the value of α_τ is known only approximately at each node, as a result of the finite number T of consensus iterations. Summarizing, the practical distributed localization algorithm works as follows.

Algorithm 1 (Distributed localization algorithm)

0. **(Initialization)** *Fix T (maximum number of consensus iterations), $\tau_{\rm wup}$ (maximum number of warm-up iterations), $\tau_{\max}$ (maximum number of gradient updates), and $\eta_{\rm abstol}$ (absolute update tolerance). Let $\tau = 0$ and set the position estimates p_i^τ, for $i = 1, \ldots, n$ to be the given initial guess for optimization. Mark all nodes as "active".*

1. **(Warm-up)**

(1.a) Fix $\alpha_\tau = \tilde{\alpha}$ (a small number, say $\tilde{\alpha} = 10^{-6}$, equal for all nodes);

(1.b) Update all node position estimates according to (14), and let $\tau = \tau + 1$;

(1.c) If $\tau < \tau_{\rm wup}$, then goto (1.a), else goto 2.

2. **(Consensus iterations)**

(2.a) Let $t = 0$, and initialize $\rho_i(\tau(t))$ and $\psi_i(\tau(t))$ as in (8), (9);

(2.b) Update ρ_i, ψ_i according to (10), (11); let $t = t + 1$;

(2.c) If $t < T$, then goto (2.a), else goto 3.

3. **(Gradient update)** *At each active node:*

(3.a) Compute $\alpha_i(\tau(t))$ according to (12);

(3.b) Update position estimate according to (14);

(3.c) If $\|p_i^{(\tau)} - p_i^{(\tau-1)}\| \leq \eta_{\rm abstol}$, then mark node i as "inactive";

4. **(Stopping criterion)** *Let $\tau = \tau + 1$. If $\tau \geq \tau_{\max}$, or all nodes are inactive, then exit; otherwise goto 2.*

It is worth noticing that the warm-up is only useful to have reasonable values for the correction in the first update ($\tau = 0$): since $p^{(-1)}$, $\nabla_i f(p^{(-1)})$ are chosen to be zero by convention, see Section 4.2.1, the first iteration can produce arbitrary values of the stepsize and can lead the local estimate outside of the region of attraction of the global minimum of the objective function. In practice, numerical tests suggest that it often suffices to fix $\tau_{\rm wup} = 1$ in order to have good convergence properties for the algorithm, see Section 5.2.

5 Numerical experiments

We now present some numerical tests on decentralized network localization with the proposed distributed gradient method. In the first experiment (Section 5.1) we verify the correctness of our theoretical analysis on a small formation, showing the convergence of the stepsizes, computed locally at each node, to a common value, which is actually the centralized stepsize α_τ. In the second experiment (Section 5.2) we investigate the performance of the distributed gradient method on a large networked system, showing the advantages of integrating a distributed averaging technique in the localization process; moreover we analyze the scalability of the proposed approach. Finally, in Section 5.3, we apply the technique to a realistic setup in which nodes in a sensor network use (eventually without any synchronization) the distributed gradient method for refining an inaccurate position guess, obtained through a preliminary semidefinite optimization-based localization method.

5.1 Example 1

Let us consider a first setup that exemplifies the case of a team of autonomous vehicles or mobile robots moving in formation where few agents are equipped with GPS, whereas others use dead-reckoning for position estimation. Using dead-reckoning progressively decreases the localization accuracy, hence the agents may periodically stop and use the relative range measurements to improve their position estimate. Distance measurements are typically obtained through acoustic beacons [21], radio frequency devices [32], or other "time of flight" sensors.

For a numerical example, we considered the case of $n = 10$ agents located on terrain according to the configuration shown in Figure 1(b). This actual configuration should be estimated autonomously by the agents using as prior knowledge the ideal position configuration shown in Figure 1(a), which is assumed as the initial position guess by the agents. Three anchor nodes are selected at the external vertices of the formation.

