Distributed self-localization for relative position sensing networks in 2D space based on gravity direction

Zhitao Wu

Hangzhou Dianzi University

Abstract: Paper [1] proposes a sensor network self-localization algorithm in a two-dimensional plane, where each sensor node can measure the relative positions of its neighboring nodes. In that work, complex linear equations are used to represent the relative positions between nodes, and the coordinates of the nodes are computed by solving a system of linear equations. We extend this two-dimensional sensor network localization algorithm to three-dimensional space. Specifically, each node is equipped with an IMU sensor to measure the gravity direction, which is then used to project the relative positions in 3D space onto the horizontal plane. This projection results in a 2D representation of the sensor network on the ground plane, allowing us to apply the method from Paper [1] to solve for the node coordinates.

Keywords: sensor networks, self localization, gravity direction.

1. Introduction

Paper [1] proposes a sensor network self-localization algorithm in a two-dimensional plane, where each sensor node is capable of measuring the relative positions of its neighboring nodes. In this paper, the relative positions between nodes are represented using complex linear equations, enabling the calculation of node coordinates through solving systems of linear equations. Paper[2] shows that a sensor network in the 2D plane can be uniquely localized in the global coordinate frame if and only if it contains at least three location-known anchor nodes and its distance graph is globally rigid.

2. Preliminary and problem formulation

2.1 Preliminary

We introduce some fundamental concepts in graph theory and algebraic graph theory. A directed graph $G = [V, E]$ consists of a non-empty set of vertices V and a set of edges $E \in V \times V$. If (j, i) is an edge in the directed graph G, then vertex j is called an in-neighbor of vertex i, and vertex i is called an out-neighbor of vertex j. We define the in-neighbor set of vertex i as $N_i = \{j \in V, (j, i) \in E\}$. Then, we introduce the complex Laplacian matrix, which is associated with the directed graph. For a directed graph, each edge $(j, i) \in E$ is assigned a complex weight ω_{ij}. The elements L_{ij} of the corresponding complex Laplacian matrix L are defined as follows:

$$L_{ij} = \begin{cases} -\omega_{ij}, \text{if } i \neq j \text{ and } j \in N_i \\ 0, \text{if } i \neq j \text{ and } j \notin N_i \\ \sum_{j \in N_i} \omega_{ij}, \text{if } i = j \end{cases}$$

2.2 Problem formulation

Let Σ_g denote a shared global coordinate system. In this section, we discuss a sensor network defined by two groups of sensor node sets. The first group of sensors is referred to as anchor nodes (denoted as$1, \dots, m$). The coordinates of these nodes in the global coordinate system Σ_g are defined as $p_a = [p_1, \dots, p_m] \in R^{3\times m}$, and these coordinates are known. The second group of sensors is referred to as free nodes (denoted as $m + 1, \dots, m + n$). The coordinates of these nodes in the global coordinate system Σ_g are defined as $p_s = [p_{m+1}, \dots, p_{m+n}] \in R^{3\times n}$, and these coordinates are unknown. The coordinates of each sensor node $p_i \in R^3$ are located in three-dimensional space. We define $p = [p_a, p_s] \in R^{3\times(m+n)}$ as the aggregated coordinates of all sensor nodes, and we refer to this set of coordinates as the configuration of the sensor network.

In this paper, we make the following assumption regarding the cumulative coordinates of all sensor nodes, denoted as $p = [p_a, p_s]$ (also referred to as the configuration of the sensor network): the configuration of the sensor network is generic. The condition for the cumulative coordinates p of the sensor network (or the configuration) to be generic is that all coordinates $(p_1, \dots, p_{m+n})$ do not satisfy any non-trivial equation with integer coefficients [3]. Intuitively, a generic configuration of sensor nodes means that it is non-degenerate; for example, no three points are collinear, and so on. This assumption is based on the fact that the positions of the sensor nodes are obtained through actual sensor measurements (which include noise).

The neighboring nodes of free node i (including both free nodes and anchor nodes) are defined as the set N_i. Free node i can measure the relative position of each neighboring node $j \in N_i$ in its local coordinate system Σ_i, denoted as $q_{ij} = [x_{ij}, y_{ij}, z_{ij}]^T$. The local coordinate system Σ_i is centered at node i, and there exists an arbitrary rotation matrix R between the coordinate axes of Σ_i and the global coordinate system Σ_g.

The meaning of the relative position q_{ij} is that the coordinates of node j in the local coordinate system of node i are $q_{ij} = [x_{ij}, y_{ij}, z_{ij}]^T$.

We constructed a directed graph $G = [V, E]$, where $V = \{1, \dots, m + n\}$, to describe the topology of these sensors. In this graph, an edge $(j, i) \in E$ indicates that node i can measure the relative position q_{ij} with respect to node j. The sensor network can thus be defined using a tuple (G, p_a, p_s). The relative position localization problem for the sensor network (G, p_a, p_s) is defined as follows.

Definition 1: Given a sensor network (G, p_a, p_s) and the relative position measurements q_{ij} in a local coordinate system, where $(j, i) \in E$, determine the coordinates of all free nodes p_s. Additionally, there exists a rotation matrix R_i that describes the transformation of a local coordinate system Σ_i relative to the global coordinate system Σ_g.

$$q_{ij} = R_i(p_j - p_i), j \in N_i$$

The measurement information obtained through a sensor network (G, p_a, p_s) can be used to compute a unique solution.

Definition 2: A sensor network (G, p_a, p_s) is jointly localizable if the following condition is satisfied: Given the coordinates of the anchor nodes p_a and the relative positions q_{ij} measured in the local coordinate system, for all $(j, i) \in E$, the relative position measurement problem admits a unique solution.

