Article

Cooperative Localization Using Distance Measurements for Mobile Nodes

Wenchao Li [1,*], Beth Jelfs, Allison Kealy [1], Xuezhi Wang [1] and Bill Moran [2]

[1] School of Science, RMIT University, Melbourne, Australia; {wenchao.li, beth.jelfs, xuezhi.wang, allison.kealy}@rmit.edu.au

[2] Department of Electrical and Electronic Engineering, University of Melbourne, Melbourne, Australia; wmoran@unimelb.edu.au

* Correspondence: wenchao.li@rmit.edu.au

Version February 23, 2021 submitted to Sensors

Abstract: This paper considers the 2D anchorless localization problem for sensor networks in GPS-denied environments. We present an efficient method, based on the multidimensional scaling (MDS) algorithm, to estimate the positions of the nodes in the network using measurements of the inter-node distances. The proposed method takes advantage of the mobility of the nodes to address the location ambiguity problem, i.e. rotation and flip ambiguity, that arises in the anchorless MDS algorithm. Knowledge of the displacement of the moving node is used to produce an analytical solution for the noise-free case. Then a least squares estimator is presented for the noisy scenario and the associated closed-form solution derived. Simulations show that the proposed algorithm accurately and efficiently estimates the locations of nodes, outperforming alternative methods.

Keywords: localization; sensor network; multidimensional scaling; position ambiguity

---

Key Notation

The following variables are used in this manuscript:

$d_{i,j}$	The distance between nodes i and j
$\mathbf{D}$	Euclidean distance matrix given the collection of locations $\boldsymbol{\mathcal{S}}$
$\mathbf{M}(\theta)$	Rotation matrix for given angle θ
$\boldsymbol{\mathcal{S}}_i = [x_i, y_i]^T$	True coordinates of the i^{th} node
$\boldsymbol{\mathcal{S}} = [\boldsymbol{\mathcal{S}}_1, \cdots, \boldsymbol{\mathcal{S}}_n]$	Collection of the true locations of all of the nodes
$\boldsymbol{\mathcal{S}}^*$	Output of multidimensional scaling given the distance matrix $\mathbf{D}$
$\boldsymbol{\mathcal{S}}^{**}$	Rotation of the output of multidimensional scaling, $\boldsymbol{\mathcal{S}}^*$, using rotation matrix $\mathbf{M}(\theta)$ with the angle, θ, obtained from the function $\theta_R(\boldsymbol{\mathcal{S}}^*, \Delta\boldsymbol{\mathcal{S}}', \mathbf{D}')$
$\theta_R(\boldsymbol{\mathcal{S}}^*, \Delta\boldsymbol{\mathcal{S}}', \mathbf{D}')$	Function to solve the possible rotation angle between the true positions and $\boldsymbol{\mathcal{S}}^*$ using the associated parameters
ψ	The diagonal elements of $\boldsymbol{\mathcal{S}}^T\boldsymbol{\mathcal{S}}$

The following notation is used to denote changes in the variables:

$(\cdot)'$	The parameter after the first movement
$(\cdot)''$	The parameter after the second movement
$\Delta(\cdot)'$	The change in the parameter after the first movement
$\Delta(\cdot)''$	The change in the parameter, relative to the original, after the second movement

For example $\boldsymbol{\mathcal{S}}'$ is the collection of true locations of all of the nodes incorporating the changes in position, i.e. $\boldsymbol{\mathcal{S}}' = \boldsymbol{\mathcal{S}} + \Delta\boldsymbol{\mathcal{S}}'$, similarly $\boldsymbol{\mathcal{S}}'' = \boldsymbol{\mathcal{S}} + \Delta\boldsymbol{\mathcal{S}}''$. It should be noted that $\boldsymbol{\mathcal{S}}'' = \boldsymbol{\mathcal{S}} + \Delta\boldsymbol{\mathcal{S}}''$ can be re-formulated as $\boldsymbol{\mathcal{S}}'' = \boldsymbol{\mathcal{S}}' + \overline{\Delta\boldsymbol{\mathcal{S}}''}$ with $\overline{\Delta\boldsymbol{\mathcal{S}}''} \triangleq \Delta\boldsymbol{\mathcal{S}}'' - \Delta\boldsymbol{\mathcal{S}}'$.

1. Introduction

Due to the increased availability of low-cost low-power sensors, smart sensors and multi-functional sensors, wireless sensor networks are becoming increasingly ubiquitous [1–4]. Wireless sensor networks are being utilized in a diverse array of tracking and monitoring applications from environmental [5] and health monitoring [6] to traffic [7] and border surveillance [8]. In many of these applications the nodes in the network are mobile and knowledge of their positions is a prerequisite for completing the task, and crucial for information sharing, data collection and scheduling [9]. For example, if the locations of the nodes are unknown or significantly incorrect, the data they have collected from surrounding environment, such as wildlife [10] or weather information [11], will be useless since the positional information is not available.

To provide the required positional information, localization algorithms estimate the locations of unknown nodes in the network using the positions of a known subset of the nodes. The most widely used localization techniques in the literature are distance-based localization algorithms such as trilateration, radio interference positioning system (RIPS) [12] and the Hop-Distance algorithm [13]. These algorithms estimate the inter-node distances and require anchors, that is nodes with known locations, to provide the locations of the remaining nodes. The location of the anchor nodes is accessed via global positioning system (GPS) or a priori information [12,14–18].

While, GPS is widely used in locating unknown nodes in many situations such as indoor, urban and forest environments, the positions of nodes are difficult to obtain from GPS [19–22]. In this case, the anchors are absent or the positions of the anchors are not available, and the above algorithms cannot be applied. This is widely regarded as the most significant challenge in the positioning and navigation field [23–25]. Therefore, there is an increasing need for anchorless localization of sensor networks for use in GPS-denied or contested environments. Using the movement of nodes and the inter-node communication, cooperative localization can be leveraged to solve this problem. This scenario arises in the field of robotics swarms, especially in the navigation and formation control of unmanned aerial vehicles swarms under a GPS-denied environment [20,25–27].

In practice, cooperative localization can be achieved by utilizing the inter-node distances [28]. However, relative localization only gives node positions which satisfy the distance constraint, meaning there could be ambiguity problems, i.e. ambiguity due to rotation and/or flip [29]. A widely used algorithm capable of tackling the anchorless localization problem is the multidimensional scaling (MDS) algorithm [30–35]. The aim of MDS is to represent the similarity (or dissimilarity) of high dimensional data in a lower dimensional map which describes the relative distances between pairs of objects (in this case sensor nodes). Like other relative localization methods MDS can also be subject to the ambiguity problem, hence, algorithms have been proposed to attempt to address this problem. In [36] a MDS-based algorithm using moving nodes is presented which constructs a cost function involving velocities and inter-node distances of all nodes at two consecutive time instants. In [37], a similar algorithm is proposed to solve the anchorless localization problem for nodes which can estimate the position via a nonlinear least square estimator, however in this case only one node is moving.

In this paper, we present an efficient algorithm for anchorless cooperative localization based on MDS. The algorithm mitigates the rotation and flip problems by taking advantage of the movement and inter-node communication of the mobile nodes. Unlike existing algorithms which operate in an iterative manner the proposed algorithm presents a closed-form solution which is computationally efficient. The algorithm is first derived in the noise-free case and the theoretical result is given. Then the noisy case is considered and the associated closed-form estimator is presented. The proposed algorithm is supported by a rigorous theoretical derivation which provides optimal parameters. Simulations results support the theoretical analysis and indicate the algorithm outperforms alternative methods.

This paper is organized as follows: Section 2 introduces the background to the MDS algorithm and the associated ambiguity problem. The theoretical solution to the ambiguity problem is then presented in Section 3. Based on the theoretical analysis, Section 4 describes the impact of noise and introduces the proposed closed-form estimator for determining positions in noisy scenarios. Section 5 presents simulations to validate the proposed algorithm and finally Section 6 concludes the paper.

2. Multidimensional Scaling Algorithm

In this section, we briefly introduce the MDS algorithm and the ambiguity problem in an ideal scenario. As stated previously the goal of MDS is to find a representation of the data that provides a low dimensional map (usually 2 or 3 dimensions) of the relative positions of the nodes based on their pairwise distances. If we consider the 2 dimensional case where there are n nodes with their true coordinates denoted by $\boldsymbol{\mathcal{S}}_i = [x_i, y_i]^T$ where $i = 1, \ldots, n$ and $n \geq 3$. Then in the noise-free case the distance between nodes i and j where $i, j = 1, \ldots, n$ and $i \neq j$ is given by $d_{i,j}$. If we assume in the ideal scenario that the nodes are able to measure the true distances between each other so that the pairwise distance between two nodes with coordinates $\boldsymbol{\mathcal{S}}_i$ and $\boldsymbol{\mathcal{S}}_j$ is given by

$$d_{i,j} = \|\boldsymbol{\mathcal{S}}_i - \boldsymbol{\mathcal{S}}_j\| = \sqrt{(\boldsymbol{\mathcal{S}}_i - \boldsymbol{\mathcal{S}}_j)^T(\boldsymbol{\mathcal{S}}_i - \boldsymbol{\mathcal{S}}_j)},$$

and furthermore, the squared distance $d^2_{i,j}$ can be written as

$$d^2_{i,j} = \boldsymbol{\mathcal{S}}_i^T\boldsymbol{\mathcal{S}}_i - 2\boldsymbol{\mathcal{S}}_i^T\boldsymbol{\mathcal{S}}_j + \boldsymbol{\mathcal{S}}_j^T\boldsymbol{\mathcal{S}}_j,$$

then we have the following symmetric Euclidean distance matrix:

$$\mathbf{D} = \begin{bmatrix} 0 & d^2_{1,2} & d^2_{1,3} & \cdots & d^2_{1,n} \\ d^2_{2,1} & 0 & d^2_{2,3} & \cdots & d^2_{2,n} \\ d^2_{3,1} & d^2_{3,2} & 0 & \cdots & d^2_{3,n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ d^2_{n,1} & d^2_{n,2} & d^2_{n,3} & \cdots & 0 \end{bmatrix}. \tag{1}$$

If $\boldsymbol{\mathcal{S}} = [\boldsymbol{\mathcal{S}}_1, \cdots, \boldsymbol{\mathcal{S}}_n]$ is the collection of all of the node coordinates and ψ is the diagonal elements of $\boldsymbol{\mathcal{S}}^T\boldsymbol{\mathcal{S}}$, i.e.

$$\psi = \mathrm{diag}(\boldsymbol{\mathcal{S}}^T\boldsymbol{\mathcal{S}}) = \left[\mathbf{s}_1^T\mathbf{s}_1, \ldots, \mathbf{s}_n^T\mathbf{s}_n\right]^T,$$

then we can rewrite $\mathbf{D}$ as

$$\mathbf{D} = \psi\mathbf{e}^T - 2\boldsymbol{\mathcal{S}}^T\boldsymbol{\mathcal{S}} + \mathbf{e}\psi^T, \tag{2}$$

where $\mathbf{e} = [1, \ldots, 1]^T$ is the vector of ones of length n. Using the centering operation $\mathbf{H} = \mathbf{I} - \mathbf{e}\mathbf{e}^T/n$ then we have

$$-\frac{1}{2}\mathbf{HDH} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^T.$$

where $\mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^T$ is the eigendecomposition of the symmetric matrix $-\frac{1}{2}\mathbf{HDH}$. Then we can recover $\boldsymbol{\mathcal{S}}$ (up to a translation and orthogonal transformation) via the following formula

$$\boldsymbol{\mathcal{S}}^* = \boldsymbol{\Lambda}^{\frac{1}{2}}\mathbf{U}^T, \tag{3}$$

MDS is an efficient algorithm for resolving the relative positions of the nodes [32]. But as is apparent from the above analysis, in the absence of anchor nodes, MDS can only give relative positions of the nodes, which can include rotation and flip ambiguity. In other words, the result of MDS maintains the relative inter-node distances, however, these calculated locations of the nodes may be flipped and/or rotated versions of the true positions of the nodes. Obviously when considering navigation of mobile nodes or formation control of the sensor network, incorrect positions of the nodes can lead to problems.