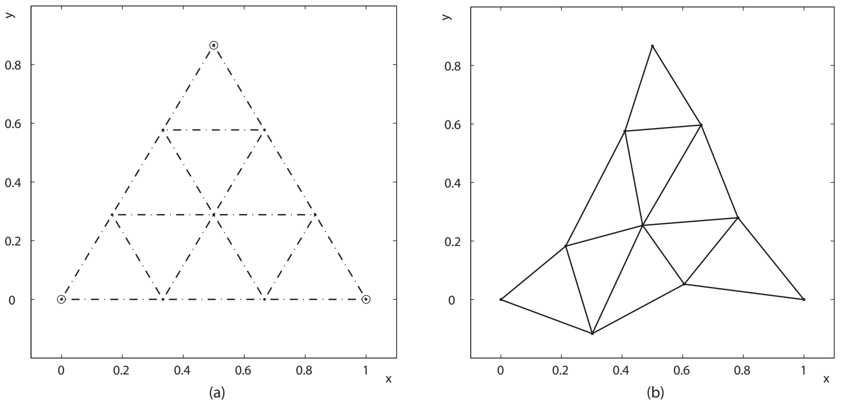


Figure 1: Triangle formation: (a) initial position guess; (b) actual nodes configuration ($n = 10$).

In order to measure the localization effectiveness, we define the *local positioning error* $\phi_i(\tau)$ at node i as the Euclidean distance between the estimated position $p_i^{(\tau)}$ at iteration τ and the true position p_i of the node. The *global positioning error* $\Phi(\tau)$ is further defined as the mean value of the local positioning errors of all the nodes in the network:

$$\Phi(\tau) = \frac{1}{n} \sum_{i=1}^{n} \|p_i - p_i^{(\tau)}\|.$$

At each update iteration, the position estimate of the i-th node is corrected according to the recursion (14), initialized with initial position estimate $p_i^{(0)}$ corresponding to the configuration in Figure 1(a). Before each update, at iteration τ each node starts inner consensus iterations $\tau(t)$, $t = 0, 1, \dots, T$, in order to compute the Barzilai-Borwein stepsize α_τ in a decentralized fashion.

Figure 2(a) shows, at some given iteration τ, how local stepsizes

$$\alpha_i(\tau(t)) = \frac{\rho_i(\tau(t))}{\psi_i(\tau(t))},$$

computed according to the scheme presented in Section 4.2.1, converge to the desired value α_τ, equal for all nodes.

It is possible to notice that, after a small and fixed number of consensus iterations, the nodes agree on a common value for the stepsizes (see Algorithm 1). Further tests on the influence of T on the performances of the algorithm are reported in Section 5.2. Figure 2(b) reports the local positioning error for each non-anchor node, using $T = 20$. As expected, in a noiseless setup, the estimated positions converge to the true location in a reasonable number of τ iterations. In

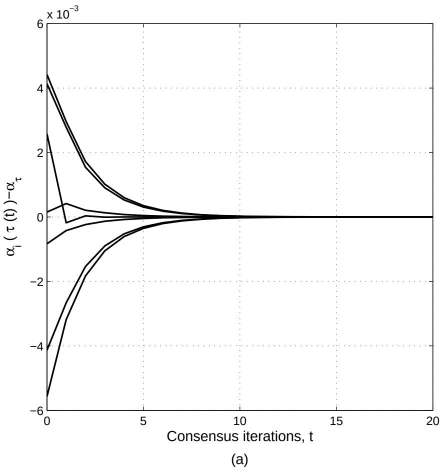


Figure 2: (a) local stepsizes $\alpha_i(\tau(t))$ of the agents compared with the centralized stepsizes α_τ at some given iteration τ. (b) Local positioning errors for the non-anchor nodes in the formation versus update iterations count τ; the number of consensus iterations is $T = 20$.

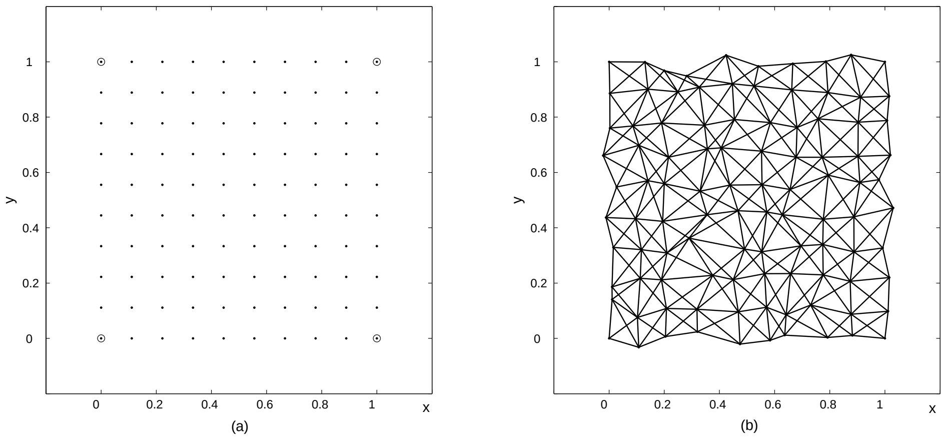


Figure 3: Lattice configuration: (a) actual nodes configuration; (b) initial position guess ($n = 100$).

practice, each node becomes inactive after a threshold condition is met, see Algorithm 1, hence the approach can be used with no need of synchronization among nodes: each node interrogates the neighbors T times for each update and it may stop to correct its position estimate autonomously, with no need of global information (further details can be found in Section 5.3).