Definition 2 might imply that obtaining the positions of all anchor nodes and all relative position measurements is required to solve the problem of relative position estimation. However, to reduce information exchange and enable each free node to compute its own position in a distributed manner, we introduce another concept termed self-localizability. In the following definitions, we will introduce several new concepts. Let U_s be an arbitrary set of free nodes. Then, the set of all other sensor nodes is defined as $U_a = V \backslash U_s$ (including anchor nodes). We further define p_{Ua} and p_{Us} as the coordinate vectors of the sensor sets U_a and U_s, respectively.

Definition 3: A sensor network (G, p_a, p_s) is said to be self-localizable if the following condition holds: Given the coordinates of anchor nodes p_{Us} in the global coordinate system Σ_g, and the relative position measurements q_{ij} in the local coordinate system for $i \in U_s, (j, i) \in E$, the sensor network must be jointly localizable for any set of free nodes U_s.

The implication of Definition 3 is that if a free node can obtain the coordinates of its immediate neighbors in the global coordinate system Σ_g, then the free node must be able to compute its own coordinates in Σ_g. The concept of self-localizability is introduced because it will play a crucial role in the fully distributed localization algorithm proposed later in this work.

This paper primarily focuses on the problem of self-localizability and proposes a distributed algorithm to compute the global coordinates of each free node in Σ_g, provided the sensor network is self-localizable.

3. The relative position localization algorithm on 2D space

Paper [1] presents a distributed iterative algorithm for solving this problem in two-dimensional cases. In the current paper, we extend the algorithm to three-dimensional space based on the approach outlined in paper [1]. Therefore, this section provides a brief description of the computational method for the two-dimensional case as detailed in paper[1].

3.1 Variable definition

The problem of relative position localization algorithms in two-dimensional (2D) and three-dimensional (3D) cases is generally described in a similar manner. Below, we provide some variable definitions for the 2D case. In the 2D case, the coordinates of sensor node i are denoted as $p_i^{2d} \in C$. The coordinates of all sensor nodes are represented using complex numbers, where the real part of the complex number corresponds to the x-axis coordinate of the sensor node, and the imaginary part corresponds to the y-axis coordinate. In the global coordinate system Σ_g, the coordinates of the anchor node set are defined as $p_a^{2d} = [p_1^{2d}, \dots, p_m^{2d}]^T \in C^n$, and the coordinates of the free node set are defined as $p_s^{2d} = [p_{m+1}^{2d}, \dots, p_{m+n}^{2d}]^T \in C^m$. We denote $p^{2d} = [{p_a^{2d}}^T, {p_s^{2d}}^T]^T \in C^{m+n}$ as the combined coordinates of all sensor nodes. A free node i can measure the relative position of each neighboring node $j \in N_i$ in its local coordinate system Σ_i, denoted as $r_{ij} \in C$.

$$r_{ij} = \rho_{ij} e^{i\theta_{ij}}$$

Here, ρ_{ij} represents the distance from node j to node i, and θ_{ij} represents the angle of node j in the local coordinate system Σ_i.

3.2 Self localization condition

Theorem 1: Under the conditions of Assumption 1, the necessary and sufficient conditions for a sensor network (G, p_a, p_s) to be self-localizable in a two-dimensional space are as follows:

(NS-1): The number of anchor nodes $m \geq 2$;

(NS-2): Every free node is 2-reachable from the anchor nodes.

3.3 Weight calculation

According to the hypothesis of this paper, each free node can obtain relative position measurements within its own local coordinate system. Then, based on Theorem 1, if a sensor network is self-localizable, every free node is 2-anchor reachable. Consequently, each free node must have at least two internal neighbors. Therefore, using the data from all internal neighbors, each free node can calculate the weights using the following formula.

$$\sum_{j \in N_i} \omega_{ij} r_{ij} = 0 \quad (1)$$

Among them, $r_{ij}, j \in N_i$, represents the data measured at node i. When $|N_i| = 2$, the parameters ω_{ij} and ω_{ik} are defined as follows. For instance, if node j is an inner neighbor of node i, the parameter is defined as:

$$\omega_{ij} = \frac{e^{-i\theta_{ij}}}{\rho_{ij}}$$

Then, if node k is another inner neighbor of node i, the parameter is defined as follows:

$$\omega_{ik} = -\frac{e^{i\theta_{ik}}}{\rho_{ik}}$$

When $|N_i| > 2$, the number of solutions to Equation (1) is greater than 1. In this case, we can randomly choose a set of parameters $\omega_{i,j1}, \dots, \omega_{i,jk}$ that satisfy Equation (1), with the only requirement being that each parameter must be non-zero. For example, we can set $\omega_{i,j1}, \dots, \omega_{i,j(k-1)}$ all equal to 1, and then use Equation (1) to compute the remaining parameter $\omega_{j,ik}$. As long as $\omega_{j,ik} \neq 0$, this set of parameters is valid. Since anchor nodes have no in-neighbors, they do not measure the relative positions of other nodes. We can then express the weight relationships among all the nodes in the following matrix form.

$$Lp^{2d} = 0$$

Let $p^{2d} = \left[{p_a^{2d}}^T, {p_s^{2d}}^T \right]^T$ represent the coordinate vector of all nodes, and let matrix L denote the Laplacian matrix of graph G, where the matrix L takes the following form:

$$L = \begin{bmatrix} 0 & 0 \\ B & H \end{bmatrix}$$

The above equation can be expanded as follows.

$$Hp_s^{2d} + Bp_a^{2d} = 0$$

Remark 2: If a sensor network is localizable, then according to Theorem 1 and Lemma 1, the matrix H is nonsingular.

