Distributed Sensor Location through Linear Programming with Triangle Inequality Constraints

Camillo Gentile, *Member, IEEE*

***Abstract*— The falling price and reduced size of sensors for monitoring spatially-sensitive environmental properties such as temperature, light, sound, and vibration have motivated research in location algorithms in recent years. To our knowledge, the algorithm which achieves the best performance refines erroneous measurements through an optimization program whose quadratic constraints force the sensors to be consistent with the geometry of the physical world. Since the program is non-convex, the authors relax the constraints to render it convex for which efficient solution methods exist. We propose solving a similar optimization program however by applying convex geometrical constraints directly, necessitating no relaxation of the constraints and in turn ensuring a solution still compliant with the physical world. We show through extensive experimentation that ours outperforms the competing algorithm across all network parameters. In addition, this paper formulates a distributed version of our algorithm which achieves the same globally optimal objective function as the centralized version, and reports the messaging overhead for its convergence.**

***Index Terms*— Simplex method, primal-dual method, quadratic programming, semidefinite programming.**

I. Introduction

The falling price and reduced size of sensors in recent years have fueled the installation of dense networks to gauge and relay environmental properties such as temperature, light, sound, and vibration in applications ranging from video surveillance and traffic control to health monitoring and industrial automation [1]. Furnishing the sensor locations proves as important as the spatially-sensitive readings in order for an external system to calibrate a network response. In particular military and public safety operations call for ad hoc deployability and self-organization to deliver high-precision location such as that of a man down in a building ablaze with zero visibility. This has launched a research area known as sensor location which seeks to aggregate potentially enormous quantities of data to achieve optimal results. Most practical systems require local distributed processing to cope with dynamic links or nodes in motion to maintain a network updated; alternatively relaying information across a large network sanctions the centralized processing of obsolete data, limiting scalability.

A recent paper conducts an exhaustive survey of the available techniques for sensor location [2]. It isolates two algorithms which achieve the best performance by refining erroneous distance measurements between neighboring nodes in a network through an optimization program whose quadratic constraints force the sensors to be consistent with the geometry of the physical world. Since the program is non-convex, the papers differ in their relaxation approaches to render it convex for which efficient solution methods exist, however relaxation alters the geometry of the problem. The approach suggested by Biswas et al. has greater applicability and yields better results than the one by Doherty et al. [2], [3]. We follow their same approach, maintaining the efficiency of convex optimization, however by applying *linear* geometrical constraints as opposed to quadratic ones. This renders the problem automatically convex, necessitating no relaxation of the original geometrical constraints and so guaranteeing a tighter solution still compliant with the physical world. Specifically, the contributions of this paper are:

- a convex linear program which ensures that the estimated link distances between neighboring nodes conform to requisite geometrical constraints;
- a distributed algorithm to solve the program over the network without compromising optimality;
- a distributed algorithm to reconstruct the locations of the sensor nodes from the estimated link distances.

The paper reads as follows: Section II defines the original problem taken from Biswas [2] to determine sensor location, and it states our linear program which proposes a solution to it. Our distributed location algorithm is divided into two stages: 1) Section III describes a centralized method to solve the program, and Section IV reduces this method to a distributed algorithm over the network; 2) Section V shows how to reconstruct the sensor locations from the solution to the program, also in a distributed manner. An extensive number of challenging tests conditions are reported in Section VI to substantiate the robustness of our algorithm to high levels of noise in comparison to the algorithm proposed by Biswas; we also report the messaging overhead of the algorithm. The last section summarizes our results.

II. Preliminaries

Consider a network with two types of nodes: n_A anchor nodes (or anchors) with known location and n_S sensor nodes (or sensors) with unknown location, for a total of $n = n_A + n_S$ nodes. For simplicity, let the nodes lie on a plane such that node i has location $\mathbf{x}_i \in \mathcal{R}^2$ indexed through i, $i = 1 \ldots n_A$ for the anchors and $i = n_A + 1 \ldots n$ for the sensors. The set N contains all pairs of nodes between which a link exists: (i, j), $i < j$; $||\mathbf{x}_i - \mathbf{x}_j||_2 < R$, where the network parameter R is known as the *radio range*. The complement

Manuscript received December 2, 2005; revised July 26, 2006 and October 12, 2006; accepted October 13, 2006. The associate editor coordinating the review of this paper and approving it for publication was K.-K. Wong.

C. Gentile is with the Wireless Communication Technologies Group at the National Institute of Standards and Technology, Gaithersburg, MD 20899 (e-mail: camillo.gentile@nist.gov).

Digital Object Identifier 10.1109/TWC.2007.05907.

set $\bar{N}$ contains all pairs of nodes between which no links exists: $(i, j),\ i < j;\ ||\mathbf{x}_i - \mathbf{x}_j||_2 \geq R$. The set M contains all triplets of nodes which form a triangle in the network: $(i, j, k),\ (i, j) \in N;\ (j, k) \in N;\ (i, k) \in N$.

Neighboring nodes i and j measure the link distance $\hat{d}_{ij}$ between them through received-signal-strength or time-of-arrival techniques [4]. Given the locations of the anchors $\mathbf{x}_i,\ i = 1...n_A$ and the *measured distances* $\hat{d}_{ij},\ \forall(i, j) \in N$ between all neighbors in the network, the original problem considered by both Biswas and Doherty to solve for the locations of the sensors $\mathbf{x}_i,\ i = n_A + 1...n$ follows:

$$\begin{aligned} \min \quad & \sum_{(i,j)\in N} |\alpha_{ij}| \\ \text{s.t.} \quad & ||\mathbf{x}_i - \mathbf{x}_j||_2 = d_{ij}, \quad \forall(i,j) \in N \\ & ||\mathbf{x}_i - \mathbf{x}_j||_2 \geq R, \quad \forall(i,j) \in \bar{N} \end{aligned} \tag{1}$$

where $d_{ij} = \hat{d}_{ij} + \alpha_{ij}$. The problem minimizes the sum of the absolute *residuals* α_{ij} between the measured distances $\hat{d}_{ij}$ and the *estimated distances* d_{ij}, provided that the latter conform to requisite geometrical constraints. The problem as defined above cannot be solved through convex optimization techniques since many of the constraints are non-convex. To overcome this obstacle, Doherty relaxes the problem by removing all the non-convex constraints, reducing it to a convex second-order cone optimization problem. This however limits the estimated locations of the sensors to lie within the convex hull formed by the anchors. Biswas avoids this limitation not by removing the non-convex constraints, but rather by relaxing the problem to a *semidefinite program*:
When the measured distances contain no error, the semidefinite program yields the identical solution to (1) with zero objective function [2]: imagine a rigid structure consisting of a set of points in $\mathcal{R}^2$ with exact distances between each other. When the measured distances contain errors, however, the distance constraints usually contradict each other and no solution exists in the plane; in this case, the semidefinite program relaxes the constraints by "lifting" the solution to a higher-dimensional space, reconfiguring the structure by setting some of the points outside of the plane. The optimal semidefinite objective function approaches zero as the dimension is allowed to increase, and so always results in a smaller value than that of the original problem. The higher-dimensional points of the semidefinite solution are then projected down into $\mathcal{R}^2$ as a suboptimal solution to (1). The projection, however, typically leads to the crowding of sensors towards the center of the network due to the large contribution to the estimated distances between neighboring points coming from the dimensions ignored [5]. In order to mitigate this limitation the authors propose: 1. adding terms to the semidefinite objective function containing a heuristic parameter whose value depends on the size and geometry of the network, which if not chosen carefully leads to an infeasible solution spreading the points too far apart; 2. post-processing the suboptimal solution through a local gradient descent method. While the combined approach improves the suboptimal solution, it reduces to a gradient descent search to the non-convex original problem with the semidefinite approach providing only an initial solution. The approach described in the sequel provides a tight solution for the sensor locations, necessitating no initial solution nor heuristic parameters.