2.1. The Ambiguity Problem

To consider the ambiguity problem, we assume a set of n nodes in 2D Euclidean space. We fix a coordinate system which, without loss of generality, places the first node $\mathbf{s}_1$ at the origin: $\mathbf{s}_1 = [0,0]^T$. We recall that knowledge of the distances provides an ambiguity up to a universal Euclidean transformation of the nodes. This fixing of node 1 at the origin removes the shift from this Euclidean transformation. Accordingly, the solution of the MDS, $\boldsymbol{\mathcal{S}}^*$, is replaced by subtracting $\mathbf{s}_1^*$ from each column, so that, with some abuse of notation, $\mathbf{s}_1^* = [0,0]^T$, and the other $\mathbf{s}_j^*$ become $\mathbf{s}_j^* - \mathbf{s}_1^*$. Once the shift is removed, the remaining ambiguity devolves to a rotation and a reflection (flip). We give the definitions of rotation and flip ambiguities as follows.

Definition 1 (Rotation ambiguity)**.** *If there exists an angle $\theta \neq 2k\pi$, $k \in \mathbb{Z}$ such that*

$$\boldsymbol{\mathcal{S}} = \mathbf{M}(\theta)\boldsymbol{\mathcal{S}}^*, \tag{4}$$

where $\mathbf{M}(\theta)$ is rotation matrix with angle θ and defined by [38]

$$\mathbf{M}(\theta) = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix}, \tag{5}$$

then rotation ambiguity occurs.

Definition 2 (Flip ambiguity)**.** *Flip ambiguity occurs if*

$$\boldsymbol{\mathcal{S}} = \mathbf{F}\boldsymbol{\mathcal{S}}^* \quad with \quad \mathbf{F} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}. \tag{6}$$

Remark 1. *The matrix $\mathbf{F}$ can be defined equivalently by $\mathbf{F} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$. This definition can be obtained via simply rotating (6) by π. In the following analysis, we use the definition of $\mathbf{F}$ as in Definition 2.*

It can be seen that rotation ambiguity and flip ambiguity can occur simultaneously. If this is the case then the true positions can be represented by

$$\boldsymbol{\mathcal{S}} = \mathbf{M}(\theta)\mathbf{F}\boldsymbol{\mathcal{S}}^*. \tag{7}$$

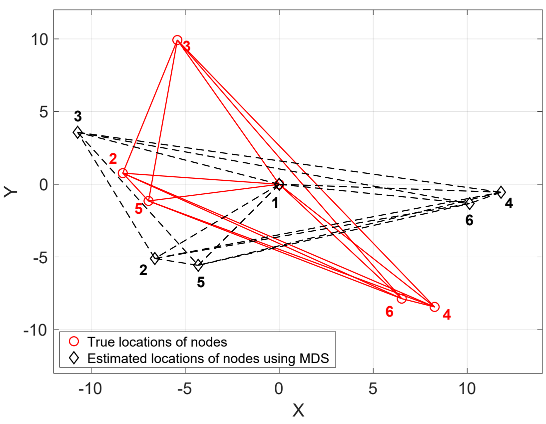
Examples of these two ambiguities are shown in Fig. 1.

3. Resolving Rotation and Flip Ambiguities

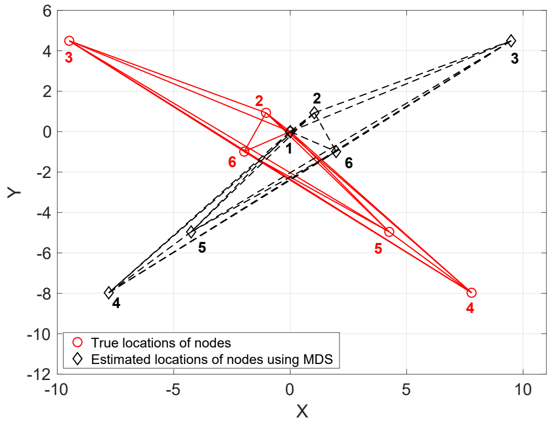
In this section, the rotation ambiguity is analyzed mathematically in the noise-free scenario and an analytical solution to the rotation and flip ambiguities presented.

3.1. Analysis of Rotation Ambiguity

Firstly, we assume that there exists only rotation ambiguity between $\boldsymbol{\mathcal{S}}$ and $\boldsymbol{\mathcal{S}}^*$, no flip ambiguity. This means that based on the coordination rotation principle [38], $\boldsymbol{\mathcal{S}}^*$ can be rotated to $\boldsymbol{\mathcal{S}}$ using an



(a)



(b)

Figure 1. Illustration of the positions of nodes calculated via MDS, where red circles are the desired position (randomly generated) and black diamonds are the output of MDS. (a) rotation ambiguity only; (b) rotation and flip ambiguities.

unknown angle θ via (4). Hence, the true (unknown) locations of the nodes $[x_i, y_i]^T$, for $i = 1, \ldots, n$, can be obtained by rotating $[x_i^*, y_i^*]^T$ by θ, i.e.

$$\boldsymbol{\mathcal{S}} = \mathbf{M}(\theta)\boldsymbol{\mathcal{S}}^*. \tag{8}$$

Therefore, to obtain the true locations of the nodes we need to know θ. To achieve this we allow the lead node to move, this movement can then be utilized to obtain information that can be used to recover the locations of other nodes. Consider mobile nodes equipped with inertial navigation systems, the lead node can move with known displacement and orientation; that is, we let the lead node move to a known position, i.e. $[\Delta x_1', \Delta y_1']^T$. Then the coordinates of all nodes after moving can be obtained as

$$\boldsymbol{\mathcal{S}}' = \boldsymbol{\mathcal{S}} + \Delta\boldsymbol{\mathcal{S}}' \tag{9}$$

$$\tag{10}$$

where the $i-$th column of $\Delta\boldsymbol{\mathcal{S}}'$ is given by

$$\Delta\boldsymbol{\mathcal{S}}_i' = \begin{cases} [\Delta x_1', \ \Delta y_1']^T & i = 1 \\ [0, \ 0]^T & i \neq 1 \end{cases}, \tag{11}$$

Accordingly, after moving the distance between the i^{th} and j^{th} nodes is

$$d_{i,j}' = \|\boldsymbol{\mathcal{S}}_i' - \boldsymbol{\mathcal{S}}_j'\|,$$

where $\boldsymbol{\mathcal{S}}_i'$ is the i^{th} column of $\boldsymbol{\mathcal{S}}'$. We can then update the distance matrix (2) with the entries $d_{i,j}'^{\,2}$ to give

$$\mathbf{D}' = \psi'\mathbf{e}^T - 2\boldsymbol{\mathcal{S}}'^T\boldsymbol{\mathcal{S}}' + \mathbf{e}\psi'^T, \tag{12}$$

where, by considering $\boldsymbol{\mathcal{S}}_1 = [0, 0]^T$ and $\Delta\boldsymbol{\mathcal{S}}_i' = [0, 0]^T$ for $i = 2, \ldots, n$,

$$\psi' = \text{diag}(\boldsymbol{\mathcal{S}}'^T\boldsymbol{\mathcal{S}}') = \text{diag}(\boldsymbol{\mathcal{S}}^T\boldsymbol{\mathcal{S}}) + \text{diag}(\Delta\boldsymbol{\mathcal{S}}'^T\Delta\boldsymbol{\mathcal{S}}') \triangleq \psi + \Delta\psi' \tag{13}$$

Substituting (9) and (13) into the distance matrix (12) we have

$$\begin{aligned}\mathbf{D}' &= (\psi + \Delta\psi')\mathbf{e}^T - 2(\boldsymbol{\mathcal{S}} + \Delta\boldsymbol{\mathcal{S}}')^T(\boldsymbol{\mathcal{S}} + \Delta\boldsymbol{\mathcal{S}}') + \mathbf{e}(\psi + \Delta\psi')^T \\ &= (\psi\mathbf{e}^T - 2\boldsymbol{\mathcal{S}}^T\boldsymbol{\mathcal{S}} + \mathbf{e}\psi^T) + (\Delta\psi'\mathbf{e}^T - 2\Delta\boldsymbol{\mathcal{S}}'^T\Delta\boldsymbol{\mathcal{S}}' + \mathbf{e}\Delta\psi'^T) - 2\Delta\boldsymbol{\mathcal{S}}'^T\boldsymbol{\mathcal{S}} - 2\boldsymbol{\mathcal{S}}^T\Delta\boldsymbol{\mathcal{S}}' \\ &= \mathbf{D} + \Delta\mathbf{D}' - 2\Delta\boldsymbol{\mathcal{S}}'^T\boldsymbol{\mathcal{S}} - 2\boldsymbol{\mathcal{S}}^T\Delta\boldsymbol{\mathcal{S}}',\end{aligned}$$

giving

$$\mathbf{0} = \mathbf{D} - \mathbf{D}' + \Delta\mathbf{D}' - 2\Delta\boldsymbol{\mathcal{S}}'^T\boldsymbol{\mathcal{S}} - 2\boldsymbol{\mathcal{S}}^T\Delta\boldsymbol{\mathcal{S}}', \tag{14}$$

where $\mathbf{0}$ is a zero matrix with dimensions $n \times n$. This equation describes the relationship between both the locations and the distance matrices pre and post the lead node moving; this information can be used to obtain the angle of rotation θ. If we break the analysis of (14) into three parts we have the following:

