# Distributed Wireless Sensor Network Localization Via Sequential Greedy Optimization Algorithm

Qingjiang Shi, *Student Member, IEEE*, Chen He, *Member, IEEE*, Hongyang Chen, and Lingge Jiang, *Member, IEEE*

***Abstract*—Node localization is essential to most applications of wireless sensor networks (WSNs). In this paper, we consider both range-based node localization and range-free node localization with uncertainties in range measurements, radio range, and anchor positions. First, a greedy optimization algorithm, named sequential greedy optimization (SGO) algorithm, is presented, which is more suitable for distributed optimization in networks than the classical nonlinear Gauss-Seidel algorithm. Then a unified optimization framework is proposed for both range-based localization and range-free localization, and two convex localization formulations are obtained based on semidefinite programming (SDP) relaxation techniques. By applying the SGO algorithm to the edge-based SDP relaxation formulation, we propose a second-order cone programming (SOCP)-based distributed node localization algorithm. Two distributed refinement algorithms are also proposed by using the SGO algorithm to nonconvex localization formulations. The proposed localization algorithms all can be implemented partially asynchronously in networks. Finally, extensive simulations are conducted to demonstrate the efficiency and accuracy of the proposed distributed localization algorithms.**

***Index Terms*—Distributed optimization, range-based node localization, range-free node localization, second-order cone programming (SOCP), semidefinite programming (SDP), sequential greedy optimization (SGO) algorithm, wireless sensor network (WSN).**

## I. Introduction

Wireless sensor networks (WSNs) consist of a large number of tiny, low-power, and randomly deployed sensor nodes that have sensing, processing, and communication capabilities [1]. Most applications of WSNs, e.g., environmental monitoring, search and rescue, target tracking, etc., require the knowledge of positions of sensor nodes. In general, for economic consideration, only a small fraction of nodes' positions are measured by global positioning system (GPS) or manual configuration (these nodes are commonly called anchors), especially for large-scale sensor networks. Hence, developing efficient node self-localization algorithms is necessary for wireless sensor networks.

The node localization problem in sensor networks includes range-based localization and range-free localization [2]. The former relies on range (or distance) measurements between nodes that can be estimated by the received signal strength (RSS) method or the time-of-arrival (TOA) method [3], while the latter uses only connectivity information (i.e., whether a node is in the transmission range of another node) [7], [10]. Correspondingly, localization algorithms can be grouped into range-based localization algorithms and range-free localization algorithms. The range-based localization algorithms can provide higher localization accuracy than the range-free localization algorithms but the latter is cheaper and simpler since they do not require special hardware for ranging. In terms of computational paradigm, the localization algorithms can also be divided into centralized algorithms and distributed algorithms [2], [8], [9]. The centralized algorithms require transmission of all range measurements or connectivity information between nodes to a fusion center (e.g., a sink node) for processing, resulting in large communication energy and bandwidth consumption and thereby shortening the lifetime of the whole network. The distributed algorithms are energy-efficient and scalable to the size of the networks, where the whole task of node localization is cooperatively carried out by all nodes with local information exchange between neighboring nodes. Hence, distributed localization algorithms are much more attractive for large-scale sensor networks.

Many localization algorithms have been proposed for sensor network localization. Among the range-free localization algorithms, [4]–[6] are heuristic, simple, and allow distributed implementation, but they are less accurate unless many anchors are used. Based on the classical multidimensional scaling (MDS) technique, a set of localization algorithms are proposed in [7], which are applicable to both range-based localization and range-free localization, and significantly outperform the heuristic algorithms. Among these algorithms, MDS-MAP (P, R) is the best for its ability to localize irregular networks. However, it is very complicated and costly because of its kernel strategy, i.e., first building a local map for each node and then merging these local maps together to form a global. Centralized computation is necessary when merging local maps. Hence, although the MDS-MAP (P, R) can be implemented in networks, it is not suitable for large-scale networks. A metric MDS-based algorithm is proposed for range-based localization in [8]. This algorithm is very suitable for distributed implementation. However, its local convergence property implies bad localization performance if

Manuscript received October 28, 2009; accepted February 04, 2010. Date of publication March 11, 2010; date of current version May 14, 2010. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Mathini Sellathurai. This work was supported in part by the National Nature Science Foundation of China by Grants 60772100, 60832009, 60872017, by the Cooperative Research Project with Fujitsu R&D Bejing Center, and by the China Scholarship Council. This work was presented in part at the IEEE Global Communications Conference, HI, December 2009.

Q. Shi, C. He, and L. Jiang are with the Department of Electronic Engineering, Shanghai Jiao Tong University, Shanghai, China, 200240 (e-mail: shiqj@sjtu.edu.cn; chenhe@sjtu.edu.cn; lgjiang@sjtu.edu.cn).

H. Chen is with the Institute of Industrial Science, The University of Tokyo, Tokyo, Japan (e-mail: hongyang@mcl.iis.u-tokyo.ac.jp).

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

Digital Object Identifier 10.1109/TSP.2010.2045416

1053-587X/$26.00 © 2010 IEEE

a good initialization is not available. In [9], a robust multilateration-based iterative localization algorithm is proposed for range-based localization. It is quite lightweight and very suitable for distributed implementation. However, there is no theoretic guarantee for the convergence of the algorithm.

Convex optimization-based localization algorithms are popular in sensor network localization due to their global convergence property. Most of them are for range-based localization [10]–[17], [19], while a few for range-free localization [10], [18]. The common philosophy of these algorithms is formulating the localization problem as a nonlinear optimization problem and then relaxing the resulting problem as a convex optimization problem solved by efficient interior-point algorithm [27]. Two convex relaxation techniques are involved in these algorithms, second-order cone programming (SOCP) relaxation in [10], [11], and [19] and semidefinite programming (SDP) relaxation in [12]–[18]. The SDP-based algorithms are very computationally expensive for large-scale networks (more than hundreds of nodes) and require centralized computation due to their complex structures. To reduce the computational load, [16] proposes a cluster-based SDP localization algorithm. Although this algorithm can be implemented in networks, it is not a good distributed algorithm since: 1) a cluster formation algorithm is necessary at the beginning of the algorithm, resulting in extra communication cost; 2) centralized computation is needed at each cluster; and 3) it may fail when there are not sufficient anchors. An alternative way to reduce the computational load is to further relax the range-based localization problem [17]. In [17], the node-based SDP relaxation and edge-based SDP relaxation are proposed. Although these two relaxations are weaker than the general SDP relaxation [13], they are both efficient and accurate in computation. Compared to the SDP relaxation, although weaker (and thus less accurate), the SOCP relaxation has simpler structure and allows efficient distributed implementation. In [19], a totally asynchronous distributed localization algorithm is proposed using SOCP for range-based localization. The idea can be easily extended to range-free localization. However, the authors of [19] show the convergence of their algorithm numerically, not theoretically.

Unlike the convex optimization-based localization algorithms, considering all constraints imposed by distance measurements and/or connectivity information, new formulations of localization problems based on nondifferentiable optimization are proposed for both range-based localization [20] and range-free localization [21]. The resulting nondifferentiable problems are solved by the normalized incremental subgradient (NIS) algorithm [20]. The NIS-based localization algorithms have much better localization performance than most previous algorithms such as the robust SDP method [14] with refinement by the dwMDS method [8], the MDS-MAP (C) method [7], etc. However, distributed implementation of these algorithms requires constructing a path across all nodes beforehand.

The localization algorithms mentioned above except [15] and [19] all assume accurate anchor positions. In fact, the anchor positions cannot be exactly known due to the limit accuracy of the civilian GPS and/or manual observation errors. In addition, there is uncertainty in the nodes' maximum radio range due to radio irregularity [6]. In this paper, both the range-based localization problem and the range-free localization problem are studied, considering the uncertainties in range measurements, radio range, and anchor positions. By using a greedy optimization algorithm, we propose two distributed localization algorithms for both range-based localization and range-free localization. The proposed localization algorithms can be implemented partially asynchronously in networks. Extensive simulations are conducted to demonstrate the efficiency and accuracy of the proposed distributed localization algorithms.

Our main contributions are threefold:

1) the sequential greedy optimization (SGO) algorithm is proposed and its convergence property is proved and analyzed. The SGO algorithm is a natural extension of the nonlinear Gauss-Seidel algorithm [30] but it is more suitable for distributed optimization in networks.
2) a unified optimization framework is proposed for sensor network localization and two convex localization formulations are obtained based on the SDP relaxation techniques. By using the SGO algorithm, we show for the first time that the edge-based SDP (ESDP) relaxation formulation can be solved in a distributed way in networks through solving a sequence of second-order cone programming.
3) distributed refinement algorithms that can further improve localization accuracy are proposed for both range-based localization and range-free localization by applying the SGO algorithm to nonconvex localization formulations.

The remainder of this paper is organized as follows. In the next section we state the SGO algorithm and demonstrate its convergence property. In Section III, we present the unified optimization framework for sensor network localization, derive two convex formulations, and develop distributed localization algorithms using ESDP formulation as well as refinement algorithms. Section IV provides numerical experiments to illustrate the performance of the proposed localization algorithms, while Section V concludes the paper.

Throughout this paper, we use upper-case bold type for matrices, lower-case bold type for vectors, regular type for scalars. $\mathbb{R}^n$ denotes the $n$-dimensional Euclidean space. $\times_{i=1}^{n} S_i$ denotes the Cartesian product over $n$ sets. $|S|$ denotes the cardinality of a set $S$. The supscript $^T$ denotes the transpose. $\mathbf{0}$ and $\boldsymbol{e}_i$, respectively, denote the zero vector and the unit column vector with its $i$th element being one, whose dimension will be clear from the context. $\mathbf{I}_m$ denotes the $m \times m$ identity matrix. For a symmetrical matrix $\mathbf{A}$, $A_{ij}$ denotes the $(i,j)$th entry of $\mathbf{A}$, and $\mathbf{A}_{(i_1,i_2,\ldots,i_k)}$ denotes the principal submatrix extracted from the rows and columns indexed by $i_1, i_2, \ldots, i_k$. For symmetrical matrices $\mathbf{A}$ and $\mathbf{B}$, $\mathbf{A} \succeq \mathbf{B}$ means that $\mathbf{A} - \mathbf{B}$ is positive semidefinite, and $\mathbf{A} \bullet \mathbf{B}$ denotes the Frobenius inner product of matrices $\mathbf{A}$ and $\mathbf{B}$. For vectors $\boldsymbol{a}$ and $\boldsymbol{b}$, $\boldsymbol{a} \succeq \boldsymbol{b}$ denotes componentwise inequality, and $\boldsymbol{a} \cdot \boldsymbol{b}$ denotes the componentwise multiplication of vectors $\boldsymbol{a}$ and $\boldsymbol{b}$.

## II. Sequential Greedy Optimization Algorithm

Many optimization algorithms have been developed for multivariate problems [26], [27]. Most of them fall into the category of iterative algorithms which are generally based on search directions. In contrast with general iterative optimization algorithms, there exists a special type of iterative algorithms, in which decision variables are partitioned into *independent* blocks and, at each iteration the objective function is minimized over one of blocks while fixing the rest. This type of algorithms are sometimes called *alternating minimization* algorithm [22], *coordinate descent* algorithm [28], [29], or *nonlinear Gauss-Seidel* (NGS) algorithm [30]. Note that, the first two are special cases of the last, hence, we discuss only the NGS algorithm in the sequel. The NGS algorithm is applicable to unconstrained optimization problems or problems with constraints of Cartesian product set. It works greedily in the sense that it can guarantee the objective function nonincreases at each iteration. Particularly, using the NGS algorithm, a complex multivariate problem can be broken down into a sequence of small-scale subproblems and the implementation can be easily realized [23]–[25], even in a distributed way (if subproblems are assigned to different processors).

In this section, we extend the NGS algorithm to a more general form. We refer to the resulting algorithm as SGO algorithm. In the SGO algorithm, the decision variables are not strictly splitted into independent blocks as in the NGS algorithm. In other words, the SGO algorithm allows some decision variables to be updated in different subproblems. This extension is natural but is important since it makes the SGO algorithm more suitable for distributed optimization in networks. The convergence property of the SGO algorithm is analyzed and proved by using first-order Karush-Kuhn-Tucker (KKT) condition, without the assumption of convexity of the objective function and/or the constraint set. In addition, we discuss the application of the SGO algorithm for general optimization problems where the constraints are coupled (i.e., not of Cartesian product type).