A. Triangle inequality constraints

Instead of relaxing the constraints in (1), we propose applying a different set of geometrical constraints while maintaining the same objective function. We exploit the triangular structure of the network such that the estimated link distances conform to the triangle inequality constraints. The problem we solve can be stated as follows:

$$\begin{aligned} \min \quad & \sum_{(i,j)\in N} |\alpha_{ij}| \\ \text{s.t.} \quad & \left.\begin{aligned} d_{ij} + d_{jk} \geq d_{ik} \\ d_{ij} + d_{ik} \geq d_{jk} \\ d_{jk} + d_{ik} \geq d_{ij} \end{aligned}\right\},\ \forall(i,j,k) \in M \end{aligned} \tag{2}$$

where $d_{ij} = \hat{d}_{ij} + \alpha_{ij}$. Consider rewriting the problem as a linear program in canonical form by removing the absolute signs and introducing two bounding constraints:

$$\begin{aligned} \min \quad & \sum_{(i,j)\in N} \alpha^+_{ij} + \alpha^-_{ij} \\ \text{s.t.} \quad & \left.\begin{aligned} d_{ij} + d_{jk} \geq d_{ik} \\ d_{ij} + d_{ik} \geq d_{jk} \\ d_{jk} + d_{ik} \geq d_{ij} \end{aligned}\right\},\ \forall(i,j,k) \in M \\ & \begin{aligned} \alpha^+_{ij} \geq 0 \\ \alpha^-_{ij} \geq 0 \end{aligned} \quad ,\ \forall(i,j) \in N \end{aligned} \tag{3}$$

where $\alpha_{ij} = \alpha^+_{ij} - \alpha^-_{ij}$. Let $\tilde{\alpha}_{ij}$ be in the optimal solution to (2) and let $\{\tilde{\alpha}^+_{ij}, \tilde{\alpha}^-_{ij}\}$ be in any feasible solution to (3) such that $\{\tilde{\alpha}^+_{ij}, \tilde{\alpha}^-_{ij}\} \geq 0$ and $\tilde{\alpha}_{ij} = \tilde{\alpha}^+_{ij} - \tilde{\alpha}^-_{ij}$. Further let $\hat{\alpha}^+_{ij} = \tilde{\alpha}^+_{ij} - \min\{\tilde{\alpha}^+_{ij}, \tilde{\alpha}^-_{ij}\}$ and $\hat{\alpha}^-_{ij} = \tilde{\alpha}^-_{ij} - \min\{\tilde{\alpha}^+_{ij}, \tilde{\alpha}^-_{ij}\}$; now $\{\hat{\alpha}^+_{ij}, \hat{\alpha}^-_{ij}\}$ are not only in a feasible solution to (3) since $\{\hat{\alpha}^+_{ij}, \hat{\alpha}^-_{ij}\} \geq 0$ and $\tilde{\alpha}_{ij} = \hat{\alpha}^+_{ij} - \hat{\alpha}^-_{ij}$, but are also optimal since $\hat{\alpha}^+_{ij} + \hat{\alpha}^-_{ij} \leq \tilde{\alpha}^+_{ij} + \tilde{\alpha}^-_{ij}$ in its objective function. Moreover since $\hat{\alpha}^+_{ij} \cdot \hat{\alpha}^-_{ij} = 0$, then $\hat{\alpha}^+_{ij} + \hat{\alpha}^-_{ij} = |\hat{\alpha}^+_{ij} - \hat{\alpha}^-_{ij}| = |\tilde{\alpha}_{ij}|$, proving the equivalence of (2) and (3) at optimality. The solution to the problem above does not directly yield the sensor locations as in (1), but only the estimated link distances. Hence the complete algorithm requires an a posteriori *location reconstruction* stage described in Section V to furnish the locations of the sensors from these distances. Note that (3) can be applied to the triangles formed in three-dimensional networks as well; this paper does not treat the reconstruction stage for such networks for the sake of brevity. The advantage of our approach lies in the linearity of the constraints which ensures the convexity of the problem without relaxing any of the original geometrical constraints.

III. The Primal-Dual Method

A. The primal problem

We denote the linear program (3) as the *primal* problem. Rewriting the primal in standard form appears as

$$
\begin{aligned}
\min \;& \sum_{(i,j)\in N} \alpha_{ij}^{+} + \alpha_{ij}^{-} \\
\text{s.t. }& \left.\begin{aligned}
a_{ij,k}^{1}(\alpha_{ij}^{+}-\alpha_{ij}^{-})+a_{jk,i}^{1}(\alpha_{jk}^{+}-\alpha_{jk}^{-})+a_{ik,j}^{1}(\alpha_{ik}^{+}-\alpha_{ik}^{-})-\alpha_{ijk}^{1}=b_{ijk}^{1}\\
a_{ij,k}^{2}(\alpha_{ij}^{+}-\alpha_{ij}^{-})+a_{jk,i}^{2}(\alpha_{jk}^{+}-\alpha_{jk}^{-})+a_{ik,j}^{2}(\alpha_{ik}^{+}-\alpha_{ik}^{-})-\alpha_{ijk}^{2}=b_{ijk}^{2}\\
a_{ij,k}^{3}(\alpha_{ij}^{+}-\alpha_{ij}^{-})+a_{jk,i}^{3}(\alpha_{jk}^{+}-\alpha_{jk}^{-})+a_{ik,j}^{3}(\alpha_{ik}^{+}-\alpha_{ik}^{-})-\alpha_{ijk}^{3}=b_{ijk}^{3}
\end{aligned}\right\}, \\
&\hfill \forall (i,j,k)\in M \\
& \left.\begin{aligned}\alpha_{ij}^{+}\ge 0\\ \alpha_{ij}^{-}\ge 0\end{aligned}\right\},\ \forall (i,j)\in N
\end{aligned}
\tag{4}
$$

where

$$
\begin{bmatrix} a_{ij,k}^{1} & a_{jk,i}^{1} & a_{ik,j}^{1}\\ a_{ij,k}^{2} & a_{jk,i}^{2} & a_{ik,j}^{2}\\ a_{ij,k}^{3} & a_{jk,i}^{3} & a_{ik,j}^{3}\end{bmatrix}=\begin{bmatrix} 1 & 1 & -1\\ 1 & -1 & 1\\ -1 & 1 & 1\end{bmatrix},\quad
\begin{bmatrix} b_{ijk}^{1}\\ b_{ijk}^{2}\\ b_{ijk}^{3}\end{bmatrix}=\begin{bmatrix} -1 & -1 & 1\\ -1 & 1 & -1\\ 1 & -1 & -1\end{bmatrix}\begin{bmatrix}\hat{d}_{ij}\\ \hat{d}_{jk}\\ \hat{d}_{ik}\end{bmatrix},
$$

and $\alpha_{ij}^{+}, \alpha_{ij}^{-}$ are the $2|N|$ primal *decision* variables associated with the links; the $3|M|$ primal *slack* variables $\alpha_{ijk}^{1}, \alpha_{ijk}^{2}, \alpha_{ijk}^{3}$ associated with the triangles serve to transform the inequalities to equations.

A *basic* solution to the primal problem contains exactly $3|M|$ nonzero (basic) variables and $2|N|$ zero (nonbasic) variables for the nondegenerate case which assumes linearly independent constraints; a *feasible* solution contains no negative variables. Since the optimal solution is necessarily both basic and feasible [6], the conventional *simplex method* pivots through basic feasible solutions of the system until finding the optimal one. A *pivot* consists of raising an *entering* variable in the nonbasic set from zero that will improve the objective function; the entering variable can rise only a certain amount until a *blocking* variable in the basic set reduces to zero; hence the entering variable becomes basic and the blocking variable nonbasic at another basic feasible solution of the system. Determining this amount necessitates global knowledge of all variables such that no variable becomes negative, and exactly $2|N|$ of them equal zero throughout the pivots. Hence the simplex method does not lend to distributed processing. *Interior-point methods* are designed for centralized computing even less than the simplex method, requiring the inversion of large sparse symmetric matrices [7].

B. The dual problem

Each primal linear program has a unique *dual* linear program. The dual problem to (4) appears as [6]:

$$
\begin{aligned}
\max \;& \underset{(i,j,k)\in M}{} b_{ijk}^{1}\beta_{ijk}^{1}+b_{ijk}^{2}\beta_{ijk}^{2}+b_{ijk}^{3}\beta_{ijk}^{3}\\
\text{s.t.}\;& \\
& \left.\begin{aligned}
\alpha_{ij}^{+}\!:\;& \underset{k,\ (i,j,k)\in M}{} a_{ij,k}^{1}\beta_{ijk}^{1}+a_{ij,k}^{2}\beta_{ijk}^{2}+a_{ij,k}^{3}\beta_{ijk}^{3}+\beta_{ij}^{+}=1\\
\alpha_{ij}^{-}\!:\;& \underset{k,\ (i,j,k)\in M}{} -a_{ij,k}^{1}\beta_{ijk}^{1}-a_{ij,k}^{2}\beta_{ijk}^{2}-a_{ij,k}^{3}\beta_{ijk}^{3}+\beta_{ij}^{-}=1
\end{aligned}\right\},\\
&\hfill \forall (i,j)\in N\\
& \left.\begin{aligned}
\alpha_{ijk}^{1}\!:\;& \beta_{ijk}^{1}\ge 0\\
\alpha_{ijk}^{2}\!:\;& \beta_{ijk}^{2}\ge 0\\
\alpha_{ijk}^{3}\!:\;& \beta_{ijk}^{3}\ge 0
\end{aligned}\right\},\ \forall (i,j,k)\in M
\end{aligned}
\tag{5}
$$

where $\beta_{ijk}^{1}, \beta_{ijk}^{2}, \beta_{ijk}^{3}$ are the dual decision variables associated with the triangles, and $\beta_{ij}^{+}, \beta_{ij}^{-}$ are the dual slack variables associated with the links. As indicated, each primal variable has a corresponding dual constraint.