1. Since only the lead node's position is changed, then the term $\mathbf{D} - \mathbf{D}'$ in (14) becomes

$$\mathbf{D} - \mathbf{D}' = \begin{bmatrix} 0 & d_{1,2}^2 - d_{1,2}'^2 & \cdots & d_{1,n}^2 - d_{1,n}'^2 \\ d_{1,2}^2 - d_{1,2}'^2 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ d_{1,n}^2 - d_{1,n}'^2 & 0 & \cdots & 0 \end{bmatrix}. \tag{15}$$

2. In (14), $\Delta\mathbf{D}$ is the distance matrix between point $\Delta\boldsymbol{\mathcal{S}}_1$ and $n-1$ origin points $[0, 0]^T$, i.e.

$$\Delta\mathbf{D}' = \begin{bmatrix} 0 & \Delta x_1'^2 + \Delta y_1'^2 & \cdots & \Delta x_1'^2 + \Delta y_1'^2 \\ \Delta x_1'^2 + \Delta y_1'^2 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \Delta x_1'^2 + \Delta y_1'^2 & 0 & \cdots & 0 \end{bmatrix}. \tag{16}$$

3. For the term $-2\Delta\boldsymbol{\mathcal{S}}'^T\boldsymbol{\mathcal{S}} - 2\boldsymbol{\mathcal{S}}^T\Delta\boldsymbol{\mathcal{S}}'$ in (14), since $\boldsymbol{\mathcal{S}}^T\Delta\boldsymbol{\mathcal{S}}'$ can be calculated by

$$\boldsymbol{\mathcal{S}}^T\Delta\boldsymbol{\mathcal{S}}' = \begin{bmatrix} 0 & 0 \\ x_2 & y_2 \\ x_3 & y_3 \\ \vdots & \vdots \\ x_n & y_n \end{bmatrix} \begin{bmatrix} \Delta x_1' & 0 & 0 & \cdots & 0 \\ \Delta y_1' & 0 & 0 & \cdots & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ x_2\Delta x_1' + y_2\Delta y_1' & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ x_n\Delta x_1' + y_n\Delta y_1' & 0 & \ldots & 0 \end{bmatrix}, \tag{17}$$

therefore

$$-\left(2\Delta\boldsymbol{\mathcal{S}}'^T\boldsymbol{\mathcal{S}} + 2\boldsymbol{\mathcal{S}}^T\Delta\boldsymbol{\mathcal{S}}'\right) = -2\left(\left(\boldsymbol{\mathcal{S}}^T\Delta\boldsymbol{\mathcal{S}}'\right)^T + \boldsymbol{\mathcal{S}}^T\Delta\boldsymbol{\mathcal{S}}'\right). \tag{18}$$

Inserting the rotation (8) into $x_i\Delta x_1' + y_i\Delta y_1'$ in (17), for $i = 2, \ldots, n$, we have

$$\begin{aligned} x_i\Delta x_1' + y_i\Delta y_1' &= x_i^*\Delta x_1'\cos(\theta) + y_i^*\Delta x_1'\sin(\theta) + y_i^*\Delta y_1'\cos(\theta) - x_i^*\Delta y_1'\sin(\theta) \\ &= (x_i^*\Delta x_1' + y_i^*\Delta y_1')\cos(\theta) + (y_i^*\Delta x_1' - x_i^*\Delta y_1')\sin(\theta). \end{aligned} \tag{19}$$

Using (19) allows us to express (18) in a way that is independent of $[x_i,\ y_i]^T$.

If we combine (15)–(19) then (14) becomes

$$\mathbf{D} - \mathbf{D}' + \Delta\mathbf{D}' - 2\Delta\boldsymbol{\mathcal{S}}'^T\boldsymbol{\mathcal{S}} - 2\boldsymbol{\mathcal{S}}^T\Delta\boldsymbol{\mathcal{S}}' = \begin{bmatrix} 0 & f_2(\theta) & \cdots & f_n(\theta) \\ f_2(\theta) & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ f_n(\theta) & 0 & \dots & 0 \end{bmatrix} = \mathbf{0}, \tag{20}$$

where

$$f_i(\theta) = a_i + b_i\cos(\theta) + c_i\sin(\theta), \qquad i = 2, \dots, n \tag{21}$$

with coefficients

$$a_i = d_{1,i}^2 - {d'}_{1,i}^2 + {\Delta x'_1}^2 + {\Delta y'_1}^2 \tag{22}$$
$$b_i = -2(x_i^*\Delta x'_1 + y_i^*\Delta y'_1) \tag{23}$$
$$c_i = 2(x_i^*\Delta y'_1 - y_i^*\Delta x'_1) \tag{24}$$

Equation (20) is equivalent to following system of equations:

$$\begin{cases} a_2 + b_2\cos(\theta) + c_2\sin(\theta) = & 0 \\ \quad\vdots & \vdots \\ a_n + b_n\cos(\theta) + c_n\sin(\theta) = & 0 \end{cases}, \tag{25}$$

Finally, the solution to (25) is the angle to resolve the rotation ambiguity. Importantly, this solution can be shown to be unique when $n \geq 3$. If we consider the case of $n = 3$ then (25) can be expressed as

$$\begin{cases} \sin(\theta) = \dfrac{a_3b_2 - a_2b_3}{b_3c_2 - b_2c_3} \triangleq W_1 \\ \cos(\theta) = \dfrac{a_2c_3 - a_3c_2}{b_3c_2 - b_2c_3} \triangleq W_2 \end{cases}. \tag{26}$$

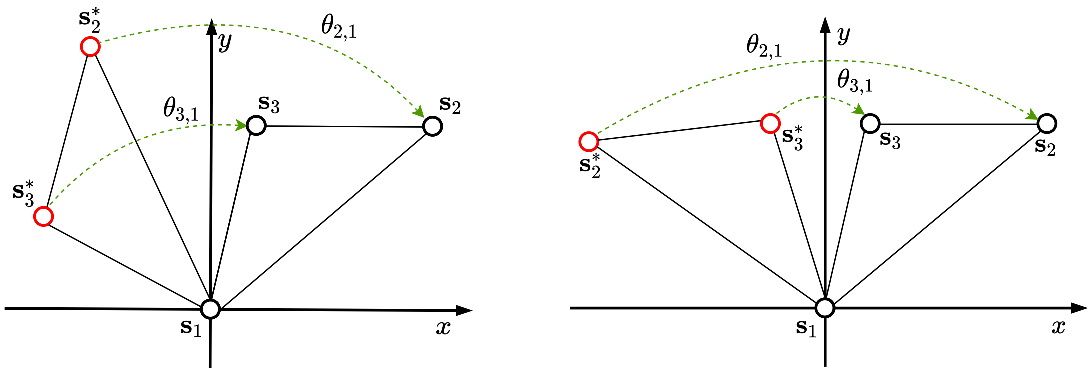
Obviously, given W_1 and W_2, (26) has a unique solution to θ within $[-\pi, \pi)$. Similarly, it is straightforward to show that, when $n \geq 3$, (25) has a unique solution within $[-\pi, \pi)$.

3.2. Analysis of Rotation and Flip Ambiguities

Having obtained a unique solution to the rotation angle when only rotation ambiguity is present, in this section, we present an analytical solution based on the analysis in Section 3.1 for when rotation and flip ambiguities occur simultaneously. The key idea behind this method is again to use the mobility of the lead node to acquire extra information in order to detect flip of the initial MDS localization result.

Firstly, in order to be able to detect flip ambiguity, we assume that there exist three non-collinear nodes. Next we note that the i^{th} equation in (25), has the following solutions $\theta_{i,1}$ and $\theta_{i,2}$:

$$\theta_{i,1,2} = \operatorname{atan2}\left(a_ib_i \pm |c_i|\sqrt{b_i^2 + c_i^2 - a_i^2}, \quad a_ic_i \mp \frac{b_i}{c_i}|c_i|\sqrt{b_i^2 + c_i^2 - a_i^2}\right), \tag{27}$$



(a) **(b)**

Figure 2. Illustration of angle $\theta_{i,1}$ given in (28) using three nodes with true locations $\mathbf{s}_i$ and solved ambiguous locations $\mathbf{s}_i^*$. (a) rotation ambiguity and $\theta_{2,1} = \theta_{3,1}$; (b) flip ambiguity and $\theta_{2,1} \neq \theta_{3,1}$

where atan2$(\cdot,\cdot) \in [-\pi, \pi)$ is the 2-argument arctangent. Therefore, the solution to (27) which is common to all values of i, $\forall i = 2, \ldots, n$ is the unique solution to (25). It can also be shown that $\theta_{i,1,2} \in [-\pi, \pi)$ given in (27) can be rewritten and rearranged into a concise form

$$\theta_{i,1} = g\Big(\text{atan2}(y_i^*, x_i^*) - \text{atan2}(y_i, x_i)\Big) \tag{28}$$

$$\theta_{i,2} = g\Big(\theta_{i,1} + 2\Theta_i\Big) \tag{29}$$

where $\Theta_i = \text{atan2}\,(y_i, x_i) + \text{atan2}\,(\Delta x_1', \Delta y_1') - \frac{\pi}{2}$ and the function $g(t) = t - 2\pi \left\lfloor \frac{t}{2\pi} + \frac{1}{2} \right\rfloor$ can wrap any arbitrary angle t in radians into range $[-\pi, \pi)$. The full derivation of these equations is given in Appendix A.

The angles $\theta_{i,1}$ in (28) represent the angles between vectors $\overrightarrow{\boldsymbol{\mathcal{S}}_1\boldsymbol{\mathcal{S}}_i^*}$ and $\overrightarrow{\boldsymbol{\mathcal{S}}_1\boldsymbol{\mathcal{S}}_i}$ for $i = 2, \ldots, n$, as shown in Fig. 2, and play an important role in the ability to detect flip ambiguity. As shown in the previous section, when there is only rotation ambiguity, because of the uniqueness of solution of (25), we have $\theta_{i,1} = \theta_{j,1}$, $\forall i, j = 2, \ldots, n$. Whereas, the angles $\theta_{i,2}$ in (29) are the summation of $\theta_{i,1}$ and the angle induced by $\Delta x_1'$ and $\Delta x_2'$. Obviously, if there exist three non-collinear nodes, $\Theta_i \neq \Theta_j$, $\forall i, j = 2, \ldots, n$ and $i \neq j$ and therefore from (29) we have $\theta_{i,2} \neq \theta_{j,2}$, $\forall i, j = 2, \ldots, n$. In contrast, it can be shown that flip ambiguity exists if and only if $\theta_{i,1} \neq \theta_{j,1}$, $\forall i, j = 2, \ldots, n$ and $i \neq j$. To illustrate why this is the case we give the counter example, assuming without loss of generality, that $n = 3$, $\theta_{2,1} = \theta_{3,1}$ and flip ambiguity exists. As shown in Fig. 2a, $\theta_{2,1} = \theta_{3,1}$ implies that $\boldsymbol{\mathcal{S}}_i^*$, $i = 2, 3$, can be rotated simultaneously to the true positions $\boldsymbol{\mathcal{S}}_i$ via either $\theta_{2,1}$ or $\theta_{3,1}$. Hence, this contradicts the assumption of the existence of flip ambiguity.