### A. Algorithm

We say an optimization problem is *sparse-constrained* if it can be written in the form

$$\begin{aligned}&\min f(\boldsymbol{x})\\ &\text{s.t. } \boldsymbol{g}_i(\boldsymbol{x}_i)\preceq \boldsymbol{0},\quad \boldsymbol{h}_i(\boldsymbol{x}_i)=\boldsymbol{0},\quad i=1,2,\ldots,n \end{aligned}\tag{1}$$

where, $\boldsymbol{x}=(\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_n)$, $\boldsymbol{x}_i\in\mathbb{R}^{d_i}$, $d_i$ is a positive integer, $i=1,2,\ldots,n$; $f:\mathbb{R}^D\to\mathbb{R}$, $D=\sum_{i=1}^n d_i$; $\boldsymbol{g}_i:\mathbb{R}^{d_i}\to\mathbb{R}^{m_i}$, $\boldsymbol{h}_i:\mathbb{R}^{d_i}\to\mathbb{R}^{l_i}$, $i=1,2,\ldots,n$, are differentiable functions; $m_i$ and $l_i$, $i=1,2,\ldots,n$, are nonnegative integers. We assume the minimum of $f(\boldsymbol{x})$ exists and is finite. Note that, $m_i=0$ $(l_i=0)$ means there is no inequality (equality) constraint on $\boldsymbol{x}_i$. Particularly, when all the $m_i$ and $l_i$ are equal to zero, the problem (1) is translated into a special sparse-constrained problem, i.e., an unconstrained optimization problem.

In the NGS algorithm, subproblems are sequentially minimized, assuming $f(\boldsymbol{x})$ is a strictly convex function of each $\boldsymbol{x}_i$. However, given a general $f(\boldsymbol{x})$, it is hard to obtain a global optimum for subproblems. Here, we will describe the SGO algorithm with a new operator "dec" instead of "min." The new operator used in an iterative algorithm is defined as

$$\begin{aligned}\boldsymbol{z}^k&=\arg\operatorname*{dec}_{\boldsymbol{z}\in\mathbb{B}}\varphi(\boldsymbol{z})\\ &=\begin{cases}\ddot{\boldsymbol{z}} & \text{if } \varphi(\boldsymbol{z}^{k-1})-\varphi(\ddot{\boldsymbol{z}})>0\\ \boldsymbol{z}^{k-1} & \text{otherwise}\end{cases}\end{aligned}\tag{2}$$

where $\varphi:\mathbb{B}\in\mathbb{R}^m\to\mathbb{R}$, is a differentiable function, $\boldsymbol{z}^k$ is the $k$th iterate, and $\ddot{\boldsymbol{z}}\in\mathbb{B}$ is a local minimizer of $\varphi(\boldsymbol{z})$ found by using a certain optimization method and such that $\varphi(\ddot{\boldsymbol{z}})\le\varphi(\boldsymbol{z}^{k-1})$. Note that, $\ddot{\boldsymbol{z}}$ is uniquely found once a specific optimization method is selected (i.e., the algorithmic rule is determined). For example, we can use descent methods [27], [30] to reach a local minimizer of $\varphi(\boldsymbol{z})$ from $\boldsymbol{z}^{k-1}$. In this sense, (2) is well defined.

In addition, we will use the following notations when describing the algorithm. Let $S_i=\{\boldsymbol{y}\in\mathbb{R}^{d_i}|\boldsymbol{g}_i(\boldsymbol{y})\preceq 0,\boldsymbol{h}_i(\boldsymbol{y})=0\}$, $i=1,2,\ldots,n$, and $S=\times_{i=1}^n S_i$. Let $\mathbb{N}_j$, $j=1,2,\ldots,p$, be subsets of $\mathbb{N}=\{1,2,\ldots,n\}$ and such that $\bigcup_{j=1}^p\mathbb{N}_j=\mathbb{N}$, and $\mathbb{L}_i$ be the $i$th index set whose components are the indexes of those $\mathbb{N}_j s$ which contain $i$. For example, if $2\in\mathbb{N}_1\bigcap\mathbb{N}_2\bigcap\mathbb{N}_5$, then $\{1,2,5\}\subseteq\mathbb{L}_2$. In the SGO algorithm, it means that the variable $\boldsymbol{x}_2$ is not only optimized in the subproblem 1, but also in the subproblems 2 and 5.

The SGO algorithm is described as follows. Assume a random initialization $\boldsymbol{x}^0$. Given $\boldsymbol{x}^{k-1}$ obtained after the $(k-1)$th iteration, it then sequentially performs $p$ greedy steps or subiterations in the $k$th iteration

$$\begin{aligned}\boldsymbol{x}^k_{\mathbb{N}_1}&=\arg\operatorname*{dec}_{\boldsymbol{x}_{\mathbb{N}_1}\in S_{\mathbb{N}_1}} f(\boldsymbol{x}_{\mathbb{N}_1},\boldsymbol{x}^{k-1}_{-\mathbb{N}_1})\\ \boldsymbol{x}^k_{\mathbb{N}_2}&=\arg\operatorname*{dec}_{\boldsymbol{x}_{\mathbb{N}_2}\in S_{\mathbb{N}_2}} f(\boldsymbol{x}_{\mathbb{N}_2},\boldsymbol{x}^{k,1}_{-\mathbb{N}_2})\\ &\vdots\\ \boldsymbol{x}^k_{\mathbb{N}_p}&=\arg\operatorname*{dec}_{\boldsymbol{x}_{\mathbb{N}_p}\in S_{\mathbb{N}_p}} f(\boldsymbol{x}_{\mathbb{N}_p},\boldsymbol{x}^{k,p-1}_{-\mathbb{N}_p})\end{aligned}\tag{3}$$

and finally gets $\boldsymbol{x}^k=\boldsymbol{x}^{k,p}$, where $\boldsymbol{x}^{k,j}$ denotes the new $\boldsymbol{x}$ obtained after the $j$th subiteration of the $k$th iteration (note:$\boldsymbol{x}^{k,0}=\boldsymbol{x}^{k-1}$); $\boldsymbol{x}_{\mathbb{N}_j}$ (respectively, $\boldsymbol{x}_{-\mathbb{N}_j}$) denotes the variable made up of $\boldsymbol{x}_i s$ indexed by the components of $\mathbb{N}_j$ (respectively, $\mathbb{N}\backslash\mathbb{N}_j$); $S_{\mathbb{N}_j}=\times_{i\in\mathbb{N}_j}S_i$.

### B. Convergence Analysis and Discussion

In general, an iterative algorithm converges means that its iterates converge. However, it cannot be established that the iterates generated by the SGO algorithm must converge, unless certain strong assumptions are made on the objective function. In what follows, we assume the SGO algorithm (3) is well defined[1] and give two propositions which indicate some properties of the SGO algorithm.

*Proposition 1:* Let $\{\boldsymbol{x}_k\}$ be the sequence generated by the SGO algorithm (3). Then the SGO algorithm converges in the sense that the sequence $\{f(\boldsymbol{x}^k)\}$ converges.

*Proof:* Let $f_{k,j}\triangleq f(\boldsymbol{x}^k_{\mathbb{N}_j},\boldsymbol{x}^{k,j-1}_{-\mathbb{N}_j})$, so we have $f_{k,p}=f(\boldsymbol{x}^k)$. From (3), we conclude that

$$f_{k,p}\le f_{k,p-1}\le\cdots\le f_{k,1}\le f_{k-1,p}.$$

[1] The assumption here means that an exact local minimum can be found at each step of the algorithm. However, it should be pointed out that, we only need to find an approximate local minimum at each step of the SGO algorithm in practical applications if an exact local minimum is not available in reasonable time.

Thus we have $f(\boldsymbol{x}^k) \leq f(\boldsymbol{x}^{k-1})$ for all $k > 0$. This implies the sequence $\{f(\boldsymbol{x}^k)\}$ is nonincreasing, and must converge since $f(\boldsymbol{x})$ is bounded below by its minimum on $S$. ■

*Proposition 2:* Assume $\bar{\boldsymbol{x}} = (\bar{\boldsymbol{x}}_1, \bar{\boldsymbol{x}}_2, \ldots, \bar{\boldsymbol{x}}_n)$ is a limit point of the sequence $\{\boldsymbol{x}_k\}$ generated by the SGO algorithm (3) and $\bar{\boldsymbol{x}} \in S$. Then $\bar{\boldsymbol{x}}$ is a KKT point of the problem (1) under some regularity conditions [31].

*Proof:* Because $\bar{\boldsymbol{x}} = (\bar{\boldsymbol{x}}_1, \bar{\boldsymbol{x}}_2, \ldots, \bar{\boldsymbol{x}}_n)$ is a limit point of the sequence $\{\boldsymbol{x}_k\}$ and $\bar{\boldsymbol{x}} \in S$, it must hold that

$$\bar{\boldsymbol{x}}_{\mathbb{N}_j} = \arg \underset{\boldsymbol{x}_{\mathbb{N}_j} \in S_{\mathbb{N}_j}}{\operatorname{dec}} f(\boldsymbol{x}_{\mathbb{N}_j}, \bar{\boldsymbol{x}}_{-\mathbb{N}_j}), j = 1, 2, \ldots, p \tag{4}$$

implying that, for each subproblem, $\bar{\boldsymbol{x}}_{\mathbb{N}_j}$ must be at least a local minimizer, otherwise we can further decrease $f(\boldsymbol{x})$ at $\bar{\boldsymbol{x}}$ by finding a local minimizer for the subproblem that doesn't hold, which contradicts the premise that $\bar{\boldsymbol{x}}$ is a limit point. Under the assumption of regularity for each subproblem, the first-order necessary condition, generally known as KKT condition, holds for each subproblem, i.e., for each $j \in \{1, 2, \ldots, p\}$ and $i \in \mathbb{N}_j$, there exist Lagrange multipliers $\boldsymbol{\lambda}_i^j \in \mathbb{R}^{m_i}$ and $\boldsymbol{\mu}_i^j \in \mathbb{R}^{l_i}$, such that[2]

$$\begin{cases} \nabla_{\boldsymbol{x}_i} f(\bar{\boldsymbol{x}}) + \boldsymbol{J}_i^T(\bar{\boldsymbol{x}}_i)\boldsymbol{\lambda}_i^j + \bar{\boldsymbol{J}}_i^T(\bar{\boldsymbol{x}}_i)\boldsymbol{\mu}_i^j = \boldsymbol{0} \\ \boldsymbol{g}_i(\bar{\boldsymbol{x}}_i) \preceq \boldsymbol{0}, \boldsymbol{h}_i(\bar{\boldsymbol{x}}_i) = \boldsymbol{0} \\ \boldsymbol{\lambda}_i^j \succeq \boldsymbol{0}, \boldsymbol{\lambda}_i^j \cdot \boldsymbol{g}_i(\bar{\boldsymbol{x}}_i) = \boldsymbol{0} \end{cases} . \tag{5}$$

where $\boldsymbol{J}_i(\boldsymbol{x}_i) = [\nabla_{\boldsymbol{x}_i} g_{i,1}(\boldsymbol{x}_i) \nabla_{\boldsymbol{x}_i} g_{i,2}(\boldsymbol{x}_i) \cdots \nabla_{\boldsymbol{x}_i} g_{i,m_i}(\boldsymbol{x}_i)]^T$ is the Jacobi matrix of $\boldsymbol{g}_i(\boldsymbol{x}_i)$ with respect to $\boldsymbol{x}_i$ and $\bar{\boldsymbol{J}}_i(\boldsymbol{x}_i) = [\nabla_{\boldsymbol{x}_i} h_{i,1}(\boldsymbol{x}_i) \nabla_{\boldsymbol{x}_i} h_{i,2}(\boldsymbol{x}_i) \cdots \nabla_{\boldsymbol{x}_i} h_{i,l_i}(\boldsymbol{x}_i)]^T$ is the Jacobi matrix of $\boldsymbol{h}_i(\boldsymbol{x}_i)$ w.r.t $\boldsymbol{x}_i$.