The Complementary Slackness Theorem [6] states that *any* feasible primal and dual solutions are optimal if the following complementary slackness conditions hold:

$$
\begin{array}{llll}
(a) & \alpha_{ijk}^{u}=0 \Rightarrow \beta_{ijk}^{u}\ge 0 & & \forall (i,j,k)\in M\\
(b) & \alpha_{ijk}^{u}>0 \Rightarrow \beta_{ijk}^{u}=0 & , & \forall u\in\{1,2,3\}\\
(c) & \alpha_{ij}^{v}=0 \Rightarrow \beta_{ij}^{v}\ge 0 & & \forall (i,j)\in N\\
(d) & \alpha_{ij}^{v}>0 \Rightarrow \beta_{ij}^{v}=0 & , & \forall v\in\{+,-\}
\end{array}
\tag{6}
$$

Rather than solve the primal problem through the simplex method, we formulate a distributed version of the *primal-dual method* in the following section. The key advantage to the latter relaxes the condition that a primal solution be basic. Our algorithm proceeds in the follow manner:

1) first a link in the network finds a feasible (not necessarily basic) primal solution locally such that all incident triangles meet the triangle inequality constraints;
2) the link then applies the complementary slackness conditions given through this primal solution, defining the *restricted* dual problem locally;
3) if the dual solution to the restricted problem is also feasible, then the primal solution is optimal through the Complementary Slackness Theorem; otherwise the primal solution is modified to improve the objective function.

Once all the links attain optimal solutions, the network achieves the globally optimal solution. Dantzig treats a full discussion on the primal-dual method [7].

IV. Distributed Link Distance Estimation

A. Network organization

The nodes in the network transmit asynchronously. If node i wakes up after node j, then n_i defaults as manager of link $_{ij}$. The link manager maintains the information on $_{ij}$: the measured distance $\hat{d}_{ij}$ and the residual α_{ij} initialized to zero. Consider three nodes n_i, n_j, n_k in a network all within radio range, where n_k woke up first and n_j second, assigning manager of $_{jk}$ to n_j. When n_i wakes up subsequently, it broadcasts a HELLO message containing its ID#: i. Nodes n_j and n_k respond with their ID#s; since n_j acts as link manager, it also broadcasts the information on the links it manages, namely $\hat{d}_{jk}$. In receiving the messages from n_j and n_k, n_i becomes manager of $_{ij}$ and $_{ik}$ and estimates $\hat{d}_{ij}$ and $\hat{d}_{ik}$; the two-way message exchange allows n_i to measure these distances asynchronously [8]. As manager, n_i then broadcasts the information on the links it manages. Now both managers n_i and n_j have access to information on all three links of $_{ijk}$ and can so compute their local primal and dual variables:

Definition: The local primal and dual variables to link $_{ij}$ are those associated with its incident triangles $\alpha_{ijk}^{u}, \beta_{ijk}^{u}$, $\forall k,\ (i,j,k)\in M$; $\forall u\in\{1,2,3\}$ and those associated with the links of those triangles $\alpha_{kl}^{v}, \beta_{kl}^{v},\ \forall l\in\{i,j\}$; $\forall v\in\{+,-\}$. When pivoting in the distributed primal-dual method, link $_{ij}$ considers only its local primal and dual variables. While we refer to the links as the processing centers for the distributed algorithm in the sequel, the actual processing of course takes place at the link managers.

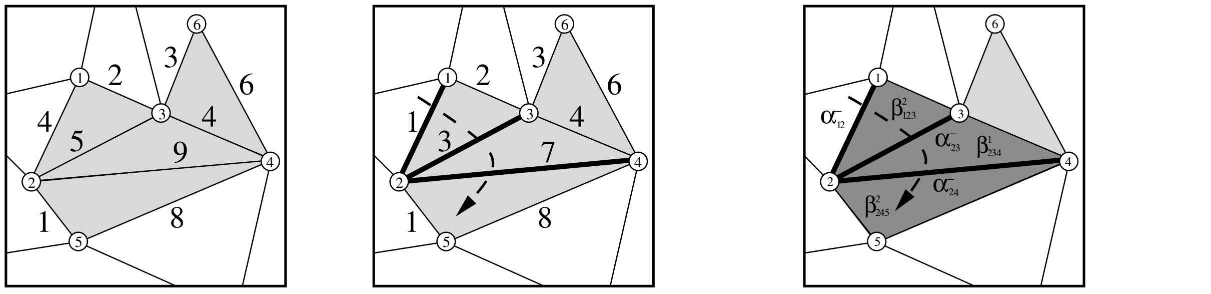

(a) The initial network.
(b) A feasible primal solution.
(c) The corresponding restricted dual problem.

Fig. 1. Propagation in the distributed primal-dual algorithm.

B. A feasible primal solution

Denote a triangle $\ _{ijk}$ as feasible if it meets all three of its triangle inequality constraints: $\alpha^u_{ijk} \geq 0,\ \forall u \in \{1,2,3\}$. Suppose that a link $\ _{ij}$ changes value, rendering one of its incident triangles infeasible: $\alpha^u_{ijk} < 0$, for u, k. Theorem I (15) shows that if any one of the constraints is bound, then the other two are feasible; so by setting $\alpha^u_{ijk} = 0$, $\ _{ij}$ restores feasibility to $\ _{ijk}$. Since the value of $\ _{ij}$ was just changed, it remains the same; rather the link selects one of the other two links on the triangle ($\ _{jk}$ or $\ _{ik}$) to set the value of α^u_{ijk} to zero, say $\ _{jk}$. Consider modifying α^v_{jk} such that $\alpha_{jk} = \alpha_{jk} + \frac{\partial \alpha^v_{jk}}{\partial \alpha^u_{ijk}}\ \delta\alpha^u_{ijk}$, where $\delta\alpha^u_{ijk} = -\alpha^u_{ijk}$ represents the necessary change in α^u_{ijk} to render the violated constraint feasible. The partial

$$\left.\frac{\partial \alpha^v_{jk}}{\partial \alpha^u_{ijk}}\right|_{\alpha_{ij},\alpha_{ik}} = \frac{1}{v a^u_{jk,i}} \tag{7}$$

while maintaining the value of the links $\ _{ij}$ and $\ _{ik}$ constant is computed through rewriting the primal constraint u in (4) as

$$a^u_{ij,k}\,\underbrace{(\alpha^+_{ij}\text{-}\alpha^-_{ij})}_{\alpha_{ij}} + a^u_{jk,i}\,\overbrace{(\alpha^+_{jk}\text{-}\alpha^-_{jk})}^{v\alpha^v_{jk}} + a^u_{ik,j}\,\underbrace{(\alpha^+_{ik}\text{-}\alpha^-_{ik})}_{\alpha_{ik}} \text{-}\alpha^u_{ijk} = b^u_{ijk}. \tag{8}$$

Note that $\alpha^v_{jk} > 0$ implies $\alpha^{\bar{v}}_{jk} = 0$. The pseudocode below summarizes the *local pivot*:

$$\left[\begin{array}{l} \text{Pivot I: Set } \alpha^u_{ijk} = 0 \text{ by modifying } \alpha^v_{jk}: \\ \alpha_{jk} = \alpha_{jk} + \frac{\partial \alpha^v_{jk}}{\partial \alpha^u_{ijk}}\ \delta\alpha^u_{ijk} \\ \left.\frac{\partial \alpha^v_{jk}}{\partial \alpha^u_{ijk}}\right|_{\alpha_{ij},\alpha_{ik}} = \frac{1}{v a^u_{jk,i}} \\ \delta\alpha^u_{ijk} = -\alpha^u_{ijk} \end{array}\right. \tag{9}$$

In restoring feasibility to $\ _{ijk}$, $\ _{jk}$ may in turn render a neighboring triangle $\ _{jkl}$ infeasible, analogous to the change in $\ _{ij}$ which rendered $\ _{ijk}$ infeasible in the previous step. This mechanism causes infeasibility to *propagate* through the triangles in the network, leaving those in its path feasible, and so obtaining a feasible primal solution locally at each step until termination at a certain triangle. In the worst case, propagation terminates at the edge of the network where $\ _{jk}$ has no neighboring $\ _{jkl}$. The following subsection describes how to select α^v_{jk} optimally in Pivot I.