Furthermore, extending to the case where $n > 3$, still assuming that $\theta_{2,1} = \theta_{3,1}$ and flip ambiguity exists. We know that, for $i = 2, 3$, $\boldsymbol{\mathcal{S}}_i^*$ can be rotated simultaneously to the true positions $\boldsymbol{\mathcal{S}}_i$ via $\theta_{2,1}$ (or $\theta_{3,1}$). Since there exist three non-collinear nodes, then three nodes with correct positions are sufficient to guarantee the localization of the whole network [39]. In this case, $\mathbf{s}_1$, $\mathbf{s}_2$ and $\mathbf{s}_3$ can be found exactly from $\theta_{2,1}$ (or $\theta_{3,1}$). Therefore, $\boldsymbol{\mathcal{S}}_i$, for $i = 4, \ldots, n$, must be solvable via rotating $\boldsymbol{\mathcal{S}}_i^*$ by angle $\theta_{2,1}$ (or $\theta_{3,1}$) and, as a result, $\theta_{i,1} = \theta_{2,1} = \theta_{3,1}$. This again contradicts the assumption of existence of flip ambiguity. On the other hand, if $\theta_{i,1} \neq \theta_{j,1}$, $\forall i, j = 2, \cdots, n$ and $i \neq j$, then it is obvious that there exists flip ambiguity since that $\mathbf{s}_i^*$ cannot be rotated to $\boldsymbol{\mathcal{S}}_i$ simultaneously. As a conclusion, there exists flip ambiguity if and only if $\theta_{i,1} \neq \theta_{j,1}$, $\forall i, j = 2, \cdots, n$ and $i \neq j$.

In what follows we assume that there exist three non-collinear nodes and based on the above analysis, we make the following conclusion:

$$\forall i,j = 2,\ldots,n \text{ and } i \neq j \begin{cases} \theta_{i,1} = \theta_{j,1}(\text{equivalently } \theta_{i,2} \neq \theta_{j,2}), & \text{If no flip ambiguity exists.} \\ \theta_{i,1} \neq \theta_{j,1}(\text{equivalently } \theta_{i,2} = \theta_{j,2}), & \text{If flip ambiguity exists.} \end{cases} \tag{30}$$

Though we cannot use (28), (29) and (30) directly to determine the existence of flip as they contain unknown true positions, those results are crucial in deriving the estimator for locations in noisy scenario.

In reality, the solution to (25) is computed using $\mathcal{S}^*$, $\Delta\mathcal{S}'$ and $\mathbf{D}'$. We denote this unique solution by $\theta_R\left(\mathcal{S}^*, \Delta\mathcal{S}', \mathbf{D}'\right)$. The same notation $\theta_R(\cdot,\cdot,\cdot)$ is used to denote a rotation angle calculated with different variables, nonetheless whatever the variables used the method is the same as described above. If we denote the positions calculated using the rotation angle $\theta_R\left(\mathcal{S}^*, \Delta\mathcal{S}', \mathbf{D}'\right)$ as $\mathcal{S}^{**}$ we have the following result

$$\mathcal{S}^{**} = \mathbf{M}\left(\theta_R(\mathcal{S}^*, \Delta\mathcal{S}', \mathbf{D}')\right)\mathcal{S}^* = \begin{bmatrix} x_1^{**}, & x_2^{**}, & \cdots, x_n^{**} \\ y_1^{**}, & y_2^{**}, & \cdots, y_n^{**} \end{bmatrix}. \tag{31}$$

If the lead node then moves to a second position $[\Delta x_1'', \Delta y_1'']$ giving a matrix formed by true positions $\mathcal{S}'' = \mathcal{S} + \Delta\mathcal{S}''$, where

$$\Delta\mathcal{S}'' = \left[\Delta\mathcal{S}_1'', \Delta\mathcal{S}_2'', \ldots, \Delta\mathcal{S}_n''\right] = \begin{bmatrix} \Delta x_1'' & 0 & \cdots & 0 \\ \Delta y_1'' & 0 & \cdots & 0 \end{bmatrix}. \tag{32}$$

After obtaining a new distance matrix $\mathbf{D}''$ at position $\mathcal{S}''$, we can solve $\theta_R(\mathcal{S}^{**}, \Delta\mathcal{S}'', \mathbf{D}'')$. If $\theta_R(\mathcal{S}^{**}, \Delta\mathcal{S}'', \mathbf{D}'') = 0$, then there is no flip ambiguity and the true positions are $\mathcal{S} = \mathcal{S}^{**}$; otherwise we move to the process of resolving the flip ambiguity. For this, according to Definition 2, all values in $\mathcal{S}^*$ along the $x-$axis are required to be flipped to obtain $\mathbf{F}\mathcal{S}^*$. Then we only need to calculate the rotation angle by using $\theta_R(\mathbf{F}\mathcal{S}^*, \Delta\mathcal{S}', \mathbf{D}')$. It should be noted that, since $\Delta\mathcal{S}'$ and $\mathbf{D}'$ are fixed, it is unnecessary to take any new measurements and the true position $\mathcal{S}$ can be resolved by

$$\mathcal{S} = \mathbf{M}\left(\theta_R(\mathbf{F}\mathcal{S}^*, \Delta\mathcal{S}', \mathbf{D}')\right)\mathbf{F}\mathcal{S}^*. \tag{33}$$

4. Proposed Algorithm Robust to Ambiguity and Noise

The analysis in the previous section assumes ideal measurements, however in practice, the measurements are corrupted by noise which can have a significant impact on the localization performance. If we consider the distances between two nodes, in the noise-free case the distance between the i^{th} and j^{th} nodes is the same regardless of which node it is measured from, i.e. $d_{i,j} = d_{j,i}$. When noise is introduced this is no longer the case, if we denote the measured distances as $\bar{d}_{i,j}$ and $\bar{d}_{j,i}$, then $\bar{d}_{i,j} \neq \bar{d}_{j,i}$ resulting in uncertainty in our estimates of the distances. In general, the measured distance between the i^{th} and j^{th} nodes can be modeled by

$$\bar{d}_{i,j} = d_{i,j} + \omega_{i,j}, \qquad i,j = 1,\ldots,n, \quad i \neq j, \tag{34}$$

where $d_{i,j}$ is the true distance and $\omega_{i,j}$ is the measurement noise.

Accordingly, the noisy Euclidean distance matrix (EDM) can be written into

$$\bar{\mathbf{D}} = \begin{bmatrix} 0 & \bar{d}_{1,2}^2 & \bar{d}_{1,3}^2 & \cdots & \bar{d}_{1,n}^2 \\ \bar{d}_{2,1}^2 & 0 & \bar{d}_{2,3}^2 & \cdots & \bar{d}_{2,n}^2 \\ \bar{d}_{3,1}^2 & \bar{d}_{3,2}^2 & 0 & \cdots & \bar{d}_{3,n}^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \bar{d}_{n,1}^2 & \bar{d}_{n,2}^2 & \bar{d}_{n,3}^2 & \cdots & 0 \end{bmatrix}. \tag{35}$$

As we know that $d_{i,j} = d_{j,i}$ when the noise is absent, therefore, in order to obtain a symmetric EDM in noisy case, we can use

$$\hat{d}_{i,j} = \hat{d}_{j,i} = \mathbb{E}(\bar{d}_{i,j}, \bar{d}_{j,i}) \tag{36}$$

where $\mathbb{E}(\bar{d}_{i,j}, \bar{d}_{j,i})$ is an estimator for the distance between i-th and j-th nodes using $\bar{d}_{i,j}$ and $\bar{d}_{j,i}$. In practice, this estimator is designed by using the knowledge of noise distribution. For example, when the noise in $\bar{d}_{i,j}$ and $\bar{d}_{j,i}$ are assumed to comprise independent and identically distributed (i.i.d.) Normal distributions, then we have $\hat{d}_{i,j} = \hat{d}_{j,i} = \frac{1}{2}\left(\bar{d}_{i,j} + \bar{d}_{j,i}\right)$ [40]. Since the study of the estimator $\mathbb{E}(\bar{d}_{i,j}, \bar{d}_{j,i})$ and estimation of the EDM is outside the scope of this article, we refer the interested reader to [40,41] for more information, including completing and estimating an EDM.

Accordingly, the estimated symmetric EDM $\widehat{\mathbf{D}}$ can be obtained using $\hat{d}_{i,j}$. Similarly, we can obtain $\widehat{\mathbf{D}}'$ by using the new position of $\boldsymbol{\mathcal{S}}' = \boldsymbol{\mathcal{S}} + \Delta\boldsymbol{\mathcal{S}}'$ as described in (11).

