A linear SLAM backend optimization algorithm based on virtual coordinate system in 3D space

Zhitao Wu

Hangzhou Dianzi University

Abstract: Common SLAM back-end optimization methods include filtering approaches and nonlinear optimization techniques. Among these, the extended Kalman filter (EKF) is a commonly used filtering method. This approach treats the robot's pose and all landmarks as state variables, which are updated using observational data. However, as the map becomes larger, the dimensionality of the state variables increases rapidly, leading to a significant growth in computational complexity. Nonlinear optimization methods, on the other hand, construct an error term by associating the spatial coordinates of observed feature points with the robot's pose. All error terms are then summed to form an error function, and the robot's pose and feature point positions are estimated by minimizing this error function. Nevertheless, this approach also suffers from high computational demands. This paper proposes an algorithm for backend optimization using systems of linear equations and gravity direction, where linear equations are utilized to represent the pose constraints between different keyframes.

Keywords: SLAM, backend optimizaiton, linear equation.

1. Introduction

Common SLAM back-end optimization methods include filtering approaches and nonlinear optimization techniques. Among these, the extended Kalman filter (EKF) is a commonly used filtering method. This approach treats the robot's pose and all landmarks as state variables, which are updated using observational data. However, as the map becomes larger, the dimensionality of the state variables increases rapidly, leading to a significant growth in computational complexity. Nonlinear optimization methods, on the other hand, construct an error term by associating the spatial coordinates of observed feature points with the robot's pose. All error terms are then summed to form an error function, and the robot's pose and feature point positions are estimated by minimizing this error function. Nevertheless, this approach also suffers from high computational demands. This paper proposes an algorithm for backend optimization using systems of linear equations and gravity direction, where linear equations are utilized to represent the pose constraints between different keyframes.

2. Representing geometric shapes using linear equations

2.1 Representing triangles using linear equations

We use linear equations to represent the shape of an arbitrary triangle in a plane. Let the coordinates of the three vertices of a triangle ijk in the plane be $p_i, p_j, p_k \in R^2$. Given that the length of edge ij is ρ_{ij}, the length of edge ik is ρ_{ik}, and the angle $\angle jik$ is θ_{jik}, we can derive the constraints for congruent triangles using these three conditions. To express the pose constraints between nodes using linear equations, we relax the congruence constraints into similarity constraints, defined as $\rho_{jik} = \rho_{ik}/\rho_{ij}, \theta_{jik}$. This relaxation ensures that the shape of triangle ijk remains unchanged, but allows the triangle to scale uniformly. Under this relaxation, the similarity constraints for the triangle can be represented using the following system of linear equations.

$$p_k - p_i = \rho_{jik} R_{jik} (p_j - p_i)$$

$$R_{jik} = \begin{bmatrix} \cos(\theta_{jik}) & -\sin(\theta_{jik}) \\ \sin(\theta_{jik}) & \cos(\theta_{jik}) \end{bmatrix}$$

The variables p_i, p_j and p_k represent the coordinates of nodes i, j, and k, respectively, and are variables that need to be constrained. The term $R(p_j - p_i)$ represents the vector ij rotated by an angle θ_{jik}, resulting in a vector that aligns in the same direction as vector ik. Moreover, there exists a proportional relationship $\rho_{jik} = \rho_{ik}/\rho_{ij}$ between $R(p_j - p_i)$ and vector ik. Here, both ρ and the rotation matrix R can be computed using the pose transformations T_{ji} and T_{ik}, which are known constants.

In [1], an equivalent form of the above equations is presented using linear equations in complex numbers. Let the complex coordinate of node i (where $i \in V$) be expressed as $p_i^c = x_i + iy_i$, where x_i and y_i are the x-axis and y-axis components of the coordinates of node i, respectively. Then, the above system of linear equations can be expressed using the following linear equations in complex numbers.

$$p_k^c - p_i^c = \omega_{jik}(p_j^c - p_i^c)$$

$$\omega_{jik} = \rho_{jik} e^{i\theta_{jik}}$$

Let p_{ki}^c represent the complex coordinate form of vector ikikik, and θ_{ki} denote the angle of vector ikikik. Similarly, let p_{ji}^c represent the complex coordinate form of vector ij, and θ_{ji} denote the angle of vector ij.

$$p_{ki}^c = p_k^c - p_i^c = \rho_{ki} e\text{^}(i\theta_{ki})$$

$$p_{ji}^c = p_j^c - p_i^c = \rho_{ji} e^{i\theta_{ji}}$$

By substituting p_{ki}^c and p_{ji}^c into the above equations in place of p_k^c, p_i^c and p_j^c, we obtain the following equations.

$$\rho_{ki} e^{i\theta_{ki}} = \omega_{jik} \rho_{ji} e^{i\theta_{ji}}$$

$$\rho_{ki} e\text{^}(i\theta_{ki}) = \rho_{jik} \rho_{ji} e^{i(\theta_{ji} + \theta_{jik})}$$

The above equations can be further transformed into constraints on the magnitude and angle, expressed as follows.

$$\rho_{jik} = \rho_{ik} / \rho_{ij}$$

$$\theta_{ji} + \theta_{jik} = \theta_{ki}$$

The constraints are exactly the same as the similarity triangle constraints for nodes i, j, and k. Therefore, the aforementioned linear equations in the complex domain can equivalently represent the geometric constraints of triangle jik. Whether using linear equations or linear equations in the complex domain, both can equivalently describe the constraints of triangle jik. In the subsequent text, we will use linear equations in the complex domain to represent the pose constraints among the three nodes, as the complex linear equations are relatively more concise, involving only a single equation. The coordinates of any node i are represented in the complex domain as $p_i^c = x_i + iy_i$.

If any two nodes in triangle ijk coincide, the equation will degenerate. Let us assume nodes i and k coincide, then $\rho_{ik} = 0$ and $\rho_{jik} = \rho_{ik} / \rho_{ij} = 0$. The parameter $\omega_{jik} = \rho_{jik} e^{i\theta_{jik}} = 0$. Consequently, the complex linear equation becomes:

$$p_k^c - p_i^c = 0 \times (p_j^c - p_i^c)$$

When nodes i and k coincide, the equation degenerates to $p_i^c = p_k^c$. Similarly, if nodes i and j coincide, the equation degenerates to $p_i^c = p_j^c$. If nodes j and k coincide, the equation becomes $p_j^c = p_k^c$. Finally, if nodes i, j, and k all coincide, the equation simplifies to $p_i^c = p_j^c = p_k^c$. Therefore, we introduce the following judgment steps.

$$p_i^c = p_j^c, \text{if } \rho_{ij} < \epsilon_{thres}$$

$$p_i^c = p_k^c, \text{if } \rho_{ik} < \epsilon_{thres}$$

$$p_j^c = p_k^c, \text{if } \rho_{jk} < \epsilon_{thres}$$

$$p_i^c = p_j^c = p_k^c, \text{if any two of } \rho_{ij}, \rho_{ik}, \rho_{jk} \text{ are less than } \epsilon_{thres}$$

Here, ϵ_{thres} is the threshold used to determine whether two nodes overlap.

We then provide the conditions for a specific localizability problem. In triangle ijk, if the coordinates of node i and node k are known, the coordinates of node j can be calculated using a linear equation under the following conditions. If nodes i and k are not coincident, node j is localizable. If nodes i and k are coincident, the linear equation degenerates into $p_i^c = p_j^c$. This equation is independent of the coordinates of node j, meaning the coordinates of node j cannot be determined using the linear equation. From a geometric perspective, we can reach the same conclusion: if nodes i and k are coincident, knowing ρ_{ij} and ρ_{kj} is insufficient to uniquely determine the coordinates of node j. In this case, node j lies on a circle.

2.2 Representing spatial triangles using linear equations

In 3D space, four nodes i, j, k, and l can form multiple triangles, such as triangle ijl, triangle ikl, and triangle jkl. If these nodes lie on the same plane, the shape of each triangle can be described using the linear equations introduced in the previous section. However, if the nodes do not lie on the same plane, we can construct the linear equations using the method proposed in [2]. This method utilizes the gravity direction provided by the IMU sensor to project the constraints of the similar triangles in space onto the horizontal plane.

We discuss here the linear equation representation of a triangle ijk located in three-dimensional space. Assume that the coordinates of the three vertices of triangle ijk in the global coordinate system Σ_g (where the z-axis direction is opposite to the direction of gravity) are p_i, p_j and $p_k \in R^3$. In the local coordinate system Σ_i, which uses node i as the origin, the relative position of node j is measured as $p_j^r = [x_{ij}^r, y_{ij}^r, z_{ij}^r]^T$, and the relative position of node k in the same local coordinate system is measured as $p_k^r = [x_{ik}^r, y_{ik}^r, z_{ik}^r]^T$. There exists an unknown rotation matrix R_i that defines the transformation between the local coordinate system Σ_i and the global coordinate system Σ_g.

We utilized the gravity direction obtained from the IMU sensor to transform the local coordinate system Σ_i into a new local coordinate system Σ_{iz}, where the z-axis is aligned opposite to the gravity direction (using a rotation matrix denoted as R_i^{cor}). Below, we provide the method for calculating the rotation matrix R_i^{cor}. In the local coordinate system Σ_i, the gravity direction measured by the IMU 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 practical applications, it is also possible to construct a robot motion model and incorporate the magnitude of gravitational acceleration. Techniques such as Kalman filtering can then be applied to estimate a more accurate gravity direction.

We define the z-axis of the global coordinate system to be opposite to the direction of gravity. For convenience, we use 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 relative position detection sensor on the sensor node is aligned with the z-axis of the IMU, which is defined as $z = [0,0,1]^T$.

We need to rotate the local coordinate system Σ_i such that the z-axis of the rotated local coordinate system Σ_{iz} is aligned opposite to the direction of gravity. Specifically, this involves rotating the local coordinate system Σ_i about the normal vector of the plane constructed by three points: "the IMU's z-axis, node i, and the direction opposite to gravity g_{inv}" (denoted as vector $n \in R^3$). The rotation angle is the angle between "the IMU's z-axis, node i, and the direction opposite to gravity g_{inv}," denoted as$\alpha \in (-\pi, \pi]$. In the local

coordinate system Σ_i, the IMU's z-axis and the direction opposite to gravity are represented as follows.

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

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

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

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

The rotation angle α can be determined using the dot product as follows.

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

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

Subsequently, the rotation vector v (where $v \in R^3$) can be computed.

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

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

R_i^cor=phi(v)

The rotation of the coordinate system itself is opposite to the rotation of the observations within the coordinate system. The relative position of node j in the corrected local coordinate system Σ_{iz} is as follows.

$$p_j^{cor} = (R_i^{cor})^{-1}\left[x_{ij}^r, y_{ij}^r, z_{ij}^r\right]^T = \left[x_{ij}^{cor}, y_{ij}^{cor}, z_{ij}^{cor}\right]^T$$

The relative position of node k in the corrected local coordinate system Σ_{iz} is as follows.

$$p_k^{cor} = (R_i^{cor})^{-1}[x_{ik}^r, y_{ik}^r, z_{ik}^r]^T = [x_{ik}^{cor}, y_{ik}^{cor}, z_{ik}^{cor}]^T$$

The corrected relative positions $p_j^{cor}, p_k^{cor} \in R^3$ are projected onto the horizontal plane, with the projection coordinates given as follows.

$$p_j^{xy} = \left[x_{ij}^{cor}, y_{ij}^{cor}\right]^T$$

$$p_k^{xy} = [x_{ik}^{cor}, y_{ik}^{cor}]^T$$

$$p_i^{xy} = [0,0]^T$$

The projection coordinates of the three vertices i,j,k of the spatial triangle ijk onto the horizontal plane, denoted as $p_i^{xy}, p_j^{xy}, p_k^{xy}$, can be expressed using the following linear equations.

$$p_k^{xy} - p_i^{xy} = \omega_{jik}^{xy}(p_j^{xy} - p_i^{xy})$$

$$\omega_{jik}^{xy} = \rho_{jik}^{xy} e^{i\theta_{jik}^{xy}}$$

The parameter ρ_{jik}^{xy} is calculated as follows.

$$\rho_{jik}^{xy} = \rho_{ik}^{xy}/\rho_{ij}^{xy}$$

$$\rho_{ij}^{xy} = \left|\left|p_j^{xy} - p_i^{xy}\right|\right|$$

$$\rho_{ik}^{xy} = \left|\left|p_k^{xy} - p_i^{xy}\right|\right|$$

The angle θ_{jik}^{xy} is calculated as follows.

$$\cos\left(\theta_{jik}^{xy}\right) = \frac{p_j^{xy} \times p_k^{xy}}{\left|\left|p_j^{xy}\right|\right|\left|\left|p_k^{xy}\right|\right|}$$

$$\theta_{jik}^{xy} = \arccos\left(\frac{p_j^{xy} \times p_k^{xy}}{\left|\left|p_j^{xy}\right|\right|\left|\left|p_k^{xy}\right|\right|}\right)$$

Next, we discuss the linear equation for the vertical components of the three vertices i, j, and k of triangle ijk. Let $p_i^z \in R$ represent the z-axis component of node i in the global coordinate system Σ_g, $p_j^z \in R$ represent the z-axis component of node j, and $p_k^z \in R$ represent the z-axis component of node k. Since the z-axis direction of the corrected local coordinate system Σ_{iz} is aligned with the z-axis direction of the global coordinate system Σ_g, both being vertical, we use the relative distances of nodes i, j, and k along the z-axis in the local coordinate system Σ_{iz} as constraint information for p_i^z, p_j^z, p_k^z.

$$p_j^z - p_i^z = p_{ij}^{cor}$$

$$p_k^z - p_i^z = p_{ik}^{cor}$$

For the spatial triangle ijk, we can use three linear equations to represent the coordinate relationships of nodes i, j, and k in the global coordinate system Σ_g. Geometrically, these three linear equations impose the following constraints on the spatial triangle ijk. First, the projection onto the horizontal plane (x-y plane) ensures that nodes i, j, and k form a similar triangle in the x-y direction, preserving their geometric shape. The shape of the triangle formed by nodes i, j, and k on the x-y plane is fixed, but its size may scale up or down. Second, the linear equation in the vertical direction ensures that the relative distances between nodes i, j, and k in the vertical (z-axis) direction are determined. Consequently, the spatial triangle ijk cannot rotate around the x-axis or y-axis, as such rotation would alter the relative distances in the vertical direction. Thus, these three equations collectively determine both the arrangement of nodes i, j, and k in the vertical direction and the shape of their projection onto the x-y plane.

The following discussion addresses the degeneracy issue of the linear equation when nodes i, j, and k are projected onto

the horizontal plane. The coordinates of the projections of nodes i, j, and k on the horizontal plane are denoted as p_i^{xy}, p_j^{xy} and p_k^{xy}, respectively. If any two of these planar coordinates coincide, the behavior of the linear equation changes as follows.

$$p_i^{xy} = p_j^{xy}, \text{if } \rho_{ij}^{xy} < \epsilon_{thres}$$

$$p_i^{xy} = p_k^{xy}, \text{if } \rho_{ik}^{xy} < \epsilon_{thres}$$

$$p_j^{xy} = p_k^{xy}, \text{if } \rho_{jk}^{xy} < \epsilon_{thres}$$

$$p_i^{xy} = p_j^{xy} = p_k^{xy}, \text{if any two of } \rho_{ij}^{xy}, \rho_{ik}^{xy}, \rho_{jk}^{xy} \text{ are less than } \epsilon_{thres}$$

Here, ρ_{ij}^{xy} represents the distance between the planar coordinates p_i^{xy} and p_j^{xy}; ρ_{ik}^{xy} is the distance between p_i^{xy} and p_k^{xy}; and ρ_{jk}^{xy} is the distance between p_j^{xy} and p_k^{xy}.

We then present the conditions for a specific locatability problem. In a triangular element ijk within three-dimensional space, if the ground plane coordinates of nodes i and k, denoted as p_i^{xy} and p_k^{xy}, are known, and the ground plane coordinates of node j, denoted as p_j^{xy}, need to be calculated using a linear equation, the conditions for determining the coordinates of node j are as follows: If nodes i and k are not coincident and do not lie on the same vertical line, then the coordinates of node j on the ground plane are locatable. Conversely, if nodes i and k are coincident, or if nodes i and k lie on the same vertical line, the linear equation for the ground plane degenerates into $p_i^{xy} = p_k^{xy}$. In this case, the linear equation is unrelated to the ground plane coordinates p_j^{xy} of node, and thus the coordinates of node j cannot be determined using the linear equation.

For the positioning problem of the vertical coordinate p_j^z of node j, there are no specific requirements. Regardless of whether nodes i and k coincide or are located on the same vertical line, we only need to use either Equation (1) or Equation (2) to calculate p_j^z.

Therefore, for a triangle ijk in three-dimensional space, if the coordinates of node i, $p_i \in R^3$, and node k, $p_k \in R^3$, are known, the condition for locating the coordinates of node jjj, p_j, is as follows: nodes i and k must not coincide, and they must not lie on the same vertical line.