The portion of a network in Fig. 1a consists of four feasible triangles (shaded) with the estimated distances $d = \hat{d}$ displayed on each link. Let d_{12} decrease to 1, rendering $\ _{123}$ infeasible. The infeasibility then propagates along the path indicated by the dashed arrow in Fig. 1b, altering the values of the boldfaced links: d_{23} decreases to 3, restoring feasibility to $\ _{123}$, but renders $\ _{234}$ infeasible; d_{24} decreases to 7, restoring feasibility to $\ _{234}$ while maintaining $\ _{245}$ feasible. The propagation terminates at $\ _{245}$ with all four triangles newly feasible.

C. The restricted dual problem

1) Defining and solving the restricted dual problem: We say that a link $\ _{ij}$ has a feasible primal solution locally if all of its incident triangles are feasible. Once a link establishes this, it can apply the complementary slackness conditions to its local variables in order to define the restricted dual locally:

- every bound primal constraint $\alpha^u_{ijk} = 0$ admits one dual decision variable (unknown) β^u_{ijk} to the restricted problem (6a), while setting the remaining dual decision variables to zero (6b). Theorem II (16) shows that a triangle with two bound links is isosceles; it readily follows a triangle with three bound links is equilateral. So a triangle may have up to three bound constraints, which in turn means that each triangle admits up to three unknowns to the dual.
- every nonzero primal decision variable $\alpha^v_{ij} > 0$ admits one bound dual constraint (equation) $\beta^v_{ij} = 0$ to the restricted problem (6d), while unbounding the remaining dual constraints (6c). A link introduces two constraints indexed as α^+_{ij} and α^-_{ij} to the dual problem in (5). Adding these constraints together yields $\beta^+_{ij} + \beta^-_{ij} = 2$; this implies that bounding one of the constraints as $\beta^v_{ij} = 0$ unbounds the other as $\beta^{\bar{v}}_{ij} = 2$, which in turn means that each link admits at most one equation to the dual.

Fig. 1c graphically represents the restricted dual problem corresponding to the feasible primal solution in Fig. 1b: each of the three darkly shaded triangles has one bound primal constraint, admitting one unknown per triangle to the dual: β^2_{123} ($d_{12}+d_{13}=d_{23}$) at $\ _{123}$, β^1_{234} ($d_{23}+d_{34}=d_{24}$) at $\ _{234}$, and β^2_{245} ($d_{24}+d_{25}=d_{45}$) at $\ _{245}$; each of the three boldfaced links with nonzero residuals ($\alpha^-_{12}=3, \alpha^-_{23}=2, \alpha^-_{24}=2$) admits one equation per link to the dual in (5): ($\text{-}\beta^2_{123}=1$) at $\ _{12}$, ($\beta^2_{123}\text{-}\beta^1_{234}=1$) at $\ _{23}$, and ($\beta^1_{234}\text{-}\beta^2_{245}=1$) at $\ _{24}$. Link $\ _{12}$

solves for its single unknown $\beta^2_{123} = -1$, and then broadcasts the value to the other two links of $_{123}$; now knowing the value of β^2_{123}, l_{23} solves for its single remaining unknown $\beta^1_{234} = 2$, and then broadcasts the value to the other two links of $_{234}$; now knowing the value of β^1_{234}, $_{24}$ solves for its single remaining unknown $\beta^2_{245} = 1$, completing the restricted dual solution, and then broadcasts the value to the other two links of $_{245}$. As demonstrated, the dual solution, like the primal, is found by propagating the solution through the triangles in the network. The dashed arrow in Fig. 1c indicates the direction of propagation.

As any solution to a nondegenerate linear system contains at least the same number of unknowns as equations, any solution to the nondegenerate primal system in (4) contains at least the same number of nonzero primal decision variables as the number of bound primal constraints. So due to complementary slackness, the restricted dual contains at least the same number of equations as the number of unknowns: a potentially overconstrained system. If not overconstrained, the unknowns in the restricted dual can be found through propagated substitution, as shown above; otherwise if the equation of a bound link $_{ij}$ is overconstrained, increase the number of unknowns at the link by admitting a dual decision variable β^u_{ijk}. This can be achieved by setting the corresponding primal slack variable $\alpha^u_{ijk} > 0$ to zero through Pivot I (9), effectively bounding another constraint of a triangle incident on $_{ij}$. The following describes how to select α^u_{ijk} optimally in Pivot I.

2) Modifying the primal solution towards optimality: If the solution to the restricted dual is feasible (i.e. all the decision and slack variables are greater than or equal to zero), then the primal solution is optimal. Otherwise it can be shown [7] that setting an infeasible dual variable to zero improves the primal objective, provided that the nonzero feasible dual variables remain admitted to the restricted dual problem.

Case i: If the restricted dual solution includes an infeasible decision variable $\beta^u_{ijk} < 0$, then raising its corresponding primal slack variable α^u_{ijk} from zero sets $\beta^u_{ijk} = 0$ through complementary slackness.

- IF entering α^u_{ijk} lowers a local blocking variable $\alpha^{\tilde{u}}_{ijk} > 0, \tilde{u} = u, \frac{\partial \alpha^u_{ijk}}{\partial \alpha^{\tilde{u}}_{ijk}} = -1$; conversely, reducing $\alpha^{\tilde{u}}_{ijk}$ to zero through Pivot I raises α^u_{ijk}. The partial
$$\left(\frac{\partial \alpha^u_{ijk}}{\partial \alpha^{\tilde{u}}_{ijk}}\right)_{\alpha_{ij},\alpha_{ik}} = \left(\frac{\partial \alpha^v_{jk}}{\partial \alpha^{\tilde{u}}_{ijk}}\right)\left(\frac{\partial \alpha^u_{ijk}}{\partial \alpha^v_{jk}}\right) = \frac{a^u_{jk,i}}{a^{\tilde{u}}_{jk,i}} \quad (10)$$
is computed through (7). Note that Pivot I results in $\alpha^v_{jk} > 0$, and in turn sets $\beta^v_{jk} = 0$ through complementary slackness; so if $\beta^v_{jk} > 0$ before the pivot, the pivot removes a nonzero feasible dual variable from the restricted dual. Select α^v_{jk} such that $\beta^v_{jk} \leq 0$.
- ELSE entering α^u_{ijk} lowers a local blocking variable $\alpha^v_{jk} > 0, \frac{\partial \alpha^u_{ijk}}{\partial \alpha^v_{jk}} = -1$; conversely, reducing α^v_{jk} to zero raises α^u_{ijk}. The reciprocal of the partial $\frac{\partial \alpha^u_{ijk}}{\partial \alpha^v_{jk}}$ appears in (7).

Case ii: If the restricted dual solution includes an infeasible slack variable $\beta^v_{ij} < 0$, then raising its corresponding primal decision variable α^v_{ij} from zero sets $\beta^v_{ij} = 0$ through complementary slackness.

- IF entering α^v_{ij} lowers a local blocking variable $\alpha^{\tilde{v}}_{jk} > 0, \frac{\partial \alpha^v_{ij}}{\partial \alpha^{\tilde{v}}_{jk}} = -1$; conversely, reducing $\alpha^{\tilde{v}}_{jk}$ to zero through Pivot II below raises α^v_{ij}. The partial $\frac{\partial \alpha^v_{ij}}{\partial \alpha^{\tilde{v}}_{jk}}$ is computed through (8).
$$\left[\begin{array}{l} \text{Pivot II: Set } \alpha^{\tilde{v}}_{jk} = 0 \text{ by raising } \alpha^v_{ij} : \\ \alpha_{ij} = \alpha_{ij} + \dfrac{\partial \alpha^v_{ij}}{\partial \alpha^{\tilde{v}}_{jk}} \; \delta\alpha^{\tilde{v}}_{jk}, \\ \dfrac{\partial \alpha^v_{ij}}{\partial \alpha^{\tilde{v}}_{jk}}_{\alpha_{ik},\alpha^u_{ijk}=0} = -\dfrac{\tilde{v}a^u_{jk,i}}{va^u_{ij,k}}, \\ \delta\alpha^{\tilde{v}}_{jk} = -\alpha^{\tilde{v}}_{jk} \end{array}\right. \quad (11)$$
Note that Pivot II affects the value of $\alpha^u_{ijk}, \forall u \in \{1,2,3\}$; so if $\beta^u_{ijk} > 0$ before the pivot, maintain $\alpha^u_{ijk} = 0$ such that the nonzero feasible dual variable remains in the restricted problem.
- ELSE entering α^v_{ij} lowers a local blocking variable $\alpha^u_{ijk} > 0, \frac{\partial \alpha^v_{ij}}{\partial \alpha^u_{ijk}} = -1$; conversely, reducing α^u_{ijk} to zero through Pivot I raises α^v_{ij}[1].