3.4 Self localization algorithm with noise

In this section, we employ a distributed algorithm to compute the coordinates of each free node in the global coordinate system Σ_g (assuming the sensor network is self-localizable). When relative position measurements with noise and rounding errors are used to compute the weights ω_{ij}, the aforementioned linear equation takes the following form.

$$(H - \Delta_H)p_s^{2d} + (B - \Delta_B)p_a^{2d} = 0$$

Here, matrices H and B are derived using Equation (1). The matrices Δ_H and Δ_B represent the error matrices caused by measurement noise and rounding errors. We define $v = \Delta_H p_s + \Delta p_a$, which transforms the above equation into the following form:

$$Hp_s^{2d} = -Bp_a^{2d} + v$$

To simplify the analysis, we assume that $v \sim CN(0, \Sigma)$; that is, v follows a complex Gaussian distribution with a mean of zero and a variance of Σ. Since the measurements and computations at different nodes are independent processes, the variance Σ is a diagonal matrix, and each node is aware of the corresponding component σ_i of the i-th node in Σ. Using the least squares estimation theory, the optimal solution for the above weighted least squares estimation can be derived as follows.

$$p_s^{2d} = \Phi^{-1}\theta$$

$$\Phi = H^T\Sigma^{-1}H$$

$$\theta = -H^T\Sigma^{-1}Bp_a^{2d}$$

Next, we propose a distributed localization scheme. This algorithm can start from any initial position and gradually converge to the optimal solution. First, we define the following matrix:

$$\widehat{H} = \Sigma^{\frac{1}{2}}B$$

$$\widehat{B} = \Sigma^{\frac{1}{2}}B$$

By introducing an auxiliary variable$\zeta \in C^n$, we propose the following iterative algorithm:

$$\zeta\,(t+1) = \widehat{H}\hat{p}_s^{2d}\,(t) + \widehat{B}p_a^{2d}$$

$$\hat{p}_s^{2d}\,(t+1) = \hat{p}_s^{2d}\,(t) - \epsilon\widehat{H}^T\zeta\,(t)$$

The above iterative algorithm can be implemented in a distributed manner at each node. The algorithm requires mutual communication between each node and its neighboring nodes. In practical scenarios, the communication range is typically larger than the sensing range of the sensors. Therefore, in most cases, if node i can measure the relative position of node j, node i can also communicate with node j.

Implementation of the iterative algorithm

1. Node i receives the estimated value $\hat{p}_j$ from its internal neighbors j.

2. Node i receives the weighted auxiliary variable $\omega_{ki}\zeta_k(t)$ from its external neighbors.

3.Node i uses the auxiliary variables to estimate its own position as follows:

$$\zeta_i\,(t+1) = \frac{1}{\sqrt{\sigma_i}}\sum_{j\in N_i}\omega_{ij}\hat{p}_j\,(t)$$

$$\hat{p}_i\,(t+1) = \hat{p}_i\,(t) - \frac{\epsilon}{\sqrt{\sigma_i}}\sum_{j\in N_k}\omega_{ki}\zeta_k\,(t)$$

4. Node i sends its position estimate $\hat{p}_i\,(t+1)$ to its external neighbors.

5. Node i sends its weighted auxiliary variable $\omega_{ki}\zeta_i\,(t+1)$ to its internal neighbors.

To ensure the convergence of the aforementioned iterative algorithm, the range of values for ϵ must satisfy the following conditions.

$$0 < \epsilon < \frac{1}{\lambda_{max}(\Phi)}$$

Using the distributed approach described below, it is possible to find feasible values for the parameter ϵ within a finite number of steps.

$$\lambda_{\max}(\Phi) \leq \left\|\widehat{H}\right\|_1 \left\|\widehat{H}\right\|_\infty$$

Thus, we can use $\frac{1}{\|\widehat{H}\|_1 \|\widehat{H}\|_\infty}$ as the upper bound for the parameter ϵ. Each node i can obtain all the values in the i-th row of the matrix $\widehat{H}$ (the values in the i-th row of $\widehat{H}$ are calculated based on the weights ω_{ij} and the variance σ_i of all in-neighbors of node i). The specific formula is $r_i = \sum_{j=1}^{n}\left|\frac{1}{\sqrt{\sigma_i}}\omega_{ij}\right|$. Then, $\left\|\widehat{H}\right\|_\infty = \max_i r_i$, which can be computed using the maximum consensus algorithm [4]. Similarly, each node i can also obtain all the values in the i-th column of the matrix $\widehat{H}$ (the values in the j-th column of $\widehat{H}$ are determined by the weights ω_{ji} and the variance σ_j of all out-neighbors of node i). The specific formula is $l_i = \sum_{j=1}^{n}\left|\frac{1}{\sqrt{\sigma_j}}\omega_{ji}\right|$. Then, $\left\|\widehat{H}\right\|_1 = \max_i l_i$, which can also be computed using the maximum consensus algorithm. Therefore, ϵ can take any value within the interval $\left(0, \frac{1}{\|\widehat{H}\|_1 \|\widehat{H}\|_\infty}\right)$, and it is guaranteed that the algorithm will converge.

4. The relative position localization algorithm on 3D space

4.1 Variable definition

The algorithm proposed in [1] solves the relative localization problem in a two-dimensional plane; we extend this algorithm to three-dimensional space. In this section, we first define some variables that will be used.

As described in the problem statement, in three-dimensional space, a free node i measures the relative position of its inner neighbor j (where $j \in N_i$) in the local coordinate system Σ_i as $q_{ij} = \left[x_{ij}, y_{ij}, z_{ij}\right]^T, j \in N_i$. Typically, sensors are equipped with MEMS-based IMUs. Although these IMUs are not highly accurate, they are sufficient for detecting the direction of gravity. Using the gravity direction obtained from the IMU sensor, we can align the local coordinate system Σ_i with a corrected local coordinate system Σ_{iz}, where the z-axis is aligned opposite to the direction of gravity (using a rotation matrix R_i^{cor}). Thus, the relative position of the inner neighbor $j \in N_i$ of free node i in the corrected local coordinate system Σ_{iz} is as follows.

$$q'_{ij} = R_i^{cor}\left[x_{ij}, y_{ij}, z_{ij}\right]^T = \left[x'_{ij}, y'_{ij}, z'_{ij}\right]^T$$

For each free node $i \in [m+1, m+n]$, we perform the same operation to align the local coordinate system Σ_i of each free node i such that it is corrected to Σ_{iz}, where the z-axis is oriented opposite to the direction of gravity. At this point, the relative position measurement q'_{ij} of the internal neighbor $j \in N_i$ of node i can be projected onto the horizontal plane, resulting in $q_{ij}^{proj} = \left[x'_{ij}, y'_{ij}\right]^T, j \in N_i$. Given the relative position measurements of all nodes on the horizontal plane, the x- and y-components of the coordinates of any free node i can be computed using the method proposed in [1].