2.3 Representing tetrahedrons using linear equations

In this section, we represent the shape of an arbitrary tetrahedron in space using linear equations. A tetrahedron consists of four triangular faces; if the shapes of three of these faces are determined, the overall shape of the tetrahedron is also fully defined. To express the constraints between nodes using linear equations, we employ the conditions of similar triangles to constrain each triangular face. For example, for a given triangular ijk, the similarity constraint can be expressed as $\rho_{jik} = \rho_{ik}/\rho_{ij}, \theta_{jik}$.

For the tetrahedron ijkl, if the similarity constraints of three of its faces are known, the shape of the entire tetrahedron becomes fully determined (up to scaling). Let the similarity constraints for the triangles ijl, ikl, and jkl be as follows.

$$\rho_{ilj} = \rho_{lj}/\rho_{li}, \theta_{ilj}$$

$$\rho_{ilk} = \rho_{lk}/\rho_{li}, \theta_{ilk}$$

$$\rho_{jlk} = \rho_{lk}/\rho_{lj}, \theta_{jlk}$$

For each of the aforementioned similar triangle constraints, we can represent them using the linear equations of triangles in 3D space. Let the coordinates of nodes i,j,k,l in the global coordinate system Σ_g be $p_i, p_j, p_k, p_l \in R^3$. The components of nodes i,j,k,l along the x-axis and y-axis are denoted as $p_i^{xy}, p_j^{xy}, p_k^{xy}, p_l^{xy} \in R^2$, and the components of nodes i,j,k,l along the z-axis are denoted as $p_i^z, p_j^z, p_k^z, p_l^z \in R$.

The relative coordinates of nodes i, j, and k in the local coordinate system Σ_l are given as $p_{li} = [x_{li}, y_{li}, z_{li}]^T, p_{lj} = [x_{lj}, y_{lj}, z_{lj}]^T$ and $p_{lk} = [x_{lk}, y_{lk}, z_{lk}]^T$, respectively. The rotation matrix that corrects the local coordinate system Σ_l to the adjusted local coordinate system Σ_{lz}, where the z-axis is aligned opposite to the direction of gravity, is denoted as R_l^{cor}. Therefore, the relative positions of nodes i, j and k in the adjusted local coordinate system Σ_{lz} are as follows.

$$p_{li}^{cor} = (R_{lz})^{-1}[x_{li}, y_{li}, z_{li}]^T = [x_{li}^{cor}, y_{li}^{cor}, z_{li}^{cor}]^T$$

$$p_{lj}^{cor} = (R_{lz})^{-1}[x_{lj}, y_{lj}, z_{lj}]^T = [x_{lj}^{cor}, y_{lj}^{cor}, z_{lj}^{cor}]^T$$

$$p_{lk}^{cor} = (R_{lz})^{-1}[x_{lk}, y_{lk}, z_{lk}]^T = [x_{lk}^{cor}, y_{lk}^{cor}, z_{lk}^{cor}]^T$$

The coordinates of triangle ijl in the global coordinate system, denoted as p_i, p_j and p_l, can be represented by the following three linear equations.

$$p_j^{xy} - p_l^{xy} = \omega_{ilj}(p_i^{xy} - p_l^{xy})$$

$$p_l^z - p_i^z = p_{il}^{cor}$$

$$p_l^z - p_j^z = p_{jl}^{cor}$$

The coordinates of triangle ikl in the global coordinate system, denoted as p_i, p_k and p_l, can be represented by the following three linear equations.

$$p_k^{xy} - p_l^{xy} = \omega_{ilk}(p_i^{xy} - p_l^{xy})$$

$$p_l^z - p_i^z = p_{il}^{cor}$$

$$p_l^z - p_k^z = p_{kl}^{cor}$$

The coordinates of triangle jkl in the global coordinate system, denoted as p_j, p_k and p_l, can be represented by the following three linear equations.

$$p_k^{xy} - p_l^{xy} = \omega_{jlk}(p_j^{xy} - p_l^{xy})$$

$$p_l^z - p_j^z = p_{jl}^{cor}$$

$$p_l^z - p_k^z = p_{kl}^{cor}$$

Equations (1) and (2) are identical, equations (3) and (4) are identical, and equations (5) and (6) are identical. Therefore, we can use six linear equations to represent the shape of an arbitrary tetrahedron ijkl. In fact, as discussed in the previous section, the linear equations for a spatial triangle not only

constrain the shape of the three nodes (denoted as nodes i, j, and k) but also impose constraints on the relative arrangement of these three nodes in the vertical direction. For the tetrahedron ijkl, it suffices to use the linear equations corresponding to two of its triangles (e.g., triangles ijl and ikl) to fully determine the shape of the tetrahedron ijkl in space as well as its vertical arrangement. However, to enhance the resistance of the linear equations to noise in all directions, we still use the linear equations of three triangles (e.g., triangles ijl, ikl, and jkl) to represent the tetrahedron ijkl.

For the tetrahedron ijkl, we use three triangular faces (e.g., triangles ijl, ikl, and jkl) to represent the relationship between the coordinates of the four nodes p_i, p_j, p_k, p_l through linear equations. Below, we present the geometric meaning of these linear equations. Based on the geometric interpretation of spatial triangles discussed in the previous section, the spatial triangle ijl corresponds to three linear equations. These three equations determine the relative vertical arrangement of nodes i, j, l and the shape of the triangle in the $x - y$ plane (similar triangles). Similarly, the tetrahedron ijkl corresponds to six linear equations, which determine the relative vertical arrangement of nodes i, j, k, l and the shape of the tetrahedron's projection in the $x - y$ plane (similar triangles).

In the tetrahedron ijkl within three-dimensional space, if the coordinates of nodes i, j, and k are known as p_i, p_j and $p_k \in R^3$, the coordinates of node l, denoted as $p_l \in R^3$, can be calculated using the linear equations derived from three triangles: triangle ijl, triangle ikl, and triangle jkl. In fact, according to the geometric positioning constraints of a triangle in space, it is generally sufficient to use the linear equation of a single triangle (e.g., triangle ijl) to determine the coordinates of node l. However, the calculation based on a single triangle requires that i and j are not coincident nodes and that i and j are not collinear with a line perpendicular to the plane of the triangle. To avoid these special cases and to improve the robustness of the linear equations against noise in all directions, we use the linear equations derived from all three triangles (triangle ijl, triangle ikl, and triangle jkl) to calculate the coordinates of node l. This approach employs six linear equations in total.

Then, we present the conditions for a specific localizability problem. In a tetrahedron ijkl in three-dimensional space, the coordinates of nodes i, j, k are given as $p_i, p_j, p_k \in R^3$. Using the linear equations derived from three triangles (triangle ijl, triangle ikl, triangle jkl), the coordinates of node l, denoted as $p_l \in R^3$, can be calculated. If the area of triangle ijk, denoted as S_{ijl}, is non-zero, then node l is localizable. The proof is as follows: Since the area of triangle ijk, S_{ijk}, is non-zero, nodes i, j, k are not collinear, and no two of the nodes i, j, k coincide. Thus, nodes i, j, k are not aligned along the same vertical line. Consequently, there exist two nodes (e.g., i and j) that are not on the same vertical line, and nodes i and j are distinct. Based on the conditions for localizability of a triangle in three-dimensional space, the coordinates of node k can be calculated using nodes i and j.

3. Representing pose transformation with linear equations

3.1 Constructing linear equations

In SLAM algorithms, there are various methods to obtain the pose transformation between two keyframes (in 3D space, represented as the rotation-translation matrix $T \in R^{4\times4}$). The table below lists different types of feature points obtained from camera sensors between the two keyframes, along with the corresponding algorithms that can be used to compute the pose transformation.

Table 1. Camera pose estimation algorithms

Feature point types	Transformation calculation algorithms
2d-2d	epipolar geometry
2d-2d(planar)	homography matrix
2d-3d	DLT, EPnP
3d-3d	ICP

Similarly, if two frames of point cloud data are obtained using a LiDAR sensor, various point cloud registration algorithms can be employed to estimate the pose transformation. In addition, the pose transformation between the two frames can also be obtained through methods such as IMU sensors, GPS sensors, wheel odometry, and loop detection.

No matter what type of sensor is used, the initial information we obtain is the pose transformation between two keyframes (nodes). The purpose of back-end optimization is to use this pose transformation information to estimate an accurate global map. Here, we use the node $i \in V$ to represent a keyframe, where each node i has its own local coordinate system, denoted as Σ_i. There exists an unknown rotation matrix R_i between the local coordinate system Σ_i of node i and the global coordinate system Σ_g. Let the 3D spatial coordinates of node i be $p_i = [x_i, y_i, z_i]^T$. Let the set of all pose transformations T_{ij} be denoted as $E = \{T_{ij}, i, j \in V\}$. We can define a directed graph $G = [V, E]$, where V represents the set of all nodes (keyframes), and E represents the set of all pose transformations.

We transform the pose transformation $T_{ij} \in R^{4\times4}$ between any two nodes i and j into a set of linear equation constraints. Suppose the coordinates of node i are represented as $p_i \in R^3$. We define a virtual node i_x located on the x-axis of the local coordinate system Σ_i with coordinates p_{ix}, satisfying $||p_{ix} - p_i|| = 1$. Similarly, we define a virtual node i_y located on the y-axis of Σ_i with coordinates p_{iy}, satisfying $||p_{iy} - p_i|| = 1$. Likewise, a virtual node i_z is defined on the z-axis of Σ_i with coordinates p_{iz}, satisfying $||p_{iz} - p_i|| = 1$. Thus, in the local coordinate system Σ_i, node i serves as the origin, while i_x, i_y and i_z are nodes representing unit vectors along the three axes of the local coordinate system.

The coordinates of the same node jjj are denoted as $p_j \in R^3$. We define a virtual node p_{jx} to lie along the x-axis of the

local coordinate system Σ_j, satisfying $||p_{jx} - p_j|| = 1$. Similarly, a virtual node p_{jy} is defined to lie along the y-axis of the local coordinate system Σ_j, satisfying $||p_{jz} - p_j|| = 1$. Furthermore, a virtual node Σ_j is defined to lie along the z-axis of the local coordinate system Σ_j, satisfying $||p_{jz} - p_j|| = 1$. In the local coordinate system Σ_j, node j serves as the origin of the coordinate system, while p_{jx}, p_{jy} and p_{jz} represent the unit coordinate vectors along the three axes of the local coordinate system.

If the coordinates of all points in the local coordinate system Σ_i are known as $(p_i, p_{ix}, p_{iy}, p_{iz})$, we need to construct linear equations to represent the coordinates of all points in another local coordinate system Σ_j as $(p_j, p_{jx}, p_{jy}, p_{jz})$. Through this approach, we can recursively determine the local coordinate system of all nodes, starting from the first node.

Given the coordinates of the three axes (p_{ix}, p_{iy}, p_{iz}) in the local coordinate system Σ_i, we first construct a linear equation for the virtual node coordinate p_{jx} in the local coordinate system Σ_j. By utilizing the linear equation representation method of a tetrahedron, we can construct linear equations for three triangles: the triangle $j_x - i_x - i_y$, the triangle $j_x - i_x - i_z$, and the triangle $j_x - i_y - i_z$.

$$p_{iy}^{xy} - p_{jx}^{xy} = \omega_{ix,jx,iy}(p_{ix}^{xy} - p_{jx}^{xy})$$
$$p_{iz}^{xy} - p_{jx}^{xy} = \omega_{ix,jx,iz}(p_{ix}^{xy} - p_{jx}^{xy})$$
$$p_{iz}^{xy} - p_{jx}^{xy} = \omega_{iy,jx,iz}(p_{iy}^{xy} - p_{jx}^{xy})$$
$$p_{jx}^{z} - p_{ix}^{z} = p_{ix,jx}^{cor}$$
$$p_{jx}^{z} - p_{iy}^{z} = p_{iy,jx}^{cor}$$
$$p_{jx}^{z} - p_{iz}^{z} = p_{iz,jx}^{cor}$$

The coordinates $p_{ix}^{xy}, p_{iy}^{xy}, p_{iz}^{xy}, p_{jx}^{xy} \in C$ represent the projections of nodes i_x, i_y, i_z, j_x onto the horizontal plane, respectively. The coordinates $p_{ix}^{z}, p_{iy}^{z}, p_{iz}^{z}, p_{jx}^{z} \in R$ represent the vertical coordinates of nodes i_x, i_y, i_z, j_x, respectively. The parameters $\omega_{ix,jx,iy}, \omega_{ix,jx,iz}, \omega_{iy,jx,iz}, p_{ix,jx}^{cor}, p_{iy,jx}^{cor}, p_{iz,jx}^{cor}$ will be specifically defined later.

Next, we construct a system of linear equations using the virtual node coordinate p_{jy} from Σ_j and the three points p_{ix}, p_{iy}, p_{iz} fromΣ. By employing the representation method of tetrahedral equations, we build linear equations for three triangles: triangle $j_y - i_x - i_y$, triangle $j_y - i_x - i_z$, and triangle $j_y - i_y - i_z$.

$$p_{iy}^{xy} - p_{jy}^{xy} = \omega_{ix,jy,iy}(p_{ix}^{xy} - p_{jy}^{xy})$$
$$p_{iz}^{xy} - p_{jy}^{xy} = \omega_{ix,jy,iz}(p_{ix}^{xy} - p_{jy}^{xy})$$
$$p_{iz}^{xy} - p_{jy}^{xy} = \omega_{iy,jy,iz}(p_{iy}^{xy} - p_{jy}^{xy})$$
$$p_{jy}^{z} - p_{ix}^{z} = p_{ix,jy}^{cor}$$
$$p_{jy}^{z} - p_{iy}^{z} = p_{iy,jy}^{cor}$$
$$p_{jy}^{z} - p_{iz}^{z} = p_{iz,jy}^{cor}$$

The coordinates $p_{ix}^{xy}, p_{iy}^{xy}, p_{iz}^{xy}, p_{jy}^{xy} \in C$ represent the projections of nodes i_x, i_y, i_z and j_y onto the horizontal plane, respectively. The coordinates $p_{ix}^{z}, p_{iy}^{z}, p_{iz}^{z}, p_{jy}^{z} \in R$ represent the vertical coordinates of nodes i_x, i_y, i_z and j_y, respectively. The parameters $\omega_{ix,jy,iy}, \omega_{ix,jy,iz}, \omega_{iy,jy,iz}, p_{ix,jy}^{cor}, p_{iy,jy}^{cor}, p_{iz,jy}^{cor}$ will be specifically defined in subsequent sections.

Next, we use the virtual node coordinate p_{jz} from Σ_j and the three points p_{ix}, p_{iy} and p_{iz} from Σ_i to construct a system of linear equations. By applying the representation methodology of tetrahedron equations, three linear equations corresponding to triangular faces (triangle $j_z - i_x - i_y$, triangle $j_z - i_x - i_z$, and triangle $j_z - i_y - i_z$) are constructed.

$$p_{iy}^{xy} - p_{jz}^{xy} = \omega_{ix,jz,iy}(p_{ix}^{xy} - p_{jz}^{xy})$$
$$p_{iz}^{xy} - p_{jz}^{xy} = \omega_{ix,jz,iz}(p_{ix}^{xy} - p_{jz}^{xy})$$
$$p_{iz}^{xy} - p_{jz}^{xy} = \omega_{iy,jz,iz}(p_{iy}^{xy} - p_{jz}^{xy})$$
$$p_{jz}^{z} - p_{ix}^{z} = p_{ix,jz}^{cor}$$
$$p_{jz}^{z} - p_{iy}^{z} = p_{iy,jz}^{cor}$$
$$p_{jz}^{z} - p_{iz}^{z} = p_{iz,jz}^{cor}$$

The coordinates $p_{ix}^{xy}, p_{iy}^{xy}, p_{iz}^{xy}, p_{jz}^{xy} \in C$ represent the projections of nodes i_x, i_y, i_z, j_z onto the horizontal plane, respectively. The coordinates $p_{ix}^{z}, p_{iy}^{z}, p_{iz}^{z}, p_{jz}^{z} \in R$ represent the vertical coordinates of nodes i_x, i_y, i_z, j_z respectively. The parameters $\omega_{ix,jz,iy}, \omega_{ix,jz,iz}, \omega_{iy,jz,iz}, p_{ix,jz}^{cor}, p_{iy,jz}^{cor}, p_{iz,jz}^{cor}$ will be specifically defined in subsequent sections.

Next, we use the node coordinates p_i in Σ_i along with the three points p_{ix}, p_{iy}, p_{iz} in Σ_i to construct a system of linear equations. Using the representation of the tetrahedral equation, we construct the linear equations for three triangles: triangle $i - i_x - i_y$, triangle $i - i_x - i_z$, and triangle $i - i_y - i_z$.

$$p_{ix}^{xy} - p_{i}^{xy} = \omega_{ix,i,iy}(p_{ix}^{xy} - p_{i}^{xy})$$
$$p_{iz}^{xy} - p_{i}^{xy} = \omega_{ix,i,iz}(p_{ix}^{xy} - p_{i}^{xy})$$
$$p_{iz}^{xy} - p_{i}^{xy} = \omega_{iy,i,iz}(p_{iy}^{xy} - p_{i}^{xy})$$
$$p_{i}^{z} - p_{ix}^{z} = p_{ix,i}^{cor}$$
$$p_{i}^{z} - p_{iy}^{z} = p_{iy,i}^{cor}$$
$$p_{i}^{z} - p_{iz}^{z} = p_{iz,i}^{cor}$$

The variables $p_{ix}^{xy}, p_{iy}^{xy}, p_{iz}^{xy}, p_{i}^{xy} \in C$ represent the coordinates of nodes i_x, i_y, i_z, i projected onto the horizontal plane, respectively. Similarly, $p_{ix}^{z}, p_{iy}^{z}, p_{iz}^{z}, p_{i}^{z} \in R$ denote the vertical coordinates of nodes i_x, i_y, i_z, i. The parameters $\omega_{ix,i,iy}, \omega_{ix,i,iz}, \omega_{iy,i,iz}, p_{ix,i}^{cor}, p_{iy,i}^{cor}, p_{iz,i}^{cor}$ will be specifically defined later in the text.