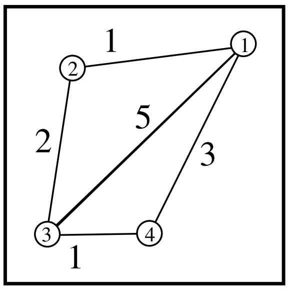
Recall in Section IV-B that the initial feasible primal solution is found in the absence of any dual restrictions, and so the arbitrary selection of a link to modify in Pivot I may lead to a sub-optimal solution. When modifying the primal solution towards optimality, a local pivot to a link may affect the primal feasibility of a non-incident triangle. And so another link on the triangle maneuvers to restore its feasibility through Pivot I, but now using dual restrictions to decide which one. The new feasible primal solution of the triangle in turn generates other dual restrictions local to this other link. As the pivots continue, the feasible primal solution becomes more and more restricted by the dual solution until reaching global optimality. An example presented in the following section substantiates these ideas.

D. Example network

Consider the example network in Fig. Ia of Table I with four nodes and five links. The measured distances $\hat{d}$ appear on each link. Table I displays the coefficients of the primal problem corresponding to this example network in $\mathbf{A}$ and $\mathbf{b}$. The blank slots indicate zeros and are left so to reduce clutter. As the nodes wake up asynchronously, let links $_{12}$ and $_{34}$ be the last two established, completing the triangles $_{123}$ and $_{234}$ respectively. Note that $_{123}$ and $_{234}$ are initially infeasible with $\alpha^1_{123} = -2$ and $\alpha^3_{134} = -1$. In solving the initial primal problem and in the absence of any dual restrictions, $_{12}$ raises its value to $7(\alpha^+_{12} = 6)$ to restore feasibility to $_{123}$, and broadcasts this value to the other two links; likewise $_{34}$ raises its value to $2(\alpha^+_{34} = 1)$ to restore feasibility to $_{234}$, and broadcasts this value to the other two links. Each network link subsequently computes its corresponding local primal decision and slack variables. This first (1) primal solution with objective 7 appears in row $\alpha^v_{ij}(1)$ and column $\alpha^u_{ijk}(1)$ of Table I. The solution is indexed according to global solutions for the sake of clarity, but as just explained, the solutions are computed locally, distributedly, and asynchronously.

[1]Changing the subscript from α^v_{ij} to α^v_{jk}.

TABLE I
THE PRIMAL-DUAL TABLE



(a) The example network.

A	α_{12}^{+}	α_{12}^{-}	α_{23}^{+}	α_{23}^{-}	α_{13}^{+}	α_{13}^{-}	α_{34}^{+}	α_{34}^{-}	α_{14}^{+}	α_{14}^{-}	**b**	$\alpha_{ijk}^{u}(1)$	$\beta_{ijk}^{u}(1)$	$\alpha_{ijk}^{u}(2)$	$\beta_{ijk}^{u}(2)$	$\alpha_{ijk}^{u}(3)$	$\beta_{ijk}^{u}(3)$
β_{123}^{1}	1	−1	1	−1	−1	1					2	4			$\boxed{1}$		$\boxed{1}$
β_{123}^{2}	1	−1	−1	1	1	−1					−4	10		4		4	
β_{123}^{3}	−1	1	1	−1	1	−1					−6		$\boxed{-1}$	4		4	
β_{234}^{1}					1	−1	1	−1	−1	1	−7	8		8		6	
β_{234}^{2}					1	−1	−1	1	1	−1	−3	2		2		2	
β_{234}^{3}					−1	1	1	−1	1	−1	1		$\boxed{1}$		$\boxed{1}$		$\boxed{0}$
$\alpha_{ij}^{v}(1)$	6			0			1		0								
$\beta_{ij}^{v}(1)$	$\boxed{}$	2	2		3	−1	$\boxed{}$	2		2							
$\alpha_{ij}^{v}(2)$	2		0				1		0								
$\beta_{ij}^{v}(2)$	$\boxed{}$	2		2	3	−1	$\boxed{}$	2		2							
$\alpha_{ij}^{v}(3)$	1		0			1											
$\beta_{ij}^{v}(3)$	$\boxed{}$	2		2	2	$\boxed{}$	1	1	1	1							

With all the triangles incident on $_{12}$ now feasible, $_{12}$ applies the complementary slackness conditions from its primal solution to define the corresponding restricted dual problem locally:

$$\alpha_{123}^{3} = 0 \Rightarrow \beta_{123}^{3} \geq 0$$
$$\{\alpha_{123}^{1}, \alpha_{123}^{2}\} > 0 \Rightarrow \{\beta_{123}^{1}, \beta_{123}^{2}\} = 0$$
$$\{\alpha_{12}^{-}, \alpha_{23}^{+}, \alpha_{23}^{-}, \alpha_{13}^{+}, \alpha_{13}^{-}\} = 0 \Rightarrow \{\beta_{12}^{-}, \beta_{23}^{+}, \beta_{23}^{-}, \beta_{13}^{+}, \beta_{13}^{-}\} \geq 0$$
$$a_{12}^{+} > 0 \Rightarrow \beta_{12}^{+} = 0$$

So $_{12}$ processes a single unknown ($\beta_{123}^{3} \geq 0$) and a single equation ($-\beta_{123}^{3} = 1$) to solve for its local dual variables. It solves for $\beta_{123}^{3} = -1$ and distributes this value to the other two links of $_{123}$. Through the same process, $_{34}$ solves for $\beta_{234}^{3} = 1$ in its restricted dual and distributes this value to the other two links of the triangle. Now that each link $_{ij}$ in the network holds the dual decision variables $\beta_{ijk}^{1}, \beta_{ijk}^{2}, \beta_{ijk}^{3}$ of all incident triangles $_{ijk}$, it can compute its slack dual variables $\beta_{ij}^{+}, \beta_{ij}^{-}$. The first (1) dual solution appears in column $\beta_{ijk}^{u}(1)$ and row $\beta_{ij}^{v}(1)$ of Table I. The table evidences the complimentary slackness structure of the primal-dual solution, where the dual slack (decision) variable is zero if a primal decision (slack) variable is nonzero; the boxes indicate the admitted unknowns and equations in each restricted dual.

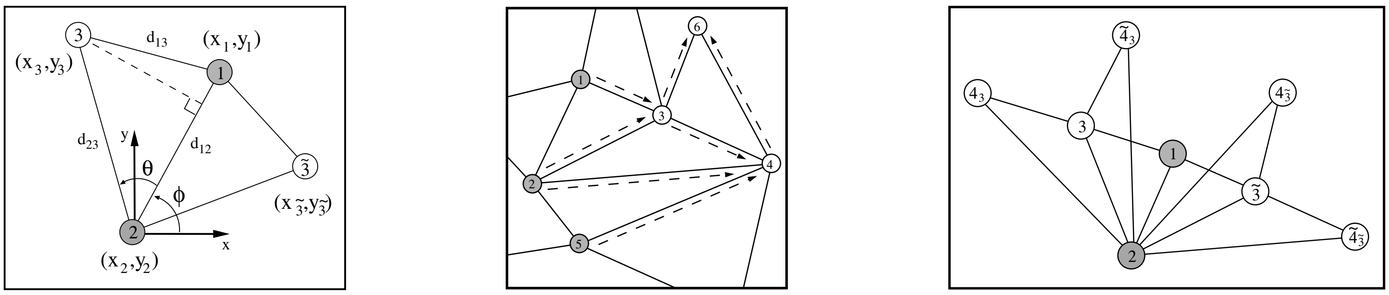
The first dual solution reveals two infeasible variables $\beta_{123}^{3} = -1$ and $\beta_{13}^{-} = -1$. Raising α_{123}^{3} or α_{13}^{-} will improve the primal objective, provided that the nonzero feasible variables remain admitted to the restricted dual problem. Link $_{12}$ raises α_{123}^{3} through Pivot I, setting the local blocking variable $\alpha_{123}^{1} = 4$, $\frac{\partial \alpha_{123}^{3}}{\partial \alpha_{123}^{1}} = -1$ to zero by modifying the value of $_{12}$ to 3 ($\alpha_{12}^{+} = 2$). Selecting to remove $\beta_{12}^{+} = 0$ from the restricted dual problem ensures that this pivot improves the primal objective. The second primal solution remains feasible after Pivot I, necessitating no additional pivots to restore feasibility. Note that the primal objective equal to 3 has indeed improved. The second primal-dual solution appears in Table I.