5.2 Example 2

For the second experiment we considered a graph with n nodes disposed in lattice configuration on the unit square $[0, 1] \times [0, 1]$. Four anchor nodes are selected on the vertices of the unit square. The actual network configuration to be estimated is taken as the regular lattice configuration (Figure 3(a)), whereas the initial position guess is generated by randomly perturbing the position of each node, see Figure 3(b). Diagonal edges are added to the lattice structure for guaranteeing rigidity of the underlying graph, hence enabling the nodes to retrieve the correct configuration.

In order to evaluate the advantages of the proposed method we compare it with other gradient-based techniques for network localization. Such techniques, that will be briefly described in the

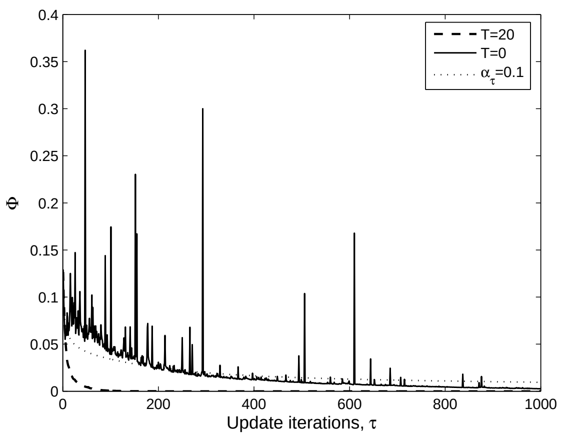


Figure 4: (a) Global positioning error versus iterations τ, for the limit case without consensus (solid line), for the proposed distributed gradient method with $T = 20$ (dashed line), and for the gradient method with fixed stepsizes $\alpha_\tau = 0.1$ (dotted line).

following, are characterized by the fact that can be all applied in a decentralized fashion. The first compared technique simply employs a fixed stepsize for gradient update: each node communicates with the neighbors, in order to compute the local gradient, and the update (3) is performed using a stepsize $\alpha_\tau = \bar{\alpha}$, $\bar{\alpha} > 0$. The second technique can be seen as a limit condition of the proposed approach, in which the agents may perform no consensus at all ($T = 0$), and each node computes its own stepsize, according to the local information:

$$\alpha_{\tau,i} = \frac{\|p_i^{(\tau)} - p_i^{(\tau-1)}\|^2}{(p_i^{(\tau)} - p_i^{(\tau-1)})^\top (\nabla_i f(p_i^{(\tau)}) - \nabla_i f(p_i^{(\tau-1)}))}. \tag{15}$$

Figure 4 shows the global positioning error as a function of update iterations τ, comparing the proposed approach (with $T = 20$) with the two gradient techniques, for a network of $n = 100$ nodes. It can be seen how the use of distributed averaging improves the convergence rate of the gradient method (results are averaged over 50 Monte Carlo runs). The use of local stepsizes, instead, causes the presence of spikes in the localization error, since there is no agreement on the correction to be applied at a single node, whereas the use of fixed stepsizes corresponds to long tails in the localization errors and poor convergence rates. The three compared techniques are local, since they require a suitable initial guess for attaining the global minimum of the non-linear objective function. In Figure 5 we report the percentage of convergent experiments for the three scenarios, with different initial guesses. The initial guess of each node is drawn from a normal distribution with mean corresponding to the actual node position, and covariance equal to $P = \sigma_I^2 I_2$, hence for increasing values of the σ_I (x-axis in the figure), the quality of the initial guess worsens. The figure highlights how the use of the proposed technique presents further advantages in terms of convergence properties with respect to the other compared gradient-based approaches. It is quite intuitive that the use of fixed stepsize is not suitable when the initial guess is far from a stationary point. In such situation, in fact, due to the structure of the objective function, the gradient can assume high values (carrying on large values for the correction in gradient update), leading the local position estimate to escape from the region of attraction of the global minimum. In literature, this problem is often avoided by using gradient methods with fixed *step length*, instead of fixed stepsize, i.e., use a stepsize $\alpha_\tau = k/\|\nabla_i f(p^{(\tau)})\|$, $k > 0$; in this way it is possible to control the norm of the local correction (in the direction of the negative gradient). This last strategy can be employed with a constant $k = \bar{k}$ or with a diminishing sequence of step length, like $k = \bar{k}/\tau$ or $k = \bar{k}/\sqrt{\tau}$ (with $\bar{k} > 0$), see [27]. The former choice (constant step length) does not allow the