In the problem description, we have already defined the coordinates of sensor node iii in three-dimensional space as $p_i = [x_i, y_i, z_i]^T \in R^3$. On the two-dimensional plane, we redefine variables to represent the coordinates of node i. Specifically, we represent sensor node i using a complex number $p_i^{proj} = x_i + y_i i$, where the real part of the complex number corresponds to the x-component of node i's coordinates, and the imaginary part corresponds to the y-component of node i's coordinates.

The global coordinate system Σ_g is originally defined in three-dimensional space. After projection, the global coordinate system on the plane, denoted as Σ_g^{proj}, is obtained. Compared to Σ_g, this coordinate system simply eliminates the z-axis, while the x-axis and y-axis directions remain the same as those in Σ_g. The local coordinate system of free node i in three-dimensional space, denoted as Σ_i, has already been aligned in a previous step such that its z-axis is opposite to the direction of gravity, forming the coordinate system Σ_{iz}. After projection, the local coordinate system of free node i on the plane is denoted as Σ_i^{proj}. This coordinate system is derived from Σ_{iz} by removing the z-axis, while the x-axis and y-axis directions remain the same as in Σ_{iz}. Then, there exists an unknown angle θ_i between the local coordinate system Σ_i^{proj} of free node i on the plane and the global coordinate system Σ_g^{proj} on the plane.

Then, we define the coordinate variables for all sensor nodes on the horizontal plane. The first group of sensors is referred to as anchor nodes (denoted as $1, \ldots, m$). In the global three-dimensional coordinate system Σ_g, the coordinates of these anchor nodes are represented as $p_a = [p_1, \ldots, p_m] \in R^{3\times m}$. After projection onto the two-dimensional global coordinate system Σ_g^{proj}, the coordinates of the anchor nodes are denoted as $p_a^{proj} = \left[p_1^{proj}, \ldots, p_m^{proj}\right]^T \in C^m$. The second group of sensors is referred to as free nodes (denoted as $m+1, \ldots, m+n$). In the three-dimensional global coordinate system Σ_g, the coordinates of these free nodes are represented as $p_s = [p_{m+1}, \ldots, p_{m+n}]^T \in R^{3\times n}$. After projection onto the two-dimensional global coordinate system Σ_g^{proj}, the coordinates of the free nodes are denoted as $p_s = \left[p_{m+1}^{proj}, \ldots, p_{m+n}^{proj}\right]^T \in C^n$. Finally, we define $p^{proj} = \left[p_a^{proj^T}, p_s^{proj^T}\right]^T \in C^{m+n}$ as the aggregated coordinates of all sensor nodes in the two-dimensional plane.

In three-dimensional space, the relative position of an interior neighbor $j \in N_i$ of a free node i, measured in the local coordinate system Σ_i, is given by $q_{ij} = \left[x_{ij}, y_{ij}, z_{ij}\right]^T, j \in N_i$. During the aforementioned projection process, the relative position measurement q'_{ij} of node i's interior neighbor $j \in N_i$ can be projected onto the horizontal plane, resulting in $q_{ij}^{proj} = \left[x'_{ij}, y'_{ij}\right]^T, j \in N_i$. This relative position measurement can be expressed as follows.

$$r_{ij}^{proj} = x'_{ij} + y'_{ij} i = \rho_{ij}^{proj} e^{i\theta_{ij}^{proj}}$$

$$\rho_{ij}^{proj} = \sqrt{{x'_{ij}}^2 + {y'_{ij}}^2}$$

$$\theta_{ij}^{proj} = \text{arctan2}(y'_{ij}, x'_{ij})$$

We constructed a directed graph $G^{proj} = [V^{proj}, E^{proj}]$, where $V^{proj} = \{1, \dots, m+n\}$, to describe the topology of the sensors. In this graph, an edge $(j, i) \in E^{proj}$ indicates that node i can measure the relative position r_{ij}^{proj}. Consequently, the sensor network can be defined as a tuple $(G^{proj}, p_a^{proj}, p_s^{proj})$.

4.2 Projection calculation

In three-dimensional space, the relative position of an inner neighbor node $j \in N_i$ in the local coordinate system Σ_i, as measured by a free node i, is $q_{ij} = [x_{ij}, y_{ij}, z_{ij}]^T$, where $j \in N_i$. To align the local coordinate system Σ_i such that its z-axis is opposite to the direction of gravity, we utilize the gravity direction obtained from an IMU sensor to correct Σ_i into the corrected local coordinate system Σ_{iz} using a rotation matrix R_{cor}^i. Consequently, the relative position of the inner neighbor node $j \in N_i$ in the corrected local coordinate system Σ_{iz} becomes: $q'_{ij} = R_i^{cor}[x_{ij}, y_{ij}, z_{ij}]^T = [x'_{ij}, y'_{ij}, z'_{ij}]^T$. The corrected relative position q'_{ij} is then projected onto the horizontal plane to obtain: $q_{ij}^{proj} = [x'_{ij}, y'_{ij}]^T, j \in N_i$.

Then, we detail the specific projection calculation method. Let the inner neighbor node of the free node i be node j, whose position in the local coordinate system Σ_j is given as $q_{ij} = [x_{ij}, y_{ij}, z_{ij}]^T$. The direction of gravity measured by the IMU in the local coordinate system Σ_i is denoted as $g_{ori} = [x_g, y_g, z_g]^T$. Here, we directly use the gravity direction measured by the IMU as g_{ori}. In practice, more accurate gravity directions can be estimated by constructing a motion model for the robot, incorporating the magnitude of gravitational acceleration, and applying methods such as Kalman filtering.