We have already used three virtual nodes, i_x, i_y and i_z in the local coordinate system Σ_i to represent node j and its three virtual nodes j_x, j_y and j_z in the local coordinate system Σ_j. Since the virtual nodes i_x, i_y and i_z are unit vectors along the

three axes of the local coordinate system Σ_i, the area of the triangle $i_x - i_y - i_z$, denoted as $S_{ix,iy,iz}$, is non-zero (satisfying the condition for the tetrahedron to be well-defined). If the coordinates of the three virtual nodes p_{ix}, p_{iy} and p_{iz} in the local coordinate system Σ_i are known, the coordinates of node j (p_j) and its three virtual nodes p_{jx}, p_{jy} and p_{jz} in the local coordinate system Σ_j can be calculated. By extending this method, all nodes (keyframes) in the SLAM problem can be connected sequentially in a chain. If the pose of the first node (the coordinates of its three virtual nodes, p_{1x}, p_{1y} and p_{1z}) is known, the pose of any node i (the coordinates of its three virtual nodes, p_{ix}, p_{iy} and p_{iz}) can be computed.

Given the coordinates of three virtual nodes, i_x, i_y and i_z, in the local coordinate system Σ_i, the coordinates of three virtual nodes, j_x, j_y and j_z, in another local coordinate system Σ_j, can be calculated using a set of linear equations. Here, we discuss the inverse problem: if the coordinates of the three virtual nodes j_x, j_y and j_z in Σ_j are known, can the coordinates of the three virtual nodes i_x, i_y and i_z in Σ_i be computed using the aforementioned linear equations? We can interpret the meaning of these linear equations from a geometric perspective. The six linear equations corresponding to the tetrahedron $i_x - i_y - i_z - j_x$ geometrically constrain the relative distances of nodes i_x, i_y, i_z, j_x in the vertical direction and the shape in the x-y plane (similar triangles). If we relax these geometric constraints by replacing the relative distances of nodes i_x, i_y, i_z, j_x in the vertical direction with the relative distance ratios, then the linear equations constrain at least the shape of the tetrahedron $i_x - i_y - i_z - j_x$, but its scale becomes flexible. Similarly, other sets of linear equations constrain the shapes of the tetrahedra $i_x - i_y - i_z - j_y$ and $i_x - i_y - i_z - j_z$, but their scales remain flexible. Consequently, the geometric shape formed by the six nodes ($i_x, i_y, i_z, j_x. j_y, j_z$) is determined, but its overall scale can vary. Therefore, given the coordinates of the virtual nodes j_x, j_y and j_z, the linear equation system can also be used to compute the coordinates of the nodes i_x, i_y and i_z. Hence, the 18 linear equations corresponding to the three tetrahedra ($i_x - i_y - i_z - j_x$, $i_x - i_y - i_z - j_y$, and $i_x - i_y - i_z - j_z$) can equivalently represent the pose transformation T_{ij} (with the scale along the x-y axes determined in subsequent calculations).

In the aforementioned system of linear equations, we use the three virtual nodes i_x, i_y, i_z in the local coordinate system Σ_i to represent the three virtual nodes j_x, j_y, j_z in the local coordinate system Σ_j. To ensure symmetry, we could similarly use the three virtual nodes j_x, j_y, j_z in Σ_j to represent the three virtual nodes i_x, i_y, i_z in Σ_i, and list the corresponding nine linear equations. However, as previously proven, the 18 linear equations that use the nodes i_x, i_y, i_z to represent j_x, j_y, j_z are already sufficient to equivalently represent the pose transformation T_{ij}. Therefore, using the nodes j_x, j_y, j_z to represent i_x, i_y, i_z again would be redundant.

In the graph $G = [V, E]$, V represents the set of all nodes (keyframes). Assume there are n nodes in the set V. For each node $i \in V$, three virtual nodes, i_x, i_y and i_z, are defined in its local coordinate system Σ_i. Consequently, we define the set of virtual nodes as $V_v = \{i_x, i_y, i_z, i \in V\}$. The coordinates of these virtual nodes are denoted as $p_{ix}, p_{iy}, p_{iz} \in R^3$ where $i_x, i_y, i_z \in V_v$. The coordinates of the virtual nodes in the plane are represented as $p_{ix}^{xy}, p_{iy}^{xy}, p_{iz}^{xy} \in C$ where $i_x, i_y, i_z \in V_v$, and the coordinates in the vertical direction are represented as $p_{ix}^{z}, p_{iy}^{z}, p_{iz}^{z}$. We define the coordinate vector of the virtual nodes as $p_v = [p_{1x}, p_{1y}, p_{1z}, \dots, p_{nx}, p_{ny}, p_{nz}] \in R^{3\times 3n}$.

The coordinate vector of the virtual nodes in the plane is defined as $p_v^{xy} = [p_{1x}^{xy}, p_{1y}^{xy}, p_{1z}^{xy}, \dots, p_{nx}^{xy}, p_{ny}^{xy}, p_{nz}^{xy}] \in C^{3n}$,

and the coordinate vector of the virtual nodes in the vertical direction is defined as

$p_v^z = [p_{1x}^z, p_{1y}^z, p_{1z}^z, \dots, p_{nx}^z, p_{ny}^z, p_{nz}^z]^T \in R^{3n}$.

Given that the set of all pose transformations T_{ij} is denoted as $E = \{T_{ij}, i, j \in V\}$, the pose transformation T_{ij} between any two nodes (node i and node j) can be expressed as 18 linear equations (9 planar coordinate equations and 9 vertical coordinate equations). Therefore, we define the system of linear equations corresponding to all pose transformations $T_{ij} \in E$ as follows。

$$A^{xy} p_v^{xy} = b^{xy}$$

$$A^{z} p_v^{xy} = b^{z}$$

In addition, we represent the coordinates of node i in the local coordinate system Σ_i using the coordinates of three virtual nodes, i_x, i_y and i_z. Once the coordinates of all virtual nodes, p_v, are computed, we can use p_v to determine the coordinates of each node $i \in V$ in the graph $G = [V, E]$. We define the coordinate vector of all nodes $i \in V$ as $p = [p_1, p_2, \dots, p_n] \in R^{3\times n}$. The planar coordinate vector of all nodes $i \in V$ is defined as $p^{xy} = [p_1^{xy}, p_2^{xy}, \dots, p_n^{xy}] \in C^n$, and the vertical coordinate vector is defined as $p^z = [p_1^z, p_2^z, \dots, p_n^z]^T \in R^n$. The equations to compute the coordinates p^{xy} and p^z of all nodes $i \in V$ using the virtual node coordinates p_v^{xy} and p_v^z are as follows.

$$C^{xy} p^{xy} = D^{xy} p_v^{xy}$$

$$C^{z} p^{z} = D^{z} p_v^{z}$$

The pseudocode of the algorithm is as follows.

Procedure 1: Representing pose transformation with linear equations

---

Input: Graph $G = [V, E]$, where V represents the set of n nodes, and E represents the set of all pose transformations

Output: Matrices $A^{xy}, b^{xy}, A^z, b^z, C^{xy}, D^{xy}, C^z, D^z$

Let the vector of all nodes be $p = [p_1, p_2, \dots, p_n] \in R^{3\times n}$.

Let the planar vector of all nodes be $p^{xy} = [p_1^{xy}, p_2^{xy}, \dots, p_n^{xy}]^T \in C$.

Let the vertical vector of all nodes be $p^z = [p_1^z, p_2^z, \dots, p_n^z]^T \in$

R.

Let the vector of all virtual nodes be $p_v = [p_{1x}, p_{1y}, p_{1z}, \ldots, p_{nx}, p_{ny}, p_{nz}] \in R^{3\times 3n}$.

Let the planar vector of all virtual nodes be $p_v^{xy} = [p_{1x}^{xy}, p_{1y}^{xy}, p_{1z}^{xy}, \ldots, p_{nx}^{xy}, p_{ny}^{xy}, p_{nz}^{xy}]^T \in R^{3\times 3n}$.

Let the vertical gravity vector of all virtual nodes be $p_v^z = [p_{1x}^z, p_{1y}^z, p_{1z}^z, \ldots, p_{nx}^z, p_{ny}^z, p_{nz}^z]^T \in R^{3\times 3n}$.

For ($T_{ij} \in V$):

Construct the planar linear equation for the tetrahedron $i_x - i_y - i_z - j_x$ and incorporate it into the linear equation system $A^{xy}p_v^{xy} = b^{xy}$.

$$p_{iy}^{xy} - p_{jx}^{xy} = \omega_{ix,jx,iy}(p_{ix}^{xy} - p_{jx}^{xy})$$
$$p_{iz}^{xy} - p_{jx}^{xy} = \omega_{ix,jx,iz}(p_{ix}^{xy} - p_{jx}^{xy})$$
$$p_{iz}^{xy} - p_{jx}^{xy} = \omega_{iy,jx,iz}(p_{iy}^{xy} - p_{jx}^{xy})$$

Construct the vertical linear equation for the tetrahedron $i_x - i_y - i_z - j_x$ and incorporate it into the linear equation system $A^z p_v^z = b^z$.

$$p_{jx}^z - p_{ix}^z = p_{ix,jx}^{cor},\ p_{jx}^z - p_{iy}^z = p_{iy,jx}^{cor},\ p_{jx}^z - p_{iz}^z = p_{iz,jx}^{cor}$$

Construct the planar linear equation for the tetrahedron $i_x - i_y - i_z - j_y$ and incorporate it into the linear equation system $A^{xy}p_v^{xy} = b^{xy}$.

$$p_{iy}^{xy} - p_{jy}^{xy} = \omega_{ix,jy,iy}(p_{ix}^{xy} - p_{jy}^{xy})$$
$$p_{iz}^{xy} - p_{jy}^{xy} = \omega_{ix,jy,iz}(p_{ix}^{xy} - p_{jy}^{xy})$$
$$p_{iz}^{xy} - p_{jy}^{xy} = \omega_{iy,jy,iz}(p_{iy}^{xy} - p_{jy}^{xy})$$

Construct the vertical linear equation for the tetrahedron $i_x - i_y - i_z - j_y$ and incorporate it into the linear equation system $A^z p_v^z = b^z$.

$$p_{jy}^z - p_{ix}^z = p_{ix,jy}^{cor},\ p_{jy}^z - p_{iy}^z = p_{iy,jy}^{cor},\ p_{jy}^z - p_{iz}^z = p_{iz,jy}^{cor}$$

Construct the planar linear equation for the tetrahedron $i_x - i_y - i_z - j_z$ and incorporate it into the linear equation system $A^{xy}p_v^{xy} = b^{xy}$.

$$p_{iy}^{xy} - p_{jy}^{xy} = \omega_{ix,jy,iy}(p_{ix}^{xy} - p_{jy}^{xy})$$
$$p_{iz}^{xy} - p_{jy}^{xy} = \omega_{ix,jy,iz}(p_{ix}^{xy} - p_{jy}^{xy})$$
$$p_{iz}^{xy} - p_{jy}^{xy} = \omega_{iy,jy,iz}(p_{iy}^{xy} - p_{jy}^{xy})$$

Construct the vertical linear equation for the tetrahedron $i_x - i_y - i_z - j_z$ and incorporate it into the linear equation system $A^z p_v^z = b^z$.

$$p_{jy}^z - p_{ix}^z = p_{ix,jy}^{cor},\ p_{jy}^z - p_{iy}^z = p_{iy,jy}^{cor},\ p_{jy}^z - p_{iz}^z = p_{iz,jy}^{cor}$$

For $i \in V$:

Construct the planar linear equation for the tetrahedron $i_x - i_y - i_z - i$ and incorporate it into the linear equation system $C^{xy}p^{xy} = D^{xy}p_v^{xy}$.

$$p_{ix}^{xy} - p_i^{xy} = \omega_{ix,i,iy}(p_{ix}^{xy} - p_i^{xy})$$
$$p_{iz}^{xy} - p_i^{xy} = \omega_{ix,i,iz}(p_{ix}^{xy} - p_i^{xy})$$
$$p_{iz}^{xy} - p_i^{xy} = \omega_{iy,i,iz}(p_{iy}^{xy} - p_i^{xy})$$

Construct the vertical linear equation for the tetrahedron $i_x - i_y - i_z - i$ and incorporate it into the linear equation system $C^z p^z = D^z p_v^z$.

$$p_i^z - p_{ix}^z = p_{ix,i}^{cor},\ p_i^z - p_{iy}^z = p_{iy,i}^{cor},\ p_i^z - p_{iz}^z = p_{iz,i}^{cor}$$

3.2 Compute the linear equation for the tetrahedron ix-iy-iz-jx

In the previous sections, we constructed the following linear equations using three triangles of the tetrahedron $i_x - i_y - i_z - j_x$ (triangle $j_x - i_x - i_y$, triangle $j_x - i_x - i_z$, and triangle $j_x - i_y - i_z$).

$$p_{iy}^{xy} - p_{jx}^{xy} = \omega_{ix,jx,iy}(p_{ix}^{xy} - p_{jx}^{xy})$$
$$p_{iz}^{xy} - p_{jx}^{xy} = \omega_{ix,jx,iz}(p_{ix}^{xy} - p_{jx}^{xy})$$
$$p_{iz}^{xy} - p_{jx}^{xy} = \omega_{iy,jx,iz}(p_{iy}^{xy} - p_{jx}^{xy})$$
$$p_{jx}^z - p_{ix}^z = p_{ix,jx}^{cor}$$
$$p_{jx}^z - p_{iy}^z = p_{iy,jx}^{cor}$$
$$p_{jx}^z - p_{iz}^z = p_{iz,jx}^{cor}$$

Here, $p_{ix}^{xy}, p_{iy}^{xy}, p_{iz}^{xy}, p_{jx}^{xy} \in C$ represent the coordinates of nodes i_x, i_y, i_z, j_x projected onto the horizontal plane, and $p_{ix}^z, p_{iy}^z, p_{iz}^z, p_{jx}^z \in R$ represent the vertical coordinates of nodes i_x, i_y, i_z, j_x. The parameters $\omega_{ix,jx,iy}, \omega_{ix,jx,iz}, \omega_{iy,jx,iz}, p_{ix,jx}^{cor}, p_{iy,jx}^{cor}, p_{iz,jx}^{cor}$ were not explicitly described in the previous sections. In the following, we provide the detailed computation methods for these parameters.

We use the gravity direction measured by the IMU in node i to adjust the local coordinate system Σ_i into a corrected local coordinate system Σ_i^{cor}, where the zzz-axis aligns opposite to the gravity direction (utilizing the rotation matrix R_i^{cor}). Assume that in node i, the gravity direction measured by the IMU is $g_{ori}^i = [x_g^i, y_g^i, z_g^i]^T$, and the direction opposite to gravity is $g_{inv}^i = -g_{ori}^i = [-x_g^i, -y_g^i, -z_g^i]^T$. Let the z-axis of the sensor used for relative position measurement in node i coincide with the z-axis of the IMU, denoted as $z = [0,0,1]^T$. The rotation matrix R_i^{cor} is defined as follows.

$$n^i = z \times g_{inv}^i$$

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

$$v^i = \frac{n^i}{||n^i||}\alpha^i$$

R_i^cor=phi(v^i)

The term R_i^{cor} represents the rotation matrix, and v_i is the rotation vector corresponding to the rotation matrix R_i^{cor}; n_i denotes the rotation axis of the rotation vector v_i, and α_i is the rotation angle of the rotation vector v_i.

In the local coordinate system Σ_i, the coordinates of the virtual nodes i_x, i_y, i_z are as follows.

$$p_{ix}^{i} = [1,0,0]^T$$
$$p_{iy}^{i} = [0,1,0]^T$$
$$p_{iz}^{i} = [0,0,1]^T$$

The superscript i in p_{ix}^{i} indicates that the coordinates are defined in the local coordinate system Σ_i. Using the rotation matrix R_i^{cor}, the coordinates of the virtual nodes i_x, i_y, i_z in the local coordinate system Σ_i can be transformed into the local coordinate system Σ_i^{cor}.

$$p_{ix}^{icor} = (R_i^{cor})^{-1}[1,0,0]^T$$
$$p_{iy}^{icor} = (R_i^{cor})^{-1}[0,1,0]^T$$
$$p_{iz}^{icor} = (R_i^{cor})^{-1}[0,0,1]^T$$

The superscript icor in p_{ix}^{icor} indicates that the coordinate is located in the local coordinate system Σ_i^{cor}. The rotation matrix R_i^{cor} represents the transformation of the local coordinate system Σ_i into the local coordinate system Σ_i^{cor}. Therefore, the observed coordinate transformation requires the use of the rotation matrix $(R_i^{cor})^{-1}$.