The second dual solution still reveals the infeasible variable $\beta_{13}^{-} = -1$. Link $_{13}$ raises α_{13}^{-} through Pivot II, setting the local blocking variable $\alpha_{34}^{+} = 1$, $\frac{\partial \alpha_{13}^{-}}{\partial \alpha_{34}^{+}} = -1$ to zero while maintaining $\alpha_{134}^{3} = 0$ such that $\beta_{134}^{3} = 1$ remains admitted to the restricted dual problem. This pivot however raises α_{123}^{1} from zero, violating a dual restriction by removing $\beta_{123}^{1} = 1$ from the same restricted dual. Link $_{13}$ sets α_{123}^{1} back to zero through Pivot I by modifying the value of $_{12}$ to 2 ($\alpha_{12}^{+} = 1$). (Alternatively modifying the value of $_{23}$ to 1 ($\alpha_{23}^{-} = 1$) would remove the nonzero feasible dual variable $\beta_{23}^{-} = 2$ from the dual problem and in consequence raise the primal objective function to 4.) The third primal-dual solution with objective 2 appears in Table I, where all the feasible dual variables indicate the optimality of this primal solution. As confirmed here, the value of the dual objective function at optimality equals that of the primal objective function [6].

V. Distributed Location Reconstruction

A. Location propagation

The reconstruction stage yields the locations of the sensors in the network from the link distances estimated through the linear program (3). The stage originates at any two anchor nodes (shaded) sharing a neighboring sensor node (unshaded), as in Fig. 2a. Given $\mathbf{x}_1 = (x_1, y_1)$ and $\mathbf{x}_2 = (x_2, y_2)$ (and so the distance d_{12} between them), and d_{13} and d_{23}, the following set of equations [9] furnishes the unknown location $\mathbf{x}_3 = (x_3, y_3)$ with respect to the reference coordinate system centered at $\mathbf{x}_2$:



(a) The geometry. (b) Location propagation from the anchor nodes. (c) Candidate mirror locations.

Fig. 2. Location reconstruction.

$$
\begin{aligned}
&(a)\quad \theta = \arccos\ \frac{d_{12}^2 + d_{23}^2 - d_{13}^2}{2d_{12}d_{23}} \\
&(b)\quad \begin{array}{c} x_3 \\ y_3 \end{array} = \begin{array}{c} x_2 \\ y_2 \end{array} + \begin{array}{c} d_{23}\cos\theta \\ s\cdot d_{23}\sin\theta \end{array} \qquad (12)\\
&(c)\quad \phi = \arctan\ \frac{y_2 - y_1}{x_2 - x_1} \\
&(d)\quad \begin{array}{c} x_{\tilde{3}} \\ y_{\tilde{3}} \end{array} = \begin{array}{c} x_2 \\ y_2 \end{array} + \begin{array}{cc} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{array} \begin{array}{c} x_3 - x_2 \\ y_3 - y_2 \end{array}
\end{aligned}
$$

In fact, the set actually furnishes two candidate locations mirrored about the line between $\mathbf{x}_1$ and $\mathbf{x}_2$ for the same given: $\mathbf{x}_3$ for $s = 1$ in (12b) and $\mathbf{x}_{\tilde{3}}$ for $s = -1$. This *mirror ambiguity* [10] arises from the use of only two anchor nodes to determine location.

Once a node determines its location, starting with the anchor nodes, it broadcasts this location to its neighboring nodes; we say that the two anchor nodes propagate their locations to the unknown sensor node. Now that the sensor node is *known*, it serves with another known sensor (or anchor) to determine the location of an unknown sensor neighboring the two. Following the arrows in Fig. 2b, n_1 and n_2 propagate their locations to n_3; nodes n_2 and n_5 propagate their locations to n_4; n_2 and n_3 propagate their locations to n_4; and n_3 and n_4 propagate their locations to n_6.

If, however, n_3 cannot resolve the mirror ambiguity after the first propagation, it computes both candidate locations $\mathbf{x}_3$ and $\mathbf{x}_{\tilde{3}}$ shown in Fig. 2c and broadcasts them both; the information it subsequently receives through its neighbors about the locations of other known sensors in the network serves to resolve that ambiguity. Note the each propagation step from the origin potentially doubles the number of candidate locations: Fig. 2c illustrates the four candidate locations $\mathbf{x}_{4_3}$, $\mathbf{x}_{\tilde{4}_3}$, $\mathbf{x}_{4_{\tilde{3}}}$, $\mathbf{x}_{\tilde{4}_{\tilde{3}}}$ for n_4 after the second propagation. Rather than double these four locations yet further to eight, node n_4 can exploit the network redundancy to dismiss candidates $\mathbf{x}_{\tilde{4}_3}$ and $\mathbf{x}_{4_3}$ which place n_1 and n_4 within radio range even though they are not neighbors; this mechanism suppresses the exponential growth of candidates. Node n_4 also separately computes two candidate mirror locations originating from anchors n_2 and n_5, but only one will be correspond to the true location $\mathbf{x}_{\tilde{4}_{\tilde{3}}}$, so dismissing the other remaining candidate $\mathbf{x}_{4_{\tilde{3}}}$. Broadcasting its unique location, node n_4 in turn enables n_3 to resolve its true location as $\mathbf{x}_{\tilde{3}}$.

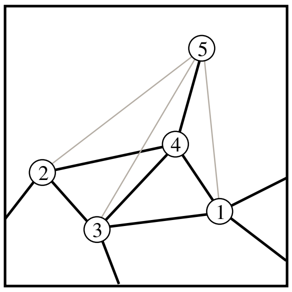


Fig. 3. Artificial links.

B. Artificial links

1) Location propagation to singly-connected sensor nodes: As described in the previous subsection, the location of a node is reconstructed through connections to it from two other nodes in the network. If a node such as n_5 in Fig. 3 has only one *normal link* (dark) to the network, we insert an *artificial link* (light) to it from a non-neighboring node: $_{15}$ now enables location propagation to it from n_1 and n_4. As any other normal link, an artificial link between $\mathbf{x}_i$ and $\mathbf{x}_j$, $(i,j) \in \bar{N}$ generates a set of new triangles in the network and so appears in the corresponding set of triangle inequality constraints in (3). Lacking an estimated distance for it, however, an artificial link is less restrained than a normal link in that we can only exploit the relationship $d_{ij} = \hat{d}_{ij} + \alpha_{ij}^{+} \geq R$. So rather than arbitrarily minimize the positive residual α_{ij}^{+} in the objective function, set $\hat{d}_{ij} = R$ and introduce only the positive bounding constraint $\alpha_{ij}^{+} \geq 0$ in the linear program ($\alpha_{ij}^{-} = 0$ since $d_{ij} < R$).

Although the single artificial link between n_1 and n_5 suffices to reconstruct $\mathbf{x}_5$, adding multiple artificial links to all neighbors of n_4 as in Fig. 3 guarantees a tighter solution to the problem since they are individually less restrained than normal links. In theory, artificial links between all non-neighboring nodes in the network guarantee an even tighter solution, however they also increase the computation complexity of the linear program; in practice, an artificial link $_{ij}$ between two nodes n_i and n_j bearing normal links to a third node n_k completes the triangle and renders satisfactory results; including any more than in this case proves superfluous.

A node completely disconnected from the network cannot gather any location information except that it lies beyond the radio range R of all other nodes in the network, hence no deterministic method exists to compute its location with any meaningful accuracy.

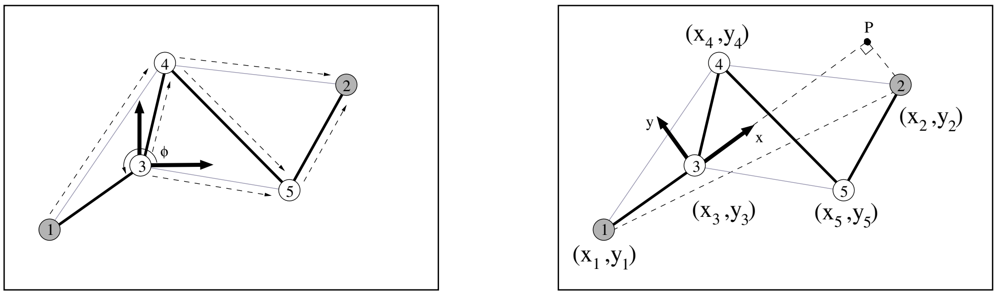


Fig. 4. Location propagation between anchors. (a) The path $n_1 \to n_3 \to n_4 \to n_5 \to n_2$ connects the two anchors. (b) Location propagation from $\mathbf{x}_1$ and $\mathbf{x}_3$ to $\mathbf{x}_4$, $\mathbf{x}_5$, and $\mathbf{x}_2$.