We define the z-axis of the global coordinate system as being opposite to the direction of gravity. To simplify the notation, we introduce the variable g_{inv} to represent the direction opposite to gravity, denoted as $g_{inv} = -g_{ori} = [-x_g, -y_g, -z_g]^T$. Furthermore, we assume that the z-axis of the sensor used for relative position detection on the sensor node aligns with the z-axis of the IMU, denoted as $z = [0,0,1]^T$.

Our goal is to rotate the local coordinate system Σ_i such that the z-axis of the rotated local coordinate system Σ_{iz} is aligned with the direction opposite to gravity. Specifically, this requires rotating the local coordinate system Σ_i around the normal vector of the plane formed by the three points: "IMU z-axis, node i, and the direction opposite to gravity (g_{inv})." This normal vector is denoted as $n \in R^3$, and the angle of rotation is denoted as$\alpha \in (-\pi, \pi]$, which represents the angle between the "IMU z-axis, node i, and the direction opposite to gravity (g_{inv})." In the local coordinate system Σ_i, the IMU z-axis and the direction opposite to gravity are given as follows.

$$z = [0,0,1]^T$$

$$g_{inv} = [-x_g, -y_g, -z_g]^T$$

The normal vector n can be calculated using the vector cross product as follows.

$$n = z \times g_{inv}$$
$$= [y_g, -x_g, 0]^T$$

The rotation angle α can be calculated using the scalar dot product as follows:

$$\cos(\alpha) = \frac{n * g_{inv}}{||n||||g_{inv}||}$$
$$= -\frac{z_g}{\sqrt{x_g^2 + y_g^2 + z_g^2}}$$

$$\alpha = \arccos\left(-\frac{z_g}{\sqrt{x_g^2 + y_g^2 + z_g^2}}\right)$$

Then, we can compute the rotation vector $v \in R^3$.

$$v = \frac{n}{||n||}\alpha$$
$$= \frac{\alpha}{\sqrt{y_g^2 + z_g^2}}[y_g, -x_g, 0]^T$$

After calculating the rotation vector v, the rotation matrix R can be derived using the following formula.

$$R = \text{phi}(v)$$

Using the aforementioned rotation matrix R, the local coordinate system Σ_i can be rotated into the local coordinate system Σ_{iz}. In the local coordinate system Σ_i at node i, the relative position of an internal neighboring node j is given as $q_{ij} = [x_{ij}, y_{ij}, z_{ij}]^T$. In the local coordinate system Σ_{iz}, the relative position of the internal neighboring node j is expressed as follows.

$$q'_{ij} = R^{-1} q_{ij}$$
$$= [x'_{ij}, y'_{ij}, z'_{ij}]^T$$

Since the z-axis of the local coordinate system Σ_{iz} is oriented opposite to the direction of gravity, the projection of q'_{ij} onto the horizontal plane can be expressed as follows (coordinates in the two-dimensional plane).

$$q_{ij}^{proj} = [x'_{ij}, y'_{ij}]^T$$

4.3 Calculation of xy component of nodes

The sensor network in three-dimensional space, denoted as (G, p_a, p_s), is projected onto the horizontal plane, resulting in a sensor network $(G^{proj}, p_a^{proj}, p_s^{proj})$. A directed graph is then used to describe the topological relationships of these sensors, represented as $G^{proj} = [V^{proj}, E^{proj}]$, where $V^{proj} = \{1, \dots, m+n\}$. In this graph, an edge $(j,i) \in E^{proj}$ indicates that node i can measure the relative position r_{ij}^{proj}. For the relative localization problem of the planar sensor network $(G^{proj}, p_s^{proj}, p_s^{proj})$, the algorithm from [1] can be applied for computation.

The weight for each free node can be calculated using the following equation.

$$\sum_{j \in N_i} \omega_{ij}^{proj} r_{ij}^{proj} = 0$$

The $r_{ij}^{proj}, j \in N_i$, represents the relative position of the neighboring node j within the local neighborhood of node i after projection. The above equation can be reorganized into the following form.

$$H^{proj} p_s^{proj} + B^{proj} p_a^{proj} = 0$$

When computing the weight ω_{ij}^{proj} under the condition of relative position measurements with noise and rounding errors, the linear equation takes the following form.

$$\left(H^{proj} - \Delta_H^{proj}\right) p_s^{proj} + \left(B^{proj} - \Delta_B^{proj}\right) p_a^{proj} = 0$$

The matrices Δ_H^{proj} and Δ_B^{proj} represent the error matrices caused by measurement noise and rounding errors. We define $v^{proj} = \Delta_H^{proj} p_s^{proj} + \Delta_B^{proj} p_a^{proj}$, and thus the above equation is transformed into the following form.

$$H^{proj} p_s^{proj} = -B^{proj} p_a^{proj} + v^{proj}$$

To simplify the analysis, we assume that $v^{proj} \sim CN(0, \Sigma^{proj})$; that is, v^{proj} follows a complex Gaussian distribution with zero mean and covariance Σ^{proj}. The component σ_i^{proj} in Σ^{proj} corresponds to the variance associated with the i-th node. Using the least squares estimation theory, the optimal solution for the above weighted least squares estimation can be expressed as follows.

$$p_s^{proj} = {\Phi^{proj}}^{-1} \theta^{proj}$$

$$\Phi^{proj} = {H^{proj}}^T {\Sigma^{proj}}^{-1} H^{proj}$$

$$\theta^{proj} = -{H^{proj}}^T {\Sigma^{proj}}^{-1} B^{proj} p_a^{proj}$$

In the distributed algorithm, the matrices $\widehat{H}^{proj}$ and $\widehat{B}^{proj}$ are defined as follows.