Let the pose transformation from node i to node j be $T_{ij} \in R^{4\times4}$.

$$T_{ij} = \begin{bmatrix} R_{ij} & t_{ij} \\ 1 & 0 \end{bmatrix} \in R^{4\times4}, R_{ij} \in R^{3\times3}$$
$$t_{ij} = [x_{ij}, y_{ij}, z_{ij}]^T \in R^3$$

Then, the position of the virtual node j_x in the local coordinate system Σ_i is as follows.

$$p_{jx}^{i} = R_{ij}[1,0,0]^T + t_{ij}$$

The position of the virtual node j_x in the corrected local coordinate system Σ_i^{icor} is as follows.

$$p_{jx}^{icor} = (R_i^{cor})^{-1} p_{jx}^{i}$$

Then, we present the relative positions of the vectors $(j_x, i_x), (j_x, i_y)$ and (j_x, i_z) in the local coordinate system Σ_i^{cor}.

$$p_{jx,ix}^{icor} = p_{ix}^{icor} - p_{jx}^{icor} = [x_{jx,ix}^{icor}, y_{jx,ix}^{icor}, z_{jx,ix}^{icor}]^T$$
$$p_{jx,iy}^{icor} = p_{iy}^{icor} - p_{jx}^{icor} = [x_{jx,iy}^{icor}, y_{jx,iy}^{icor}, z_{jx,iy}^{icor}]^T$$
$$p_{jx,iz}^{icor} = p_{iz}^{icor} - p_{jx}^{icor} = [x_{jx,iz}^{icor}, y_{jx,iz}^{icor}, z_{jx,iz}^{icor}]^T$$

Let the projections of the vector coordinates $p_{jx,ix}^{icor}, p_{jx,iy}^{icor}$ and $p_{jx,iz}^{icor}$ onto the horizontal plane be as follows.

$$p_{jx,ix}^{icor^{xy}} = [x_{jx,ix}^{icor}, y_{jx,ix}^{icor}]^T$$
$$p_{jx,iy}^{icor^{xy}} = [x_{jx,iy}^{icor}, y_{jx,iy}^{icor}]^T$$
$$p_{jx,iz}^{icor^{xy}} = [x_{jx,iz}^{icor}, y_{jx,iz}^{icor}]^T$$

The coordinates in this section involve numerous subscripts and superscripts. We will redefine the meanings of these subscripts and superscripts to clarify. In the coordinate notation $p_{jx,ix}^{icor^{xy}}$, the subscript jx, ix indicates that the coordinate belongs to the vector (j_x, i_x). The superscript icor signifies that this coordinate is expressed in the local coordinate system Σ_i^{cor}, while the superscript xy denotes that the coordinate has been projected onto the horizontal plane, resulting in planar coordinates.

Next, we calculate the parameters $\omega_{ix,jx,iy}, \omega_{ix,jx,iz}$ and $\omega_{iy,jx,iz}$. The lengths between the coordinates $p_{jx,ix}^{icor^{xy}}, p_{jx,iy}^{icor^{xy}}$ and $p_{jx,iz}^{icor^{xy}}$ are determined as follows.

$$\rho_{ix} = \left|\left|p_{jx,ix}^{icor^{xy}}\right|\right|$$
$$\rho_{iy} = \left|\left|p_{jx,iy}^{icor^{xy}}\right|\right|$$
$$\rho_{iz} = \left|\left|p_{jx,iz}^{icor^{xy}}\right|\right|$$

The ratios of magnitudes are as follows.

$$\rho_{ix,jx,iy} = \rho_{iy}/\rho_{ix}$$
$$\rho_{ix,jx,iz} = \rho_{iz}/\rho_{ix}$$
$$\rho_{iy,jx,iz} = \rho_{iz}/\rho_{iy}$$

The angle calculations are as follows.

$$\theta_{ix,jx,iy} = \arccos\left(\frac{p_{jx,ix}^{icor^{xy}} p_{jx,iy}^{icor^{xy}}}{\left|\left|p_{jx,ix}^{icor^{xy}}\right|\right| \left|\left|p_{jx,iy}^{icor^{xy}}\right|\right|}\right)$$
$$\theta_{ix,jx,iz} = \arccos\left(\frac{p_{jx,ix}^{icor^{xy}} p_{jx,iz}^{icor^{xy}}}{\left|\left|p_{jx,ix}^{icor^{xy}}\right|\right| \left|\left|p_{jx,iz}^{icor^{xy}}\right|\right|}\right)$$
$$\theta_{iy,jx,iz} = \arccos\left(\frac{p_{jx,iy}^{icor^{xy}} p_{jx,iz}^{icor^{xy}}}{\left|\left|p_{jx,iy}^{icor^{xy}}\right|\right| \left|\left|p_{jx,iz}^{icor^{xy}}\right|\right|}\right)$$

The parameters $\omega_{ix,jx,iy}, \omega_{ix,jx,iz}$ and $\omega_{iy,jx,iz}$ are as follows.

$$\omega_{ix,jx,iy} = \rho_{ix,jx,iy} e^{i\theta_{ix,jx,iy}}$$
$$\omega_{ix,jx,iz} = \rho_{ix,jx,iz} e^{i\theta_{ix,jx,iz}}$$
$$\omega_{iy,jx,iz} = \rho_{iy,jx,iz} e^{i\theta_{iy,jx,iz}}$$

Then, we calculate the numerical values of the parameters $p_{ix,jx}^{cor}, p_{iy,jx}^{cor}, p_{iz,jx}^{cor}$.

$$p_{ix,jx}^{cor} = z_{jx,ix}^{icor}$$
$$p_{iy,jx}^{cor} = z_{jx,iy}^{icor}$$
$$p_{iz,jx}^{cor} = z_{jx,iz}^{icor}$$

Here, $z_{jx,ix}^{icor}, z_{jx,iy}^{icor}, z_{jx,iz}^{icor}$ have already been derived in the above steps while determining the relative positions of the vectors (j_x, i_x), (j_x, i_y) and (j_x, i_z) in the local coordinate system Σ_i^{cor}.

Using the pose transformation T_{ij} from node i to node j, as well as the gravity direction in node iii's coordinate system $g_{ori}^{i} = \left[x_g^i, y_g^i, z_g^i\right]^T$, the linear equation of the tetrahedron $i_x - i_y - i_z - j_x$ can be computed. The specific pseudocode is as follows.

Procedure 1: Compute the linear equation for the tetrahedron ix-iy-iz-jx

---

Input: Transformation matrix T_{ij}, gravity direction $g_{ori}^{i} = \left[x_g^i, y_g^i, z_g^i\right]^T$, coordinates of the virtual node j_x in the local coordinate system Σ_j as $p_{jx}^{j} = [1,0,0]^T$

Output: parameters of the linear equation $\omega_{ix,jx,iy}, \omega_{ix,jx,iz}, \omega_{iy,jx,iz}, p_{ix,jx}^{cor}, p_{iy,jx}^{cor}, p_{iz,jx}^{cor}$

Define the vector $z = [0,0,1]^T$.

Compute the rotation axis $n^i = z \times g_{inv}^{i}$.

Compute the rotation angle $\alpha^i = \arccos\left(-\frac{z_g^i}{\sqrt{{x_g^i}^2+{y_g^i}^2+{z_g^i}^2}}\right)$.

Compute the rotation vector $v^i = \frac{n^i}{||n^i||}\alpha^i$.

Compute the corrected rotation matrix R_i^cor=phi(v^i).

Positions of virtual nodes i_x, i_y, i_z in the local coordinate system Σ_i^{cor}:

$p_{ix}^{icor} = (R_i^{cor})^{-1}[1,0,0]^T$

$p_{iy}^{icor} = (R_i^{cor})^{-1}[0,1,0]^T$

$p_{iz}^{icor} = (R_i^{cor})^{-1}[0,0,1]^T$

Position of virtual node j_x in local coordinate system Σ_i: $p_{jx}^{i} = R_{ij}p_{jx}^{j} + t_{ij}$.

Position of virtual node j_x in the corrected coordinate system Σ_i^{icor}: $p_{jx}^{icor} = (R_i^{cor})^{-1}p_{jx}^{i}$.

Relative positions of vectors $(j_x, i_x), (j_x, i_y), (j_x, i_z)$ in Σ_i^{cor}:

$p_{jx,ix}^{icor} = p_{ix}^{icor} - p_{jx}^{icor} = \left[x_{jx,ix}^{icor}, y_{jx,ix}^{icor}, z_{jx,ix}^{icor}\right]^T$

$p_{jx,iy}^{icor} = p_{iy}^{icor} - p_{jx}^{icor} = \left[x_{jx,iy}^{icor}, y_{jx,iy}^{icor}, z_{jx,iy}^{icor}\right]^T$

$p_{jx,iz}^{icor} = p_{iz}^{icor} - p_{jx}^{icor} = \left[x_{jx,iz}^{icor}, y_{jx,iz}^{icor}, z_{jx,iz}^{icor}\right]^T$

The projections of $p_{jx,ix}^{icor}, p_{jx,iy}^{icor}, p_{jx,iz}^{icor}$ onto the horizontal plane:

$p_{jx,ix}^{icor^{xy}} = \left[x_{jx,ix}^{icor}, y_{jx,ix}^{icor}\right]^T$

$p_{jx,iy}^{icor^{xy}} = \left[x_{jx,iy}^{icor}, y_{jx,iy}^{icor}\right]^T$

$p_{jx,iz}^{icor^{xy}} = \left[x_{jx,iz}^{icor}, y_{jx,iz}^{icor}\right]^T$

The Euclidean norms (magnitudes) of the vectors $p_{jx,ix}^{icor}, p_{jx,iy}^{icor}, p_{jx,iz}^{icor}$.

$\rho_{ix} = \left|\left|p_{jx,ix}^{icor^{xy}}\right|\right|, \rho_{iy} = \left|\left|p_{jx,iy}^{icor^{xy}}\right|\right|, \rho_{iz} = \left|\left|p_{jx,iz}^{icor^{xy}}\right|\right|$

The ratios of the magnitudes are defined as:

$\rho_{ix,jx,iy} = \rho_{iy}/\rho_{ix}, \rho_{ix,jx,iz} = \rho_{iz}/\rho_{ix}, \rho_{iy,jx,iz} = \rho_{iz}/\rho_{iy}$

Calculate angle $\theta_{ix,jx,iy}, \theta_{ix,jx,iz}, \theta_{iy,jx,iz}$:

$$\theta_{ix,jx,iy} = \arccos\left(\frac{p_{jx,ix}^{icor^{xy}} p_{jx,iy}^{icor^{xy}}}{\left|\left|p_{jx,ix}^{icor^{xy}}\right|\right|\left|\left|p_{jx,iy}^{icor^{xy}}\right|\right|}\right)$$

$$\theta_{ix,jx,iz} = \arccos\left(\frac{p_{jx,ix}^{icor^{xy}} p_{jx,iz}^{icor^{xy}}}{\left|\left|p_{jx,ix}^{icor^{xy}}\right|\right|\left|\left|p_{jx,iz}^{icor^{xy}}\right|\right|}\right)$$

$$\theta_{iy,jx,iz} = \arccos\left(\frac{p_{jx,iy}^{icor^{xy}} p_{jx,iz}^{icor^{xy}}}{\left|\left|p_{jx,iy}^{icor^{xy}}\right|\right|\left|\left|p_{jx,iz}^{icor^{xy}}\right|\right|}\right)$$

Calculate parameters $\omega_{ix,jx,iy}, \omega_{ix,jx,iz}, \omega_{iy,jx,iz}$:

$\omega_{ix,jx,iy} = \rho_{ix,jx,iy}e^{i\theta_{ix,jx,iy}}$

$\omega_{ix,jx,iz} = \rho_{ix,jx,iz}e^{i\theta_{ix,jx,iz}}$

$\omega_{iy,jx,iz} = \rho_{iy,jx,iz}e^{i\theta_{iy,jx,iz}}$

Calculate parameters $p_{ix,jx}^{cor}, p_{iy,jx}^{cor}, p_{iz,jx}^{cor}$:

$p_{ix,jx}^{cor} = z_{jx,ix}^{icor}, p_{iy,jx}^{cor} = z_{jx,iy}^{icor}, p_{iz,jx}^{cor} = z_{jx,iz}^{icor}$

3.3 Compute the linear equations for the tetrahedra ix-iy-iz-jy, ix-iy-iz-jz, ix-iy-iz-i

In the previous sections, we constructed the following linear equations using the three triangles of the tetrahedron $i_x - i_y - i_z - j_y$ (triangle $j_y - i_x - i_y$, triangle $j_y - i_x - i_z$, and triangle $j_y - i_y - i_z$):

$$p_{iy}^{xy} - p_{jy}^{xy} = \omega_{ix,jy,iy}(p_{ix}^{xy} - p_{jy}^{xy})$$

$$p_{iz}^{xy} - p_{jy}^{xy} = \omega_{ix,jy,iz}(p_{ix}^{xy} - p_{jy}^{xy})$$

$$p_{iz}^{xy} - p_{jy}^{xy} = \omega_{iy,jy,iz}(p_{iy}^{xy} - p_{jy}^{xy})$$

$$p_{jy}^{z} - p_{ix}^{z} = p_{ix,jy}^{cor}$$

$$p_{jy}^{z} - p_{iy}^{z} = p_{iy,jy}^{cor}$$

$$p_{jy}^{z} - p_{iz}^{z} = p_{iz,jy}^{cor}$$

The variables $p_{ix}^{xy}, p_{iy}^{xy}, p_{iz}^{xy}, p_{jy}^{xy} \in \mathbb{C}$ represent the coordinates of nodes i_x, i_y, i_z, j_y projected onto the ground plane, respectively. The variables $p_{ix}^{z}, p_{iy}^{z}, p_{iz}^{z}, p_{jy}^{z} \in \mathbb{R}$ denote the vertical coordinates of nodes i_x, i_y, i_z, j_y, respectively. The parameters $\omega_{ix,jy,iy}, \omega_{ix,jy,iz}, \omega_{iy,jy,iz}, p_{ix,jy}^{cor}, p_{iy,jy}^{cor}, p_{iz,jy}^{cor}$ have not been explicitly described in terms of their computation in the previous sections. Below, we provide the method for calculating these parameters.

Since the linear equation computation for the tetrahedron $i_x - i_y - i_z - j_y$ is similar to that of the tetrahedron $i_x - i_y -$

$i_z - j_x$, we only need to modify the input condition for "Procedure 2: Calculation of tetrahedron $i_x - i_y - i_z - j_x$." Specifically, replace the condition "the local coordinate of the virtual node j_x in the local coordinate system Σ_j is $p_{jx}^j = [1,0,0]^T$" with "the local coordinate of the virtual node j_y in the local coordinate system Σ_j is $p_{jy}^j = [0,1,0]^T$." Using this modification, the output of the algorithm will yield the parameters $\omega_{ix,jy,iy}, \omega_{ix,jy,iz}, \omega_{iy,jy,iz}, p_{ix,jy}^{cor}, p_{iy,jy}^{cor}, p_{iz,jy}^{cor}$。

In the previous sections, we constructed the following linear equations using the three triangles of the tetrahedron $i_x - i_y - i_z - j_z$ (triangle $j_z - i_x - i_y$, triangle $j_z - i_x - i_z$, and triangle $j_z - i_y - i_z$):

$$p_{iy}^{xy} - p_{jz}^{xy} = \omega_{ix,jz,iy}(p_{ix}^{xy} - p_{jz}^{xy})$$
$$p_{iz}^{xy} - p_{jz}^{xy} = \omega_{ix,jz,iz}(p_{ix}^{xy} - p_{jz}^{xy})$$
$$p_{iz}^{xy} - p_{jz}^{xy} = \omega_{iy,jz,iz}(p_{iy}^{xy} - p_{jz}^{xy})$$
$$p_{jz}^{z} - p_{ix}^{z} = p_{ix,jz}^{cor}$$
$$p_{jz}^{z} - p_{iy}^{z} = p_{iy,jz}^{cor}$$
$$p_{jz}^{z} - p_{iz}^{z} = p_{iz,jz}^{cor}$$

The variables $p_{ix}^{xy}, p_{iy}^{xy}, p_{iz}^{xy}, p_{jz}^{xy} \in C$ represent the projections of nodes i_x, i_y, i_z, j_z onto the horizontal plane, respectively. Similarly, $p_{ix}^{z}, p_{iy}^{z}, p_{iz}^{z}, p_{jz}^{z} \in R$ denote the vertical coordinates of nodes i_x, i_y, i_z, j_z, respectively. The parameters $\omega_{ix,jz,iy}, \omega_{ix,jz,iz}, \omega_{iy,jz,iz}, p_{ix,jz}^{cor}, p_{iy,jz}^{cor}, p_{iz,jz}^{cor}$ were not explicitly described in detail in earlier sections. In the following, we provide the computational methods for these parameters.