2) Location propagation between anchor nodes: An anchor node in the network can be incorporated in the linear program (3) by including the constraints associated with all triangles formed between it, another anchor node, and a sensor node neighboring both anchors. Set the residual of the link distance between the anchor pair to zero since this distance is known. In general network topologies, especially those with low anchor density, no single sensor node connects any two anchors, as in Fig. **??**a. Consider as an alternative the connection between the two anchors through the minimum-hop path $n_1 \to n_3 \to n_4 \to n_5 \to n_2$ found during network organization. Adding the three artificial links $_{14}$, $_{35}$, and $_{24}$ enables location propagation between the two anchors according to the arrows: from n_1 and n_3 to n_4, from n_3 and n_4 to n_5, and from n_4 and n_5 to n_2.

In contrast to location propagation from two anchor nodes as described in the previous subsection, here propagation originates from one known anchor node n_1 and one unknown sensor node n_3. Since $\mathbf{x}_3$ is unknown, the value of ϕ in Fig. **??**a is also unknown. Consider instead establishing the relative coordinate system illustrated in Fig. **??**b oriented along the line between n_1 and n_3 and centered at $\mathbf{x}_3$. Now propagate $\mathbf{x}_1$ and $\mathbf{x}_3$ to $\mathbf{x}_4$, $\mathbf{x}_5$, and $\mathbf{x}_2$ as in Subsection V-A, however rather than in terms of real values and with respect to the reference coordinate system, here in terms of the variables (x_3, y_3) and with respect to the relative coordinate system. These locations assume the following form with individual real-valued terms (d^x, d^y) computed pairwise and stepwise through (12): $(x_4, y_4) = (x_3 + d^x_{34}, y_3 + d^y_{34})$, $(x_5, y_5) = (x_4 + d^x_{45}, y_4 + d^y_{45})$, and $(x_2, y_2) = (x_5 + d^x_{25}, y_5 + d^y_{25})$, or compactly

$$\begin{array}{c} x_2 \\ y_2 \end{array} = \begin{array}{c} x_3 + d_{3P} \\ y_3 + d_{2P} \end{array}, \tag{13}$$

where $d_{3P} = d^x_{34} + d^x_{45} + d^x_{25}$ and $d_{2P} = d^y_{34} + d^y_{45} + d^y_{25}$ represent the distances from n_3 and n_2 respectively to the imaginary point P in the relative coordinate system. Once the propagation reaches n_2, backward substitute (x_2, y_2) in (13) to solve for the unknown values (x_3, y_3), which in turn yield the locations of the other sensor nodes along the minimum-hop path. Now that all nodes along the path have known neighbors with which to propagate location to other unknown sensors, propagation can continue as in the previous subsection. Note that the distances d_{3P} and d_{2P} in Fig. **??**b must conform to the Pythagorean Theorem $(d_{13} + d_{3P})^2 + d^2_{2P} = d^2_{12}$, where d_{13} and d_{12} are known values. This relationship serves to resolve any mirror ambiguities in propagation along the path up to the one degree of freedom associated with the use of only two anchor nodes.

VI. Experimental Setup and Results

A. Gentile vs. Biswas

In order to quantify the performance of our algorithm in comparison to Biswas, we conduct experiments on a network with the same structure as in [2]: the network contains 50 static sensor nodes uniformly distributed throughout a 1×1 unit area. The three varying parameters are the number of anchor nodes, the radio range, and the noisy factor of the link distances. As Biswas, the ground-truth link distances $\bar{d}_{ij}$ between neighboring nodes i and j are perturbed with zero-mean unit-variance Gaussian noise $\mathcal{N}(0,1)$ and the varying parameter $noise$, so the algorithm accepts as input the noisy link distances $\hat{d}_{ij} = \bar{d}_{ij} * (1 + \mathcal{N}(0,1) * noise)$. Interior-point methods, with an expected complexity of $\mathcal{O}(\sqrt{n} \ln n)$ where n denotes the number of constraints [11], have recently been shown to solve both linear and semidefinite programs more efficiently than the simplex method, hence the computational complexities of Gentile and Biswas are comparable.

Biswas reports the results of a single trial network for the six tests described in [2]. The quantitative measure for each is the average location error over the sensor nodes

$$\frac{1}{n_S} \overset{n}{\underset{i=n_A+1}{}} ||\bar{\mathbf{x}}_i - \mathbf{x}_i||_2, \tag{14}$$

where $\bar{\mathbf{x}}_i$ and $\mathbf{x}_i$ denote the ground-truth and estimated locations. Our paper includes a more extensive superset of their tests, spanning a much higher range of noise, for a total of 38. In addition, for each test we conduct ten trials of randomly distributed sensor networks rather than one, totaling 380 trials. Table II contains the results for 36 tests as the cross product of $\#anchor = \{3, 5, 7\}$, $R = \{0.20, 0.25, 0.30\}$, and $noise = \{0.0, 0.1, 0.2, 0.3\}$. For each slot in the table, the result of our algorithm is reported on the top line as an average and variance over the ten trials μ_{Gentile}, $\sigma^2_{\text{Gentile}}$, and if available, the corresponding result (μ_{Biswas}, σ^2_{Biswas}) in [2] is shown in parentheses on the middle line in the same slot, as indicated by the legend. For perfect range measurements with three anchor nodes, our algorithm performs only 12% better for $R = 0.25$ in terms of average error, but furnishes an average nearly

TABLE II
NUMERICAL RESULTS FOR EXPERIMENTS

noise	R=0.20			R=0.25			R=0.30		
	3	5	7	3	5	7	3	5	7
0.0	.0427, .0058 (.0800, .0900) 377.2, 561.0	.0419, .0111 369.5, 570.5	.0408, .0116 366.0, 584.2	.0067, .0007 (.0076, .0021) 433.4, 573.1	.0058, .0004 421.6, 577.0	.0058, .0004 412.3, 584.1	$<1e^{-6}$, $<1e^{-12}$ ($1.8e^{-4}$, $4.3e^{-5}$) 484.1, 551.9	$<1e^{-6}$, $<1e^{-13}$ 475.1, 556.1	$<1e^{-6}$, $<1e^{-13}$ 462.8, 557.0
0.1	.0754, .0087 395.1, 561.0	.0644, .0725 388.4, 570.5	.0638, .0101 382.2, 584.2	.0526, .0029 372.4, 573.1	.0436, .0025 463.4, 577.0	.0270, .0035 457.2, 584.1	0.0447, .0026 667.4, 551.9	0.0362, .0027 657.3, 556.1	0.0245, .0024 (0.0640, .0120) 643.1, 557.0
0.2	.0846, .0049 422.9, 561.0	.0801, .0094 415.8, 570.5	.0676, .0101 409.7, 584.2	.0764, .0035 556.2, 573.1	.0649, .0074 539.3, 577.0	.0493, .0057 527.5, 584.1	.0570, .0030 1051.2, 551.9	.0566, .0034 1021.7, 556.1	.0458, .0041 1015.5, 557.0
0.3	.1063, .0105 455.6, 561.0	.0877, .0075 453.9, 570.5	.0873, .0117 444.6, 584.2	.0954, .0089 847.0, 573.1	.0825, .0105 833.7, 577.0	.0772, .0067 827.9, 584.1	.0767, .0050 1405.4, 551.9	.0736, .0055 1384.3, 556.1	.0459, .0032 1373.1, 557.0

LEGEND
μ_{Gentile}, $\sigma^2_{\text{Gentile}}$
$\left(\mu_{\text{Biswas}}, \sigma^2_{\text{Biswas}}\right)$
$\#me_{\text{Dist.}}$, $\#me_{\text{Cntr.}}$

half the size as Biswas for $R = 0.20$, and on the order of 100 times smaller for $R = 0.30$. For seven anchor nodes, $R = 0.30$, and $noise = 0.1$, the average for Biswas is 161% greater than ours. In fact, in the same column their average .0640 for $noise = 0.1$ is still 39% greater than our average .0459 for $noise = 0.3$; this shows that our algorithm is much more robust to noise. The fifth test condition not appearing in the table included in [2] is for seven anchors, $R = 0.30$, and $noise = 0.05$, yielding $\mu_{\text{Gentile}} = .0162$, $\sigma^2_{\text{Gentile}} = .0021$ and ($\mu_{\text{Biswas}} = .0540$, $\sigma^2_{\text{Biswas}} = .0140$). The last competing test condition is for seven anchors, $R = 0.40$, and $noise = 0.1$, yielding $\mu_{\text{Gentile}} = .0162$, $\sigma^2_{\text{Gentile}} = .0007$ and ($\mu_{\text{Biswas}} = .0500$, $\sigma^2_{\text{Biswas}} = .0120$). Our error variances fall about a magnitude smaller than theirs, showing that the proposed algorithm not only turns out a smaller average error in all tests, but also a smaller error throughout all the nodes in the network.