Combining the KKT conditions of all subproblems, we have

$$\begin{cases} \nabla_{\boldsymbol{x}_i} f(\bar{\boldsymbol{x}}) + \boldsymbol{J}_i^T(\bar{\boldsymbol{x}}_i)\boldsymbol{\lambda}_i + \bar{\boldsymbol{J}}_i^T(\bar{\boldsymbol{x}}_i)\boldsymbol{\mu}_i = \boldsymbol{0} \\ \boldsymbol{g}_i(\bar{\boldsymbol{x}}_i) \preceq \boldsymbol{0}, \boldsymbol{h}_i(\bar{\boldsymbol{x}}_i) = \boldsymbol{0} \\ \boldsymbol{\lambda}_i \succeq \boldsymbol{0}, \boldsymbol{\lambda}_i \cdot \boldsymbol{g}_i(\bar{\boldsymbol{x}}_i) = \boldsymbol{0} \\ i = 1, 2, \ldots, n \end{cases} \tag{6}$$

where $\boldsymbol{\lambda}_i = (1)/(|\mathbb{L}_i|) \sum_{j \in \mathbb{L}_i} \boldsymbol{\lambda}_i^j$, $\boldsymbol{\mu}_i = (1)/(|\mathbb{L}_i|) \sum_{j \in \mathbb{L}_i} \boldsymbol{\mu}_i^j$. This implies that for the point $\bar{\boldsymbol{x}}$, there exist Lagrange multipliers $\boldsymbol{\lambda} = (\boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2, \ldots, \boldsymbol{\lambda}_n)$ and $\boldsymbol{\mu} = (\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_n)$ satisfying the KKT condition of the primal problem. Therefore, $\bar{\boldsymbol{x}}$ is a KKT point of the primal problem.[3]

Proposition 2 implies that if the problem (1) is a convex optimization problem, then the limit point $\bar{\boldsymbol{x}}$ is a global optimum.

Some remarks are made as follows.

*Remark 1:* At the limit point $\bar{\boldsymbol{x}}$, the objective function value, i.e., $f(\bar{\boldsymbol{x}})$, is equal to the limit of the function sequence $\{f(\boldsymbol{x}^k)\}$ since $f(\boldsymbol{x})$ is continuous. Hence, although the convergence of the SGO algorithm cannot be established, an approximate limit point of the sequence $\{\boldsymbol{x}^k\}$ can be easily found in terms of the reduction amount of the objective function. Specifically, $\boldsymbol{x}^k$ is an approximate limit point (and, thus, an approximate KKT point) if $f(\boldsymbol{x}^{k-1}) - f(\boldsymbol{x}^k) <= \eta$ holds for a given sufficiently small and positive scalar $\eta$.

*Remark 2:* When $\mathbb{N}_i = \{i\}$ and $p = n$, the SGO algorithm (3) reduces to $(i = 1, 2 \cdots, n)$

$$\boldsymbol{x}_i^k = \arg \underset{\boldsymbol{x}_i \in S_i}{\operatorname{dec}} f\left(\boldsymbol{x}_1^k, \ldots, \boldsymbol{x}_{i-1}^k, \boldsymbol{x}_i, \boldsymbol{x}_{i+1}^{k-1}, \ldots, \boldsymbol{x}_n^{k-1}\right) \tag{7}$$

which coincides with the iterative form of the nonlinear Gauss-Seidel algorithm [30]. The SGO algorithm of (7) is more often used in practice although it is a special case of (3).

*Remark 3:* Compared to (7), the algorithm (3) can be easily exploited to devise the distributed algorithms for in-network processing. Considering for an extreme example that, two nodes in a network, each with an unknown state variable $\boldsymbol{x}_i \in S_i$ $(i = 1, 2)$ and a coupled unknown parameter $\boldsymbol{x}_{12} \in S_3$ that is related to both, try to estimate their state variables in a decentralized way by minimizing a cost function of the form $f(\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_{12})$. Using (3), each node minimizes $f$ jointly over its state variable and the coupled variable once the update of the other node's state variable is received, implying that the message-passing on the coupled variable is not needed between two nodes. However, it is readily known that using (7) will incur more computation and/or communication overhead.

*Remark 4:* Optimization problems with coupled constraints often appear in distributed signal processing in networks. Directly using the SGO type algorithm (i.e., sequentially optimize the objective function over some variables while fixing others) to solve such problems is possible (due to its greedy nature) but not safe. For example, the following problem:

$$\begin{aligned} &\min x_1 + x_2 \\ &\text{s.t. } x_1 x_2 \geq 1, x_1 \geq 0, x_2 \geq 0 \end{aligned} \tag{8}$$

has unique global solution $x_1 = x_2 = 1$, however, the SGO algorithm would converge to any point such that $x_1 x_2 = 1$.

For problems (particularly for convex problems) with coupled constraints, we can use[4] barrier method (e.g., log-barrier method [27]), to eliminate all coupled constraints (respectively, all constraints) and derive a sparse-constrained (respectively, unconstrained) approximated problem. It is interesting to see that, solving the resulting approximated problem by using the SGO algorithm is equivalent to solving the primal problem using a SGO type algorithm with each subproblem being *approximately* (not perfectly) solved by the barrier method or other equivalent methods (e.g., primal-dual interior point method [27]). In other words, it is safe to directly use a SGO type algorithm to the problems with coupled constraints while each subproblem is approximately solved.

To clarify the argument above, revisit the example above. Directly solving (8) by using the SGO algorithm with perfectly solving two subproblems will lead to an unexpected convergence point. However, by the log-barrier method, the approximated problem is

$$\min_{x_1 \geq 0, x_2 \geq 0} x_1 + x_2 - \epsilon \log(x_1 x_2 - 1)$$

where $\epsilon$ is a very small positive scalar which controls the approximation accuracy, and the SGO solution (which is the same

[2] When $m_i = 0$ $(l_i = 0)$, the inequality (equality) constraints-related terms in (5) should be canceled. For example, when $m_i = 0$ and $l_i \neq 0$, (5) can be simplified as $\{\nabla_{\boldsymbol{x}_i} f(\bar{\boldsymbol{x}}) + \bar{\boldsymbol{J}}_i^T(\bar{\boldsymbol{x}}_i)\boldsymbol{\mu}_i^j = \boldsymbol{0}, \boldsymbol{h}_i(\bar{\boldsymbol{x}}_i) = \boldsymbol{0}\}$. In this paper, for simplicity, we write the first-order necessary condition in a unified framework.

[3] This KKT point is often also a local optimum since the objective function values keep nonincreasing.

[4] Lagrange dual method can also be used to transform a problem with coupled constraints to a sparse-constrained problem.

as the solution achieved by directly using the SGO algorithm to (8) but each subproblem being approximately solved by the same log-barrier method) to this sparse-constrained problem is $x_1 = x_2 = (\epsilon + \sqrt{4+\epsilon})/(2) \approx 1 + (5/8)\epsilon$, which can arbitrarily approach the global solution of the problem by choosing an infinitely small positive $\epsilon$.

*Remark 5:* The order of the subproblems can be varied at different iterations. Moreover, it is not necessary to run all the subproblems at each iteration. These will make the algorithm much more suitable for asynchronous implementation[5] in networks but not influence the convergence of the algorithm.

## III. Distributed Sensor Network Localization Algorithm

In this section, we first propose a unified optimization framework for sensor network localization. Then we derive two centralized localization formulations, named FullSDP and ESDP, based on convex relaxation techniques. Next, by using the SGO algorithm, we show that the ESDP can be implemented in a distributed way with each subproblem being an SOCP. Finally, we propose distributed refinement algorithms for sensor network localization.

### A. Problem Formulation

Assume a sensor network in $\mathbb{R}^d$ ($d = 2$ or 3) has $N$ sensor nodes in total, with $m$ anchor nodes whose locations are known (maybe inaccurate) and $N - m$ sensor nodes whose locations are unknown $(m < N - m)$, and the maximum radio ranges of the nodes are all $R$. Let $\boldsymbol{x}_i$ and $\boldsymbol{a}_i$, $i = 1, 2, \ldots, m$, denote the exact locations and the inaccurate locations of anchor nodes, respectively, and let $\boldsymbol{x}_i$, $i = m+1, m+2, \ldots, N$, denote the locations of sensor nodes. In range-free localization, only the connectivity information of any two nodes, i.e., whether they can communicate with each other directly (without relay), are obtained. We call such two nodes one-hop neighboring nodes, and the set of all one-hop neighboring nodes' index pairs $(i,j)$ with $i < j$ is denoted by $\mathcal{N}^1$. In range-based localization, (noisy) range measurements are taken by one-hop neighboring nodes, denoted by $d_{ij}$ for neighboring nodes $i$ and $j$. In addition, we assume that the location errors of anchors are bounded by $\delta$, i.e., $\|\boldsymbol{x}_i - \boldsymbol{a}_i\| \le \delta$, $i = 1, 2, \ldots, m$. Then the range-based localization problem with anchor position uncertainty is to find a realization of $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N$ such that

$$\begin{aligned} &\|\boldsymbol{x}_i - \boldsymbol{x}_j\| \approx d_{ij}, \forall (i,j) \in \mathcal{N}^1 \\ &\|\boldsymbol{x}_i - \boldsymbol{x}_j\| > R, \forall (i,j) \notin \mathcal{N}^1 \\ &\|\boldsymbol{x}_i - \boldsymbol{a}_i\| \le \delta, i = 1, 2, \ldots, m \end{aligned} \tag{9}$$

and the range-free localization problem with anchor position uncertainty is to find a realization such that

$$\begin{aligned} &\|\boldsymbol{x}_i - \boldsymbol{x}_j\| \le R, \quad \forall (i,j) \in \mathcal{N}^1 \\ &\|\boldsymbol{x}_i - \boldsymbol{x}_j\| > R, \forall (i,j) \notin \mathcal{N}^1 \\ &\|\boldsymbol{x}_i - \boldsymbol{a}_i\| \le \delta, i = 1, 2, \ldots, m. \end{aligned} \tag{10}$$

Note that, there are $(N^2 - N)/(2) + m$ constraints in total. To achieve distributed localization with high accuracy and reduce the computational complexity, we will neglect all '>' constraints in range-based localization, while in range-free localization, we will neglect all '>' constraints but the ones that are related to two-hop neighboring nodes. Here, two-hop neighboring nodes are the nodes that cannot directly communicate with each other but can communicate with just one relay. Hence, for node $i$ and node $j$ who are two-hop neighbors, their distance satisfies $R < \|\boldsymbol{x}_i - \boldsymbol{x}_j\| \le 2R$. Correspondingly, the set of their index pairs $(i,j)$ with $i < j$ are denoted by $\mathcal{N}^2$.

Here we formulate sensor network localization problems in a unified framework as the following minimization problem:

$$\begin{aligned} \min \sum_{(i,j)\in\mathbb{S}} (\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2 - t_{ij})^2 + \sum_{i=1}^{m} \|\boldsymbol{x}_i - \boldsymbol{a}_i\|^2 \\ \text{s.t. } \|\boldsymbol{x}_i - \boldsymbol{a}_i\| \le \delta, i = 1, 2, \ldots, m \end{aligned} \tag{11}$$

where $\mathbb{S}$ is the one-hop neighboring nodes set $\mathcal{N}^1$ and $t_{ij} = d_{ij}^2$ is known in range-based localization; in range-free localization, $\mathbb{S}$ is the union of $\mathcal{N}^1$ and $\mathcal{N}^2$, and $t_{ij}$ is unknown but $0 \le t_{ij} \le R^2$ if $\mathbb{S} = \mathcal{N}^1$ and $R^2 \le t_{ij} \le 4R^2$ if $S = \mathcal{N}^2$. (11) can be interpreted as a least-squares localization formulation considering all measurement uncertainty including the uncertainty of the maximum radio range $R$, range measurements, and anchor positions[6]. Moreover, it should be pointed out that, (11) will become ill-posed if the last term in the objective function is not included.

### B. Convex Relaxations of Sensor Network Localization

The problem (11) is a very complex nonconvex problem that is difficult to solve. We below relax the problem as two convex optimization problems by using SDP relaxation techniques in [12] and [17], respectively, named FullSDP and ESDP as in [17]. For simplicity, we will use $t_{ij} \in T_{ij}$ to denote the linear constraint on $t_{ij}$.

Let $\mathbf{X} = [\boldsymbol{x}_1 \, \boldsymbol{x}_2 \cdots \boldsymbol{x}_N]$. We first equivalently write (11) as

$$\begin{aligned} \min_{\mathbf{Y},\mathbf{X},t_{ij}\text{s}} & \sum_{(i,j)\in\mathbb{S}} (Y_{ii} - 2Y_{ij} + Y_{jj} - t_{ij})^2 \\ & + \sum_{i=1}^{m} \left(Y_{ii} - 2\boldsymbol{a}_i^T \boldsymbol{x}_i + \|\boldsymbol{a}_i\|^2\right) \\ \text{s.t. } & Y_{ii} - 2\boldsymbol{a}_i^T \boldsymbol{x}_i + \|\boldsymbol{a}_i\|^2 \le \delta^2, i = 1, 2, \ldots, m \\ & \mathbf{Y} = \mathbf{X}^T\mathbf{X} \\ & t_{ij} \in T_{ij}, \quad \forall (i,j) \in \mathbb{S}. \end{aligned} \tag{12}$$

By relaxing $\mathbf{Y} = \mathbf{X}^T\mathbf{X}$ as $\mathbf{Y} \succeq \mathbf{X}^T\mathbf{X}$ which means

$$\mathbf{Z} = \begin{bmatrix} \mathbf{Y} & \mathbf{X}^T \\ \mathbf{X} & \mathbf{I}_d \end{bmatrix} \succeq 0 \tag{13}$$

[5] In implementing the SGO algorithm, it is not allowed that two subproblems, which contain coupled variables (i.e., when $\mathbb{N}_i \bigcap \mathbb{N}_j \neq 0$), run concurrently. Hence, "asynchronous" here doesn't mean "totally asynchronous" [30]. To distinguish from the latter, we call it "partially asynchronous."