Since the computation of the linear equations for the tetrahedron $i_x - i_y - i_z - j_z$ is similar to that for the tetrahedron $i_x - i_y - i_z - j_x$, the only modification required is to replace the input condition of "Process 2, computation of tetrahedron $i_x - i_y - i_z - j_x$," where the coordinates of the virtual node j_x in the local coordinate system Σ_j are given as $p_{jx}^j = [1,0,0]^T$, with "the coordinates of the virtual node j_z in the local coordinate system Σ_j are given as $p_{jz}^j = [0,0,1]^T$." With this modification, the output of the algorithm will yield the parameters $\omega_{ix,jz,iy}, \omega_{ix,jz,iz}, \omega_{iy,jz,iz}, p_{ix,jz}^{cor}, p_{iy,jz}^{cor}, p_{iz,jz}^{cor}$.

In the previous sections, we constructed the following linear equations using the three triangles of the tetrahedron $i_x - i_y - i_z - i$ (triangle $i - i_x - i_y$, triangle $i - i_x - i_z$, and triangle $i - i_y - i_z$):

$$p_{iy}^{xy} - p_{i}^{xy} = \omega_{ix,i,iy}(p_{ix}^{xy} - p_{i}^{xy})$$
$$p_{iz}^{xy} - p_{i}^{xy} = \omega_{ix,i,iz}(p_{ix}^{xy} - p_{i}^{xy})$$
$$p_{iz}^{xy} - p_{i}^{xy} = \omega_{iy,i,iz}(p_{iy}^{xy} - p_{i}^{xy})$$
$$p_{i}^{z} - p_{ix}^{z} = p_{ix,i}^{cor}$$
$$p_{i}^{z} - p_{iy}^{z} = p_{iy,i}^{cor}$$
$$p_{i}^{z} - p_{iz}^{z} = p_{iz,i}^{cor}$$

The variables $p_{ix}^{xy}, p_{iy}^{xy}, p_{iz}^{xy}, p_{i}^{xy} \in C$ represent the projections of nodes i_x, i_y, i_z, i onto the horizontal plane, respectively. Similarly, $p_{ix}^{z}, p_{iy}^{z}, p_{iz}^{z}, p_{i}^{z} \in R$ denote the vertical coordinates of nodes i_x, i_y, i_z, i. The parameters $\omega_{ix,i,iy}, \omega_{ix,i,iz}, \omega_{iy,i,iz}, p_{ix,i}^{cor}, p_{iy,i}^{cor}, p_{iz,i}^{cor}$ were not explicitly described in terms of their calculation methods in the previous sections. Below, we provide the calculation methods for these parameters.

Since the linear equation computation for the tetrahedron $i_x - i_y - i_z - i$ is similar to that of the tetrahedron $i_x - i_y - i_z - j_x$, we can adapt the input conditions from "Process 2: Calculation of tetrahedron $i_x - i_y - i_z - j_x$" as follows: replace the condition "the virtual node j_x in the local coordinate system Σ_j has coordinates $p_{jx}^j = [1,0,0]^T$" with "the node i in the local coordinate system Σ_i has coordinates $p_i^i = [0,0,0]^T$". Additionally, replace the expression for the virtual node (j_x's position in the local coordinate system Σ_i, $p_{jx}^i = R_{ij}p_{jx}^j + t_{ij}$, with the expression for the node i's position in the local coordinate system Σ_i: $p_i^i = [0,0,0]^T$.

By applying these modifications, the output parameters of the algorithm become $\omega_{ix,i,iy}, \omega_{ix,i,iz}, \omega_{iy,i,iz}, p_{ix,i}^{cor}, p_{iy,i}^{cor}, p_{iz,i}^{cor}$.

4. Pose transformation of different sensors

In this section, we discuss how data obtained from different types of sensors can be converted into pose transformations between various nodes. Pose transformations between two frames can be derived using cameras, LiDAR, IMU sensors, GPS sensors, wheel odometers, and loop closure detection. The following subsections will individually examine the pose transformations obtained from each type of sensor.

4.1 Non-GPS sensors

We first discuss the case of camera and LiDAR sensors. These two types of sensors can provide the relative pose transformation T_{ij} between two specific nodes (e.g., node i and node j). Typically, node i and node j represent two consecutive keyframes or two keyframes that are spatially very close to each other.

For IMU sensors and wheel odometers, these sensors generally provide the relative pose transformation $T_{i-1,i}$ between two consecutive keyframes.

Similarly, in the loop closure detection module, the relative pose transformation T_{ij} between two nodes (node i and node j) can also be obtained. Usually, node i and node j are adjacent nodes identified after the robot completes a loop and returns to the starting point. While node i and node j are far apart in terms of the robot's motion timeline, they are spatially close to each other.

The pose transformation T_{ij} between two nodes (node i and node j) obtained from non-GPS sensors can be converted into a set of linear equations between two groups of virtual nodes (virtual nodes i_x, i_y, i_z and virtual nodes j_x, j_y, j_z).

4.2 GPS sensors

The processing method for GPS sensor data has already been provided in [3]. However, for the sake of completeness in this SLAM solution, we will still elaborate on the specific approach to handling GPS data.

The GPS sensor is somewhat unique in nature, as it directly provides data in terms of latitude and longitude. Using established formulas, these latitude and longitude values can be converted into specific distance measurements. Consequently, the position detected by the GPS sensor is always defined in a fixed latitude-longitude coordinate system, or equivalently, a fixed global coordinate system. In contrast, other types of sensors, such as cameras and LiDAR, can only provide the relative pose transformation, T_{ij}, between two nodes (e.g., node i and node j). To compute the pose transformation between two distant nodes (e.g., node i and node $i+n$), it is necessary to iteratively accumulate transformations as follows: $T_{i,i+n} = T_{i,i+1}T_{i+1,i+2}\dots T_{i+n-1,i+n}$. However, the cumulative error in the transformation $T_{i,i+n}$ obtained through this iterative process will gradually increase. In contrast, for the GPS sensor, since it operates within a fixed latitude-longitude coordinate system, the translational component $t_{i,i+n}$ between two distant nodes (e.g., node i and node $i+n$) can still maintain a very high level of accuracy. This makes GPS sensor data highly valuable for preserving the overall shape and accuracy of large-scale maps.

Another special case of the GPS sensor is that GPS data provides only displacement information without any rotation information. Suppose we obtain the GPS data of node i (its position in the latitude-longitude coordinate system) and then obtain the GPS data of node j. In this case, we can only calculate the translational displacement $t_{ij} \in R^2$ from node i to node j in the latitude-longitude coordinate system, without any rotation information. Therefore, the translational vector $t_{ij} \in R^2$ obtained from the GPS sensor cannot be combined with the pose transformation $T_{ij} \in R^{3\times3}$ obtained from other sensors to construct linear equation constraints. Although we can construct nonlinear pose graph constraints, this approach differs from the linear equation-based optimization algorithm we aim to develop.

The data obtained from GPS sensors typically includes only latitude and longitude, without altitude information. Even when altitude data is available, its accuracy is significantly lower compared to that of latitude and longitude. If the GPS sensor provides only latitude and longitude data, the planar translation $t_{ij} \in R^2$ between two nodes (node i and node j) can be determined. To construct a linear equation, three nodes (i, j, k) can be selected to form a planar triangle; the specific steps will be provided later. On the other hand, if the GPS sensor provides both latitude, longitude, and altitude data, the spatial translation $t_{ij} \in R^3$ between two nodes (node i and node j) can be determined. Similarly, we can construct a linear equation by selecting three nodes (i, j, k) to form a spatial triangle. The process for this case is analogous to that of the planar triangle, and the derivation can be carried out in a similar manner. Therefore, the specific formulas for this scenario will not be detailed here.

In the graph $G = [V, E]$, there are n nodes (keyframes). The update rate of GPS data is generally much lower than that of camera sensors or IMU sensors; therefore, not every node has corresponding GPS data. Let the set of indices of nodes with GPS data be denoted as $G_{index} = \{g_i, i \in [1, m]\}$. Each index value g_i in this set satisfies $g_i \in [1, n]$, and the set contains m elements, corresponding to m GPS latitude and longitude data points. Let the set of these m GPS latitude and longitude data points be denoted as

$$G = \{pos_{g(i)} = \left[lon_{g(i)}, lat_{g(i)}\right]^T, g(i) \in [1, n], i \in [1, m]\}.$$

Let the i-th GPS latitude and longitude data be $pos_{g(i)} = \left[lon_{g(i)}, lat_{g(i)}\right]^T$, and the j-th GPS latitude and longitude data be $pos_{g(j)} = \left[lon_{g(j)}, lat_{g(j)}\right]^T$. Using the spherical distance formula or the UTM planar coordinate system, we can transform the latitude and longitude data of nodes g_i and g_j into a relative translation vector.

$$t_{g(i),g(j)} = f(pos_{g(i)}, pos_{g(j)})$$

The function f converts the latitude and longitude of two nodes, $pos_{g(i)}$ and $pos_{g(j)}$, into a translation vector $t_{g(i),g(j)}$.

After obtaining GPS sensor data, we construct two types of linear equations: linear equations for adjacent GPS data and linear equations for cross-node GPS data. The following section first introduces the linear equations for adjacent GPS data.

Let the GPS data at the i-th timestamp be denoted as $pos_{g(i)}$, the GPS data at the (i+1)-th timestamp as $pos_{g(i+1)}$, and the GPS data at the (i+2)-th timestamp as $pos_{g(i+2)}$. Using the node g_{i+1} as an intermediate node, the relative position from node g_{i+1} to node g_i is represented as: $t_{g(i+1),g(i)} = f(pos_{g(i+1)}, pos_{g(i)})$. Similarly, the relative position from node g_i to node g_{i+2} is given by: $t_{g(i),g(i+2)} = f(pos_{g(i)}, pos_{g(i+2)})$. Finally, the relative position from node g_i to node g_{i+2} is: $t_{g(i),g(i+2)} = f(pos_{g(i+2)}, pos_{g(i)})$. It is first necessary to ensure that there are no overlapping nodes among these three nodes. The method to determine this is described as follows.

$$\rho_{g(i+1),g(i)} = \left|\left|t_{g(i+1),g(i)}\right|\right| > \epsilon_{thres}$$

$$\rho_{g(i+1),g(i+2)} = \left|\left|t_{g(i+1),g(i+2)}\right|\right| > \epsilon_{thres}$$

$$\rho_{g(i),g(i+2)} = \left|\left|t_{g(i),g(i+2)}\right|\right| > \epsilon_{thres}$$

After ensuring that there are no overlapping nodes, we can construct the following linear equations.

$$p^{xy}_{g(i+2)} - p^{xy}_{g(i+1)} = \omega_{i+1}(p^{xy}_{g(i)} - p^{xy}_{g(i+1)})$$

$$\omega_{i+1} = \rho_{i+1}e^{i\theta_{i+1}}$$

The parameters ρ_{i+1} and θ_{i+1} can be calculated as follows.

$$\rho_{i+1} = \frac{\rho_{g(i+1),g(i+2)}}{\rho_{g(i+1),g(i)}}$$

$$\cos(\theta_{i+1}) = \frac{t_{g(i+1),g(i)} t_{g(i+1),g(i+2)}}{\left||t_{g(i+1),g(i)}|\right| \left||t_{g(i+1),g(i+2)}|\right|}$$

In addition, we can construct linear equations based on GPS data across different nodes. The translation vector $t_{g(i),g(i+1)}$, calculated using any two adjacent GPS data points, cannot fully utilize the information provided by GPS. This is because the strength of GPS data lies in its ability to maintain high accuracy in measuring the relative positions between two nodes that are far apart. Therefore, in addition to constructing the aforementioned linear equations, we also need to use GPS data to establish linear equations for nodes that are far apart.

For this purpose, we need to identify three such nodes (nodes g_i, g_j and g_k) within the GPS data set G, ensuring that there are no overlapping nodes among them.

$$\rho_{g(i),g(j)} > \epsilon_{thres}$$

$$\rho_{g(i),g(k)} > \epsilon_{thres}$$

$$\rho_{g(j),g(k)} > \epsilon_{thres}$$

The distances between these three nodes are approximately equal, or we can describe this using the following formula.

$$\frac{\rho_{g(i),g(j)}}{\rho_{g(i),g(k)}} < p_{thres}, \text{if } \rho_{g(i),g(j)} \geq \rho_{g(i),g(k)}$$

$$\frac{\rho_{g(i),g(k)}}{\rho_{g(i),g(j)}} < p_{thres}, \text{if } \rho_{g(i),g(j)} < \rho_{g(i),g(k)}$$

$$\frac{\rho_{g(j),g(i)}}{\rho_{g(j),g(k)}} < p_{thres}, \text{if } \rho_{g(j),g(i)} \geq \rho_{g(j),g(k)}$$

$$\frac{\rho_{g(j),g(k)}}{\rho_{g(j),g(i)}} < p_{thres}, \text{if } \rho_{g(j),g(i)} < \rho_{g(j),g(k)}$$

$$\frac{\rho_{g(k),g(i)}}{\rho_{g(k),g(j)}} < p_{thres}, \text{if } \rho_{g(k),g(i)} \geq \rho_{g(k),g(j)}$$

$$\frac{\rho_{g(k),g(j)}}{\rho_{g(k),g(i)}} < p_{thres}, \text{if } \rho_{g(k),g(i)} < \rho_{g(k),g(j)}$$

Where p_{thres} represents the ratio of the two sides; we need to ensure that the ratio of any two sides in the triangle formed by $g_i g_j g_k$ is less than p_{thres} (the ratio of the longer side to the shorter side).

Since nodes g_i, g_j and g_k are randomly selected across the entire map, the distances between these three points are generally large. Using these three points to form a similar triangle and construct a linear equation provides a strong constraint on the overall shape of the map. To closely satisfy the similar triangle formed by these three points, the positions of nodes g_i, g_j and g_k must undergo uniform and relatively significant adjustments.

To determine as many such triangular combinations as possible from the GPS dataset G, we adopt the following approach: randomly select three points from the GPS dataset and evaluate whether they satisfy the proportional conditions of the triangle sides. If the conditions are not met, a new set of three points is selected. We can define a maximum search limit f_{max} and the required number of triangles Δ_{req}. Once three nodes g_i, g_j and g_k satisfying the conditions are identified, the following linear equations can be constructed.

$$p^{xy}_{g(k)} - p^{xy}_{g(i)} = \omega_{g(j),g(i),g(k)} (p^{xy}_{g(j)} - p^{xy}_{g(i)})$$

$$\omega_{g(j),g(i),g(k)} = h(t_{g(i),g(j)}, t_{g(i),g(k)})$$

We formulate the aforementioned linear equations into a system of equations, denoted as follows.

$$E^{xy} p^{xy} = 0$$

Let $p^{xy} \in C^n$ represent the planar coordinate vector of all nodes $i \in V$. For the GPS data set G, we construct two types of linear equations: the linear equations for adjacent GPS data and the linear equations for cross-node GPS data. Below, we first introduce the linear equations for adjacent GPS data, followed by an explanation of the specific methods using pseudocode.

Process 3: Constructing linear equations from adjacent GPS data

---

Input: a set of GPS data, denoted as $G = \{pos_{g(i)} = [lon_{g(i)}, lat_{g(i)}]^T, g_i \in [1, n], i \in [1, m]\}$; overlap threshold distance ϵ_{thres}

Output: plane coordinate vectors $p^{xy} = [p^{xy}_1, p^{xy}_2, \ldots, p^{xy}_n]^T$; matrix E^{xy}

For (i in $[1, m-2]$):

Compute the relative translation between nodes g_i and g_{i+1}: $t_{g(i),g(i+1)} = f(pos_{g(i)}, pos_{g(i+1)})$, with distance $\rho_{g(i),g(i+1)}$.

Compute the relative translation between nodes g_{i+1} and g_{i+2}: $t_{g(i+1),g(i+2)} = f(pos_{g(i+1)}, pos_{g(i+2)})$, with distance $\rho_{g(i+1),g(i+2)}$.

Compute the relative translation between nodes g_i and g_{i+2}, with distance $\rho_{g(i),g(i+2)}$.

If($\rho_{g(i+1),g(i)} > \epsilon_{thres}$):

break

If($\rho_{g(i+1),g(i+2)} > \epsilon_{thres}$):

break

If($\rho_{g(i),g(i+2)} > \epsilon_{thres}$):

break

Construct a linear equation: $p^c_{g(i+2)} - p^c_{g(i+1)} = \omega_{i+1}(p^c_{g(i)} - p^c_{g(i+1)})$ and append it to the matrix E^{xy}.

Process 4: Constructing linear equations from cross-node GPS data

---

Input: a set of GPS data $G = \{pos_{g(i)} = [lon_{g(i)}, lat_{g(i)}]^T$, edge ratio threshold k_{thres}, maximum number of iterations

f_{max}, required number of triangles Δ_{req}

Output: a planar coordinate vector $p^{xy} = [p_1^{xy}, p_2^{xy}, \ldots, p_n^{xy}]^T$, matrix E^{xy}

Iteration counter $iter_{num} = 0$, number of triangles found $\Delta_{num} = 0$.

For ($iter_{num} < f_{max}$ and $\Delta_{num} < \Delta_{req}$):

$iter_{num} = iter_{num} + 1$.

Randomly select three points g_i, g_j, g_k from the GPS data set G.