In (25), in the noise-free case, the solution is unique and easy to find. However, in the noisy environment, the theoretical unique solution to (25) is not guaranteed. Therefore, a key issue in estimating θ is to find a value which satisfies a certain objective, i.e.

$$\widehat{\theta}_R\left(\boldsymbol{\mathcal{S}}^*, \Delta\boldsymbol{\mathcal{S}}', \mathbf{D}'\right) = \arg\min_{\theta\in[-\pi,\pi)} Obj\left(\theta;\ \boldsymbol{\mathcal{S}}^*, \Delta\boldsymbol{\mathcal{S}}', \widehat{\mathbf{D}}'\right), \tag{37}$$

where $Obj\left(\theta;\ \boldsymbol{\mathcal{S}}^*, \Delta\boldsymbol{\mathcal{S}}', \mathbf{D}'\right)$ is an objective function of θ given $\boldsymbol{\mathcal{S}}^*$, $\Delta\boldsymbol{\mathcal{S}}'$ and $\widehat{\mathbf{D}}'$. In general, the least square estimator provides a good choice of objective function as it is a well defined computationally efficient estimator. The objective function based on the least square estimator is given by

$$Obj\left(\theta;\ \boldsymbol{\mathcal{S}}^*, \Delta\boldsymbol{\mathcal{S}}', \mathbf{D}'\right) = \sum_{i=1}^{n}\left(a_i + b_i\cos(\theta) + c_i\sin(\theta)\right)^2. \tag{38}$$

Solutions to (38) can be obtained by taking the derivative of the objective function with respect to θ and equating to zero, from this we obtain a quartic equation (see Appendix B for details) which has at most 4 real roots giving the corresponding collection of angles as

$$\boldsymbol{\theta}_\lambda = \left\{\arcsin(\lambda_1), g(\pi - \arcsin(\lambda_1)), \ldots, \arcsin(\lambda_m), g(\pi - \arcsin(\lambda_m))\right\} \in [-\pi, \pi),$$

where $\lambda_j \in [-1, 1]$, $j = 1, \ldots, m$, and m is the number of solutions, such that $1 \leq m \leq 4$. Then (38) becomes

$$\widehat{\theta}_R\left(\boldsymbol{\mathcal{S}}^*, \Delta\boldsymbol{\mathcal{S}}', \widehat{\mathbf{D}}'\right) = \arg\min_{\theta\in\boldsymbol{\theta}_\lambda} Obj\left(\theta;\ \boldsymbol{\mathcal{S}}^*, \Delta\boldsymbol{\mathcal{S}}', \widehat{\mathbf{D}}'\right), \tag{39}$$

and accordingly, by (31)

$$\boldsymbol{\mathcal{S}}^{**} = \mathbf{M}\left(\widehat{\theta}_R\left(\boldsymbol{\mathcal{S}}^*, \Delta\boldsymbol{\mathcal{S}}', \widehat{\mathbf{D}}'\right)\right)\boldsymbol{\mathcal{S}}^*. \tag{40}$$

In order to detect flip ambiguity, we can follow the same method as described in Section 3, i.e. let the lead node move to another position $[\Delta x_1'', \Delta y_1'']^T$ and solve $\widehat{\theta}_R\left(\boldsymbol{\mathcal{S}}^{**}, \Delta\boldsymbol{\mathcal{S}}'', \widehat{\mathbf{D}}''\right)$ via (39) using the estimated distance matrix $\widehat{\mathbf{D}}''$ obtained at the new position $\boldsymbol{\mathcal{S}}'' = \boldsymbol{\mathcal{S}} + \Delta\boldsymbol{\mathcal{S}}''$. However, becasue of the presence of noise $\widehat{\theta}_R\left(\boldsymbol{\mathcal{S}}^{**}, \Delta\boldsymbol{\mathcal{S}}'', \widehat{\mathbf{D}}''\right)$ is not necessarily equal to 0 when there is no flip ambiguity. To handle this, we need to create a detector for the flip. For this, we use a straightforward threshold detector:

$$\begin{cases} \text{no flip,} & \text{if } \left|\widehat{\theta}_R\left(\boldsymbol{\mathcal{S}}^{**}, \Delta\boldsymbol{\mathcal{S}}'', \widehat{\mathbf{D}}''\right)\right| \leq |l| \\ \text{flip,} & \text{if } \left|\widehat{\theta}_R\left(\boldsymbol{\mathcal{S}}^{**}, \Delta\boldsymbol{\mathcal{S}}'', \widehat{\mathbf{D}}''\right)\right| > |l|, \end{cases} \tag{41}$$

The following proposition, the proof of which is given in Appendix C, allows us to efficiently find an optimal threshold in (41).

Proposition 1. *Under the noise-free case, if there exists flip ambiguity, then*

$$\theta_R\left(\boldsymbol{\mathcal{S}}^{**}, \Delta\boldsymbol{\mathcal{S}}'', \mathbf{D}''\right) = g\Big(-2\,\mathrm{atan2}(\Delta x_1', \Delta y_1') + 2\,\mathrm{atan2}(\Delta x_1'', \Delta y_1'')\Big), \tag{42}$$

where $g(t) = t - 2\pi\left\lfloor \frac{t}{2\pi} + \frac{1}{2} \right\rfloor$.

From Prop. 1, under the noise-free case, we know the value of $\theta_R\left(\boldsymbol{\mathcal{S}}^{**}, \Delta\boldsymbol{\mathcal{S}}'', \mathbf{D}''\right)$ when a flip occurs, therefore, in the noisy case, the optimal threshold is

$$l = \frac{1}{2} g\Big(-2\,\mathrm{atan2}(\Delta x_1', \Delta y_1') + 2\,\mathrm{atan2}(\Delta x_1'', \Delta y_1'')\Big).$$

As a conclusion, the positions of nodes can estimated using the following formula:

$$\widehat{\boldsymbol{\mathcal{S}}} = \begin{cases} \mathbf{M}\left(\widehat{\theta}_R\left(\boldsymbol{\mathcal{S}}^{*}, \Delta\boldsymbol{\mathcal{S}}', \widehat{\mathbf{D}}'\right)\right)\boldsymbol{\mathcal{S}}^{*}, & \text{if } \left|\widehat{\theta}_R\left(\boldsymbol{\mathcal{S}}^{**}, \Delta\boldsymbol{\mathcal{S}}'', \widehat{\mathbf{D}}''\right)\right| \le |l| \\ \mathbf{M}\left(\widehat{\theta}_R\left(\mathbf{F}\boldsymbol{\mathcal{S}}^{*}, \Delta\boldsymbol{\mathcal{S}}', \widehat{\mathbf{D}}'\right)\right)\mathbf{F}\boldsymbol{\mathcal{S}}^{*}, & \text{if } \left|\widehat{\theta}_R\left(\boldsymbol{\mathcal{S}}^{**}, \Delta\boldsymbol{\mathcal{S}}'', \widehat{\mathbf{D}}''\right)\right| > |l| \end{cases} \tag{43}$$

The algorithm to estimate locations of nodes with noisy measurements is summarized in Algorithm 1.

Algorithm 1: Algorithm to estimate locations of mobile nodes.

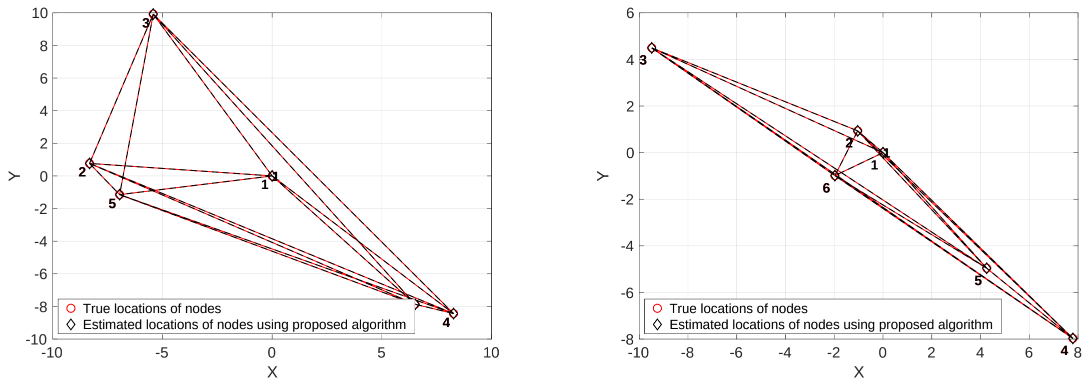
Result: Estimated locations of nodes
Initialization;
begin /* First step */
  Collect $\bar{d}_{i,j}$ for $i, j = 1, \cdots, n$ and estimate $\widehat{\mathbf{D}}$ via (36);
  Calculate $\boldsymbol{\mathcal{S}}^*$ via MDS algorithm;
end
begin /* Second step */
  Let node 1 move to position $[\Delta x_1', \Delta y_1']^T$;
  Collect $d_{i,j}'$ for $i, j = 1, \ldots, n$ at the new position and estimate $\widehat{\mathbf{D}}'$;
  Calculate $\widehat{\theta}_R\left(\boldsymbol{\mathcal{S}}^{*}, \Delta\boldsymbol{\mathcal{S}}', \widehat{\mathbf{D}}'\right)$ according to (39);
end
begin /* Third step */
  Let node 1 move to position $[\Delta x_1'', \Delta y_1'']^T$;
  Collect $d_{i,j}''$ for $i, j = 1, \ldots, n$ at the new position and estimate $\widehat{\mathbf{D}}''$;
  Calculate $\widehat{\theta}_R\left(\boldsymbol{\mathcal{S}}^{**}, \Delta\boldsymbol{\mathcal{S}}'', \widehat{\mathbf{D}}''\right)$ according to (39) and (40) using variables $\boldsymbol{\mathcal{S}}^{**}$, $\Delta\boldsymbol{\mathcal{S}}''$ and $\widehat{\mathbf{D}}''$;
  Use (43) to estimate the locations, i.e. $\widehat{\boldsymbol{\mathcal{S}}}$;
end
if $\left|\widehat{\theta}_R\left(\boldsymbol{\mathcal{S}}^{**}, \Delta\boldsymbol{\mathcal{S}}'', \mathbf{D}''\right)\right| < |l|$ **then**
  Determine locations of nodes via $\mathbf{F}$ and (22)–(24);
else
  Determine locations of nodes via (31);
end

---

5. Simulations

To test the performance of the proposed algorithm, networks with randomly generated nodes were used. In all of the following simulations the position of the first node is fixed to $[0, 0]$ and the positions of the other nodes are uniformly generated within $[-20, 20]$.

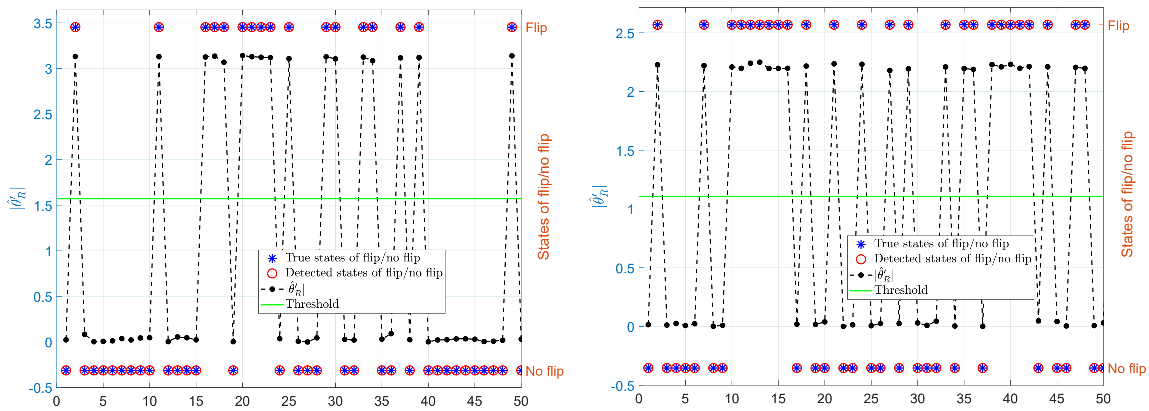
To validate the proposed approach can address the ambiguity problem we use the same noise-free scenarios as given in Fig. 1 the results of which are shown in Fig. 3. Unlike the results obtained from the MDS alone, shown in Fig. 1, when we compare the locations of the true and estimated nodes in Fig. 3, we can see that using the method presented in Section 3, the rotation and flip ambiguities are solved correctly and therefore the positions of the nodes are successfully recovered.