[6] The range-free localization problem is often formulated as a feasibility problem. However, the uncertainty in maximum radio range will lead to the problem infeasible, implying the method such as [18] or [10] that assume the problem has feasible solution may fail.

according to Schur complement [12], [27], we then obtain the FullSDP

$$\begin{aligned}\min_{\mathbf{Z},t_{ij}\text{s}} & \sum_{(i,j)\in\mathbb{S}} (\mathbf{E_{ij}}\bullet\mathbf{Z}-t_{ij})^2+\sum_{i=1}^{m}\mathbf{F_i}\bullet\mathbf{Z}\\ \text{s.t. } & \mathbf{F_i}\bullet\mathbf{Z}\le\delta^2, i=1,2,\ldots,m\\ & \mathbf{Z}\succeq 0\\ & \mathbf{Z}_{(N+1,\ldots,N+d)}=\mathbf{I}_d\\ & t_{ij}\in T_{ij},\quad\forall(i,j)\in\mathbb{S}\end{aligned}\tag{14}$$

where

$$\mathbf{E_{ij}}=\begin{bmatrix}\boldsymbol{e}_i-\boldsymbol{e}_j\\ \mathbf{0}_d\end{bmatrix}\begin{bmatrix}\boldsymbol{e}_i-\boldsymbol{e}_j\\ \mathbf{0}_d\end{bmatrix}^T,\ \mathbf{F_i}=\begin{bmatrix}\boldsymbol{e}_i\\ -\boldsymbol{a}_i\end{bmatrix}\begin{bmatrix}\boldsymbol{e}_i\\ -\boldsymbol{a}_i\end{bmatrix}^T.$$

Also, using the edge-based SDP relaxation technique [17], we can relax (11) as the following convex optimization problem, i.e., ESDP, which decomposes the single constraint of large positive semidefinite matrix $\mathbf{Z}$ in (14) into a number of constraints of small positive semidefinite matrices.

$$\begin{aligned}\min_{\mathbf{Z},t_{ij}\text{s}} & \sum_{(i,j)\in\mathbb{S}} (\mathbf{E_{ij}}\bullet\mathbf{Z}-t_{ij})^2+\sum_{i=1}^{m}\mathbf{F_i}\bullet\mathbf{Z}\\ \text{s.t. } & \mathbf{F_i}\bullet\mathbf{Z}\le\delta^2, i=1,2,\ldots,m\\ & \mathbf{Z}_{(i,j,N+1,\ldots,N+d)}\succeq 0,\forall(i,j)\in\mathbb{S}\\ & \mathbf{Z}_{(N+1,\ldots,N+d)}=\mathbf{I}_d\\ & t_{ij}\in T_{ij},\quad\forall(i,j)\in\mathbb{S}.\end{aligned}\tag{15}$$

Notice that a constraint in the form of $\|\boldsymbol{x}\|^2\le h$ can be turned into a second-order cone constraint

$$\sqrt{4\|\boldsymbol{x}\|^2+(h-1)^2}\le h+1.$$

Hence, the FullSDP and ESDP can be equivalently written as

$$\begin{aligned}\min_{h,\mathbf{Z},t_{ij}\text{s}} & \ h+\sum_{i=1}^{m}\mathbf{F_i}\bullet\mathbf{Z}\\ \text{s.t. } & \sqrt{4\sum_{(i,j)\in\mathbb{S}}(\mathbf{E_{ij}}\bullet\mathbf{Z}-t_{ij})^2+(h-1)^2}\le(h+1)\\ & \mathbf{F_i}\bullet\mathbf{Z}\le\delta^2, i=1,2,\ldots,m\\ & \mathbf{Z}\succeq 0,\\ & \mathbf{Z}_{(N+1,\ldots,N+d)}=\mathbf{I}_d\\ & t_{ij}\in T_{ij},\quad\forall(i,j)\in\mathbb{S}\end{aligned}\tag{16}$$

and

$$\begin{aligned}\min_{h,\mathbf{Z},t_{ij}\text{s}} & \ h+\sum_{i=1}^{m}\mathbf{F_i}\bullet\mathbf{Z}\\ \text{s.t. } & \sqrt{4\sum_{(i,j)\in\mathbb{S}}(\mathbf{E_{ij}}\bullet\mathbf{Z}-t_{ij})^2+(h-1)^2}\le(h+1)\\ & \mathbf{F_i}\bullet\mathbf{Z}\le\delta^2, i=1,2,\ldots,m\\ & \mathbf{Z}_{(i,j,N+1,\ldots,N+d)}\succeq 0,\quad\forall(i,j)\in\mathbb{S}\\ & \mathbf{Z}_{(N+1,\ldots,N+d)}=\mathbf{I}_d\\ & t_{ij}\in T_{ij},\quad\forall(i,j)\in\mathbb{S}\end{aligned}\tag{17}$$

respectively. (16) and (17) can be further written in standard cone programming forms and solved by using some efficient convex cone programming solvers such as SeDuMi [32] or SDPT3 [33] in a centralized way.

### *C. Distributed Localization via Edge-Based SOCP (ESOCP)*

Centralized solving the fullSDP or the ESDP formulation is quite computationally intensive when the problem has large size. It requires numerous memory resource, especially for the range-free localization formulation which involves two-hop constraints. In the following, we show that the ESDP formulation (15) can be solved in a distributed fashion by using the SGO algorithm. Particularly, each subproblem is an SOCP. We call such an SOCP-based localization method ESOCP.

The edge-based formulation (15) can also be written as

$$\begin{aligned}\min_{\mathbf{Y},\mathbf{X},t_{ij}\text{s}} & \sum_{(i,j)\in\mathbb{S}}(Y_{ii}-2Y_{ij}+Y_{jj}-t_{ij})^2\\ & +\sum_{i=1}^{m}\left(Y_{ii}-2\boldsymbol{a}_i^T\boldsymbol{x}_i+\|\boldsymbol{a}_i\|^2\right)\\ \text{s.t. } & Y_{ii}-2\boldsymbol{a}_i^T\boldsymbol{x}_i+\|\boldsymbol{a}_i\|^2\le\delta^2, i=1,2,\ldots,m\\ & \begin{bmatrix}Y_{ii} & Y_{ij}\\ Y_{ij} & Y_{jj}\end{bmatrix}\succeq\begin{bmatrix}\|\boldsymbol{x}_i\|^2 & \boldsymbol{x}_i^T\boldsymbol{x}_j\\ \boldsymbol{x}_i^T\boldsymbol{x}_j & \|\boldsymbol{x}_j\|^2\end{bmatrix},\forall(i,j)\in\mathbb{S}\\ & t_{ij}\in T_{ij},\quad\forall(i,j)\in\mathbb{S}.\end{aligned}\tag{18}$$

For a positive semidefinite matrix, all of the leading principal minors are nonnegative. Hence, (18) is equivalent to

$$\begin{aligned}\min_{\mathbf{Y},\boldsymbol{x},t_{ij}\text{s}} & \sum_{(i,j)\in\mathbb{S}}(Y_{ii}-2Y_{ij}+Y_{jj}-t_{ij})^2+\\ & \sum_{i=1}^{m}\left(Y_{ii}-2\boldsymbol{a}_i^T\boldsymbol{x}_i+\|\boldsymbol{a}_i\|^2\right)\\ \text{s.t. } & Y_{ii}-2\boldsymbol{a}_i^T\boldsymbol{x}_i+\|\boldsymbol{a}_i\|^2\le\delta^2, i=1,2,\ldots,m\\ & \|\boldsymbol{x}_i\|^2\le Y_{ii}, i=1,2,\ldots,N\\ & \left(Y_{ij}-\boldsymbol{x}_i^T\boldsymbol{x}_j\right)^2\le c_j(Y_{ii}-\|\boldsymbol{x}_i\|^2),\quad\forall(i,j)\in\mathbb{S}\\ & t_{ij}\in T_{ij},\quad\forall(i,j)\in\mathbb{S}\end{aligned}\tag{19}$$

where $c_j=Y_{jj}-\|\boldsymbol{x}_j\|^2$.

In (19), $Y_{ii}$ and $\boldsymbol{x}_i$ are related to the node $i$ only while $Y_{ij}s$ and $t_{ij}s$ are coupled variables related to both node $i$ and node $j$. Hence, (19) is not sparse-constrained. However, it can be solved by using the SGO algorithm according to the Remarks 3 and 4. The subproblem at each anchor node $i$ is

$$\begin{aligned}\min_{\boldsymbol{x}_i,Y_{ii},Y_{ij}\text{s},t_{ij}\text{s}} & \sum_{j\in\mathbb{S}_i}(Y_{ii}-2Y_{ij}+Y_{jj}-t_{ij})^2\\ & +\left(Y_{ii}-2\boldsymbol{a}_i^T\boldsymbol{x}_i+\|\boldsymbol{a}_i\|^2\right)\\ \text{s.t. } & Y_{ii}-2\boldsymbol{a}_i^T\boldsymbol{x}_i+\|\boldsymbol{a}_i\|^2\le\delta^2\\ & \|\boldsymbol{x}_i\|^2\le Y_{ii}\\ & \left(Y_{ij}-\boldsymbol{x}_i^T\boldsymbol{x}_j\right)^2+c_j\|\boldsymbol{x}_i\|^2\le c_jY_{ii},\quad\forall j\in\mathbb{S}_i\\ & t_{ij}\in T_{ij},\quad\forall j\in\mathbb{S}_i,\end{aligned}\tag{20}$$

and at each sensor node $i$ is

$$\begin{aligned}&\min_{\boldsymbol{x}_i, Y_{ii}, Y_{ij}\text{s}, t_{ij}\text{s}} \sum_{j\in\mathbb{S}_i}(Y_{ii}-2Y_{ij}+Y_{jj}-t_{ij})^2\\ &\text{s.t. } \|\boldsymbol{x}_i\|^2 \le Y_{ii}\\ &\quad \left(Y_{ij}-\boldsymbol{x}_i^T\boldsymbol{x}_j\right)^2 + c_j\|\boldsymbol{x}_i\|^2 \le c_j Y_{ii}, \quad \forall j\in\mathbb{S}_i\\ &\quad t_{ij}\in T_{ij},\quad \forall j\in \mathbb{S}_i. \end{aligned}\tag{21}$$

Here $\mathbb{S}_i = \{j|(i,j)\in\mathbb{S} \text{ or } (j,i)\in\mathbb{S}\}$. Note, when solving (20) or (21), node $i$ requires only the estimates of its neighbors, i.e., $\mathbf{Y}_{jj}$ and $\boldsymbol{x}_j$ $(j\in\mathbb{S}_i)$. Hence, the ESOCP can be implemented in a distributed way in networks with neighborhood communication.