$$\widehat{H}^{proj} = {\Sigma^{proj}}^{\frac{1}{2}} H^{proj}$$

$$\widehat{B}^{proj} = {\Sigma^{proj}}^{\frac{1}{2}} B^{proj}$$

Introduce an auxiliary variable $\zeta^{proj} \in C^n$, and the iterative algorithm is expressed as follows.

$$\zeta^{proj}(t+1) = \widehat{H}^{proj} \hat{p}_s^{proj}(t) + \widehat{B}^{proj} p_a^{proj}$$

$$\hat{p}_s^{proj}(t+1) = \hat{p}_s^{proj}(t) - \epsilon^{proj} {\widehat{H}^{proj}}^T \zeta^{proj}(t)$$

To ensure the convergence of the above iterative algorithm, the value range of ϵ must satisfy the following conditions:

$$0 < \epsilon^{proj} < \frac{1}{\lambda_{max}(\Phi^{proj})}$$

Using the following distributed approach, it is possible to determine feasible values for the parameter ϵ^{proj} within a finite number of steps.

$$\lambda_{max}(\Phi^{proj}) \le \left\| \widehat{H}^{proj} \right\|_1 \left\| \widehat{H}^{proj} \right\|_\infty$$

4.4 Calculation of z component of nodes

We have already computed the x and y components of each node in the three-dimensional sensor network (G, p_a, p_s), while the remaining z component has not yet been calculated. At this point, we can arbitrarily select a direction on the horizontal plane as the y-axis (e.g., by using the direction defined by two anchor nodes), and then project all sensor nodes onto the y-z plane. Since we have already acquired the y-axis coordinates for each sensor node, the task now is to compute the z-axis coordinates. This can similarly be achieved using relative positioning algorithms in the two-dimensional plane. However, in this approach, the selected y-axis on the horizontal plane needs to be broadcast to every sensor node using methods such as the maximum consensus algorithm, which entails significant computational overhead. To address this, we propose a much simpler method to compute the z-axis component of each node.

In the projection calculation phase, we can utilize the gravitational direction obtained from the IMU sensors to adjust the local coordinate system Σ_i into a corrected local coordinate system Σ_{iz}, where the z-axis is aligned oppositely to the gravitational direction (denoted by the rotation matrix R_i^{cor}). In the local coordinate system Σ_i of node i, the relative position of an internal neighbor node j is observed as $q_{ij} = \left[x_{ij}, y_{ij}, z_{ij}\right]^T$, and the relative position of another internal neighbor node k is observed as $q_{ik} = \left[x_{ij}, y_{ik}, z_{ik}\right]^T$. In the corrected local coordinate system Σ_{iz}, the relative positions of internal neighbor nodes j and k are as follows.

$$q'_{ij} = R_i^{cor} q_{ij} = \left[x'_{ij}, y'_{ij}, z'_{ij}\right]^T$$

$$q'_{ik} = R_i^{cor} q_{ik} = [x'_{ik}, y'_{ik}, z'_{ik}]^T$$

The coordinates of node i in three-dimensional space are given as $p_i = [x_i, y_i, z_i]^T$, the coordinates of node j are $p_j = \left[x_j, y_j, z_j\right]^T$, and the coordinates of node k are $p_k = [x_k, y_k, z_k]^T$. Since the z-axis of the local coordinate system Σ_{iz} is oriented opposite to the direction of gravity, the z-axis components of the node coordinates z_i, z_j, z_k satisfy the following relationship with the observed values z'_{ij} and z'_{ik}.

$$z_j - z_i = z'_{ij}$$

$$z_k - z_i = z'_{ik}$$

Each free node must have at least two inner neighbors. If node i has exactly two inner neighbors, the following linear equation can be constructed.

$$\frac{1}{z'_{ij}}(z_j - z_i) - \frac{1}{z'_{ik}}(z_k - z_i) = 0$$

$$(z_j - z_i)\omega^z_{ij} + (z_k - z_i)\omega^z_{ik} = 0$$

The parameter $\omega^z_{ij} = \frac{1}{z'_{ij}}$, and $\omega^z_{ik} = -\frac{1}{z'_{ik}}$.

If the number of in-neighbors $j \in N_i$ of node i is greater than 2, the parameter ω^z_{ij} can be calculated using the following equation, where r^z_{ij} represents the vertical distance difference between node i and node j. Since the number of in-neighbors of node i exceeds 2, this equation does not have a unique solution. We only need to find a set of non-zero parameters. For instance, we can set $\omega^z_{i,j(1)}, \ldots, \omega^z_{ij(|N_i|-1)}$ to 1, and then use this equation to compute $\omega^z_{i,j(|N_i|)}$.

$$\sum_{j\in N_i} \omega^z_{ij} r^z_{ij} = 0 \qquad (2)$$

Since the anchor nodes lack internal neighbors, they do not measure the relative positions of other nodes. Consequently, the weight relations among all nodes can be expressed in the following matrix form.

$$L^z p^z = 0$$

Let $p^z = \left[p_a^{z\,T}, p_s^{z\,T}\right]^T$, where $p^z_a = [z_1, \ldots, z_n]^T$ and $p_s = [z_{n+1}, \ldots, z_{n+m}]^T$ represent the z-axis coordinate vectors of all nodes. Here, p^z denotes the z-axis coordinate vector of all nodes, and the matrix L^z is the Laplacian matrix, which takes the following form.

$$L^z = \begin{bmatrix} 0 & 0 \\ B^z & H^z \end{bmatrix}$$

The above equation can be expanded as follows.

$$H^z p^z_s + B^z p^z_a = 0$$

Remark 2: If a sensor network is self-localizable, then according to Theorem 1 and Lemma 1, the matrix H is non-singular.