(a) **(b)**

Figure 3. Estimation of the positions of nodes in noise-free scenarios. (a) Rotation only; (b) Rotation and flip.

Next we consider the noisy scenario, as discussed in Section 4, in the noisy scenario the detection of whether flip has occurred is no longer straightforward and requires an appropriate choice of threshold. Firstly, to demonstrate the ability of the proposed algorithm to detect flip or no flip states we tested two different network configurations: one with the number of nodes $n = 6$ and the other with $n = 10$. The noise is assumed to be Gaussian distributed with mean 0 and standard deviation, σ, of 0.01. For each of the configurations, the outcomes of 50 simulations are shown in Fig. 4. The results show that the proposed algorithm can correctly detect flip ambiguity in the noisy scenario.



(a) **(b)**

Figure 4. Illustration of the detection of flip/no flip states and the corresponding values of $\left|\widehat{\theta}_R\left(\boldsymbol{\mathcal{S}}^{**}, \Delta\boldsymbol{\mathcal{S}}'', \widehat{\mathbf{D}}''\right)\right|$. The left y-axes represent the values of $\left|\widehat{\theta}_R\left(\boldsymbol{\mathcal{S}}^{**}, \Delta\boldsymbol{\mathcal{S}}'', \widehat{\mathbf{D}}''\right)\right|$ and the threshold while the right y-axes show the true and detected states of flip/no flip with $\sigma = 0.01$. (a) The number of nodes $n = 6$, $[\Delta x_1', \Delta y_1'] = [1, 0]$ and $[\Delta x_1'', \Delta y_1''] = [0, 1]$; (b) The number of nodes $n = 10$, $[\Delta x_1', \Delta y_1'] = [1, 0.5]$ and $[\Delta x_1'', \Delta y_1''] = [0, 1]$.

Having established that the proposed algorithm can successfully deal with the ambiguity problem we now consider the performance of the algorithm in terms of the accuracy of the localization. For the sake of simplicity, we define the noise level, standard variance of the noise, as $\delta = -20\log_{10}\sigma$ with δ chosen to range from 20 to 50 in steps of 5 and σ calculated accordingly. For each noise level the Root-Mean-Square Error (RMSE) from 2000 simulations is used to evaluate the performance of the localization. The RMSE is defined by

$$\text{RMSE} = \frac{1}{n}\sum_{i=1}^{n}\sqrt{\sum_{j=1}^{m}\frac{(\hat{x}_{i,j}-x_{i,j})^2+(\hat{y}_{i,j}-y_{i,j})^2}{m}} \tag{44}$$

where $(x_{i,j}, y_{i,j})$ and $(\hat{x}_{i,j}, \hat{y}_{i,j})$ are the true and estimated location of i-th, $i = 1, \cdots, n$, node in j-th, $j = 1, \cdots, m$, Monte Carlo simulation. It is assumed that $(x_{i,j}, y_{i,j})$ is generated within region $[-100, 100] \times [-100, 100]$ uniformly in each Monte Carlo simulation.

To provide a comparison with the proposed algorithm we implemented the nonlinear least squares (NLS) estimator presented in [37], solving the NLS using an optimization method. Additionally, the proposed algorithm essentially takes advantage of the mobility of the node to create virtual anchors for localizing the unknown nodes. As a result, the alignment method of relative locations proposed in [42,43] can be potentially applied in this scenario. As a comparison, the performance of this conventional method are also given.

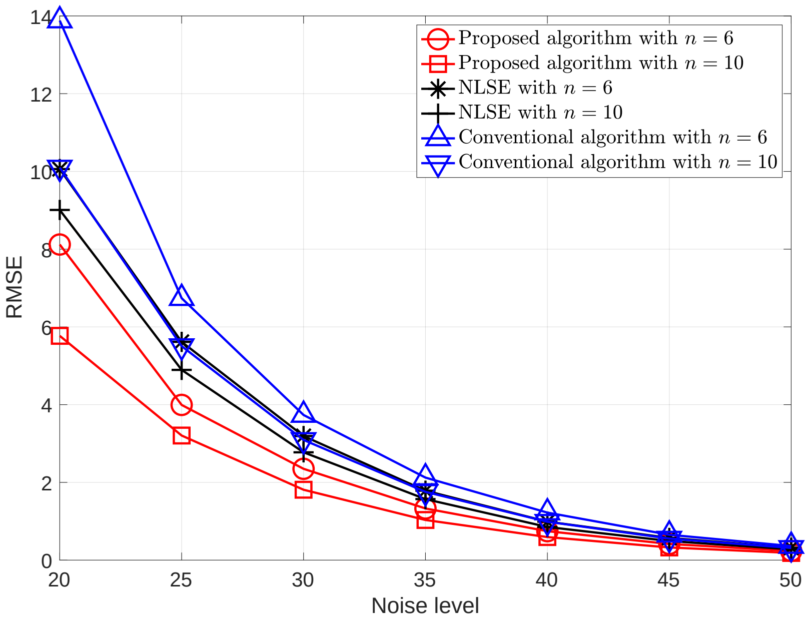


Figure 5. The RMSE for the proposed algorithm, the NLS estimator [37], NLSE, and conventional algorithm proposed in [42,43] for different numbers of nodes and different noise levels.

From the simulation results shown in Fig. 5, it can be seen that the proposed algorithm has better performance in terms of the RMSE for different noise levels and numbers of nodes than the NLS estimator with the conventional method. The localization error are arise from two effects: 1. MDS localization error; 2. mis-alignment error, i.e. the error from inaccurately aligning the positions. Additionally, it should also be noted that the algorithm proposed in this paper can estimate the positions of all of the nodes simultaneously, which cannot be achieved using the algorithm presented in [37]. As an indication of the computational efficiency of the proposed algorithm, our simulations indicate a ratio of required CPU time of the proposed algorithm relative to NLSE is Proposed algorithm : NLSE = 1 : 8.83.

6. Conclusion

In this paper, we have presented an efficient cooperative localization algorithm based on MDS. The algorithm provides a practical solution for anchorless localization of mobile nodes using noisy measurements. Unlike traditional MDS algorithms which suffer from an ambiguity problem, the algorithm presented here can solve the flip and rotation ambiguities and accurately estimate the positions of nodes in 2D space. Simulation results demonstrate the accuracy of the algorithm, showing that it outperforms alternative methods. At the same time the proposed algorithm provides greater efficiency than alternative solutions which operate in an iterative manner by providing a closed-form solution. We point out that the main limitations of this algorithm are twofold. Firstly, as mentioned above, this algorithm has been developed in a 2D scenario, which limits its application in more complex situations. Though one may follow a similar procedure to derive the corresponding algorithm for a more general case i.e. 3D space, this is non-trivial as the geometry of the network has more degrees of freedom in the 3D space. Secondly, and in similar vein to other algorithms, the algorithm proposed here requires inertial navigation to provide the displacement of the moving node. As a result, a deeper analysis of the error arising from the inertial navigation system should be taken into account in improving this algorithm. These issues will be addressed in future work.

Author Contributions: Conceptualization, Wenchao Li, Allison Kealy and Bill Moran; Data curation, Wenchao Li; Formal analysis, Wenchao Li; Investigation, Beth Jelfs and Bill Moran; Methodology, Wenchao Li; Project administration, Allison Kealy; Software, Wenchao Li; Supervision, Allison Kealy; Validation, Wenchao Li, Beth Jelfs and Bill Moran; Visualization, Wenchao Li; Writing–original draft, Wenchao Li; Writing–review and editing, Beth Jelfs, Allison Kealy, Xuezhi Wang and Bill Moran. All authors have read and agreed to the published version of the manuscript

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

Proposition A1. *Function $g(t) = t - 2\pi \left\lfloor \frac{t}{2\pi} + \frac{1}{2} \right\rfloor$ can wrap arbitrary angle t in radians into range $[-\pi, \pi)$ and it satisfies following properties: P.1 $g(t_1 \pm t_2) = g(g(t_1) \pm g(t_2))$; and P.2 $g(-t) = -g(t)$.*

Lemma A1. *Let $g(t) = t - 2\pi \left\lfloor \frac{t}{2\pi} + \frac{1}{2} \right\rfloor$, then*

$$\mathrm{atan2}(\beta_1\alpha_2 \pm \beta_2\alpha_1,\ \alpha_1\alpha_2 \mp \beta_1\beta_2) = g\left(\mathrm{atan2}(\beta_1, \alpha_1) \pm \mathrm{atan2}(\beta_2, \alpha_2)\right), \tag{A1}$$

From (22)-(24)), we have

$$\begin{aligned} a_i &= d_{1,i}^2 - {d'}_{1,i}^2 + {\Delta x'_1}^2 + {\Delta y'_1}^2 \\ &= \left(x_i^2 + y_i^2\right) - \left((x_i - \Delta x'_1)^2 + (y_i - \Delta y'_1)^2\right) + {\Delta x'_1}^2 + {\Delta y'_1}^2 \\ &= 2\left(x_i \Delta x'_1 + y_i \Delta y'_1\right) \end{aligned}$$

and

$$\begin{aligned} b_i^2 + c_i^2 - a_i^2 =& 4(x_i^* \Delta x'_1 + y_i^* \Delta y'_1)^2 + 4(y_i^* \Delta x'_1 - x_i^* \Delta y'_1)^2 - 4\left(x_i \Delta x'_1 + y_i \Delta y'_1\right)^2 \\ =& 4\left(\left(d_{1,i}^2 - d_{1,i}^2 + y_i^2\right){\Delta x'_1}^2 + \left(d_{1,i}^2 - d_{1,i}^2 + x_i^2\right){\Delta y'_1}^2 - 2x_i y_i \Delta x'_1 \Delta y'_1\right) \\ =& 4\left(x_i \Delta y'_1 - y_i \Delta x'_1\right)^2. \end{aligned} \tag{A2}$$

Then from Lemma A1 and (A2), we have

$$\begin{aligned}
\theta_{i,1,2} &= \operatorname{atan2}\left(-a_i c_i \mp \frac{b_i}{c_i}|c_i|\sqrt{b_i^2+c_i^2-a_i^2},\ -a_i b_i \pm |c_i|\sqrt{b_i^2+c_i^2-a_i^2}\right) \\
&= -g\left(\operatorname{atan2}(b_i,c_i) + \operatorname{atan2}\left(a_i,\ \pm\frac{|c_i|}{c_i}\sqrt{b_i^2+c_i^2-a_i^2}\right)\right) \qquad \text{(A3)} \\
&= -g\Big(\operatorname{atan2}(b_i,c_i) + \operatorname{atan2}\left(a_i,\ \pm 2\left(x_i\Delta y_1 - y_i\Delta x_1\right)\right)\Big). \qquad \text{(A4)}
\end{aligned}$$

From (A3) to (A4), the sign of $\pm\frac{|c_i|}{c_i}\sqrt{4\left(x_i\Delta y_1 - y_i\Delta x_1\right)^2}$ is interpreted by $\pm$, which implies that the values of $\theta_{i,1}$ and $\theta_{i,2}$ may be simply exchanged.