Likewise, by transforming the constraints into the second-order cone constraints, (20) and (21) can be, respectively, equivalently written as

$$\begin{aligned}&\min_{\boldsymbol{x}_i, Y_{ii}, h, Y_{ij}\text{s}, t_{ij}\text{s}} h + \left(Y_{ii}-2\boldsymbol{a}_i^T\boldsymbol{x}_i+\|\boldsymbol{a}_i\|^2\right)\\ &\text{s.t. } \sqrt{4\sum_{j\in\mathbb{S}_i}(Y_{ii}-2Y_{ij}+Y_{jj}-t_{ij})^2+(h-1)^2} \le h+1\\ &\quad \sqrt{4\|\boldsymbol{x}_i\|^2+(Y_{ii}-1)^2}\le Y_{ii}+1\\ &\sqrt{4\left(Y_{ij}-\boldsymbol{x}_i^T\boldsymbol{x}_j\right)^2+4c_j\|\boldsymbol{x}_i\|^2+(c_jY_{ii}-1)^2}\\ &\qquad\qquad \le c_jY_{ii}+1,\quad \forall j\in\mathbb{S}_i\\ &\quad Y_{ii}-2\boldsymbol{a}_i^T\boldsymbol{x}_i+\|\boldsymbol{a}_i\|^2\le\delta^2\\ &\quad t_{ij}\in T_{ij},\quad \forall j\in\mathbb{S}_i \end{aligned}\tag{22}$$

and

$$\begin{aligned}&\min_{\boldsymbol{x}_i, Y_{ii}, h, Y_{ij}\text{s}, t_{ij}\text{s}} h\\ &\text{s.t. } \sqrt{4\sum_{j\in\mathbb{S}_i}(Y_{ii}-2Y_{ij}+Y_{jj}-t_{ij})^2+(h-1)^2} \le h+1\\ &\quad \sqrt{4\|\boldsymbol{x}_i\|^2+(Y_{ii}-1)^2}\le Y_{ii}+1\\ &\sqrt{4\left(Y_{ij}-\boldsymbol{x}_i^T\boldsymbol{x}_j\right)^2+4c_j\|\boldsymbol{x}_i\|^2+(c_jY_{ii}-1)^2}\\ &\qquad\qquad \le c_jY_{ii}+1,\quad \forall j\in\mathbb{S}_i\\ &\quad t_{ij}\in T_{ij},\quad \forall j\in\mathbb{S}_i \end{aligned}\tag{23}$$

which are SOCPs and also can be efficiently solved by the interior-point algorithm with the solver SeDuMi [32] or SDPT3 [33].

Compared to the fullSDP or ESDP formulation, each subproblem in the ESOCP has much smaller size (which depends on the number of neighboring nodes). Hence, the ESOCP can be implemented in a processor with limited memory resource, regardless of the network size (i.e., the number of nodes). This is a significant advantage over the fullSDP and ESDP which will fail when the network size is very large. On the other hand, compared to the subproblems of the distributed SOCP [19], the subproblems of the ESOCP have the same computational complexity of $O(n_i^3)$ in the range-based localization, while the computational complexity of the subproblems is $O(m_i^3)$ in the range-free localization. Here, $n_i$ denotes the total number of one-hop neighboring nodes of node $i$ and $m_i$ denotes the total number of both one-hop and two-hop neighboring nodes of node $i$. In general, $n_is$ and $m_is$ are small (typically less than one hundred), hence the subproblems can be efficiently solved. For example, for a network with connectivity level (i.e., the average number of one-hop neighboring nodes of each node) of about 11.7, solving a range-based localization subproblem averagely costs about 0.071 second by the solver SeDuMi with the desired accuracy of $1e-4$ while solving a range-free localization subproblem costs about 0.085 second in our simulations.

### D. Distributed Refinement Algorithms

The convex relaxation-based methods can only approximate the localization problem. In this subsection, we propose distributed refinement algorithms based on the SGO algorithm for both range-based localization and range-free localization. We name them as nonconvex sequential greedy (NCSG) localization algorithms.

*1) Range-Based Refinement Algorithm:* For the range-based localization problem, it can also be formulated as

$$\begin{aligned}&\min_{X} \sum_{(i,j)\in\mathcal{N}^1}(\|\boldsymbol{x}_i-\boldsymbol{x}_j\|-d_{ij})^2+\sum_{i=1}^{m}\|\boldsymbol{x}_i-\boldsymbol{a}_i\|^2\\ &\text{s.t. } \|\boldsymbol{x}_i-\boldsymbol{a}_i\|\le\delta, i=1,2,\ldots,m.\end{aligned}\tag{24}$$

Clearly, (24) is nonconvex and sparse-constrained, so it can be directly solved by the SGO algorithm with local convergence. The subproblem at the anchor node $i$ is

$$\begin{aligned}&\min_{\boldsymbol{x}_i} \sum_{j\in\mathcal{N}_i^1}(\|\boldsymbol{x}_i-\boldsymbol{x}_j\|-d_{ij})^2+\|\boldsymbol{x}_i-\boldsymbol{a}_i\|^2\\ &\text{s.t. } \|\boldsymbol{x}_i-\boldsymbol{a}_i\|\le\delta\end{aligned}\tag{25}$$

and at the sensor node $i$ is

$$\min_{\boldsymbol{x}_i} \sum_{j\in\mathcal{N}_i^1}(\|\boldsymbol{x}_i-\boldsymbol{x}_j\|-d_{ij})^2.\tag{26}$$

Upon receiving the neighbors' estimates, i.e., $\boldsymbol{x}_j$ $(j\in\mathcal{N}_i^1)$, the above two subproblems can be solved by using some nonlinear optimization methods, e.g., Newton method or Sequential Quadratic Programming. Interestingly, these subproblems can be turned into sparse-constrained ones and thus solved by the SGO algorithm.

By introducing intermediate variables $\boldsymbol{v}_js$, (25) can be equivalently written in a sparse-constrained form as follows:

$$\begin{aligned}&\min_{\boldsymbol{x}_i,\boldsymbol{v}_j\text{s}} \sum_{j\in\mathcal{N}_i^1}\|\boldsymbol{x}_i-\boldsymbol{v}_j\|^2+\|\boldsymbol{x}_i-\boldsymbol{a}_i\|^2\\ &\text{s.t. } \|\boldsymbol{x}_i-\boldsymbol{a}_i\|\le\delta\\ &\quad \|\boldsymbol{v}_j-\boldsymbol{x}_j\|=d_{ij}, j\in\mathcal{N}_i^1.\end{aligned}\tag{27}$$

The equivalence between (25) and (27) can be proved by noticing that, for any fixed $\boldsymbol{x}_i\ne\boldsymbol{x}_j$, we must have

$$\boldsymbol{v}_j=\boldsymbol{x}_j+d_{ij}\frac{\boldsymbol{x}_i-\boldsymbol{x}_j}{\|\boldsymbol{x}_i-\boldsymbol{x}_j\|}\tag{28}$$

to minimize $\|\boldsymbol{x}_i-\boldsymbol{v}_j\|^2$ over $\boldsymbol{v}_j$ subject to $\|\boldsymbol{v}_j-\boldsymbol{x}_j\|^2=d_{ij}^2$.

Now we can apply the SGO algorithm to (27). Fixing $\boldsymbol{x}_i$, we have (28), while fixing $\boldsymbol{v}_j s$, we get

$$\boldsymbol{x}_i = \begin{cases} \breve{\boldsymbol{x}}_i & \text{if } \|\breve{\boldsymbol{x}}_i - \boldsymbol{a}_i\| \leq \delta \\ \boldsymbol{a}_i + \delta \frac{\breve{\boldsymbol{x}}_i - \boldsymbol{a}_i}{\|\breve{\boldsymbol{x}}_i - \boldsymbol{a}_i\|} & \text{otherwise} \end{cases} \tag{29}$$

where $\breve{\boldsymbol{x}}_i = (1)/(|\mathcal{N}_i^1| + 1)(\sum_{j \in \mathcal{N}_i^1} \boldsymbol{v}_j + \boldsymbol{a}_i)$.

Hence, starting from $\boldsymbol{x}_i^0 \neq \boldsymbol{x}_j$ ($\forall j \in \mathcal{N}_i^1$), a stepsize-free iterative algorithm for (25) can be obtained, i.e.

$$\boldsymbol{x}_i^k = \begin{cases} \breve{\boldsymbol{x}}_i & \text{if } \|\breve{\boldsymbol{x}}_i - \boldsymbol{a}_i)\| \leq \delta \\ \boldsymbol{a}_i + \delta \frac{\breve{\boldsymbol{x}}_i - \boldsymbol{a}_i}{\|\breve{\boldsymbol{x}}_i - \boldsymbol{a}_i\|} & \text{otherwise} \end{cases} \tag{30}$$

with $\breve{\boldsymbol{x}}_i = (1)/(|\mathcal{N}_i^1| + 1)(\sum_{j \in \mathcal{N}_i^1} (\boldsymbol{x}_j + d_{ij}(\boldsymbol{x}_i^{k-1} - \boldsymbol{x}_j)/(\|\boldsymbol{x}_i^{k-1} - \boldsymbol{x}_j\|)) + \boldsymbol{a}_i)$.

Similarly, (26) can also be written in a sparse-constrained form and then solved by the SGO algorithm, yielding the following simple iterative solution:

$$\boldsymbol{x}_i^k = \frac{1}{|\mathcal{N}_i^1|} \sum_{j \in \mathcal{N}_i^1} \left( \boldsymbol{x}_j + d_{ij} \frac{\boldsymbol{x}_i^{k-1} - \boldsymbol{x}_j}{\|\boldsymbol{x}_i^{k-1} - \boldsymbol{x}_j\|} \right). \tag{31}$$

*2) Range-Free Refinement Algorithm:* For the range-free localization problem, we can formulate it as the following sparse-constrained problem:

$$\begin{aligned} \min_X & \sum_{(i,j) \in \mathcal{N}^1} (\|\boldsymbol{x}_i - \boldsymbol{x}_j\| - R)_+^2 \\ & + \sum_{(i,j) \in \mathcal{N}^2} (R - \|\boldsymbol{x}_i - \boldsymbol{x}_j\|)_+^2 \\ & + \sum_{(i,j) \in \mathcal{N}^2} (\|\boldsymbol{x}_i - \boldsymbol{x}_j\| - 2R)_+^2 + \sum_{i=1}^{m} \|\boldsymbol{x}_i - \boldsymbol{a}_i\|^2 \\ \text{s.t.} & \|\boldsymbol{x}_i - \boldsymbol{a}_i\| \leq \delta, \ i = 1, 2, \ldots, m \end{aligned} \tag{32}$$

where $(\cdot)_+ = \max(\cdot, 0)$, and similarly solve it by using the SGO algorithm in a distributed way.

The subproblem at the anchor node $i$ is

$$\begin{aligned} \min_{\boldsymbol{x}_i} & \sum_{j \in \mathcal{N}_i^1} (\|\boldsymbol{x}_i - \boldsymbol{x}_j\| - R)_+^2 \\ & + \sum_{j \in \mathcal{N}_i^2} (R - \|\boldsymbol{x}_i - \boldsymbol{x}_j\|)_+^2 \\ & + \sum_{j \in \mathcal{N}_i^2} (\|\boldsymbol{x}_i - \boldsymbol{x}_j\| - 2R)_+^2 + \|\boldsymbol{x}_i - \boldsymbol{a}_i\|^2 \\ \text{s.t.} & \|\boldsymbol{x}_i - \boldsymbol{a}_i\| \leq \delta \end{aligned} \tag{33}$$

and its equivalent sparse-constrained formulation is

$$\begin{aligned} \min_{\boldsymbol{x}_i, \boldsymbol{u}_j \text{s}, \boldsymbol{v}_j \text{s}, \boldsymbol{w}_j \text{s}} & \sum_{j \in \mathcal{N}_i^1} \|\boldsymbol{x}_i - \boldsymbol{u}_j\|^2 \\ & + \sum_{j \in \mathcal{N}_i^2} \|\boldsymbol{x}_i - \boldsymbol{v}_j\|^2 \\ & + \sum_{j \in \mathcal{N}_i^2} \|\boldsymbol{x}_i - \boldsymbol{w}_j\|^2 + \|\boldsymbol{x}_i - \boldsymbol{a}_i\|^2 \\ \text{s.t.} \ & \|\boldsymbol{x}_i - \boldsymbol{a}_i\| \leq \delta \\ & \|\boldsymbol{u}_j - \boldsymbol{x}_j\| \leq R, \quad \forall j \in \mathcal{N}_i^1 \\ & \|\boldsymbol{v}_j - \boldsymbol{x}_j\| \geq R, \quad \forall j \in \mathcal{N}_i^2 \\ & \|\boldsymbol{w}_j - \boldsymbol{x}_j\| \leq 2R, \quad \forall j \in \mathcal{N}_i^2. \end{aligned} \tag{34}$$