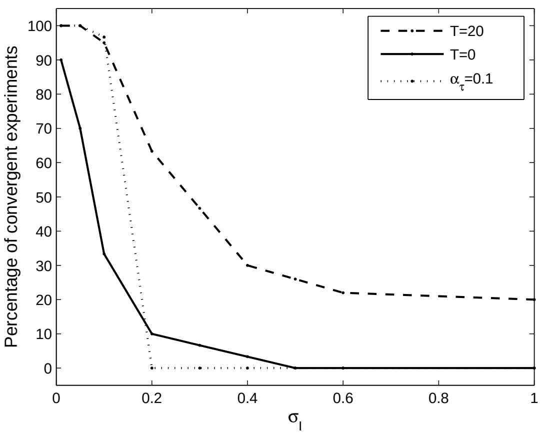


Figure 5: Percentage of convergent experiments for the case with no consensus (solid line), for the proposed distributed gradient method with $T = 20$ (dashed line), and for the gradient method with fixed stepsizes $\alpha_\tau = 0.1$ (dotted line).

estimate to get arbitrarily close to the stationary point (also in the convex case it is possible to assess only convergence to within some range of the optimal value [27]). The latter, instead, shows poor convergence rates as we will see in a while. Further tests on the convergence properties on local techniques for network localization can be found in [6].

We now want to evaluate the total number of update iterations required for network localization using the above mentioned techniques. The total number of iteration is connected to the amount of computation and the communication burden required to the nodes, hence it is crucial in low-cost low-power applications. We consider the value $\eta_{abstol} = 10^{-10}$ as threshold for a node to stop the update iterations, and $\tau_{max} = 10^5$, see Algorithm 1; we define τ_{end} as the number of update iteration after which all the nodes are inactive, i.e., after how many update iterations the algorithm stops. In Table 1 we report the values of τ_{end} when using the proposed technique, a gradient method with local BB-stepsizes (15), a gradient method with fixed stepsizes and a gradient method with specified step lengths. Obviously, due to the stopping criterion for the update iterations, the number of updates performed using specified step lengths, can be known *a-priori*; when using $k = \bar{k}/\sqrt{\tau}$, for instance, the iterations will stop when $\tau \geq (\bar{k}/\eta_{abstol})^2$.

	Required update iterations (τ_{end})	Communication rounds
Distributed gradient method with $T = 20$	262	5240
Gradient method with local stepsizes ($T = 0$)	6071	6071
Gradient method with stepsizes $\alpha_\tau = 0.1$	23491	23491
Gradient method with stepsizes $\alpha_\tau = 0.01$	10^5(*)	10^5(*)
Gradient method with step length $k = 10/\sqrt{\tau}$	10^5(*)	10^5(*)

Table 1: Computational effort and communication burden for the limit case without consensus, for the proposed distributed gradient method with $T = 20$, for gradient method with fixed stepsizes and for gradient method with specified step lengths. Results are averaged over 50 Monte Carlo runs. (*) Iterations are stopped because the maximum number of allowed updates (τ_{max}) is reached.

In Table 1 we also report the communication burden for the three scenarios mentioned before. The proposed approach requires each node to communicate T times with the neighbors, for each update (each communication allows a generic node i to broadcast the local information to the neighbors). On the other hand the compared gradient methods (local stepsizes, fixed stepsizes,

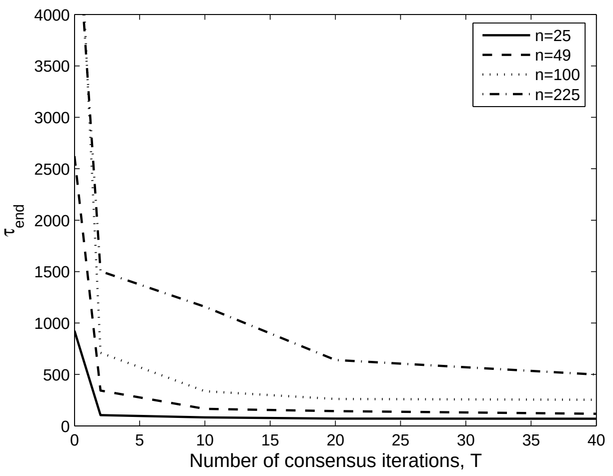


Figure 6: Number of update iterations τ_{end} required for network localization using different values of T. Results are provided for lattice configurations with $n = 25, 49, 100, 225$ nodes.

specified step lengths) require nodes to communicate with their neighbors only one time for each update, in order to compute the local gradient $\nabla_i f(p^{(\tau)})$. It is possible to observe that, despite the communication load due to consensus, the proposed approach is still advantageous over the compared techniques.