Relative translation between nodes g_i and g_j: $t_{g(i),g(j)} = f(pos_{g(i)}, pos_{g(j)})$, with distance $\rho_{g(i),g(j)}$.

Relative translation between nodes g_i and g_k: $t_{g(i),g(k)} = f(pos_{g(i)}, pos_{g(k)})$, with distance $\rho_{g(i),g(k)}$.

Relative translation between nodes g_j and g_k: $t_{g(j),g(k)} = f(pos_{g(j)}, pos_{g(k)})$, with distance $\rho_{g(j),g(k)}$.

If($\rho_{g(i),g(j)} < \epsilon_{thres}$):

break

If($\rho_{g(i),g(k)} < \epsilon_{thres}$):

break

If($\rho_{g(j),g(k)} < \epsilon_{thres}$):

break

If$\left(\frac{\rho_{g(i),g(j)}}{\rho_{g(i),g(k)}} > p_{thres}$ and $\rho_{g(i),g(j)} \geq \rho_{g(i),g(k)}\right)$:

break

If$\left(\frac{\rho_{g(i),g(k)}}{\rho_{g(i),g(j)}} > p_{thres}$ and $\rho_{g(i),g(j)} < \rho_{g(i),g(k)}\right)$:

break

If$\left(\frac{\rho_{g(j),g(i)}}{\rho_{g(j),g(k)}} > p_{thres}$ and $\rho_{g(j),g(i)} \geq \rho_{g(j),g(k)}\right)$:

break

If$\left(\frac{\rho_{g(j),g(k)}}{\rho_{g(j),g(i)}} > p_{thres}$ and $\rho_{g(j),g(i)} < \rho_{g(j),g(k)}\right)$:

break

If$\left(\frac{\rho_{g(k),g(i)}}{\rho_{g(k),g(j)}} < p_{thres}$ and $\rho_{g(k),g(i)} \geq \rho_{g(k),g(j)}\right)$:

break

If$\left(\frac{\rho_{g(k),g(j)}}{\rho_{g(k),g(i)}} < p_{thres}$ and $\rho_{g(k),g(i)} < \rho_{g(k),g(j)}\right)$:

break

Using node g_i, node g_j, and node g_k, a linear equation is constructed as $p_{g(k)}^{xy} - p_{g(i)}^{xy} = \omega_{g(j),g(i),g(k)}(p_{g(j)}^{xy} - p_{g(i)}^{xy})$, where this equation is incorporated into the matrix E^{xy}.

$\Delta_{num} = \Delta_{num} + 1$.

4.3 Variance of pose transformation

The relative pose transformation between two frames can be obtained using cameras, LiDAR, IMU sensors, GPS sensors, wheel encoders, and loop closure detection. Typically, the pose transformations obtained from different sensors exhibit varying levels of variance. The variance of each sensor can be pre-defined based on the characteristics of the sensor itself. Alternatively, the variance of the pose transformation T_{ij} can be computed during the process of determining the transformation in the front-end module of a SLAM algorithm. The pose transformation matrix T_{ij} is a 4×4 matrix. For simplicity, we use a single variance σ_{ij}^2 to represent the variance of the entire pose transformation matrix.

The linear equations between each node $i \in V$ and the virtual nodes $i_x, i_y, i_z \in V_v$ are given as follows.

$$A^{xy}p_v^{xy} = b^{xy} + v_1$$
$$A^{z}p_v^{z} = b^{z} + v_2$$
$$C^{xy}p^{xy} = D^{xy}p_v^{xy} + v_3$$
$$C^{z}p^{z} = D^{z}p_v^{z} + v_4$$
$$E^{xy}p^{xy} = 0 + v_5$$

The error vectors are distributed as follows: $v_1 \sim N(0, \Sigma_1)$, $v_2 \sim N(0, \Sigma_2)$, $v_3 \sim N(0, \Sigma_3)$, $v_4 \sim N(0, \Sigma_4)$, and $v_5 \sim N(0, \Sigma_5)$. By neglecting the correlations between different linear equations, the covariance matrix Σ becomes a diagonal matrix.

The error vectors are distributed as follows: $v_1 \sim N(0, \Sigma_1), v_2 \sim N(0, \Sigma_2), v_3 \sim N(0, \Sigma_3), v_4 \sim N(0, \Sigma_4)$ and $v_5 \sim N(0, \Sigma_5)$. By neglecting the correlations between different linear equations, the covariance matrix Σ becomes a diagonal matrix.

For the variances of the error vectors v_1 and v_2 (i.e., Σ_1 and Σ_2), we use the variance σ_{ij}^2 of the pose transformation T_{ij} corresponding to each linear equation as the variance of the respective linear equation. For the variances of the error vectors v_3 and v_4 (i.e., Σ_3 and Σ_4), they can be set to 0, as these two sets of linear equations express the virtual coordinates i_x, i_y, i_z of node $i \in V$ in terms of its actual coordinates without introducing any observational data. For the error vector v_5, each linear equation is constructed using two pose transformations (e.g., T_{ij} and T_{ik}). For simplicity, we set the variance of the corresponding linear equation to $\sigma^2 = \sigma_{ij}^2 + \sigma_{ik}^2$.

5. Map scale calculation

5.1 Without using GPS sensors

In the graph $G = [V, E]$, V represents the set of all nodes (keyframes). Assume that the set V contains n nodes. In the local coordinate system Σ_i of each node $i \in V$, three virtual nodes i_x, i_y, i_z are defined. The set of virtual nodes is denoted as $V_v = \{i_x, i_y, i_z, i \in V\}$. The coordinate vector of the virtual nodes is $p_v = [p_{1x}, p_{1y}, p_{1z}, \ldots, p_{nx}, p_{ny}, p_{nz}] \in R^{3\times n}$. The

coordinate vector of the virtual nodes projected onto the plane is $p_v^{xy} = \left[p_{1x}^{xy}, p_{1y}^{xy}, p_{1z}^{xy}, \ldots, p_{nx}^{xy}, p_{ny}^{xy}, p_{nz}^{xy}\right] \in C^n$, and the coordinate vector of the virtual nodes in the vertical direction is $p_v^z = \left[p_{1x}^z, p_{1y}^z, p_{1z}^z, \ldots, p_{nx}^z, p_{ny}^z, p_{nz}^z\right]^T \in R^n$.

The set of all pose transformations T_{ij} is denoted as $E = \{T_{ij}, i, j \in V\}$. The pose transformation T_{ij} between any two nodes (node i and node j) can be converted into 18 linear equations (9 planar coordinate equations and 9 vertical coordinate equations). Therefore, we define the system of linear equations corresponding to all pose transformations $T_{ij} \in E$ as follows.

$$A^{xy} p_v^{xy} = b^{xy}$$

$$A^z p_v^z = b^z$$

We define the local coordinate system Σ_1 of the first node as the global coordinate system Σ_g and use it to calculate the coordinates of other nodes. If we use another coordinate system Σ_i as the global coordinate system Σ_g, the corresponding pose transformation T_{1i} can be used to transform all coordinates into the coordinate system Σ_i. Assuming that the local coordinate system Σ_1 is set as the global coordinate system Σ_g, the coordinates of node 1 are $p_1 = [0,0,0]^T$; the virtual node 1_x has coordinates $p_{1x} = [1,0,0]^T$, the virtual node 1_y has coordinates $p_{1y} = [0,1,0]^T$, and the virtual node 1_z has coordinates $p_{1z} = [0,0,1]^T$.

In the vertical coordinate vector $p_v^z \in R^{3n}$, the vertical coordinate of the virtual node 1_x is $p_{1x}^z = 0$, the vertical coordinate of the virtual node 1_y is $p_{1y}^z = 0$, and the vertical coordinate of the virtual node 1_z is $p_{1z}^z = 1$. According to the node localizability conditions, as long as all the coordinates of a single node (e.g., the coordinates of node 1, node 1_x, node 1_y, and node 1_z) are determined, the coordinates of all other nodes can be calculated. Additionally, due to the relative distances between nodes in the vertical direction being constrained by the linear equations in the vertical direction, there is no issue of scale ambiguity. Therefore, by substituting p_{1x}^z, p_{1y}^z and p_{1z}^z into the coordinate vector p_v^z, the remaining node coordinates in the vector p_v^z can be computed using the linear equation $A^z p_v^z = b^z$.

In the planar coordinate vector $p_v^{xy} \in C^{3n}$, the planar coordinates of the virtual node 1_x are given as $p_{1x}^{xy} = 1 + i \times 0$, the planar coordinates of the virtual node 1_y are $p_{1y}^z = 0 + i \times 1$, and the planar coordinates of the virtual node 1_z are $p_{1z}^z = 0 + i \times 0$. According to the node localizability conditions, once the coordinates of one node (e.g., the coordinates of nodes $1, 1_x, 1_y$ and 1_z) are determined, the coordinates of all other nodes can be derived. Substituting p_{1x}^{xy}, p_{1y}^{xy} and p_{1z}^{xy} into the coordinate vector p_v^{xy}, we can compute the coordinates of all other nodes in the vector p_v^{xy} by solving the linear equation $A^{xy} p_v^{xy} = b^{xy}$, where A^{xy} and b^{xy} are predefined matrices and vectors, respectively.

The geometric meaning of the linear equation $A^{xy} p_v^{xy} = b^{xy}$ in the plane is that the shape of the projection of all nodes $i \in V$ on the plane is determined, but the scale can vary. This is because each linear equation is equivalent to a constraint of a similar triangle. If the coordinates of other nodes in p_v^{xy} are computed using the three planar coordinates p_{1x}^{xy}, p_{1y}^{xy} and p_{1z}^{xy}, it is effectively equivalent to determining the entire planar shape's scale using these three planar coordinates. However, as this scale information propagates outward from node 1 to other nodes, the error gradually accumulates. For example, if we set the unit vector lengths of the three axes in the local coordinate system Σ_1 to 1 (i.e., $\rho_{i,ix} = 1, \rho_{i,iy} = 1, \rho_{i,iz} = 1$), the magnitude of the vectors representing the three axes in the local coordinate system Σ_n—formed by node n and its adjacent nodes n_x, n_y and n_z—after calculating the coordinates of node n, may no longer necessarily remain 1 (i.e., $\rho_{n,nx}, \rho_{n,ny}, \rho_{n,nz}$ may deviate). To address this, we can adopt the following method to determine a more accurate map scale.

Let the planar coordinates of Node 1 be $p_1^{xy} = 0 + i \times 0$, where the planar coordinates of the virtual nodes are defined as follows: $p_{1x}^{xy} = \rho + i \times 0$, $p_{1y}^{xy} = 0 + i \times \rho$, and $p_{1z}^{xy} = 0 + i \times 0$. The vertical coordinates of Node 1 and the virtual nodes $1_x, 1_y, 1_z$ remain unchanged, denoted as $p_1^z = 0, p_{1x}^z = 0, p_{1y}^z = 0$ and $p_{1z}^z = 1$. Therefore, the computation of the virtual node vector p_v^z and the node vector p^z is not affected. Let the planar coordinate vector of the virtual nodes $i_x, i_y, i_z \in V_v/(1_x, 1_y, 1_z)$ be represented as: $p_{sub}^{xy} = \left[p_{2x}^{xy}, p_{2y}^{xy}, p_{2z}^{xy}, \ldots, p_{nx}^{xy}, p_{ny}^{xy}, p_{nz}^{xy}\right] \in C^{3n-3}$. By substituting the planar coordinates $p_{1x}^{xy}, p_{1y}^{xy}, p_{1z}^{xy}$ into the equations, the linear system $A^{xy} p_v^{xy} = b^x y$ is transformed as follows.

$$A_{sub}^{xy} p_{sub}^{xy} = b_{sub}^{xy} + v_{sub1}$$

The term $v_{sub1} \sim N(0, \Sigma_{sub1})$ represents the noise associated with each linear equation. The variance of each linear equation is represented by the variance σ_{ij}^2 of the corresponding pose transformation T_{ij}. The number of linear equations in the system remains unchanged, and the variance associated with each linear equation does not vary. Therefore, the error vector $v_{sub1} = v_1$, and the covariance matrix $\Sigma_{sub1} = \Sigma_1$. The least-squares solution for this system of linear equations is as follows.

$$p_{sub}^{xy} = \left(A_{sub}^{xy\ T} \Sigma_1^{-1} A_{sub}^{xy}\right)^{-1} A_{sub}^{xy\ T} \Sigma_1^{-1} b_{sub}^{xy}$$

The planar coordinate vector p_{sub}^{xy} represents the complex coordinates of the $3(n-1)$ virtual nodes from node 2 to node n. Each complex coordinate (e.g., $p_{ix}^{xy}, p_{iy}^{xy}, p_{iz}^{xy}$) is a linear function of the unknown variable ρ, with both the real and imaginary parts being first-order terms of ρ.

$$p_{ix}^{xy} = (a_{ix}\rho + b_{ix}) + i(c_{ix}\rho + d_{ix})$$

$$p_{iy}^{xy} = (a_{iy}\rho + b_{iy}) + i(c_{iy}\rho + d_{iy})$$

$$p_{iz}^{xy} = (a_{iz}\rho + b_{iz}) + i(c_{iz}\rho + d_{iz})$$

The complex coordinates of the three virtual nodes of Node 1, namely p_{1x}^{xy}, p_{1y}^{xy} and p_{1z}^{xy}, are linear functions of the unknown variable ρ for both the real and imaginary parts.

Consequently, for all virtual nodes $i_x, i_y, i_z \in V_v$, the real and imaginary parts of their complex coordinates are expressed as linear functions of the unknown variable ρ.

It is known that the vertical coordinates of Node 1 and its virtual nodes 1_x, 1_y and 1_z are $p_1^z = 0, p_{1x}^z = 0, p_{1y}^z = 0, p_{1z}^z = 1$. By substituting the plane coordinates p_{1x}^z, p_{1y}^z and p_{1z}^z into the equations, the linear system $A^z p_v^z = b^z + v_2$ transforms into the following form.

$$A_{sub}^z p_{sub}^z = b_{sub}^z + v_{sub2}$$

The coordinate vector is defined as $p_{sub}^z = [p_{2x}^z, p_{2y}^z, p_{2z}^z, \dots, p_{nx}^z, p_{ny}^z, p_{nz}^z] \in R^{3n-3}$. The term $v_{sub2} \sim N(0, \Sigma_{sub2})$ represents the noise associated with each linear equation. The number of linear equations in the system remains unchanged, and the variance corresponding to each linear equation also remains constant. Therefore, the error vector $v_{sub2} = v_2$, and the covariance matrix $\Sigma_{sub2} = \Sigma_2$. The least squares solution to this linear equation system is expressed as follows.

$$p_{sub}^z = \left({A_{sub}^z}^T \Sigma_2^{-1} A_{sub}^z\right)^{-1} {A_{sub}^z}^T \Sigma_2^{-1} b_{sub}^z$$

The vertical component of the virtual node coordinates, $p_v^z \in R^{3n-3}$, is a determined value and does not contain the unknown variable ρ. The coordinates of node i can be expressed using the coordinates of the three virtual nodes i_x, i_y and i_z within the local coordinate system Σ_i. Once the coordinates of all virtual nodes p_v are computed, we can use p_v to calculate the coordinates of each node $i \in V$ in the graph $G = [V, E]$. The coordinate vector of all nodes $i \in V$ is denoted as $p = [p_1, p_2, \dots, p_n] \in R^{3n}$. The planar coordinate vector of all nodes $i \in V$ is represented as $p^{xy} = [p_1^{xy}, p_2^{xy}, \dots, p_n^{xy}] \in C^n$, and the vertical coordinate vector of all nodes $i \in V$ is denoted as $p^z = [p_1^z, p_2^z, \dots, p_n^z]^T \in R^n$. Therefore, the equations for computing the coordinates p^{xy} and p^z of all nodes $i \in V$ using the virtual node coordinates p_v^{xy} and p_v^z are as follows.

$$C^{xy} p^{xy} = D^{xy} p_v^{xy}$$

$$C^z p^z = D^z p_v^z$$

The least-squares solutions of the two equations above are as follows.

$$p^{xy} = \left({C^{xy}}^T C^{xy}\right)^{-1} {C^{xy}}^T D^{xy} p_v^{xy}$$

$$p^z = \left({C^z}^T C^z\right)^{-1} D^z p_v^z$$

The coordinate vector $p^{xy} \in C^n$ contains the complex coordinates of each node $i \in V$, where the real and imaginary parts of each complex coordinate are linear functions of the unknown variable ρ.

$$p_i^{xy} = (a_i \rho + b_i) + i(c_i \rho + d_i)$$

The coordinate vector $p^z \in R^n$ specifies a definite value for the coordinate of each node $i \in V$, and it does not contain the unknown variable ρ.