The equivalence between the above two problems can be established in a similar way for (25) and (27). Applying the SGO algorithm to (34) (first computing $\boldsymbol{u}_j s$, $\boldsymbol{v}_j s$, and $\boldsymbol{w}_j s$ while fixing $\boldsymbol{x}_i$, then conversely) yields the following stepsize-free iterative algorithm:

$$\boldsymbol{x}_i^k = \begin{cases} \breve{\boldsymbol{x}}_i & \text{if } \|\breve{\boldsymbol{x}}_i - \boldsymbol{a}_i\| \leq \delta \\ \boldsymbol{a}_i + \delta \frac{\breve{\boldsymbol{x}}_i - \boldsymbol{a}_i}{\|\breve{\boldsymbol{x}}_i - \boldsymbol{a}_i\|} & \text{otherwise} \end{cases} \tag{35}$$

with

$$\breve{\boldsymbol{x}}_i = \frac{1}{|\mathcal{N}_i^1| + 2|\mathcal{N}_i^2| + 1} \times \left( \sum_{j \in \mathcal{N}_i^1} \boldsymbol{u}_j^k + \sum_{j \in \mathcal{N}_i^2} \boldsymbol{v}_j^k + \sum_{j \in \mathcal{N}_i^2} \boldsymbol{w}_j^k + \boldsymbol{a}_i \right)$$

where

$$\boldsymbol{u}_j^k = \begin{cases} \boldsymbol{x}_i^{k-1} & \text{if } \|\boldsymbol{x}_i^{k-1} - \boldsymbol{x}_j\| \leq R \\ \boldsymbol{x}_j + R \frac{\boldsymbol{x}_i^{k-1} - \boldsymbol{x}_j}{\|\boldsymbol{x}_i^{k-1} - \boldsymbol{x}_j\|} & \text{otherwise} \end{cases} \tag{36}$$

$$\boldsymbol{v}_j^k = \begin{cases} \boldsymbol{x}_i^{k-1} & \text{if } \|\boldsymbol{x}_i^{k-1} - \boldsymbol{x}_j\| \geq R \\ \boldsymbol{x}_j + R \frac{\boldsymbol{x}_i^{k-1} - \boldsymbol{x}_j}{\|\boldsymbol{x}_i^{k-1} - \boldsymbol{x}_j\|} & \text{otherwise} \end{cases} \tag{37}$$

$$\boldsymbol{w}_j^k = \begin{cases} \boldsymbol{x}_i^{k-1} & \text{if } \|\boldsymbol{x}_i^{k-1} - \boldsymbol{x}_j\| \leq 2R \\ \boldsymbol{x}_j + 2R \frac{\boldsymbol{x}_i^{k-1} - \boldsymbol{x}_j}{\|\boldsymbol{x}_i^{k-1} - \boldsymbol{x}_j\|} & \text{otherwise.} \end{cases} \tag{38}$$

Analogously, the stepsize-free iterative algorithm for the subproblem at the sensor node $i$ can be obtained as follows

$$\boldsymbol{x}_i^k = \frac{1}{|\mathcal{N}_i^1| + 2|\mathcal{N}_i^2|} \left( \sum_{j \in \mathcal{N}_i^1} \boldsymbol{u}_j^k + \sum_{j \in \mathcal{N}_i^2} \boldsymbol{v}_j^k + \sum_{j \in \mathcal{N}_i^2} \boldsymbol{w}_j^k \right)$$

where $\boldsymbol{u}_j^k$, $\boldsymbol{v}_j^k$, and $\boldsymbol{w}_j^k$ are the same as (36), (37), and (38), respectively.

## IV. Simulation Results

In this section, we conduct simulations using MATLAB with the solver SeDuMi [32] on a laptop of 2 GB Memory and 2 GHz CPU to evaluate the performance of the proposed localization algorithms. Simulations are performed on planar networks of 100 randomly (uniformly) distributed nodes in a square region of [01] × [01] with eight anchor nodes positioned on the boundary of the networks (their locations are, respectively, [0.1 0.1], [0.1 0.5], [0.1 0.9], [0.5 0.1], [0.5 0.9], [0.9 0.1], [0.9 0.5], and [0.9 0.9]). The ideal maximum radio range of each node is set to be 0.22, i.e., $R = 0.22$, leading to an average connectivity level of about 12. In the simulations of range-based localization, the noisy range measurements are generated according to

$$\hat{d} = d \times |1 + \text{nf} \times \text{randn}| \tag{39}$$

where $d$ denotes the true range measurement, nf is the noise factor, randn is a standard normal random variable, while in the simulations of range-free localization, the nonideal maximum radio ranges of nodes are randomly (uniformly) chosen from the range $[R(1-(\tau)/(2)), R(1+(\tau)/(2))]$, where $\tau$ denotes the degree of radio irregularity. The inaccurate anchor positions are randomly (uniformly) chosen from the disks centered at the true anchor locations with the radius of $\delta$.

We provide the localization performance of the distributed SOCP algorithm for comparison. In the distributed SOCP algorithm, we use the SGO algorithm to solve the following SOCP relaxation formulation[7]

$$\begin{aligned} &\min_{\mathbf{X}, y_{ij}\text{s}} \sum_{(i,j)\in\mathcal{N}^1} (y_{ij}-t_{ij})^2 + \sum_{i=1}^{m} \|\boldsymbol{x}_i - \boldsymbol{a}_i\|^2 \\ &\text{s.t. } \|\boldsymbol{x}_i - \boldsymbol{x}_j\| \le y_{ij}, \quad \forall (i,j) \in \mathcal{N}^1 \\ &\qquad \|\boldsymbol{x}_i - \boldsymbol{a}_i\| \le \delta,\ i=1,2,\ldots,m \end{aligned} \tag{40}$$

where $t_{ij} = d_{ij}$ for range-based localization and $t_{ij} = R$ for range-free localization. It is noted that (40) is equivalent to the following convex sparse-constrained formulation

$$\begin{aligned} &\min_{\mathbf{X}} \sum_{(i,j)\in\mathcal{N}^1} (\|\boldsymbol{x}_i - \boldsymbol{x}_j\| - t_{ij})_+^2 + \sum_{i=1}^{m} \|\boldsymbol{x}_i - \boldsymbol{a}_i\|^2 \\ &\text{s.t. } \|\boldsymbol{x}_i - \boldsymbol{a}_i\| \le \delta,\ i=1,2,\ldots,m \end{aligned} \tag{41}$$

since

$$\min_{\|\boldsymbol{x}_i - \boldsymbol{x}_j\| \le y_{ij}} (y_{ij}-t_{ij})^2 \equiv (\|\boldsymbol{x}_i - \boldsymbol{x}_j\| - t_{ij})_+^2$$

while $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are given. Hence, the distributed SOCP algorithm can globally solve (40). The resulting subproblem at each node can be solved by either the SGO algorithm like the range-free refinement algorithm or the interior-point algorithm by formulating it as an SOCP. In this paper, we use the interior-point algorithm to solve each subproblem for fast convergence. Note that, this distributed SOCP algorithm is different from that in [19] where the convergence of the algorithm is not guaranteed theoretically although its distributed implementation is totally asynchronous.

We emphasize that, according to Remark 1, when applying the SGO algorithm to node localization, we can achieve convergence by terminating the procedure at each node in terms of the reduction amount of the subproblem objective function value. However, we find from simulations that the iterates generated by all SGO-based localization algorithms always converge. To verify this and also for simplicity, iterates-related termination criterion is used instead of function value-related criterion in simulations.

## A. Convergence Performance

In this set of simulations, we examine the convergence performance of the ESOCP and NCSG. The convergence comparison between the ESOCP and the SOCP will be presented in the next subsection. The ESOCP is randomly initialized while the NCSG is initialized from the intermediate result of the ESOCP. For simplicity, the subproblems in all SGO-based localization algorithms are run in the order of nodes' indices. Particularly, to clearly demonstrate the convergence performance, we here set $\delta = 0$, and $\text{nf} = 0$ in the range-based localization and $\tau = 0$ in the range-free localization, so that the minimum cost function values of (18), (24), and (32) are all 0.

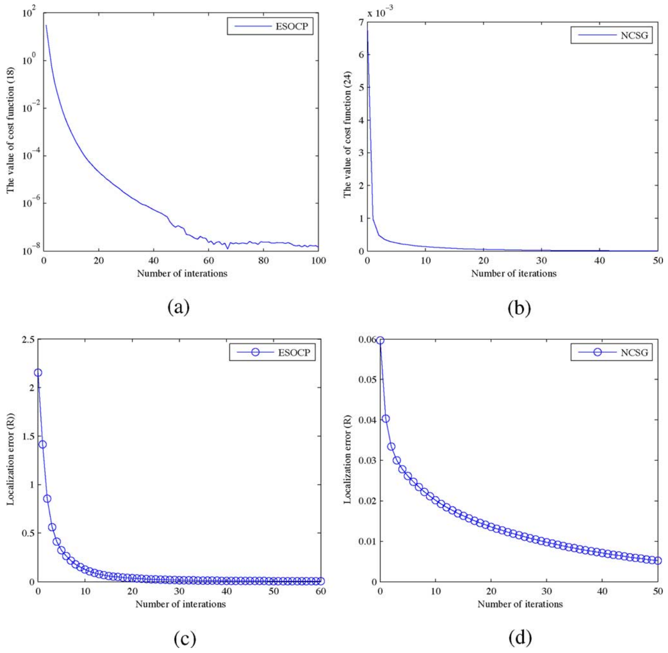
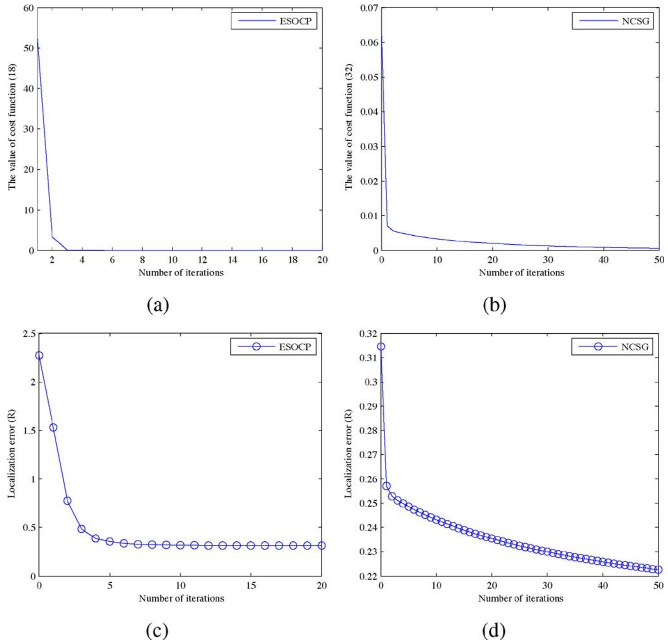


Fig. 1. The convergence performance of the range-based distributed localization algorithms. (a) Cost function value. (b) Cost function value. (c) Localization error. (d) Localization error.

The convergence results of two algorithms are presented in Fig. 1 for range-based localization and Fig. 2 for range-free localization. In the figures, the localization errors are calculated according to $(1)/(N)\sum_{i=1}^{N} \|\boldsymbol{x}_i^* - \boldsymbol{x}_i\|$ where $\boldsymbol{x}^*$ denotes the true location. From Fig. 1(a), (b), Fig. 2(a), and (b), it can be observed that the ESOCP and the NCSG converge with the property of nonincreasing of cost function at each iteration[8]. Moreover, we can find that these algorithms converge much faster at the beginning of iterations than at the subsequent iterations. Further, we can find from Fig. 1(c) and Fig. 2(c) that the ESOCP algorithm can converge well within 20 iterations in the range-based localization and within 10 iterations in the range-free localization, which also can be seen from Fig. 3(a), (b) and Fig. 4(a), (b), where the known anchors' positions are denoted by (blue) diamonds, the true nodes' positions are denoted by (green) circles and their estimates by (red) asterisks, and the lines indicate the estimation errors. The reason why the ESOCP converges faster in the range-free localization case than in the range-based case is that the range-free case has a larger solution set.

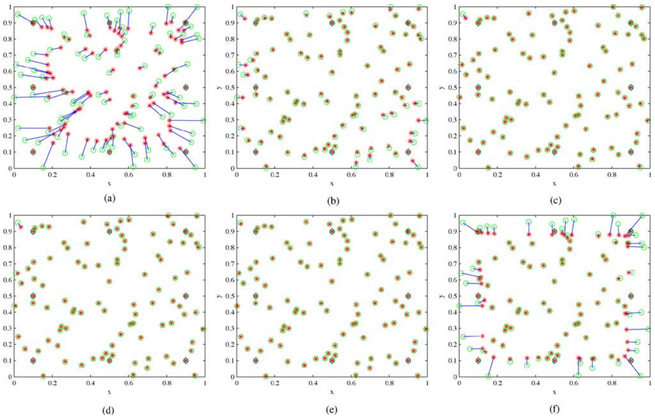
[7] The term $\sum_{i=1}^{m} \|\boldsymbol{x}_i - \boldsymbol{a}_i\|^2$ is added in the least-squares sense, which leads to that only a single localization phase is required to solve the problem, otherwise three localization phases like that in [19] are needed.