In the previous experiments we considered $T = 20$ consensus iterations for each update. Now we discuss such choice, reporting some results on how the total number of update iterations varies in function of the number of consensus iterations T. In Figure 6 we show the value of τ_{end} for different values of T and for different network sizes. It is worth noticing that, also for small values of T, the introduction of the consensus phase contributes to remarkably reduce the total amount of updates, and there is no need to consider a large number of consensus iterations (convergence is exponential, see Remark 1).

As a conclusion of this section we discuss the scalability of the proposed approach. Scalability issues are particularly relevant when dealing with sensor networks in which a large number of nodes has to perform localization with limited bandwidth and reduced computational resources. For evaluating the scalability of the approach we tested the distributed gradient algorithm for different network sizes. Figure 7 reports the total number of iterations τ_{end} for networks with n nodes and $T = 20$.

5.3 Example 3

In this section we evaluate the use of the proposed technique as a refinement phase, after semidefinite programming (SDP) is used for computing a rough estimate of node position. It is well known that the SDP relaxation can be inaccurate in presence of measurements noise, see [4, 5], hence we use the output of SDP as initial guess for a distributed gradient-based refinement. The idea of a local refinement of a global solution is not new, see [16, 24], although our approach has the advantages of being fully distributed and effective in terms of communication and computational requirements.

In order to work on more realistic networks, we now consider *random geometric graphs*, that is graphs in which nodes are deployed at random in the unit square, and an edge exists between a pair of nodes if and only if their geometrical distance is smaller than a *sensing radius R*. It has been proved in [8] that if $R > 2\sqrt{2}\sqrt{\frac{\log(n)}{n}}$, the graphs produced by the previous technique are generically globally rigid with high probability. An example of geometric graph with $R = 0.3$ and

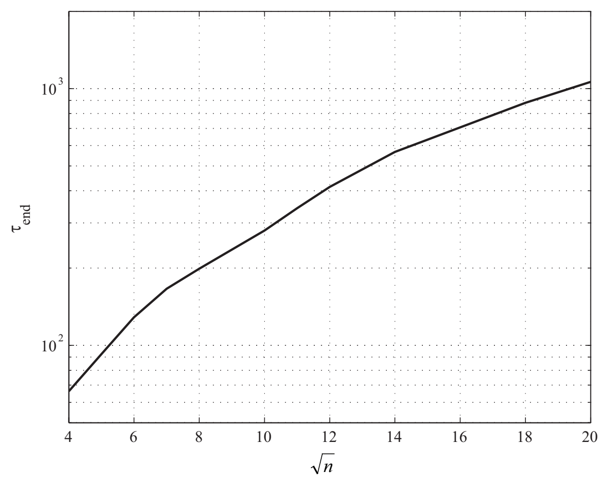


Figure 7: Total number of update iterations τ_{end} for increasing network sizes.

			ν_d		
			0.05	0.1	0.15
n_a	20	SDP	0.010	0.017	0.023
		SDP+DG	0.006	0.013	0.018
	10	SDP	0.020	0.033	0.040
		SDP+DG	0.013	0.023	0.032
	3	SDP	0.094	0.145	0.170
		SDP+DG	0.046	0.127	0.156

Table 2: Global localization error when using semidefinite programming (SDP) and SDP with distributed gradient refinement (SDP-DG); the table shows results for different values of the noise factor (ν_d) and for different number of anchor nodes (n_a).

$n = 50$ is shown in Figure 8.

We consider the configuration generated as previously described as the "true" configuration (which is of course unknown in practice), and then, we use the distance measurements from this configuration as the data for the numerical tests; n_a anchor nodes are randomly selected among the nodes. Two nodes are connected by an edge if their distance is smaller than 0.3 and distance measurements are affected by noise in the form:

$$\bar{d}_{ij} = |d_{ij} + \epsilon_{ij}^d| \;\; \forall \;\; (i,j) \in \mathcal{E} \tag{16}$$

where d_{ij} is the true distance among node i and node j, $\bar{d}_{ij}$ is the corresponding measured quantity and ϵ_{ij}^d is a zero mean white noise with standard deviation σ_{ij}^d. According to [5] we choose $\sigma_{ij}^d = \nu_d d_{ij}$, where the constant ν_d, often referred to as *noise factor*, defines how measurement accuracy decreases with the actual distance. The parameters used for the distributed gradient method are $T = 20, \eta_{abstol} = 10^{-15}, \tau_{max} = 10^5$.