We then define the following error function J_1. This error function represents the condition that, in the local coordinate system Σ_i of each node i, the magnitude of the vector from node i to the virtual node i_x should be 1, the magnitude of the vector from node i to node i_y should also be 1, and the magnitude of the vector from node i to node i_z should likewise be 1.

$$\begin{aligned} J_1 &= \sum_{i \in V} \left(\left(\left| |p_1 - p_{1x}| \right|^2 - 1 \right)^2 + \left(\left| |p_1 - p_{1y}| \right|^2 - 1 \right)^2 \right. \\ &\left. + \left(\left| |p_1 - p_{1z}| \right|^2 - 1 \right)^2 \right) \\ &= \sum_{i \in V} \left(((a_i \rho - a_{ix} \rho)^2 + (b_i - b_{ix})^2 + (p_i^z - p_{ix}^z)^2 - 1)^2 \right. \\ &+ \left((a_i \rho - a_{iy} \rho)^2 + (b_i - b_{iy})^2 + (p_i^z - p_{iy}^z)^2 - 1 \right)^2 \\ &\left. + ((a_i \rho - a_{iz} \rho)^2 + (b_i - b_{iz})^2 + (p_i^z - p_{iz}^z)^2 - 1)^2 \right) \\ &= A_1 \rho^4 + B_1 \rho^3 + C_1 \rho^2 + D_1 \rho + E_1 \end{aligned}$$

The error function J_2 is defined as follows: If there exists a pose transformation $T_{ij} \in R^{4 \times 4}$ between node iii and node jjj, where the translational component is $t_{ij} \in R^3$, then the length between node i and node j is $\rho_{ij} = ||t_{ij}||$. Meanwhile, the length can also be computed using the solved coordinates p_i of node i and p_j of node j (as a linear function of the unknown variable ρ). The error function is then defined as the difference between these two lengths.

$$\begin{aligned} J_2 &= \sum_{T_{ij} \in E} \left(\left| |p_i - p_j| \right|^2 - \rho_{ij}^2 \right)^2 \\ &= \sum_{T_{ij} \in E} \left((a_i \rho - a_j \rho)^2 + (b_i - b_j)^2 + (p_i^z - p_j^z)^2 - \rho_{ij}^2 \right)^2 \\ &= A_2 \rho^4 + B_2 \rho^3 + C_2 \rho^2 + D_2 \rho + E_2 \end{aligned}$$

By summing the error functions J_1 and J_2, we obtain the total error function J.

$$\begin{aligned} J &= J_1 + J_2 \\ &= A\rho^4 + B\rho^3 + C\rho^2 + D\rho + E \end{aligned}$$

To minimize this error function, we compute the derivative of J.

$$\frac{dJ}{d\rho} = 4A\rho^3 + 3B\rho^2 + 2C\rho + D = 0$$

Then, the three roots ρ_1, ρ_2 and ρ_3 are calculated using the formula for solving cubic equations. After discarding any complex roots among the three, the second derivative of J is used to evaluate the remaining roots.

$$\frac{dJ^2}{d\rho^2} = 12A\rho^2 + 6B\rho + 2C$$

If the second derivative of $\rho_i, i \in [1,3]$ is positive, then ρ_i is the minimum of the function J. Conversely, if the second derivative of ρ_i is negative, then ρ_i is a local maximum of the function J. After determining the optimized map scale ρ, we can substitute it into the coordinates $p_v^{xy} \in C^{3n}$ and $p^{xy} \in C^n$. Using the planar coordinate vector of virtual nodes $p_v^{xy} \in C^{3n}$

and the vertical coordinate vector $p_v^z \in R^{3n}$, we can obtain the coordinate vector of the virtual nodes $p_v \in R^{3\times 3n}$. Similarly, for a node $i \in V$, using its planar coordinate vector $p^{xy} \in C^n$ and vertical coordinate vector $p^z \in R$, we can derive the coordinate vector of the node $i \in V$ as $p \in R^{3\times n}$. The pseudocode for computing the map scale and node coordinates is as follows.

Procedure 5: Calculation of map scale and node coordinates (without using GPS data)

Input: Linear equation systems $A^{xy}p_v^{xy} = b^{xy}, A^z p_v^z = b^z, C^{xy}p^{xy} = D^{xy}p_v^{xy}, C^z p^z = D^z p_v^z$, variance matrices Σ_1, Σ_2

Output: Virtual node coordinate vector $p_v \in R^{3\times 3n}$, node coordinate vector $p \in R^{3\times n}$, map scale ρ

Let the coordinate vector of node $i \in V$ be: $p = [p_1, p_2, \ldots, p_n] \in R^{3\times n}$.

Let the planar coordinate vector of node $i \in V$ be: $p^{xy} = \left[p_1^{xy}, p_2^{xy}, \ldots, p_n^{xy}\right] \in C^n$.

Let the vertical coordinate vector of node $i \in V$ be: $p^z = [p_1^z, p_2^z, \ldots, p_n^z]^T \in R^n$.

Let the virtual node coordinate vector be: $p_v = [p_{1x}, p_{1y}, p_{1z}, \ldots, p_{nx}, p_{ny}, p_{nz}] \in R^{3\times n}$.

Let the planar coordinate vector of virtual nodes be: $p_v^{xy} = \left[p_{1x}^{xy}, p_{1y}^{xy}, p_{1z}^{xy}, \ldots, p_{nx}^{xy}, p_{ny}^{xy}, p_{nz}^{xy}\right] \in C^n$.

Let the vertical coordinate vector of virtual nodes be: $p_v^z = \left[p_{1x}^z, p_{1y}^z, p_{1z}^z, \ldots, p_{nx}^z, p_{ny}^z, p_{nz}^z\right]^T \in R^n$.

Let the planar coordinates of Node 1 be denoted as $p_1^{xy} = 0 + i \times 0$, the planar coordinates of the virtual node 1_x as $p_{1x}^{xy} = \rho + i \times 0$, the planar coordinates of the virtual node 1_y as $p_{1y}^{xy} = 0 + i \times \rho$, and the planar coordinates of the virtual node 1_z as $p_{1z}^{xy} = 0 + i \times 0$.

Let the planar coordinate vector of virtual nodes $i_x, i_y, i_z \in V_v/(1_x, 1_y, 1_z)$ be denoted as: $p_{sub}^{xy} = \left[p_{2x}^{xy}, p_{2y}^{xy}, p_{2z}^{xy}, \ldots, p_{nx}^{xy}, p_{ny}^{xy}, p_{nz}^{xy}\right] \in C^{3n-3}$.

Substituting the planar coordinates $p_{1x}^{xy}, p_{1y}^{xy}, p_{1z}^{xy}$ into the equations, the linear system $A^{xy}p_v^{xy} = b^{xy}$ is reduced to $A_{sub}^{xy}p_{sub}^{xy} = b_{sub}^{xy}$.

The planar components p_{sub}^{xy} of the virtual nodes can then be computed using the linear system: $A_{sub}^{xy}p_{sub}^{xy} = b_{sub}^{xy}$, where the least-squares solution for the system is: $p_{sub}^{xy} = \left(A_{sub}^{xy\ T}\Sigma_1^{-1}A_{sub}^{xy}\right)^{-1}A_{sub}^{xy\ T}\Sigma_1^{-1}b_{sub}^{xy}$, which is a linear function of ρ.

Let the vertical coordinate vector of virtual nodes $i_x, i_y, i_z \in V_v/(1_x, 1_y, 1_z)$ be denoted as: $p_{sub}^z = \left[p_{2x}^z, p_{2y}^z, p_{2z}^z, \ldots, p_{nx}^z, p_{ny}^z, p_{nz}^z\right]^T \in R^{3n-3}$.

Substituting the vertical coordinates $p_{1x}^z, p_{1y}^z, p_{1z}^z$ into the equations, the linear system $A^z p_v^z = b^z$ is reduced to $A_{sub}^z p_{sub}^z = b_{sub}^z$.

The vertical components p_{sub}^z of the virtual nodes can then be computed using the linear system: $A_{sub}^z p_{sub}^z = b_{sub}^z$, where the least-squares solution is: $p_{sub}^z = \left(A_{sub}^{z\ T}\Sigma_2^{-1}A_{sub}^z\right)^{-1}A_{sub}^{z\ T}\Sigma_2^{-1}b_{sub}^z$, which is a fixed value independent of ρ.

The planar components p^{xy} of node $i \in V$ are calculated using the linear system of equations $C^{xy}p^{xy} = D^{xy}p_v^{xy}$. The least-squares solution of this linear system is given by: $p^{xy} = \left(C^{xy\,T}C^{xy}\right)^{-1}C^{xy\,T}D^{xy}p_v^{xy}$ (a linear function of ρ).

The vertical component p^z of node $i \in V$ is calculated using the linear system of equations $C^z p^z = D^z p_v^z$. The least-squares solution of this system is given by: $p^z = \left(C^{z\,T}C^z\right)^{-1}D^z p_v^z$ (a constant value).

Define the error function J_1 as: $J_1 = \sum_{i\in V}\left(\left(\left||p_1 - p_{1x}|\right|^2 - 1\right)^2 + \left(\left||p_1 - p_{1y}|\right|^2 - 1\right)^2 + \left(\left||p_1 - p_{1z}|\right|^2 - 1\right)^2\right) = A_1\rho^4 + B_1\rho^3 + C_1\rho^2 + D_1\rho + E_1$.

Define the error function J_2 as: $J_2 = \sum_{T_{ij}\in E}\left(\left||p_i - p_j|\right|^2 - \rho_{ij}^2\right)^2 = A_2\rho^4 + B_2\rho^3 + C_2\rho^2 + D_2\rho + E_2$.

Combine the two error functions to obtain the total error function J: $J = J_1 + J_2 = A\rho^4 + B\rho^3 + C\rho^2 + D\rho + E$.

To determine the parameter ρ that minimizes the error function, compute the derivative of J with respect to ρ: $\frac{dJ}{d\rho} = 4A\rho^3 + 3B\rho^2 + 2C\rho + D = 0$.

Solve the above cubic equation using the cubic formula to find the three roots ρ_1, ρ_2, ρ_3. Discard any complex roots, and select the root ρ_i corresponding to the minimum value of J using the second derivative of the error function.

Substitute the optimal parameter ρ into the coordinates p_v^{xy}, p^{xy} to compute the final coordinates p^v, p.

5.2 Using GPS sensors

When GPS sensors are not utilized, the linear equations between each node $i \in V$ and the virtual nodes $i_x, i_y, i_z \in V_v$ are as follows.

$$A^{xy}p_v^{xy} = b^{xy} + v_1$$

$$A^z p_v^z = b^z + v_2$$

$$C^{xy}p^{xy} = D^{xy}p_v^{xy} + v_3$$

$$C^z p^z = D^z p_v^z + v_4$$

We define the coordinates as $p_1^{xy} = 0 + i \times 0, p_{1x}^{xy} = \rho + i \times 0, p_{1y}^{xy} = 0 + i \times \rho, p_1^z = 0, p_{1z}^{xy} = 0 + i \times 0, p_1^z = 0, p_{1x}^z = 0, p_{1y}^z = 0, p_{1z}^z = 1$. Using equations (1) and (2), the coordinate vectors p_v^{xy} and p_v^z are calculated. Then, by

substituting the coordinate vectors p_v^{xy} and p_v^z into equations (3) and (4), the final coordinate vectors p^{xy} and p^z are obtained.

When using GPS sensors, the linear equations between each node $i \in V$ and the virtual nodes $i_x, i_y, i_z \in V_v$ need to be further extended as follows.

$$E^{xy}p^{xy} = 0 + v_5$$

At this point, after setting the coordinates $p_1^{xy}, p_{1x}^{xy}, p_{1y}^{xy}, p_{1z}^{xy}, p_1^z, p_{1x}^z, p_{1y}^z, p_{1z}^z$, it is necessary to simultaneously solve equations (1), (3), and (5) to compute the node vectors p_v^{xy} and p^{xy}. We define the coordinate vector $p_{acc}^{xy} = \left[p_v^{xyT}, p^{xyT}\right]^T \in C^{4\times n}$ as the merged vector of p_v^{xy} and p_{xy}. Subsequently, equations (1), (3), and (5) are combined as follows.

$$F^{xy}p_{acc}^{xy} = G^{xy} + v_6$$

The error vector v_6 is defined as $v_6 = [v_1^T, v_3^T, v_5^T]^T$ with $v_6 \sim N(0, \Sigma_6)$. By ignoring the correlations between the individual linear equations, the covariance matrix Σ_6 becomes a diagonal matrix. It satisfies the following relationship: $\Sigma_6 = \mathrm{diag}\{\Sigma_1, \Sigma_3, \Sigma_5\}, \Sigma_3 = 0\_(|v3| \times |v3|)$.

Substituting the planar coordinates $p_1^{xy}, p_{1x}^{xy}, p_{1y}^{xy}, p_{1z}^{xy}$ into the above equation, the equation takes the following form.

$$F_{sub}^{xy}p_{sub,acc}^{xy} = G_{sub}^{xy} + v_{sub6}$$

The coordinate vector $p_{sub,acc}^{xy} \in C^{4n-4}$ is the vector p_{acc}^{xy} with the elements $p_1^{xy}, p_{1x}^{xy}, p_{1y}^{xy}, p_{1z}^{xy}$ removed. The error vector $v_{sub6} \sim N(0, \Sigma_{sub6})$ represents the noise corresponding to each linear equation. The number of linear equations in the system of linear equations remains unchanged, and the variance corresponding to each linear equation also remains unchanged. Therefore, the error vector satisfies $v_{sub6} = v_6$, and the covariance matrix satisfies $\Sigma_{sub6} = \Sigma_6$. The least-squares solution of this system of linear equations is as follows.

$$p_{sub,acc}^{xy} = \left(F_{sub}^{xy\ T}\Sigma_6^{-1}F_{sub}^{xy}\right)^{-1} F_{sub}^{xy\ T}\Sigma_6^{-1}G_{sub}^{xy}$$

The pseudocode for calculating the map scale and node coordinates is as follows.

Procedure 6: Calculation of map scale and node coordinates (using GPS data)

Input: Systems of linear equations $A^{xy}p_v^{xy} = b^{xy}, A^z p_v^z = b^z, C^{xy}p^{xy} = D^{xy}p_v^{xy}, C^z p^z = D^z p_v^z$, variance matrices Σ_1, Σ_2

Output: Virtual node coordinate vector $p_v \in R^{3\times 3n}$, node coordinate vector $p \in R^{3\times n}$, map scale ρ

Let the coordinate vector of node $i \in V$ be: $p = [p_1, p_2, \ldots, p_n] \in R^{3\times n}$.

Let the planar coordinate vector of node $i \in V$ be: $p^{xy} = \left[p_1^{xy}, p_2^{xy}, \ldots, p_n^{xy}\right] \in C^n$.

Let the vertical coordinate vector of node $i \in V$ be: $p^z = [p_1^z, p_2^z, \ldots, p_n^z]^T \in R^n$.

Let the virtual node coordinate vector be: $p_v = \left[p_{1x}, p_{1y}, p_{1z}, \ldots, p_{nx}, p_{ny}, p_{nz}\right] \in R^{3\times n}$.

Let the planar coordinate vector of virtual nodes be: $p_v^{xy} = \left[p_{1x}^{xy}, p_{1y}^{xy}, p_{1z}^{xy}, \ldots, p_{nx}^{xy}, p_{ny}^{xy}, p_{nz}^{xy}\right] \in C^n$.

Let the vertical coordinate vector of virtual nodes be: $p_v^z = \left[p_{1x}^z, p_{1y}^z, p_{1z}^z, \ldots, p_{nx}^z, p_{ny}^z, p_{nz}^z\right]^T \in R^n$.

Let the planar coordinates of Node 1 be $p_1^{xy} = 0 + i \times 0$, the planar coordinates of the virtual node 1_x be $p_{1x}^{xy} = \rho + i \times 0$, the planar coordinates of the virtual node 1_y be $p_{1y}^{xy} = 0 + i \times \rho$, and the planar coordinates of the virtual node 1_z be $p_{1z}^{xy} = 0 + i \times 0$.

Let the planar coordinate vector of virtual nodes $i_x, i_y, i_z \in V_v/(1_x, 1_y, 1_z)$ be $p_{sub}^{xy} = \left[p_{2x}^{xy}, p_{2y}^{xy}, p_{2z}^{xy}, \ldots, p_{nx}^{xy}, p_{ny}^{xy}, p_{nz}^{xy}\right] \in C^{3n-3}$.

Substituting the coordinates $p_{1x}^{xy}, p_{1y}^{xy}, p_{1z}^{xy}$ into the equation, the linear equation system $A^{xy}p_v^{xy} = b^{xy}$ is reduced to $A_{sub}^{xy}p_{sub}^{xy} = b_{sub}^{xy}$.

Define the coordinate vector $p_{acc}^{xy} = \left[p_v^{xyT}, p^{xyT}\right]^T \in C^{4\times n}$. By combining equations (1), (3), and (5), the system can be written as $F^{xy}p_{acc}^{xy} = G^{xy}$

Let the coordinate vector $p_{sub,acc}^{xy} \in C^{4n-4}$ denote the vector p_{acc}^{xy} after removing $p_1^{xy}, p_{1x}^{xy}, p_{1y}^{xy}, p_{1z}^{xy}$. The equation then becomes $F_{sub}^{xy}p_{sub,acc}^{xy} = G_{sub}^{xy}$.

The planar components of the nodes, $p_{sub,acc}^{xy}$, can be computed using the linear equation system $F_{sub}^{xy}p_{sub,acc}^{xy} = G_{sub}^{xy}$. The least-squares solution to this system is given by:

$p_{sub,acc}^{xy} = \left(F_{sub}^{xy\ T}\Sigma_6^{-1}F_{sub}^{xy}\right)^{-1} F_{sub}^{xy\ T}\Sigma_6^{-1}G_{sub}^{xy}$ (a first-order function of ρ).