The form of the above equation is very similar to the relative position localization algorithm in a two-dimensional plane. In the localization algorithm for the 2D plane, each component of the vector p_a is expressed as a complex number, representing the position of a node in the plane. In the equation for z-axis localization, each component of the vector p^z_a is a single scalar value, which represents the position of a node along the z-axis. Similarly, we employ a distributed algorithm to compute the z-axis coordinates of each free node in the global coordinate system Σ_g. When relative position measurements are affected by noise and weight coefficients ω^z_{ij} are computed under rounding error conditions, the form of the above linear equation becomes as follows.

$$(H^z - \Delta^z_H)p^z_s + (B^z - \Delta^z_B)p^z_a = 0$$

The matrices H^z and B^z are obtained through Equation (2), while the matrices Δ^z_H and Δ^z_B represent error matrices caused by measurement noise and rounding errors. We define $v^z = \Delta^z_H p^z_s + \Delta^z p^z_a$, which reformulates the above equation into the following form.

$$H^z p^z_s = -B^z p^z_a + v^z$$

To simplify the analysis, we assume $v^z \sim N(0, \Sigma^z)$; that is, v^z follows a complex Gaussian distribution with a mean of zero and a variance of Σ^z. Since the measurements and computations of different nodes are independent processes, the variance Σ^z is a diagonal matrix, and each node knows the corresponding component σ^z_i of the i-th node in Σ^z. Using the least squares estimation theory, the optimal solution for the above weighted least squares estimation can be derived as follows.

$$p^z_s = \Phi^{z-1}\theta^z$$

$$\Phi^z = H^{z^T}\Sigma^{z-1}H^z$$

$$\theta^z = -H^{z^T}\Sigma^{z-1}B^z p^z_a$$

Then, we propose a distributed localization scheme. This algorithm can start from any initial position and gradually converge to the optimal solution. First, we define the following matrix.

$$\widehat{H}^z = \Sigma^{z\frac{1}{2}}H^z$$

$$\widehat{B}^z = \Sigma^{z\frac{1}{2}}B^z$$

By introducing an auxiliary variable $\zeta^z \in R^n$, we propose the following iterative algorithm.

$$\zeta^z(t+1) = \widehat{H}^z\hat{p}^z_s(t) + \widehat{B}^z p^z_a$$

$$\hat{p}^z_s(t+1) = \hat{p}^z_s(t) - \epsilon^z \widehat{H}^{z^T}\zeta^z(t)$$

The iterative algorithm described above can be implemented in a distributed manner at each node. This algorithm requires mutual communication between each node and its neighboring nodes. In practical scenarios, the communication range typically exceeds the sensing range of the sensors. Therefore, under normal circumstances, if node i can measure the relative position of node j, node i can also exchange information with node j.

Implementation of the iterative algorithm

1. Node i obtains the estimated value of p^z_j, denoted as $\hat{p}^z_j$, from its inner neighbors j.

2. Node i receives the weighted auxiliary variable $\omega^z_{ki}\zeta^z_k(t)$ from its outer neighbors.

3. Node i uses the auxiliary variables to estimate its own position as follows:

$$\zeta^z_i(t+1) = \frac{1}{\sqrt{\sigma^z_i}}\sum_{j\in N_i} \omega^z_{ij}\hat{p}^z_j\ (t)$$

$$\hat{p}^z_i(t+1) = \hat{p}^z_i\ (t) - \frac{\epsilon^z}{\sqrt{\sigma^z_i}}\sum_{j\in N_k} \omega^z_{ki}\zeta^z_k\ (t)$$

4. Node i sends its position estimate $\hat{p}^z_i\ (t+1)$ to its outer neighbors.

5. Node i sends its weighted auxiliary variable $\omega^z_{ki}\zeta^z_i\ (t+1)$ to its inner neighbors.

To ensure the convergence of the iterative algorithm, the value of ϵ^z must satisfy the following conditions (proof provided in Appendix 1).

$$0 < \epsilon^z < \frac{1}{\lambda_{\max}(\Phi^z)}$$

Using the following distributed approach, we can determine feasible values for the parameter ϵ^z within a finite number of steps.

$$\lambda_{\max}(\Phi^z) \le \left\|\widehat{H}^z\right\|_1 \left\|\widehat{H}^z\right\|_\infty$$

We can adopt $\frac{1}{\left\|\widehat{H}^z\right\|_1 \left\|\widehat{H}^z\right\|_\infty}$ as the upper bound for the parameter ϵ^z. Each node i can compute all the elements of the i-th row of the matrix $\widehat{H}^z$ (the values in the i-th row of $\widehat{H}^z$ are computed based on the weights ω_{ij}^z and the variance σ_j^z of all its inward neighbors). The specific formula is $r_i^z = \sum_{j=1}^n \left| \frac{1}{\sqrt{\sigma_i^z}} \omega_{ij}^z \right|$, and then $\left\|\widehat{H}^z\right\|_1 = \max_i l_i^z$, which can be computed using the max-consensus algorithm [4]. Similarly, each node i can compute all the elements of the i-th column of the matrix $\widehat{H}^z$ (the values in the j-th column of $\widehat{H}^z$ are computed based on the weights ω_{ji}^z and the variance σ_j^z of all its outward neighbors). The specific formula is $l_i^z = \sum_{j=1}^n \left| \frac{1}{\sqrt{\sigma_j^z}} \omega_{ji}^z \right|$, and then $\left\|\widehat{H}^z\right\|_1 = \max_i l_i^z$, which can also be computed using the max-consensus algorithm. Thus, ϵ^z can take any value in the interval $(0, \frac{1}{\left\|\widehat{H}^z\right\|_1 \left\|\widehat{H}^z\right\|_\infty})$, and all such values will ensure the convergence of the algorithm.