In Table 2 we report the global localization error for different ν_d, considering the network in Figure 8. It is possible to see that the introduction of the distributed gradient refinement improves localization accuracy, leading to an error reduction of $15 \div 50\%$. In particular the distributed gradient method can drastically improve localization performance in case few anchor nodes are present in the formation or when the anchors lie in the interior on the formation: in such situation

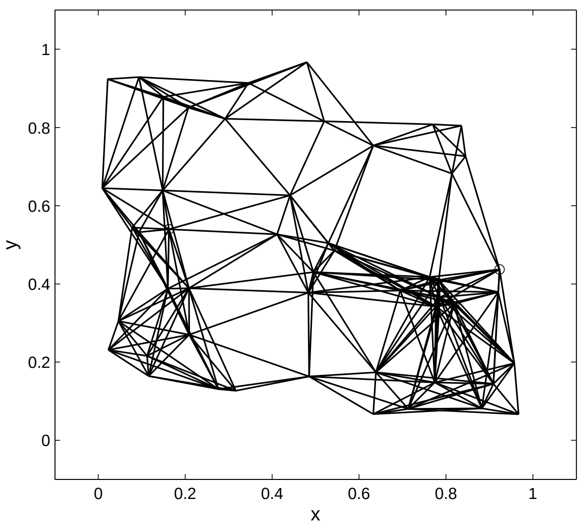


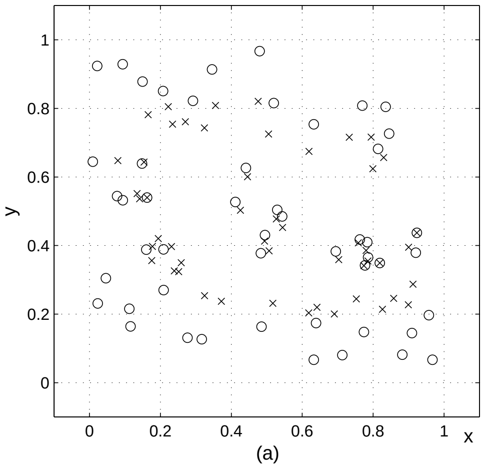
Figure 8: Example of geometric random graph with $n = 50$ nodes in the unit square and sensing radius $R = 0.3$.

the SDP localization is known to be inaccurate and the local refinement is essential for practical use (see Figure 9 for a meaningful example).

It is worth noticing that the proposed distributed technique assumes that some sort of synchronization among the nodes in the network exists, i.e., there is a global clock, that we will call t^g, so that for $t^g = 0$ all the nodes start the warm-up phase, and at $t^g = kT,\ k = 1, 2, \ldots$, they all update local estimates, iterating consensus at each discrete time in the interval $[(k-1)T; kT]\,,\ k = 1, 2, \ldots$. Although such underlying assumption is common to several distributed techniques (see [2], for instance), we now want to evaluate what happens when relaxing this hypothesis. For this purpose we consider a setup in which each node may start iterations at a different time t_i^g, and alternates consensus and update phases, based on a local clock t_i^l; after the warm-up each node will interrogate neighbors T times (during which, possibly, some neighbors may change their estimates) and then apply a local gradient update, according to (14). Before introducing our numerical tests we define the *maximum clock skew* $\Delta t_{max} = \max_{i,j} |t_i^g - t_j^g|$. Obviously the case of $\Delta t_{max} = 0$ corresponds to a perfect synchronization among the nodes. On the other hand for increasing values of the maximum clock skew, the delays between the nodes in the network can be relevant.

For numerical tests we studied the behavior of the distributed gradient method considering different maximum skew times as follow: given a Δt_{max}, the starting time t_i^g of the i-th node is randomly sampled in the interval $[0; \Delta t_{max}]$ and the local clock is initialized accordingly (when the local clock starts the node becomes *active*). Before starting consensus iterations, a node verifies that also the neighbors are active, otherwise it remains in the warm-up phase (such procedure assures that the results of the consensus is meaningful, since inactive nodes cannot properly define the quantities (8) and (9)). In the reported tests, each node uses the value $T = 20$ for the consensus iterations and autonomously stops when the correction of the local estimates drops below $\eta_{abstol} = 10^{-15}$.