B. Distributed vs. Centralized

We compare distributed and centralized versions of our algorithm by the number of transmitted messages required for each to converge for the same tests as in the previous subsection. As indicated in the legend, the bottom line of each slot in Table II displays the number of messages $\#me_{\text{Dist.}}$ and $\#me_{\text{Cntr.}}$ for each. The solutions in both the distributed and centralized versions match up one-to-one for each trial, confirming our results in the table and moreover demonstrating that our distributed version indeed finds the global optimum.

Distributed version: We initialize the network as in Subsection IV-A by placing a HELLO message on the queue of each node at t=0. A node processes any message on its queue immediately and generates a new message containing information pertaining to network organization, primal-dual pivots, or location propagation. Before broadcasting the new message to its neighbors, it waits a random variable $t \in (0, 1)$ ms uniformly distributed within the interval; the random delay mimics asynchronous node operation. The distributed algorithm combines both stages of link estimation and location reconstruction to furnish each sensor node with its estimated location upon convergence.

Centralized version: The centralized version organizes and sends messages in the same asynchronous fashion as described for the distributed version, but the link estimation and location reconstruction instead occur at a *processor* node. The processor is designated a priori as the one closest to the geometric center of the network to foster the shortest multi-hop routes in communicating between all the other nodes. Once it gathers all the measured distances from the sensors and the anchor locations, it generates the unknown locations and broadcasts them back to the sensors.

The distributed version outperforms the centralized version up to $noise = 0.2$ across all network parameters (except for $R = 0.30$). Larger noise causes more initially infeasible triangles, necessitating more iterations to solve the linear program: the distributed version dissipates those iterations in primal-dual messages while the centralized version places a greater burden on the processor but witnesses no increased messaging. Larger radio range increases the number of triangles in the network and in turn the number of constraints in the linear program, again increasing the primal-dual messages in the distributed version and the burden on the processor in the centralized version, but actually lowers the number of messages in the latter since nodes can communicate to the processor through fewer hops. The number of anchors does not significantly affect either version of the algorithm. In our test network, the number of hops in the centralized algorithm averaged from only 2.8 for $R = 0.2$ to 1.8 for $R = 0.30$, and so the distributed version could prove advantageous in larger networks with more hops to the processor. Indeed in the distributed version the messages remain local to the node, curbing latency issues in forwarding packets across the network, especially when the application necessitates frequent updating; moreover local messaging reduces the reliability on multiple links to the processor: if the node is cut off from the processor, or if the processor fails, it cannot recover its location. The centralized version may also require special units with enhanced processing capabilities.

The IEEE 802.15.4a working standard [12] provides a specification for low-cost Ultra Wideband communication devices with ranging capabilities to measure the distance between a pair. The radio transmits at a power of $P_{R=0.20} = 33$ nW for a range of roughly 20 m in line-of-sight conditions, at $P_{R=0.25}=52$ nW for a range of 25 m, and in accordance with the FCC unlicensed mask at a maximum power of $P_{R=0.30}=74$ nW for a range of 30 m. The three radio ranges can be associated with $R = 0.20$, $R = 0.25$, and $R = 0.30$ in our experiments. The packet length of a message has an average duration of 1 ms due to the long header which enables submeter ranging, so the network transmission energy consumed in running the algorithm can be computed for the tests in Table II as $\#me \times P_R \times 1$ ms. In both the centralized and distributed versions of the algorithms across all parameters, the shortest execution period is $t = 6.52$ ms for the distribution version with seven anchors, $R = 0.20$, and $noise = 0.0$ and the longest execution period is $t = 23.46$ ms for the distributed version with three anchors, $R = 0.30$, and $noise = 0.3$.

VII. Conclusions

This paper proposes a distributed algorithm to determine the locations of sensors in a network. Drawing on previous approaches employing complex optimization, our approach provides a tighter solution to the problem than its competitors by applying triangle inequality geometrical constraints to the network. In order to substantiate its performance, we run an extensive set of experiments in comparison with the published results for the best competing algorithm. Our algorithm outperforms the competing algorithm and proves robust even in the presence of high levels of noise in the measured link distances. We also report the number of distributed messages for convergence of our algorithm, and confirm that it achieves the same optimal objective function as the centralized version.

Appendix

Theorem I: If any one of the three inequality constraints of a triangle is bound, then the other two are feasible.
Proof: Assume without loss of generality that the first inequality constraint is bound:

$$d_{ij} + d_{jk} = d_{ik} \Rightarrow \begin{array}{l} -d_{ij} + d_{ik} = d_{jk} \\ -d_{jk} + d_{ik} = d_{ij} \end{array},$$

but $\begin{array}{l} d_{ij} + d_{ik} \geq -d_{ij} + d_{ik} \\ d_{jk} + d_{ik} \geq -d_{jk} + d_{ik} \end{array}$ since $d_{ij}, d_{jk} \geq 0$,

so $$\begin{array}{l} d_{ij} + d_{ik} \geq d_{jk} \\ d_{jk} + d_{ik} \geq d_{ij} \end{array} .\square \qquad (15)$$

Theorem II: If any two of the three inequality constraints of a triangle are bound, then the triangle is isosceles.
Proof: Assume without loss of generality that the first two inequality constraint are bound:

$$\begin{array}{l} d_{ij} + d_{jk} = d_{ik} \\ d_{ij} + d_{ik} = d_{jk} \end{array} \Rightarrow d_{jk} = d_{ik} \qquad (16)$$

by subtracting one equation from the other. □

References

[1] P. F. Gorder, "Sizing up smart dust," *IEEE J. Computing in Sciences and Engineering*, vol. 5, no. 6, pp. 6-9, Nov. 2003.
[2] P. Biswas and Y. Ye, "Semidefinite programming for ad hoc wireless sensor network localization," in *Proc. IEEE Conf. on Information Processing in Sensor Networks* pp. 46-54, April 2004.
[3] L. Doherty, K. S. J. Pister, and L. El Ghaoui, "Convex position estimation in wireless sensor networks," in *Proc. IEEE Conf. on Information Theory and Communications*, pp. 1655-1663, April 2001.
[4] J. Hightower and G. Borriello, "Location systems for ubiquitous computing," *IEEE Computer*, vol. 34, no. 8, pp. 57-66, Aug. 2001.
[5] P. Biswas, T.-C. Liang, K.-C. Toh, T.-C. Wang, and Y. Ye, "Semidefinite programming approaches for sensor network localization with noisy distance measurements," http://www.stanford.edu/~yyye/ieee_tase5.pdf.
[6] M. S. Bazaraa, J. J. Jarvis, and H. D. Sherali, *Linear Programming and Network Flows*. John Wiley & Sons, Inc., 1990.
[7] G. B. Dantzig, *Linear Programming and Extensions*. Princeton University Press, 1963.
[8] J. Werb and C. Lanzi, "Designing a positioning system for finding things and people indoors," *IEEE Spectrum*, vol. 35, no. 9, pp. 71-78, Sept. 1998.
[9] S. Capkun, M. Hamdi, and J.-P. Hubaux, "GPS-free positioning in mobile ad-hoc networks," in *Proc. IEEE Hawaii Conf. on System Sciences*, pp. 255-264, Jan. 2001.
[10] D. Niculescu and B. Nath, "Ad hoc positioning system (APS)," in *Proc. IEEE Conf. on Global Communications*, pp. 2926-2931, Nov. 2001.
[11] R. Shamir, "Probabilistic analysis in linear programming," *Statistical Sciences*, vol. 8, no. 1, 57-64, Feb. 1993.
[12] http://www.caba.org/standard/802154a.html.

Camillo Gentile (S'96-M'96-SM'00) received the B.S. and M.S. degrees from Drexel University, Philadelphia, PA and the Ph.D. degree from the Pennsylvania State University, College Park, PA, all in electrical engineering.

He has been a researcher in the Wireless Communication Technologies Group at the National Institute of Standards and Technology (NIST), Gaithersburg, MD since 2001. His interests include RF location systems, UWB channel modeling, and mobile ad-hoc networks. He is currently serving as the technical editor for the location annex of the IEEE 802.15.4a standard. In the past, he has worked in the areas of computer vision, pattern recognition, and neural networks.