Appendix 1. proof of range for parameter ϵ^z

The iterative formula of the algorithm is as follows.

$$\zeta^z(t+1) = \widehat{H}^z \hat{p}_s^z(t) + \widehat{B}^z p_a^z$$

$$\hat{p}_s^z(t+1) = \hat{p}_s^z(t) - \epsilon^z {\widehat{H}^z}^T \zeta^z(t)$$

Let the matrices $\widehat{H}^z = {\Sigma^z}^{\frac{1}{2}} H^z$ and $\widehat{B}^z = {\Sigma^z}^{\frac{1}{2}} B^z$, and define $\Phi^z = {H^z}^T {\Sigma^z}^{-1} H^z$. We need to prove that, for the algorithm to converge, the range of values for ϵ^z should satisfy the following condition.

$$0 < \epsilon^z < \frac{1}{\lambda_{\max}(\Phi^z)}$$

The iterative formula of the above algorithm can be written in matrix form as follows.

$$\begin{bmatrix} \zeta^z(t+1) \\ \hat{p}_s^z(t+1) \end{bmatrix} = \begin{bmatrix} 0 & \widehat{H^z} \\ -\epsilon^z \widehat{H}^z & I \end{bmatrix} \begin{bmatrix} \zeta^z(t) \\ \hat{p}_s^z(t) \end{bmatrix} + \begin{bmatrix} \widehat{B}^z \\ 0 \end{bmatrix} p_a^z$$

Let λ^z be the eigenvalue of the matrix $A^z = \begin{bmatrix} 0 & \widehat{H^z} \\ -\epsilon^z \widehat{H}^z & I \end{bmatrix}$, and let $\begin{bmatrix} x \\ y \end{bmatrix}$ be the corresponding eigenvector. Then, we obtain the following equation.

$$\widehat{H}^z y = \lambda^z x$$

$$-\epsilon^z {\widehat{H}^z}^T x + y = \lambda^z y$$

The variable x is eliminated from the above equation, resulting in the following equation.

$$\epsilon^z \Phi^z y = (-{\lambda^z}^2 + \lambda^z) y$$

In the above equation, the vector y exactly satisfies the form of the eigenvector of Φ^z, with the corresponding eigenvalue being $(-{\lambda^z}^2 + \lambda^z)/\epsilon^z$. Let the eigenvalues of the matrix Φ^z be denoted as γ^z. Since Φ^z is a symmetric positive definite matrix, it follows that $\gamma^z > 0$.

$$\gamma^z = \frac{-{\lambda^z}^2 + \lambda^z}{\epsilon^z}$$

The transformed equation is expressed as follows.

$${\lambda^z}^2 + \lambda^z + \epsilon^z \gamma^z = 0$$

This equation takes the form of a quadratic equation, and the variable λ^z can be expressed in terms of ϵ^z and γ^z.

$$\lambda^z = \frac{1 \pm \sqrt{1 - 4\epsilon^z \gamma^z}}{2}$$

To ensure the iterative equation converges, the spectral radius $\rho(A^z)$ must satisfy $\rho(A^z) < 1$, which implies that the modulus of any λ^z must be less than 1. Since the parameter ϵ^z represents the step size in gradient descent, it must satisfy $\epsilon^z > 0$. The parameter γ^z denotes the eigenvalue of a symmetric positive-definite matrix, which is also a positive value. Thus, the term $4\epsilon^z\gamma^z$ is non-negative. For the term $1 - 4\epsilon^z\gamma^z$, we can analyze it under two cases: when it is greater than or equal to zero, and when it is less than zero. If $1 - 4\epsilon^z\gamma^z \ge 0$, then this term is necessarily less than or equal to 1. Consequently, the parameter λ^z is guaranteed to satisfy $\lambda^z < 1$, ensuring$\rho(A^z) < 1$. In this case, the parameter ϵ^z does not require a specific range of values. If $1 - 4\epsilon^z\gamma^z < 0$, the modulus of λ^z is given as follows.

$$\left\|\lambda^z\right\|^2 = \frac{1 + i\sqrt{4\epsilon^z\gamma^z - 1}}{2} \times \frac{1 - i\sqrt{4\epsilon^z\gamma^z - 1}}{2}$$
$$= \frac{1 + 4\epsilon^z\gamma^z - 1}{4}$$
$$= \epsilon^z\gamma^z$$

To ensure the convergence of the iterative algorithm, the spectral radius $\rho(A^z)$ must satisfy$\rho(A^z) < 1$. Consequently, the modulus of any eigenvalue λ_z must adhere to the following condition.

$$\left\|\lambda^z\right\|^2 = \epsilon^z\gamma^z < 1$$

$$\epsilon^z < \frac{1}{\gamma^z}$$

Since ϵ^z must account for all possible eigenvalues γ^z, it follows that $\epsilon^z < \frac{1}{\gamma_{\max}^z}$ or equivalently $\epsilon^z < \frac{1}{\rho(\Phi^z)}$, where $\gamma_{\max}^z$ is the largest eigenvalue. At the same time, since ϵ^z serves as the step size in the gradient descent algorithm, it must satisfy $\epsilon^z > 0$. Therefore, the feasible range for the parameter ϵ^z is as follows.

$$0 < \epsilon^{z} < \frac{1}{\rho(\Phi^{z})}$$

REFERENCES

[1] Z. Lin, M. Fu, and Y. Diao, "Distributed Self Localization for Relative Position Sensing Networks in 2D Space," IEEE Transactions on signal processing, vol. 63, no. 14, july 15, 2015.

[2] T. Eren, O. K. Goldenberg, W. Whiteley, Y. R. Yang, A. S. Morse, B. D. O. Anderson, and P. N. Belhumeur, "Rigidity, computation, and randomization in network localization," in Proc. 23rd Annu. Joint Conf. IEEE Comput. Commun. Soc., 2004, pp. 2673–2684.

[3] P. Barooah and J. P. Hespanha, "Estimation from relative measurements: Electrical analogy and large graphs," IEEE Trans. Signal Process., vol. 56, no. 6, pp. 2181–2193, 2008.

[4] S. Gortler, A. D. Healy, and D. P. Thurston, "Characterizing generic global rigidity," Amer. J. Math.., vol. 132, pp. 1–42, 2010.