Table 3 reports the global localization error and the average number of updates for the distributed gradient method. It is possible to see that there is no remarkable degradation in performance hence confirming that the algorithm is suitable for decentralized asynchronous operation.



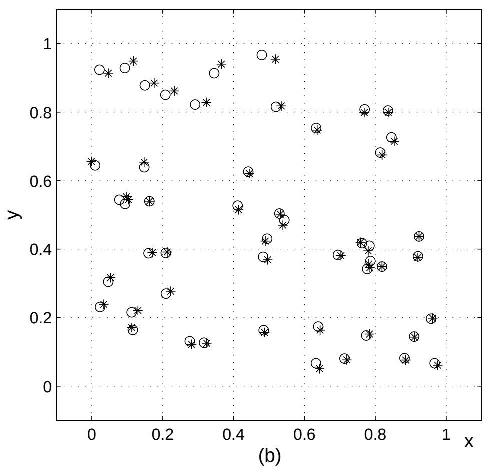


Figure 9: (a) estimated versus actual configuration for SDP localization (circles denote actual node positions, whereas crosses correspond to SDP estimates); (b) estimated versus actual configuration for SDP localization with distributed gradient refinement (circles denotes actual node positions, whereas stars correspond to estimates after refinement). The reported estimates are obtained in the same Monte Carlo run with $\nu_d = 0.05$, $T = 20$, $\eta_{abstol} = 10^{-15}$.

	Δt_{max}				
	10	20	30	50	100
Φ for SDP	0.096	0.095	0.099	0.097	0.094
Φ for SDP+DG	0.053	0.053	0.055	0.060	0.063
$\bar{\tau}_{max}$	1681	1915	1859	1787	1882

Table 3: Global localization error Φ for semidefinite programming (SDP) and SDP with distributed gradient refinement (SDP-DG); the table also reports the average number of updates $\bar{\tau}_{max}$ performed by the nodes in the network. The results are averaged over 30 Monte Carlo runs, considering 3 anchor nodes and noise factor $\nu_d = 0.05$.

6 Conclusion

In this paper we presented a distributed gradient method for solving the network localization problem from relative distance measurements. First, the localization problem is formulated as minimization of a non-linear cost, and the formulation of a centralized gradient method is recalled, focusing the attention on the use of Barzilai-Borwein stepsizes. Then, a distributed scheme is proposed for allowing each node to autonomously correct its position estimate, by communicating only with its neighbors. The decentralized technique iterates two phases: a *consensus* phase and an *update* phase. In the consensus phase each node iteratively consults and exchanges information with its neighbors in order to converge approximately to some parameter value that must be common and available to all nodes (the α_τ parameter in the gradient method) for an update step to be possible. In the update phase, this common parameter is used to actually update the current position estimate. The distributed gradient method is proved to converge to the same solution of its centralized counterpart, while providing the benefits of a fully decentralized scheme that can be implemented autonomously by the networked agents. Moreover extensive numerical experiments highlighted the advantages of using the proposed technique, in terms of convergence properties, computational cost and communication burden for the nodes in the network. The algorithm is also suitable for the case in which no synchronization exists among the nodes and it is shown to be scalable in the network size, since it requires inexpensive computation and low memory storage.

References

[1] P. Barooah. Estimation and control with relative measurements: Algorithms and scaling laws. *Ph.D. Thesis*, University of California, Santa Barbara, 2007.

[2] P. Barooah and J.P. Hespanha. Estimation on graphs from relative measurements. *IEEE Control Systems Magazine*, 27(4):57–74, 2007.

[3] J. Barzilai and J.M. Borwein. Two-point step size gradient methods. *IMA J. Numer. Anal.*, 8:141–148, 1988.

[4] P. Biswas, T. Liang, K. Toh, T. Wang, and Y. Ye. Semidefinite programming approaches for sensor network localization with noisy distance measurements. *IEEE Transactions on Automation Science and Engineering*, 3(4):360–371, 2006.

[5] P. Biswas and Y. Ye. Semidefinite programming for ad hoc wireless sensor network localization. In *Proceedings of the Third International Symposium on Information Processing in Sensor Networks (IPSN)*, pages 2673–2684, 2004.

[6] G.C. Calafiore, L. Carlone, and M. Wei. Position estimation from relative distance measurements in multi-agents formations. In *Proceedings of the 2010 18th Mediterranean Conference on Control and Automation (MED)*, pages 148–153, 2010.