[8] In the ideal range-based localization case, $c_j = Y_{jj} - \|\boldsymbol{x}_j\|^2$ must be equal to zero if the network is uniquely localized. However, due to the numerical error, $c_j$s may become a very small negative scalar as the iterations proceed. In our simulations, we set $c_j = \max(Y_{jj} - \|\boldsymbol{x}_j\|^2, 0)$, which leads to the small variation of the cost function value of the ESOCP after 40 iterations in Fig. 1(a).



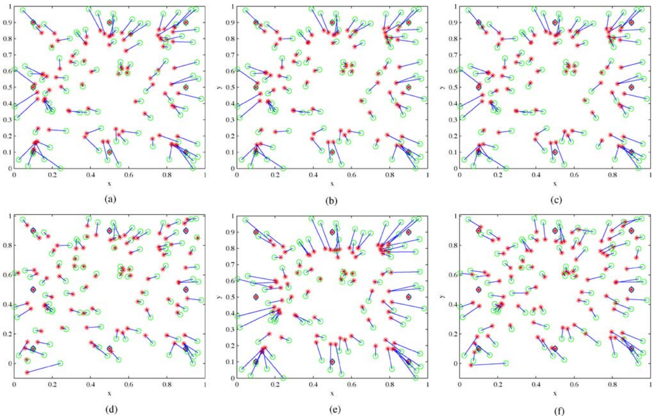
(a) (b) (c) (d)

Fig. 2. The convergence performance of the range-free distributed localization algorithms.



(a) (b) (c) (d) (e) (f)

Fig. 3. Range-based localization results. (a) ESOCP, after 5 iterations, $\mathrm{error} = 32.43\%R$. (b) ESOCP, after 20 iterations, $\mathrm{error} = 3.55\%R$. (c) ESOCP, after 50 iterations, $\mathrm{error} = 0.61\%R$. (d) ESDP, $\mathrm{error} = 0.73\%R$. (e) fullSDP, $\mathrm{error} = 0.15\%R$. (f) SOCP, $\mathrm{error} = 11.84\%R$.



(a) (b) (c) (d) (e) (f)

Fig. 4. Range-free localization results. (a) ESOCP, after 5 iterations, $\mathrm{error} = 35.44\%R$. (b) ESOCP, after 10 iterations, $\mathrm{error} = 31.78\%R$. (c) ESDP, $\mathrm{error} = 32.02\%R$. (d) $\mathrm{ESOCP{+}NCSG}$, $\mathrm{error} = 22.26\%R$. (e) SOCP, $\mathrm{error} = 42.62\%R$. (f) SOCP+NCSG, $\mathrm{error} = 31.69\%R$.

### *B. Localization Performance*

In this subsection, we compare the localization performance of the ESOCP and the NCSG with the other localization algorithms (including ESDP, fullSDP, and SOCP) in terms of localization errors. Note that, the fullSDP often fails in the range-free localization case for networks of more than 100 nodes since much more constraints involved in the fullSDP lead to an out-of-memory issue. Hence, its localization performance is not evaluated in the range-free localization case. The localization results of various localization algorithms in ideal environment are presented in Figs. 3 and 4. In Fig. 4, ESOCP+NCSG (or SOCP+NCSG) denotes that the NCSG is initialized from the result of ESCOP (or SOCP). We can observe from two figures that, the ESOCP can yield almost the same localization performance as the ESDP, and better localization performance than the SOCP. We also can find as shown in [17] that the ESDP relaxation-based localization algorithms can work almost as well as the fullSDP in the range-based localization case although the ESDP relaxation is weaker than the fullSDP relaxation. In addition, it can be observed from Fig. 4 that the refinement algorithm can significantly improve the localization performance of the ESOCP or the SOCP, and that the ESOCP can provide better initialization than the SOCP.

Below we study the effect of the distance measurement noises, the anchor position errors, and the radio irregularity on the average localization performance. In simulations, each result is averaged over 30 randomly generated and connected networks. All distributed algorithms terminate if either $\max_i \|\boldsymbol{x}_i^k - \boldsymbol{x}_i^{k-1}\| \leq 0.01R$ or the total number of iterations $K > K_{\max}$. We set $K_{\max} = 30$ for ESOCP and SOCP while $K_{\max} = 10$ for NCSG. Particularly, we demonstrate the localization performance of the NIS algorithm [20] as a benchmark. Considering anchor position errors, we use the NIS algorithm to solve the following minimization problem which involves all constraints

$$\begin{aligned}\min_{X} & \sum_{(i,j)\in\mathcal{N}^1} f_{ij}(\boldsymbol{x}_i,\boldsymbol{x}_j) + \sum_{(i,j)\notin\mathcal{N}^1} (R - \|\boldsymbol{x}_i - \boldsymbol{x}_j\|)_+^2 \\ & + \sum_{i=1}^{m} \|\boldsymbol{x}_i - \boldsymbol{a}_i\|^2 \\ \text{s.t. } & \|\boldsymbol{x}_i - \boldsymbol{a}_i\| \leq \delta, i = 1, 2, \ldots, m \end{aligned} \tag{42}$$

where $f_{ij}(\boldsymbol{x}_i,\boldsymbol{x}_j) = (\|\boldsymbol{x}_i - \boldsymbol{x}_j\| - d_{ij})^2$ for range-based localization and $f_{ij}(\boldsymbol{x}_i,\boldsymbol{x}_j) = (\|\boldsymbol{x}_i - \boldsymbol{x}_j\| - R)_+^2$ for range-free localization.

Simulation results are presented in Figs. 5, 6, 7, and 8. From these four figures, we can see that the ESOCP always outperforms the SOCP and the refinement algorithms can substantially improve the performance of both the ESOCP and the SOCP. Moreover, the ESOCP with refinement is the best among the distributed localization algorithms with the closest performance to the NIS algorithm which considers all constraints. Additionally, it is observed that the performance gap between the ESOCP

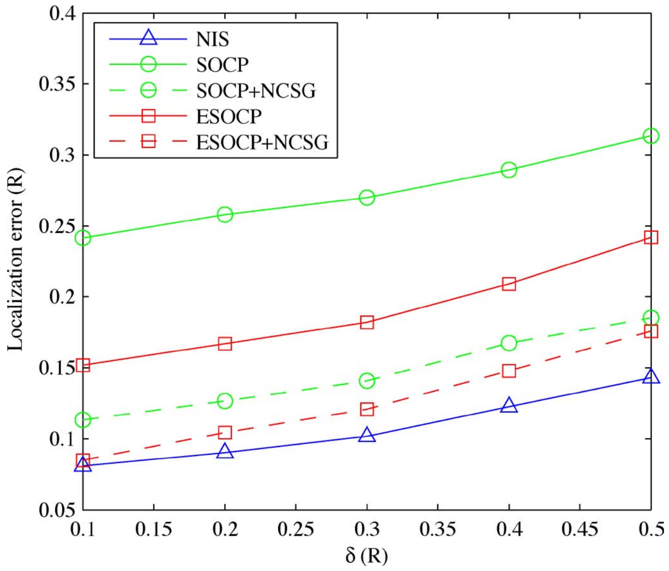


Fig. 5. Range-based localization error versus anchor position error when $\text{nf} = 0.1$.

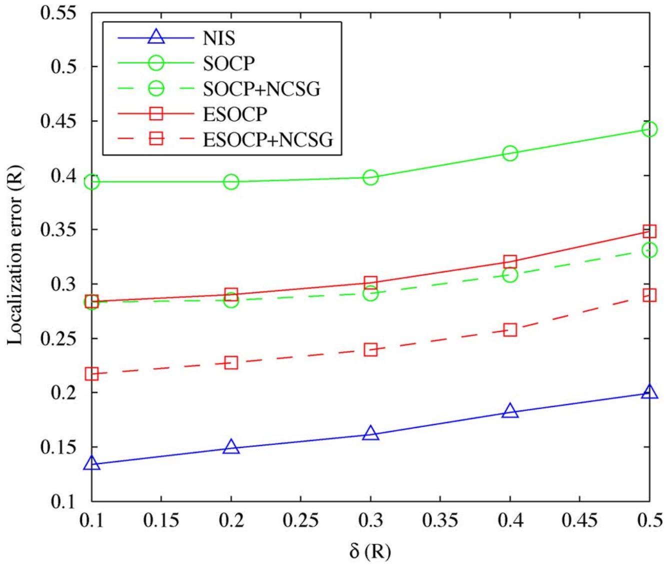


Fig. 7. Range-free localization error versus anchor position error when $\tau = 0.2$.

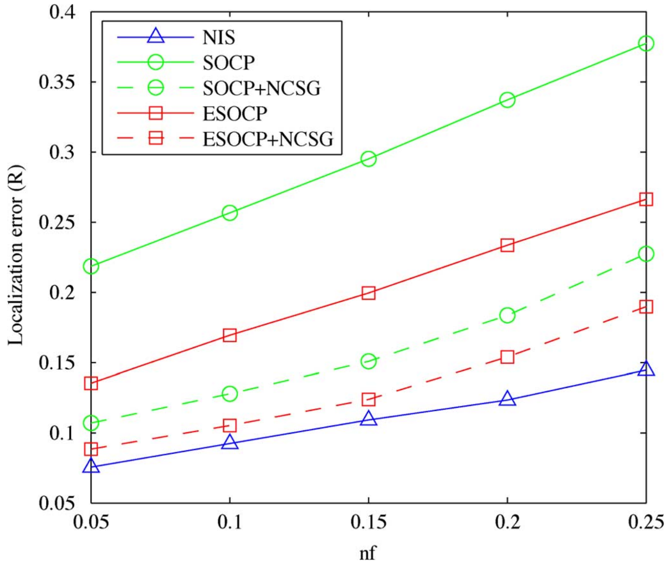


Fig. 6. Range-based localization error versus noise factor when $\delta = 0.2R$.

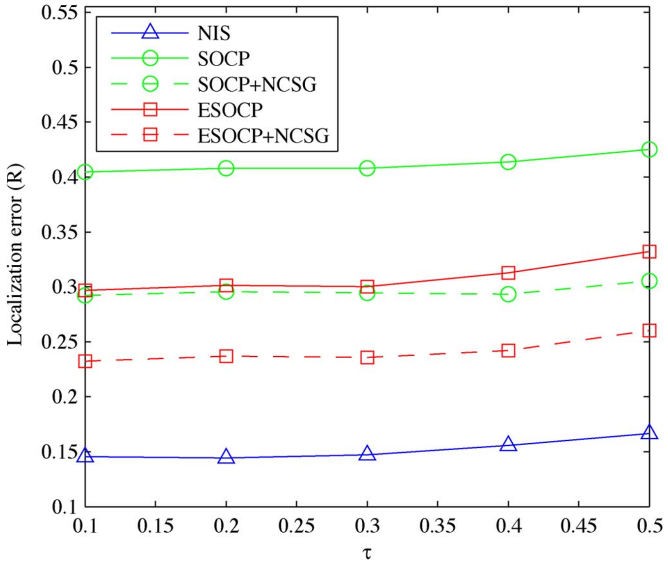


Fig. 8. Range-free localization error versus radio irregularity when $\delta = 0.2R$.

with refinement and the NIS in the range-based localization is smaller than that in the range-free localization.

From Figs. 5 and 6, we observe that the localization errors of the ESOCP and the SOCP increase more sharply with the distance measurement noise factor increasing than with the anchor position errors, implying that the effect of the distance measurement noise on the localization performance of the ESOCP and the SOCP is more dramatic than the effect of the anchor position errors in the range-based localization. Furthermore, it is observed that the performance of the ESOCP with refinement is very close to that of the NIS algorithm in the range-based localization when the distance measurement noises and the anchor position errors are small.

From Figs. 7 and 8, we can see that, the effect of the anchor position errors on the localization performance of the ESOCP and the SOCP is slightly more dramatic than the effect of the radio irregularity in the range-free localization. Particulary, it is shown in Fig. 8 that the range-free localization performance of various algorithms are almost invariable when $\tau$ is small ($\leq 0.3$). Furthermore, we can see from Figs. 7 and 8 that, the localization performance of the ESOCP is close to that of the SOCP with refinement in the range-free localization, which is not the case in the range-based localization.