Furthermore,

$$\begin{aligned}
\operatorname{atan2}(b_i,c_i) &= \operatorname{atan2}\Big(-\left(x_i^*\Delta x_1' + y_i^*\Delta y_1'\right),\ x_i^*\Delta y_1' - y_i^*\Delta x_1'\Big) \\
&= -\operatorname{atan2}\left(x_i^*\Delta x_1' + y_i^*\Delta y_1',\ x_i^*\Delta y_1' - y_i^*\Delta x_1'\right) \\
&= g\Big(-\operatorname{atan2}\left(y_i^*,x_i^*\right) - \operatorname{atan2}\left(\Delta x_1',\Delta y_1'\right)\Big) \qquad \text{(A5)}
\end{aligned}$$

and

$$\begin{aligned}
\operatorname{atan2}\left(a_i,\ \pm 2\left(x_i\Delta y_1' - y_i\Delta x_1'\right)\right) &= g\Big(\operatorname{atan2}\left(2\left(x_i\Delta x_1' + y_i\Delta y_1'\right),\ \pm 2\left(x_i\Delta y_1' - y_i\Delta x_1'\right)\right)\Big) \\
&= \begin{cases} g\Big(\operatorname{atan2}(y_i,x_i) + \operatorname{atan2}(\Delta x_1',\Delta y_1')\Big) \\ g\Big(-\operatorname{atan2}(y_i,x_i) - \operatorname{atan2}(\Delta x_1',\Delta y_1')\Big) \end{cases} . \qquad \text{(A6)}
\end{aligned}$$

Substituting (A6) and (A5) into (A4) leads to

$$\begin{aligned}
\theta_{i,1} &= g\Big(\operatorname{atan2}(y_i^*,x_i^*) - \operatorname{atan2}(y_i,x_i)\Big) \\
\theta_{i,2} &= g\Big(\operatorname{atan2}(y_i^*,x_i^*) + \operatorname{atan2}(y_i,x_i) + 2\operatorname{atan2}(\Delta x_1',\Delta y_1') - \pi\Big) \\
&= g\Big(\theta_{i,1} + 2\Theta_i\Big)
\end{aligned}$$

where $\Theta_i = \operatorname{atan2}(y_i,x_i) + \operatorname{atan2}(\Delta x_1',\Delta y_1') - \frac{\pi}{2}$.

Appendix B

Taking the derivative of $Obj\left(\theta;\ \boldsymbol{\mathcal{S}}^*, \Delta\boldsymbol{\mathcal{S}}', \mathbf{D}'\right)$ with respect to θ, we have

$$\begin{aligned}
&\frac{\partial}{\partial\theta}\sum_{i=1}^{n}\left(a_i + b_i\cos(\theta) + c_i\sin(\theta)\right)^2 \\
=&\sum_{i=1}^{n} 2\left(c_i\cos(\theta) - b_i\sin(\theta)\right)\left(a_i + b_i\cos(\theta) + c_i\sin(\theta)\right) \\
=&\sum_{i=1}^{n}\Big(-a_i b_i\sin(\theta) + a_i c_i\cos(\theta) + b_i c_i\cos^2(\theta) - b_i c_i\sin^2(\theta) + (c_i^2 - b^2)\sin(\theta)\cos(\theta)\Big) \\
=&2\Big(-\alpha_n\sin(\theta) + \beta_n\cos(\theta) + 2\gamma_n\cos^2(\theta) + \delta_n\sin(\theta)\cos(\theta) - \gamma_n\Big), \qquad \text{(A7)}
\end{aligned}$$

where

$$\alpha_n = \sum_{i=1}^{n} a_i b_i, \qquad \beta_n = \sum_{i=1}^{n} a_i c_i,$$
$$\gamma_n = \sum_{i=1}^{n} b_i c_i, \qquad \delta_n = \sum_{i=1}^{n} (c_i^2 - b_i^2).$$

If we let $\sin(\theta) = \lambda \in [-1, 1]$ and $\cos(\theta) = \pm\sqrt{1-\lambda^2}$. Then setting (A7) to 0 leads to:

$$\gamma_n \pm \beta_n \sqrt{1-\lambda^2} - 2\gamma_n \lambda^2 \pm \delta_n \lambda \sqrt{1-\lambda^2} - \alpha_n \lambda = 0. \tag{A8}$$

Rearranging (A8) and squaring both sides, we have:

$$\begin{aligned} &\left(\pm(\beta_n + \delta_n \lambda)\sqrt{1-\lambda^2}\right)^2 = \left(-\gamma_n + 2\gamma_n\lambda^2 + \alpha_n\lambda\right)^2 \\ \implies &A_n\lambda^4 + B_n\lambda^3 + C_n\lambda^2 + D_n\lambda + E_n = 0, \end{aligned} \tag{A9}$$

where

$$A_n = 4\gamma_n^2 + \delta_n^2, \qquad B_n = 2(2\alpha_n\gamma_n + \beta_n\delta_n), \qquad C_n = \alpha_n^2 + \beta_n^2 - 4\gamma_n^2 - \delta_n^2,$$
$$D_n = 2(-\alpha_n\gamma_n - \beta_n\delta_n), \qquad E_n = -\beta_n^2 + \gamma_n^2.$$

Equation (A9) is a quartic equation [44] and has at most 4 real roots. Suppose that we have $1 \le m \le 4$ solution(s) from (A9) satisfying $\lambda_j \in [-1, 1]$, $j = 1, \ldots, m$, then the corresponding collection of angles is

$$\boldsymbol{\theta}_\lambda = \left\{ \arcsin(\lambda_1), g(\pi - \arcsin(\lambda_1)), \ldots, \arcsin(\lambda_m), g(\pi - \arcsin(\lambda_m)) \right\} \in [-\pi, \pi).$$

223 ## Appendix C

Proof of Proposition 1. If we recall from (30) that in the case of flip ambiguity we have

$$\theta_R\left(\boldsymbol{\mathcal{S}}^*, \Delta\boldsymbol{\mathcal{S}}', \mathbf{D}'\right) = g\Big(\operatorname{atan2}(y_i^*, x_i^*) + \operatorname{atan2}(y_i, x_i) + 2\operatorname{atan2}(\Delta x_1', \Delta y_1') - \pi\Big), \tag{A10}$$

then we can rearrange to give

$$\begin{aligned} \theta_R\left(\boldsymbol{\mathcal{S}}^*, \Delta\boldsymbol{\mathcal{S}}', \mathbf{D}'\right) =& g\Big(\operatorname{atan2}(y_i^*, x_i^*) + \operatorname{atan2}(y_i, x_i) + 2\operatorname{atan2}(\Delta x_1', \Delta y_1') - \pi \\ &+ \operatorname{atan2}(y_i^{**}, x_i^{**}) - \operatorname{atan2}(y_i^{**}, x_i^{**})\Big) \\ =& g\Big(\operatorname{atan2}(y_i^{**}, x_i^{**}) + \operatorname{atan2}(y_i, x_i) + 2\operatorname{atan2}(\Delta x_1', \Delta y_1') - \pi\Big) \\ &+ g\Big(-\operatorname{atan2}(y_i^{**}, x_i^{**}) + \operatorname{atan2}(y_i^*, x_i^*)\Big). \end{aligned} \tag{A11}$$

From the definition of $\boldsymbol{\mathcal{S}}^{**}$ in (31) it can be shown that

$$g\Big(-\operatorname{atan2}(y_i^{**}, x_i^{**}) + \operatorname{atan2}(y_i^*, x_i^*)\Big) = \theta_R\left(\boldsymbol{\mathcal{S}}^*, \Delta\boldsymbol{\mathcal{S}}', \mathbf{D}'\right),$$

giving

$$\begin{aligned}\theta_R\left(\boldsymbol{\mathcal{S}}^{*},\Delta\boldsymbol{\mathcal{S}}',\mathbf{D}'\right)=&g\Big(\operatorname{atan2}(y_i^{**},x_i^{**})+\operatorname{atan2}(y_i,x_i)+2\operatorname{atan2}(\Delta x_1',\Delta y_1')-\pi\Big)+\\&\theta_R\left(\boldsymbol{\mathcal{S}}^{*},\Delta\boldsymbol{\mathcal{S}}',\mathbf{D}'\right)\\0=&g\Big(\operatorname{atan2}(y_i^{**},x_i^{**})+\operatorname{atan2}(y_i,x_i)+2\operatorname{atan2}(\Delta x_1',\Delta y_1')-\pi\Big).\end{aligned}\tag{A12}$$

Incorporating (A12) into the equation for $\theta_R\left(\boldsymbol{\mathcal{S}}^{**},\Delta\boldsymbol{\mathcal{S}}'',\mathbf{D}''\right)$ we have

$$\begin{aligned}\theta_R\left(\boldsymbol{\mathcal{S}}^{**},\Delta\boldsymbol{\mathcal{S}}'',\mathbf{D}''\right)=&g\Big(\operatorname{atan2}(y_i^{**},x_i^{**})+\operatorname{atan2}(y_i,x_i)+2\operatorname{atan2}(\Delta x_1'',\Delta y_1'')-\pi\Big)\\=&g\Big(\operatorname{atan2}(y_i^{**},x_i^{**})+\operatorname{atan2}(y_i,x_i)+2\operatorname{atan2}(\Delta x_1'',\Delta y_1'')-\pi\\&+2\operatorname{atan2}(\Delta x_1',\Delta y_1')-2\operatorname{atan2}(\Delta x_1',\Delta y_1')\Big)\\=&g\Big(\operatorname{atan2}(y_i^{**},x_i^{**})+\operatorname{atan2}(y_i,x_i)+2\operatorname{atan2}(\Delta x_1',\Delta y_1')-\pi\Big)\\&+g\Big(-2\operatorname{atan2}(\Delta x_1',\Delta y_1')+2\operatorname{atan2}(\Delta x_1'',\Delta y_1'')\Big)\\=&g(0)+g\Big(-2\operatorname{atan2}(\Delta x_1',\Delta y_1')+2\operatorname{atan2}(\Delta x_1'',\Delta y_1'')\Big)\\=&g\Big(-2\operatorname{atan2}(\Delta x_1',\Delta y_1')+2\operatorname{atan2}(\Delta x_1'',\Delta y_1'')\Big).\end{aligned}\tag{A13}$$