Let the vertical coordinate vector of virtual nodes $i_x, i_y, i_z \in V_v/(1_x, 1_y, 1_z)$ be $p_{sub}^z = \left[p_{2x}^z, p_{2y}^z, p_{2z}^z, \ldots, p_{nx}^z, p_{ny}^z, p_{nz}^z\right]^T \in R^{3n-3}$.

Substituting the vertical coordinates $p_{1x}^z, p_{1y}^z, p_{1z}^z$ into the equation, the linear equation system $A^z p_v^z = b^z$ is reduced to $A_{sub}^z p_{sub}^z = b_{sub}^z$.

The vertical component of the virtual node coordinates, denoted as p_{sub}^z, is calculated using the linear system $A_{sub}^z p_{sub}^z = b_{sub}^z$. The least-squares solution to this linear system is given by: $p_{sub}^z = \left(A_{sub}^{z\ T}\Sigma_2^{-1}A_{sub}^z\right)^{-1} A_{sub}^{z\ T}\Sigma_2^{-1}b_{sub}^z$(fixed value).

The vertical component p^z for each node $i \in V$ is computed using the linear system $C^z p^z = D^z p_v^z$. The least-squares solution to this system is: $p^z = \left(C^{zT}C^z\right)^{-1}D^z p_v^z$ (fixed value)

The error function J_1 is defined as: $J_1 = \sum_{i\in V}\Big(\big(||p_1 -$

$p_{1x}||^2 - 1\Big)^2 + \Big(\big||p_1 - p_{1y}\big|\big|^2 - 1\Big)^2 + \Big(||p_1 - p_{1z}||^2 - 1\Big)^2\Big) = A_1\rho^4 + B_1\rho^3 + C_1\rho^2 + D_1\rho + E_1.$

The error function J_2 is defined as: $J_2 = \sum_{T_{ij}\in E}\Big(\big||p_i - p_j\big|\big|^2 - \rho_{ij}^2\Big)^2 = A_2\rho^4 + B_2\rho^3 + C_2\rho^2 + D_2\rho + E_2.$

By summing the two, the total error function J is: $J = J_1 + J_2 = A\rho^4 + B\rho^3 + C\rho^2 + D\rho + E.$

To determine the parameter ρ that minimizes the error function, the derivative of J with respect to ρ is computed:

$\frac{dJ}{d\rho} = 4A\rho^3 + 3B\rho^2 + 2C\rho + D = 0.$

The cubic equation is solved using the standard formula to find its three roots ρ_1, ρ_2, ρ_3. Any complex roots are discarded, and the second derivative of J is used to select the root ρ_i that corresponds to the minimum value of J.

The parameter ρ is then substituted into the coordinates p_v^{xy} and p^{xy} to compute the final coordinates p^v and p.

---

6. Outlier removal algorithm

In this paper, the linear equations are constructed based on the pose transformations T_{ij} between nodes. If the set of pose transformations $E = \{T_{ij}, i, j \in V\}$ in the graph $G = [V, E]$ contains outliers, the corresponding coefficients in the linear equations will also exhibit outliers. When solving the linear system using the least squares method, each linear equation corresponds to an error term. Outliers can cause significant deviations in the solution of the equations.

In the nonlinear optimization-based SLAM framework, each pose transformation T_{ij} is used to construct an error function $h(T_{ij})$ through an observation function. These error functions are then used to form error terms $e(T_{ij}) = h(T_{ij})^2$, and by summing these error terms, the overall error function J is obtained. However, it is common to apply a robust function to the error terms, as shown below.

$$\rho(x) = \begin{cases} x^2, \text{if } -1 < x < 1 \\ x, \text{if } x \geq 1 \\ -x, \text{if } x \leq -1 \end{cases}$$

The robust function ρ increases rapidly near 0 and grows much more slowly beyond a certain range. After incorporating the robust function, the error function J is expressed as follows.

$$J = \sum_{T_{ij}\in E_T} \rho\left(h(T_{ij})^2\right)$$

Due to the presence of the robust function, even if a specific pose transformation T_{ij} is classified as an outlier, the corresponding error term will not have a large magnitude. Consequently, the impact of outliers on the minimum value of the overall error function is relatively limited. Additionally, the original error term $e(T_{ij}) = h(T_{ij})^2$ is a nonlinear function, and the error term after adding the robust function, $\rho\left(h(T_{ij})^2\right)$, remains a nonlinear function. We still employ numerical optimization methods to solve for the minimum value of the error function.

However, the linear equations in this paper cannot adopt a robust function approach. If a robust function is applied to the residual terms of each linear equation, the equations would become nonlinear. The purpose of constructing pose transformations as linear equations is to reduce computational complexity. If nonlinear equations are used, this goal cannot be achieved.

Paper [4[]] proposes a method to remove outliers in pose transformations. Since the method involves several detailed steps, we provide only a brief summary of the approach. The algorithm takes as input a graph $G = [V, E]$ consisting of all pose transformations $E = \{T_{ij}, i, j \in V\}$, and outputs a set of pose transformations with outliers removed $E' = \{T_{ij}, i, j \in V\}$. By using the pose transformations in the set E' to construct the linear equations, the influence of outliers on the equations can be avoided.

The algorithm requires assigning each pose transformation T_{ij} in the input set $E = \{T_{ij}, i, j \in V\}$ a prior probability p_{ij} of being an outlier. This probability does not need to be an absolutely precise value. Instead, it can be roughly estimated using the variance σ_{ij}^2 associated with the pose transformation T_{ij}. For instance, the following scheme can be adopted.

$$p_{ij} = \begin{cases} 0.1, \text{if } \sigma_{ij}^2 < \sigma_{thres1}^2 \\ 0.2, \text{if } \sigma_{thres1}^2 \leq \sigma_{ij}^2 < \sigma_{thres2}^2 \\ 0.3, \text{if } \sigma_{thres2}^2 \leq \sigma_{ij}^2 \end{cases}$$

The probability p_{ij} in the outlier removal algorithm is used to find the route between different nodes (node i and node j); the pose transformation T_{ij} represents the edge in the path. The algorithm aims to find a path composed of edges with fewer outliers to achieve the transformation from node i to node j.

The basic approach of this outlier removal algorithm is as follows: Assume that in $G = [V, E]$, there exist two nodes $p_1, p_2 \in V$; then, there are three rotation and translation matrices $T_{12}^1, T_{12}^2, T_{12}^3$ between nodes p_1 and p_2.

At the same time, nodes p_1 and p_2 can also form a connection through another intermediate node (e.g., p_3, p_4, p_5). In this case, the pose transformation from p_1 to p_2 can be calculated as follows.

$$T_{12}^4 = T_{13}T_{32}$$
$$T_{12}^5 = T_{14}T_{42}$$
$$T_{12}^6 = T_{15}T_{52}$$

Then, the nodes p_1 and p_2 can form a connection through two additional nodes (e.g., p_6, p_7, or p_8, p_9, or p_{10}, p_{11}). The pose transformation from p_1 to p_2 can be calculated as follows.

$$T_{12}^7 = T_{16}T_{67}T_{72}$$

$$T_{12}^{8} = T_{18}T_{89}T_{92}$$

$$T_{12}^{9} = T_{1,10}T_{10,11}T_{11,2}$$

Thus, we obtain a total of 9 groups of pose transformations between node 1 and node 2, denoted as T_{12}^{i}, where $i \in [1,9]$. If all the pose transformations used (set E) contain no outliers, then these pose transformations T_{12}^{i} should be very similar. However, if one of the pose transformations used is an outlier, the corresponding T_{12}^{i} will differ significantly compared to the other transformations. We can convert the pose transformations T_{12}^{i} into translation vectors and rotation vectors, and then apply a statistics-based IQR (Interquartile Range) algorithm to each component in order to remove outliers.

7. Calculation of local coordinate system

We have calculated the coordinates of each node in graph $G = [V, E]$ in the global coordinate system Σ_g, denoted as $p = [p_1, p_2, \ldots, p_n] \in R^{3\times n}$. Let the global coordinates of node i be $p_i = [x_i, y_i, z_i]^T$, and let the neighboring nodes of node i be $j \in N_i$, where the global coordinates of node j in the global coordinate system Σ_g are $p_j = [x_j, y_j, z_j]^T$. The relative coordinates of node j with respect to node i are given by $p_j^r = [x_j - x_i, y_j - y_i, z_j - z_i]^T$. The relative coordinate vectors of all neighboring nodes in the global coordinate system are represented as $P_i^n = [p_{j1}^r, \ldots, p_{jn}^r] \in R^{3\times|N_i|}$, where $j_1, \ldots, j_n \in N_i$. Let the pose transformation from node i to its neighboring node $j \in N_i$ be denoted as $T_{ij} = \begin{bmatrix} R_{ij} & t_{ij} \\ 1 & 0 \end{bmatrix} \in R^{4\times4}$, where the translation vector is $t_{ij} = [x_{ij}, y_{ij}, z_{ij}]^T$, with $j \in N_i$. Then, the relative position vectors of the neighboring nodes $j \in N_i$ in the local coordinate system Σ_i are expressed as $Q_i^n = [t_{ij1}, \ldots, t_{ijn}] = [q_{ij1}, \ldots, q_{ijn}] \in R^{3\times|N_i|}$, where $j_1, \ldots, j_n \in N_i$.

We have already calculated the global coordinates of each node $i \in V$ and its corresponding virtual nodes $i_x, i_y, i_z \in V_v$ in the global coordinate system Σ_g, denoted as $p_{ix} = [x_{ix}, y_{ix}, z_{ix}]^T$, $p_{iy} = [x_{iy}, y_{iy}, z_{iy}]^T$ and $p_{iz} = [x_{iz}, y_{iz}, z_{iz}]^T$. The global coordinates of node i are given as $p_i = [x_i, y_i, z_i]^T$. The relative coordinates of the virtual nodes i_x, i_y, i_z with respect to the node i can be expressed as follows.

$$p_{ix}^r = [x_{ix} - x_i, y_{ix} - y_i, z_{ix} - z_i]^T$$

$$p_{iy}^r = [x_{iy} - x_i, y_{iy} - y_i, z_{iy} - z_i]^T$$

$$p_{iz}^r = [x_{iz} - x_i, y_{iz} - y_i, z_{iz} - z_i]^T$$

The relative coordinate vectors of the virtual nodes i_x, i_y, i_z in the global coordinate system are $P_i^v = [p_{ix}^r, p_{iy}^r, p_{iz}^r]$. The coordinates of the virtual nodes i_x, i_y, i_z in the local coordinate system Σ_i are given as $p_{ix}^i = [1,0,0]^T$, $p_{iy}^i = [0,1,0]^T$ and $p_{iz}^i = [0,0,1]^T$. Let the coordinate vector of the virtual nodes i_x, i_y, i_z in the local coordinate system Σ_i be denoted as $Q_i^v = [p_{ix}^i, p_{iy}^i, p_{iz}^i]$.

The coordinate vectors P_i^n and P_i^v are concatenated to form $P_i = [P_i^n, P_i^v] \in R^{3\times|N_i|+3}$, and the coordinate vectors Q_i^n and Q_i^v are concatenated to form $Q_i = [Q_i^n, Q_i^v] \in R^{3\times|N_i|+3}$. Subsequently, we can compute the rotation matrix R_i of the local coordinate system for node i using a point set registration method.

Here, we utilize the neighboring nodes $j \in N_i$ of node i and the virtual nodes i_x, i_y, i_z to construct the coordinate vectors P_i and Q_i. Although it is possible to include the virtual coordinates j_x, j_y, j_z, corresponding to the neighboring nodes $j \in N_i$, into the coordinate vectors, we have opted not to include them for the following reasons. First, as long as there are three nodes (e.g., the virtual nodes i_x, i_y, i_z), the rotation matrix R_i can be determined. Second, the information of the neighboring nodes $j \in N_i$ already sufficiently represents the positions of these nodes. Including the virtual nodes j_x, j_y, j_z of $j \in N_i$ might unnecessarily increase computational complexity.

Given a set of points with position vectors $P_i \in R^{3\times|N_i|+3}$ in the global coordinate system Σ_g, and their corresponding position vectors $Q_i \in R^{3\times|N_i|+3}$ in the local coordinate system Σ_i, we aim to find a rotation matrix R_i by constructing the following error function.

$$E(R_i) = \sum_{p_i^j \in P_i, q_i^j \in Q_i} \left|\left| R_i p_i^j - q_i^j \right|\right|^2$$

We seek the rotation matrix $E(R_i)$ that minimizes the error function $R_i \in R^{3\times3}$. This problem falls into the category of point cloud registration. The matrix R_i can be computed using the singular value decomposition (SVD) method within the Iterative Closest Point (ICP) algorithm. The matrix R_i represents the rotation that transforms the point set from the global coordinate system Σ_g to the local coordinate system Σ_i. Since the rotation of the coordinate system itself is opposite to the rotation of the point set, the rotation matrix of the local coordinate system Σ_i relative to the global coordinate system Σ_g is given by $(R_i)^{-1}$. Using the SVD-based method from the ICP algorithm, the rotation matrix R_i that minimizes the error function $E(R_i)$ is derived as follows.

$$H = P_i Q_i^T = U\Sigma V^T$$

$$R_i = VU^T$$

The singular value decomposition (SVD) of the matrix $H \in R^{3\times3}$ is computed, yielding the orthogonal matrices U and V. The rotation matrix is then calculated as $R_i = VR^T$. However, if the point set P_i or Q_i is degenerate, the resulting matrix R_i may become a reflection matrix (i.e., $|R_i| = -1$). To address this, the reflection matrix R_i can be adjusted into a proper rotation matrix using the following steps.

$$R_i = V\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{bmatrix}U^T$$

The complete formula for computing the rotation matrix is as follows.

$$R_i = V\begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & |VU^T|\end{bmatrix}U^T$$

Process 7: Calculation of local coordinate system

Input: Coordinates of node i, $p_i \in R^3$; neighboring nodes of i, $j \in N_i$; coordinates of neighboring nodes $j \in N_i$, $p_j \in R^3$; pose transformations T_{ij} of neighboring nodes $j \in N_i$; coordinates of virtual nodes i_x, i_y, i_z, denoted as $p_{ix}, p_{iy}, p_{iz} \in R^3$

Output: The rotation matrix $(R_i)^{-1}$ of the local coordinate system Σ_i relative to the global coordinate system Σ_g

Relative coordinates of a neighboring node $j \in N_i$ with respect to node i are given by: $p_j^r = [x_j - x_i, y_j - y_i, z_j - z_i]^T$.

Coordinates of neighbors $j \in N_i$ in the global coordinate system Σ_g: $P_i^n = [p_{j1}^r, \ldots, p_{jn}^r] \in R^{3\times|N_i|}, j_1, \ldots, j_n \in N_i$.

Coordinates of neighbors $j \in N_i$ in the local coordinate system Σ_i: $Q_i^n = [t_{ij1}, \ldots, t_{ijn}] = [q_{ij1}, \ldots, q_{ijn}] \in R^{3\times|N_i|}, j_1, \ldots, j_n \in N_i$.

Relative coordinates of virtual nodes i_x, i_y, i_z with respect to node i, $p_{ix}^r = [x_{ix} - x_i, y_{ix} - y_i, z_{ix} - z_i]^T, p_{iy} = [x_{iy} - x_i, y_{iy} - y_i, z_{iy} - z_i]^T, p_{iz} = [x_{iz} - x_i, y_{iz} - y_i, z_{iz} - z_i]^T$, combine these into a vector $P_i^v = [p_{ix}^r, p_{iy}^r, p_{iz}^r] \in R^{3\times3}$.

Coordinates of virtual nodes i_x, i_y, i_z in the local coordinate system Σ_i: $p_{ix}^i = [1,0,0]^T, p_{iy}^i = [0,1,0]^T, p_{iz}^i = [0,0,1]^T$, combine these into a vector $Q_i^v = [p_{ix}^i, p_{iy}^i, p_{iz}^i] \in R^{3\times3}$.

Combine the coordinates of neighbors and virtual nodes P_i^n and P_i^v into $P_i = [P_i^n, P_i^v] \in R^{3\times(3n+3)}$, combine the coordinates of neighbors and virtual nodes Q_i^n and Q_i^v into $Q_i = [Q_i^n, Q_i^v] \in R^{3\times(3n+3)}$.

Perform point set registration to compute the rotation matrix R_i. First, compute the matrix: $H = P_i Q_i^T$.

Apply singular value decomposition (SVD) to H: $H = U\Sigma V^T$.

The rotation matrix R_i is then computed as: $R_i = V \times \mathrm{diag}([1,1,|VU^T|]) \times U^T$.

The rotation matrix of the local coordinate system Σ_i relative to the global coordinate system Σ_g is $(R_i)^{-1}$.

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] Z. Wu, "Distributed self-localization for relative position sensing networks in 2D space based on gravity direction, " 10.5281/zenodo.14468023

[3] Z. Wu, "A linear SLAM backend optimization algorithm on 2D space, " 10.5281/zenodo.14468254

[4] Z. Wu, "Outlier Removal Algorithm in Pose Transformation, " 10.5281/zenodo.14467742