Table I shows the convergence rate of various localization algorithms (i.e., the total number of iterations), where all results are averaged over 30 networks. The first three lines of the table present the results of the range-based localization, while the last three lines present the results of the range-free localization. We can see from the table that the ESOCP needs about one (respectively, four) more iterations than the SOCP to converge in the range-based (respectively, range-free) localization. Furthermore, it can be observed that, less iterations are needed for the

TABLE I
THE CONVERGENCE RATE OF VARIOUS LOCALIZATION ALGORITHMS

| | SOCP | SOCP+NCSG | ESOCP | ESOCP+NCSG |
|---|---|---|---|---|
| $\delta = 0.2R,\ nf = 0.1$ | 20.20 | 29.27 | 21.77 | 29.00 |
| $\delta = 0.4R,\ nf = 0.1$ | 22.73 | 32.23 | 22.47 | 30.10 |
| $\delta = 0.2R,\ nf = 0.2$ | 18.07 | 27.93 | 19.27 | 28.00 |
| $\delta = 0.2R,\ \tau = 0.2$ | 11.5 | 15.27 | 15.4 | 18.53 |
| $\delta = 0.4R,\ \tau = 0.2$ | 14.23 | 17.90 | 18.43 | 21.43 |
| $\delta = 0.2R,\ \tau = 0.4$ | 12.34 | 15.10 | 15.60 | 18.76 |

refinement over the ESOCP than over the SOCP whether in the range-based localization or in the range-free localization.

## V. CONCLUSION

We have presented a unified optimization framework for sensor network localization problems with uncertainties in anchor positions, distance measurements and radio range of nodes, and derived two SDP relaxation formulations, the fullSDP formulation and the ESDP formulation. In addition, we have proposed the sequential greedy optimization algorithm which is very suitable for distributed optimization in networks. Based on the SGO algorithm, we obtain distributed ESDP localization algorithms (i.e., ESOCP) as well as distributed refinement algorithms (i.e., NCSG) for both range-based localization and range-free localization. Simulation results show that the ESOCP has good convergence performance, and it can work almost as well as the centralized fullSDP and ESDP and much better than the SOCP algorithm. The distributed refinement algorithms can provide a substantial improvement in the localization accuracy for both the SOCP and the ESOCP.

In simulations, we assume anchors are positioned at the boundary of networks to achieve highly accurate localization. In practice, however, as other relaxation-based localization methods, the ESOCP also behaves poorly when most sensor nodes in a network are outside the convex hull of the anchor nodes [14]. On the other hand, it is known that the values of $c_i$ can characterize well the position estimation error of nodes, i.e., nodes with smaller $c_i$ are more accurately localized [13]. In our future work, we will exploit this fact to improve the localization performance when anchors are randomly placed in the networks.

## ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers and Prof. Z.-Q. Luo with his smart research group of University of Minnesota for their valuable comments.

## REFERENCES

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, “Wireless sensor networks: A survey,” *Comput. Netw.*, vol. 38, no. 4, pp. 393–422, 2002.

[2] G. Mao, B. Fidan, and B. D. O. Anderson, “Wireless sensor network localization techniques,” *Comput. Netw.*, vol. 51, pp. 2529–2553, 2007.

[3] N. Patwari, A. O. Hero III, M. Perkins, N. S. Correal, and R. J. ODea, “Relative location estimation in wireless sensor networks,” *IEEE Trans. Signal Process.*, vol. 51, no. 8, pp. 2137–2148, Aug. 2003.

[4] N. Bulusu, J. Heidemann, and D. Estrin, “GPS-less low-cost ourdoor localization for very small devices,” *IEEE Pers. Commun.*, vol. 7, no. 5, pp. 28–34, 2000.

[5] D. Niculescu and B. Nath, “Ad hoc positioning system (APS),” in *Proc. IEEE GLOBECOM*, San Antonio, TX, 2001, vol. 7, pp. 2926–2931.

[6] T. He, C. Huang, B. Blum, J. Stankovic, and T. Abdelzaher, “Range-free localization schemes for large scale sensor networks,” in *Proc. ACM Mobicom*, San Diego, CA, 2003, pp. 81–95.

[7] Y. Shang, W. Ruml, Y. Zhang, and M. Fromherz, “Localization from connectivity in sensor networks,” *IEEE Trans. Parallel Distrib. Syst.*, vol. 15, no. 11, pp. 961–974, Nov. 2004.

[8] J. A. Costa, N. Patwari, and A. O. Hero III, “Distributed multidimensional scaling with adaptive weighting for node localization in sensor networks,” *ACM Trans. Sens. Netw.*, vol. 2, no. 1, pp. 39–64, Feb. 2006.

[9] J. Liu, Y. Zhang, and F. Zhao, “Robust distributed node localization with error management,” in *Proc. ACM MobiHoc*, Florence, Italy, May 2006, pp. 250–261.

[10] L. Doherty, L. E. Ghaoui, and S. J. Pister, “Convex position estimation in wireless sensor networks,” in *Proc. IEEE INFOCOM*, Anchorage, AK, Apr. 2002, vol. 3, pp. 1655–1663.

[11] P. Tseng, “Second-order cone programming relaxation of sensor network localization,” *SIAM J. Optim.*, vol. 8, no. 1, pp. 156–185, Feb. 2007.

[12] P. Biswas and Y. Ye, “Semidefinite programming for ad hoc wireless sensor network localization,” in *Proc. ACM IPSN*, Berkeley, CA, 2004, pp. 46–54.

[13] P. Biswas, T. C. Lian, T. C. Wang, and Y. Ye, “Semidefinite programming based algorithms for sensor network localization,” *ACM Trans. Sens. Netw.*, vol. 2, no. 2, pp. 188–220, 2006.

[14] P. Biswas, T. C. Liang, K. C. Toh, Y. Ye, and T. C. Wang, “Semidefinite programming approaches for sensor network localization with noisy distance measurements,” *IEEE Trans. Autom. Sci. Eng.*, vol. 3, no. 4, pp. 360–371, Oct. 2006.

[15] K. W. K. Lui, W.-K. Ma, H. C. So, and F. K. W. Chan, “Semi-definite programming algorithms for sensor network node localization with uncertainties in anchor positions and/or propagation speed,” *IEEE Trans. Signal Process.*, vol. 57, no. 2, pp. 752–763, Feb. 2009.

[16] P. Biswas and Y. Ye, “A distributed method for solving semidefinite programs arising from ad hoc wireless sensor network localization,” in *Multiscale Optimization Methods and Applications*, W. W. Hager, S.-J. Huang, P. M. Pardalos, and O. A. Prokopyev, Eds. New York: Springer, 2006, pp. 69–84.

[17] Z. Wang, S. Zheng, Y. Ye, and S. Boyd, “Further relaxations of the semidefinite programming approach to sensor network localization,” *SIAM J. Optim.*, vol. 19, no. 2, pp. 655–673, Jul. 2008.

[18] Q. Shi and C. He, “A SDP approach for range-free localization in wireless sensor networks,” in *Proc. IEEE ICC*, Beijing, China, May 2008, pp. 4214–4218.

[19] S. Srirangarjan, A. H. Tewfik, and Z.-Q. Luo, “Distributed sensor network localization using SOCP relaxation,” *IEEE Trans. Wireless Commun.*, vol. 7, no. 12, pp. 4886–4895, Dec. 2008.

[20] Q. Shi, C. He, and L. Jiang, “Normalized incremental subgradient algorithm and its application,” *IEEE Trans. Signal Process.*, vol. 57, no. 10, Oct. 2009.

[21] Q. Shi, C. He, L. Jiang, and J. Luo, “Sensor network localization via nondifferentiable optimization,” in *Proc. IEEE GLOBECOM*, New Orleans, LA, Dec. 2008, pp. 1–5.

[22] I. Csiszár and G. Tusnády, E. J. Dudewicz, Ed., “Information geometry and alternating minimization procedures,” *Statistic and Decisions, Suppl. Issue No. 1*, pp. 205–237, 1984.

[23] W. S. Wang, J. C. Juang, and Y. Y. Sun, “On the application of alternating projection under the constraint of linear matrix inequalities,” *IMA J. Math. Control Inf.*, vol. 16, no. 1, pp. 15–22, 1999.

[24] J. Platt, “Fast training of support vector machines using sequential minimial optimization,” in *Advances in Kernel Methods-Support Vector Learning*, B. Schölkopf, C. Burges, and A. Smola, Eds. Cambridge, MA: MIT Press, 1999, ch. 12, pp. 185–208.

[25] A. Tkacenko and P. P. Vaidyannathan, “Iterative greedy algorithm for solving the FIR paraunitary approximation problem,” *IEEE Trans. Signal Process.*, vol. 54, no. 1, pp. 144–160, Jan. 2006.

[26] J. E. Dennis and R. B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Englewood Cliffs, NJ: Prentice-Hall, 1983.

[27] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[28] T. Abatzoglou and B. O’ Donnell, “Minimization by coordiante descent,” *J. Optim. Theory Appl.*, vol. 36, pp. 163–174, 1982.

[29] Z.-Q. Luo and P. Tseng, “On the convergence of coordinate descent method for convex differentiable minimization,” *J. Optim. Theory Appl.*, vol. 72, no. 1, pp. 7–35, Jan. 1992.

[30] D. P. Berteskas and J. N. Tsitsiklis, *Parrell and Distributed Computation: Numerical Methods*. Englewood Cliffs, NJ: Prentice-Hall, 1989.

[31] M. Kuhn, The Karush-Kuhn-Tucker Theorem Nov. 2006 [Online]. Available: http://en.wikipedia.org/wiki/Karush-Kuhn-Tucker\conditions. Also: http://webrum.uni-mannheim.de/vwl/mokuhn/public/KarushKuhn-Tucker.pdf, [online]. Available:

[32] J. F. Sturm, "Using SeDuMi 1.02, a Matlab toolbox for optimization over symmetric cones," *Optim. Methods Softw.*, vol. 11–12, pp. 625–653, 1999.

[33] K. C. Toh, M. J. Todd, and R. H. Tutuncu, "SDPT3-A MATLAB software package for semidefinite programming," *Optim. Methods Softw.*, vol. 11, pp. 545–581, 1999.

**Qingjiang Shi** (S'09) received the B.S. degree in electronic engineering in 2003, and the M.S. degree in control engineering in 2006, both from the China University of Petroleum, Shangdong, China.

He is currently pursuing the Ph.D. degree in electronic engineering at Shanghai Jiao Tong University, Shanghai, China. Since September 2009, he has been a visiting researcher with the University of Minnesota, Twin Cities. His interests include distributed signal processing in sensor network, and optimization for wireless communication.

Mr. Shi received the Best Paper Award from IEEE PIMRC'09 conference.

**Chen He** (S'93-M'96) received the B.E. and M.E. degrees in electronic engineering from Southeast University of China in 1982 and 1985, respectively, and the Ph.D. degree in electronics system from Tokushima University of Japan in 1994.

He was with the Department of Electronic Engineering, Southeast University of China, from 1985 to 1990. He joined the Department of Electronic Engineering, Shanghai Jiao Tong University of China, Shanghai, China, in 1996. He visited the Tokushima University of Japan as a foreign researcher from October 1990 to September 1991, and visited the Communication Research Laboratory of Japan from December 1999 to December 2000 as a research fellow. He has published more than 200 journal papers and over 80 conference papers. Currently, he is a Professor and Vice Director of Advanced Communication Institute, Shanghai Jiao Tong University. His current research interests are 4G wireless communication systems, wireless sensor network, and signal processing.

Dr. He received the best paper award in Globecom 2007.

**Hongyang Chen** was born in China. He received the B.S. degree from Southwest Jiaotong University, Chengdu, China, in 2003 and the M.S. degree from the Institute of Mobile Communications, Southwest Jiaotong University, in 2006.

Currently, he is a Ph.D. student with the Graduate School of Information Science and Technology, University of Tokyo, Tokyo, Japan. In 2009, he was a Visiting Researcher with the UCLA Adaptive Systems Laboratory, University of California, Los Angeles, under the supervision of Prof. A. H. Sayed. His research interests include wireless localization, wireless sensor networks, statistical signal processing. He has published some referred journal and conference papers in the ACM *Transactions on Sensor Networks*, IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, IEICE *Transactions on Fundamentals*, IEEE MILCOM, IEEE GlOBECOM, IEEE ICC, and more.

Mr. Chen has served as a TPC member for some flagship conferences, including IEEE PIMRC, IEEE/IFIP EUC, IEEE VTC, IEEE CCNC, IEEE WCNC, IEEE GLOBECOM, and IEEE ICC. He organized a Session on WSNs at IEEE MILCOM 2008. He received the Best Paper Award from IEEE PIMRC'09 conference. He has been listed in Marquis *Who's Who in the World*.

**Lingge Jiang** (M'03) received the B.E. degree in radio engineering from the Southeast University of China in 1982, and the M.E. and Ph.D. degrees in electrical engineering from Tokushima University of Japan, in 1993 and 1996, respectively.

She joined the Department of Electronic Engineering, Shanghai Jiao Tong University, Shanghai, China, in 1996. Currently, she is a professor with Shanghai Jiao Tong University. Her current research interests include wireless communication systems, wireless sensor networks, and intelligent information processing.