[7] R. Diestel. *Graph Theory.* Springer, New York, 2005.

[8] T. Eren, D.K. Goldenberg, W. Whiteley, Y.R. Yang, A.S. Morse, B.D.O. Anderson, and P.N. Belhumeur. Rigidity, computation, and randomization in network localization. In *IEEE INFOCOM*, volume 4, pages 2673–2684, 2004.

[9] W.H. Foy. Position-location solutions by Taylor-series estimation. In *IEEE Transaction on Aerospace and Electronic Systems AES-12 (2)*, pages 187–194, 1976.

[10] A. Friedlander, J.M. Martinez, and M. Raydan. A new method for large-scale box constrained convex quadratic minimization problems. *Optim. Methods and Software*, 5:57–74, 1995.

[11] C. Godsil and G. Royle. *Algebraic Graph Theory.* Springer, New York, 2001.

[12] B. Hendrickson. The molecule problem: Exploiting structure in global optimization. *SIAM Journal on Optimization*, 5(4):835–857, 1995.

[13] R.A. Horn and C.R. Johnson. *Matrix Analysis.* Cambridge University Press, UK, 1985.

[14] U.A. Khan, S. Kar, and J.M.F. Moura. Distributed sensor localization in random environments using minimal number of anchor nodes. *IEEE Transactions on Signal Processing*, 57:2000–2016, 2009.

[15] U.A. Khan, S. Kar, and J.M.F. Moura. Diland: An algorithm for distributed sensor localization with noisy distance measurements. *IEEE Transactions on Signal Processing*, 58:1940–1947, 2010.

[16] T.C. Liang, T.C. Wang, and Y. Ye. A gradient search method to round the semidefinite programming relaxation solution for ad hoc wireless sensor network localization. In *Tech. rep., Dept of Management Science and Engineering, Stanford University*, 2004.

[17] G. Mao, B. Fidan, and B.D.O. Anderson. Wireless sensor network localization techniques. *Computer Networks*, 51(10):2529–2553, 2007.

[18] D. Moore, J. Leonard, D. Rus, and S. Teller. Robust distributed network localization with noisy range measurements. In *Proceedings of the Second ACM Conference on Embedded Networked Sensor Systems (SenSys '04)*, pages 50–61, 2004.

[19] J. Nocedal and S.J. Wright. *Numerical Optimization.* Springer, 2006.

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

[21] E. Olson, J.J. Leonard, and S. Teller. Robust range-only beacon localization. *IEEE Journal of Oceanic Engineering*, 31(4):949–958, 2006.

[22] N.B. Priyantha, H. Balakrishnan, E. Demaine, and S. Teller. Anchor-free distributed localization in sensor networks. In *Proceedings of the 1st international conference on Embedded networked sensor systems*, pages 340–341, 2003.

[23] M. Raydan. The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem. *SIAM J. Optim.*, 7(1):26–33, 1997.

[24] C. Savarese, J. Rabaey, and K. Langendoen. Robust positioning algorithms for distributed ad-hoc wireless sensor networks. In *USENIX Annual Technical Conference*, pages 317–327, 2002.

[25] J.B. Saxe. Embeddability of weighted graphs in k-space is strongly NP-hard. In *Proceedings of the 17th Allerton Conference in Communications, Control and Computing*, pages 480–489, 1979.

[26] I. Shames, B. Fidan, and B.D.O. Anderson. Minimization of the effect of noisy measurements on localization of multi-agent autonomous formations. *Automatica*, 45(4):1058–1065, 2009.

[27] N.Z. Shor. *Minimization Methods for Non-differentiable Functions.* Springer Series in Computational Mathematics, 1985.

[28] R.G. Stanfield. Statistical theory of DF finding. *Journal of IEE*, 94(5):762–770, 1947.

[29] L. Xiao and S. Boyd. Fast linear iterations for distributed averaging. *Systems and Control Letters*, 53:65–78, 2004.

[30] L. Xiao, S. Boyd, and S. Lall. A scheme for robust distributed sensor fusion based on average consensus. In *International Conference on Information Processing in Sensor Networks (IPSN 2005)*, pages 63–70, 2005.

[31] C. Xunxue, S. Zhiguan, and L. Jianjun. Distributed localization for anchor-free sensor networks. *Journal of Systems Engineering and Electronics*, 19(3):405–418, 2008.

[32] J. Zhou and J. Shi. Performance evaluation of object localization based on active radio frequency identification technology. *Computers in Industry*, 60(9):669–676, 2009.