■

References

1. Ramson, S.R.J.; Moni, D.J. Applications of Wireless Sensor Networks – A Survey. International Conference on Innovations in Electrical, Electronics, Instrumentation and Media Technology (ICEEIMT), 2017, pp. 325–329.
2. Yick, J.; Mukherjee, B.; Ghosal, D. Wireless Sensor Network Survey. *Computer Networks* **2008**, *52*, 2292–2330.
3. Mao, G.; Fidan, B.; Anderson, B.D. Wireless Sensor Network Localization Techniques. *Computer Networks* **2007**, *51*, 2529–2553.
4. Chong, C.Y.; Kumar, S. Sensor Networks: Evolution, Opportunities, and Challenges. *Proceedings of the IEEE* **2003**, *91*, 1247–1256.
5. Lombardo, L.; Corbellini, S.; Parvis, M.; Elsayed, A.; Angelini, E.; Grassini, S. Wireless Sensor Network for Distributed Environmental Monitoring. *IEEE Transactions on Instrumentation and Measurement* **2018**, *67*, 1214–1222.
6. Alemdar, H.; Ersoy, C. Wireless Sensor Networks for Healthcare: A Survey. *Computer Networks* **2010**, *54*, 2688–2710.
7. Kafi, M.A.; Challal, Y.; Djenouri, D.; Doudou, M.; Bouabdallah, A.; Badache, N. A Study of Wireless Sensor Networks for Urban Traffic Monitoring: Applications and Architectures. *Procedia Computer Science* **2013**, *19*, 617–626.
8. Hammoudeh, M.; Al-Fayez, F.; Lloyd, H.; Newman, R.; Adebisi, B.; Bounceur, A.; Abuarqoub, A. A Wireless Sensor Network Border Monitoring System: Deployment Issues and Routing Protocols. *IEEE Sensors Journal* **2017**, *17*, 2572–2582.
9. Moran, B.; Suvorova, S.; Howard, S. Sensor Management for Radar: A Tutorial. In *Advances in Sensing with Security Applications*; Byrnes, J.; Ostheimer, G., Eds.; Springer, Dordrecht, 2006; Vol. 2, *NATO Security Through Science Series*, pp. 269–291.
10. Garcia-Sanchez, A.J.; Garcia-Sanchez, F.; Losilla, F.; Kulakowski, P.; Garcia-Haro, J.; Rodríguez, A.; López-Bao, J.V.; Palomares, F. Wireless Sensor Network Deployment for Monitoring Wildlife Passages. *Sensors* **2010**, *10*, 7236–7262.
11. Abdullah, S.; Bertalan, S.; Masar, S.; Coskun, A.; Kale, I. A Wireless Sensor Network for Early Forest Fire Detection and Monitoring as a Decision Factor in the Context of a Complex Integrated Emergency

Response System. IEEE Workshop on Environmental, Energy, and Structural Monitoring Systems (EESMS), 2017.

12. Maróti, M.; Völgyesi, P.; Dóra, S.; Kusý, B.; Nádas, A.; Lédeczi, Á.; Balogh, G.; Molnár, K. Radio Interferometric Geolocation. Proceedings of the 3rd international conference on Embedded networked sensor systems (SenSys). ACM Press, 2005.
13. Nath, S.; Ekambaram, V.N.; Kumar, A.; Kumar, P.V. Theory and Algorithms for Hop-Count-Based Localization with Random Geometric Graph Models of Dense Sensor Networks. *ACM Transactions on Sensor Networks* **2012**, *8*, 1–38.
14. Wang, X.; Moran, B.; Brazil, M. Hyperbolic Positioning Using RIPS Measurements for Wireless Sensor Networks. 15th IEEE International Conference on Networks, 2007, pp. 425–430.
15. Wymeersch, H.; Lien, J.; Win, M.Z. Cooperative Localization in Wireless Networks. *Proceedings of the IEEE* **2009**, *97*, 427–450.
16. Yaghoubi, F.; Abbasfar, A.A.; Maham, B. Energy-Efficient RSSI-Based Localization for Wireless Sensor Networks. *IEEE Communications Letters* **2014**, *18*, 973–976.
17. Tomic, S.; Beko, M.; Dinis, R. RSS-Based Localization in Wireless Sensor Networks Using Convex Relaxation: Noncooperative and Cooperative Schemes. *IEEE Transactions on Vehicular Technology* **2015**, *64*, 2037–2050.
18. Goel, S.; Kealy, A.; Lohani, B. Cooperative UAS Localization Using Lowcost Sensors. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences* **2016**, *III-1*, 183–190.
19. Ahrens, S.; Levine, D.; Andrews, G.; How, J.P. Vision-Based Guidance and Control of a Hovering Vehicle in Unknown, GPS-Denied Environments. IEEE International Conference on Robotics and Automation, 2009, pp. 2643–2648.
20. Bachrach, A.; Prentice, S.; He, R.; Roy, N. RANGE-Robust autonomous navigation in GPS-denied environments. *Journal of Field Robotics* **2011**, *28*, 644–666.
21. Zhang, L.; Ye, M.; Anderson, B.D.; Sarunic, P.; Hmam, H. Cooperative localisation of UAVs in a GPS-denied environment using bearing measurements. IEEE 55th Conference on Decision and Control (CDC), 2016, pp. 4320–4326.
22. Balamurugan, G.; Valarmathi, J.; Naidu, V.P.S. Survey on UAV navigation in GPS denied environments. 2016 International Conference on Signal Processing, Communication, Power and Embedded System (SCOPES), 2016, pp. 198–204.
23. de Paula Veronese, L.; Cheein, F.A.; Bastos-Filho, T.; Souza, A.F.D.; de Aguiar, E. A Computational Geometry Approach for Localization and Tracking in GPS-denied Environments. *Journal of Field Robotics* **2015**, *33*, 946–966.
24. Schnipke, E.; Reidling, S.; Meiring, J.; Jeffers, W.; Hashemi, M.; Tan, R.; Nemati, A.; Kumar, M. Autonomous Navigation of UAV through GPS-Denied Indoor Environment with Obstacles. AIAA Infotech @ Aerospace, 2015.
25. Russell, J.S.; Ye, M.; Anderson, B.D.; Hmam, H.; Sarunic, P. Cooperative Localisation of a GPS-Denied UAV in 3-Dimensional Space Using Direction of Arrival Measurements. *IFAC-PapersOnLine* **2017**, *50*, 8019–8024.
26. Singh, S.; Sujit, P. Landmarks based path planning for UAVs in GPS-denied areas. *IFAC-PapersOnLine* **2016**, *49*, 396 – 400. 4th IFAC Conference on Advances in Control and Optimization of Dynamical Systems ACODS 2016.
27. Power, W.; Pavlovski, M.; Saranovic, D.; Stojkovic, I.; Obradovic, Z., Autonomous Navigation for Drone Swarms in GPS-Denied Environments Using Structured Learning. In *Artificial Intelligence Applications and Innovations*; Springer International Publishing, 2020; pp. 219–231.
28. Cao, M.; Anderson, B.D.O.; Morse, A.S. Sensor Network Localization With Imprecise Distances. *Systems & Control Letters* **2006**, *55*, 887–893.
29. Beck, B.; Baxley, R. Anchor Free Node Tracking Using Ranges, Odometry, and Multidimensional Scaling. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2014.
30. Ji, X.; Zha, H. Sensor Positioning in Wireless Ad-Hoc Sensor Networks Using Multidimensional Scaling. IEEE INFOCOM 2004, 2004, Vol. 4, pp. 2652–2661.
31. Borg, I.; Groenen, P.J.F. *Modern Multidimensional Scaling*; Springer, New York, 2005.
32. Costa, J.A.; Patwari, N.; Hero, A.O. Distributed Weighted-Multidimensional Scaling for Node Localization in Sensor Networks. *ACM Transactions on Sensor Networks* **2006**, *2*, 39–64.

33. Rajan, R.T.; van der Veen, A.J. Joint Ranging and Synchronization for an Anchorless Network of Mobile Nodes. *IEEE Transactions on Signal Processing* **2015**, *63*, 1925–1940.
34. Wei, M.; Aragues, R.; Sagues, C.; Calafiore, G.C. Noisy Range Network Localization Based on Distributed Multidimensional Scaling. *IEEE Sensors Journal* **2015**, *15*, 1872–1883.
35. Saeed, N.; Nam, H.; Al-Naffouri, T.Y.; Alouini, M. A State-of-the-Art Survey on Multidimensional Scaling-Based Localization Techniques. *IEEE Communications Surveys Tutorials* **2019**, *21*, 3565–3583.
36. Di Franco, C.; Melani, A.; Marinoni, M. Solving Ambiguities in MDS Relative Localization. International Conference on Advanced Robotics (ICAR), 2015, pp. 230–236.
37. Guo, K.; Qiu, Z.; Meng, W.; Xie, L.; Teo, R. Ultra-wideband based cooperative relative localization algorithm and experiments for multiple unmanned aerial vehicles in GPS denied environments. *International Journal of Micro Air Vehicles* **2017**, *9*, 169–186.
38. Arfken, G.B.; Weber, H.J.; Harris, F.E. *Mathematical Methods for Physicists*, 7th ed.; Elsevier, Boston, 2013.
39. Anderson, B.D.O.; Shames, I.; Mao, G.; Fidan, B. Formal Theory of Noisy Sensor Network Localization. *SIAM Journal on Discrete Mathematics* **2010**, *24*, 684–698.
40. Zhang, H.; Liu, Y.; Lei, H. Localization From Incomplete Euclidean Distance Matrix: Performance Analysis for the SVD–MDS Approach. *IEEE Transactions on Signal Processing* **2019**, *67*, 2196–2209.
41. Dokmanic, I.; Parhizkar, R.; Ranieri, J.; Vetterli, M. Euclidean Distance Matrices: Essential theory, algorithms, and applications. *IEEE Signal Processing Magazine* **2015**, *32*, 12–30.
42. Ji, X.; Zha, H. Sensor positioning in wireless ad-hoc sensor networks using multidimensional scaling. IEEE INFOCOM 2004. IEEE, 2004, Vol. 4, pp. 2652–2661.
43. Di Franco, C.; Bini, E.; Marinoni, M.; Buttazzo, G.C. Multidimensional Scaling Localization with Anchors. IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC), 2017, pp. 49–54.
44. Strobach, P. The Fast Quartic Solver. *Journal of Computational and Applied Mathematics* **2010**, *234*, 3007–3024.