SLAM-homework1

YA JU
juya@shanghaitech.edu.cn
2021533088

March 2024

1 Scale drift-aware pose graph optimization

In this part, I implement scale drift-aware pose graph optimization in C++, with g2o library.

1.1 optimization items

Each vertice contains the following items: x, y, z, qx, qy, qz, qw, s. They are optimized iteratively.

1.2 error function

The error function is simple,

$$e_{i,j}(x_i, x_j) = z_{i,j} - t2v((T_i^{-1} \cdot T_j)) = z_{i,j} - t2v((v2t(x_i)^{-1} \cdot v2t(x_j))$$

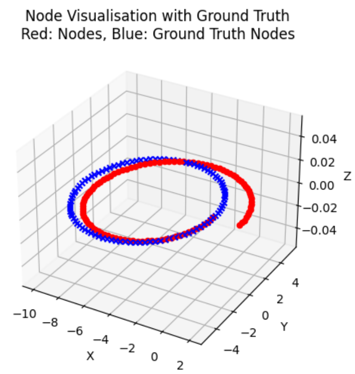
Where $z_{i,j}$ is the edge measurement vector and x_i, x_j are the respective vertices Function $v2t$ and $t2v$ are defined as:

$$v2t(x_i) = \begin{bmatrix} s_i R_i & t_i \\ 0 & 1 \end{bmatrix}$$

$$t2v(T_i) = (x_i, y_i, z_i, qx_i, qy_i, qz_i, qw_i, s_i)$$

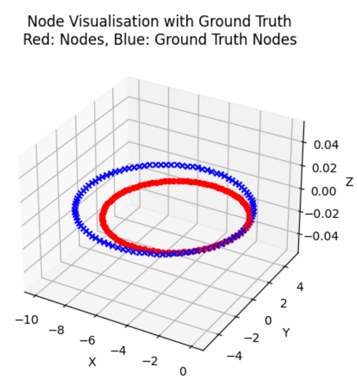
scale of the transform is extracted by taking the norm of any row or column in T_i, the normalized rotaion matrix is then obtained by dividing that scale s, a quaternion is then transformed from that rotation matrix.

1.3 optimization results



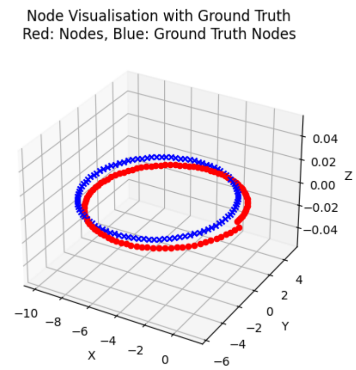
Original trajectory:

The whole system is optimized by distributing the main error of loop closure over all nodes, so there are two approaches:
1. fixing the first node so that the nodes will not drift
2. do not fix the first node, enable the system to drift a little in order to distribute the error more evenly.



Optimized trajectory(first node fixed):

we can see that the tail of the trajectory is shifted back to the position of head, achieving good loop closure, but because of the distribution of energy, the trajectory shape does not get recovered as same as ground truth.



Optimized trajectory(first node free):

we can see that although the nodes drift a little, the shape of trajectory is much closer to the ground truth shape. The tail does not overlaps with the head, presenting a bad loop closure.

2 Scale jump-aware pose graph optimization

2.1 modification of error function

According to the paper "Scale jump-aware pose graph relaxation for monocular SLAM with re-initializations", we can use another version of error function when the scale of an edge is uncertain or having scale jump, specifically in our homework data, having a -1 scale factor.
In my implementation, the new error function is:

$$e_{i,j}(x_i, x_j) = \left[z_{i,j} - t2v((v2t(x_i)^{-1} \cdot v2t(x_j))\right] \begin{bmatrix} \mathbf{I}_{7x7} & \mathbf{0} \\ \mathbf{0}^T & 0 \end{bmatrix}$$

This error function is used when the scale of measurement is -1.

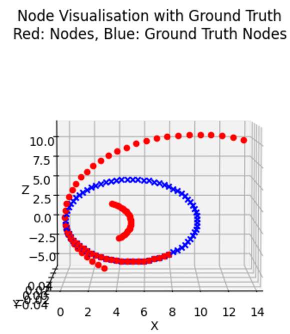
2.2 Numerical rank verification

According to the paper, we can verify if a global scale can be reconciled by checking the rank of deficiency the matrix A in the following formula:

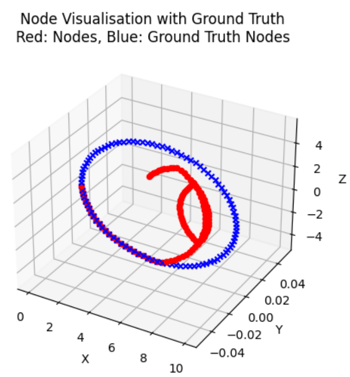
$$A = \begin{bmatrix} I_{3x3N(s1,e1)} & \mathbf{v}_1 & & \\ \vdots & & \ddots & \\ I_{3x3N(s_B,e_B)} & & & \mathbf{v}_B \\ [\mathbf{I}_{3x3} \ \mathbf{0}_{3x3(N-1)}] & \mathbf{0} & \dots & \mathbf{0} \end{bmatrix}, \ \mathbf{x} = \begin{bmatrix} \mathbf{p_1} \\ \vdots \\ \mathbf{p_N} \\ \lambda_1 \\ \vdots \\ \lambda_B \end{bmatrix}$$

$$\mathbf{Ax} = \mathbf{0}$$

2.3 Jump-aware pose graph optimization attempts



scale_jump_circle3 optimization result:



scale_jump_circle4 optimization result:

Both attempts failed, my optimizer didn't recover the global scale. Then I checked the rank of deficiency of these two senarios.

For scale_jump_circle3:
The rank of A is: 10
The rank deficiency of A is: 2

For scale_jump_circle4:
The rank of A is: 13
The rank deficiency of A is: 2

According to the theroem in the paper, large rank of deficiency might account for the optimization result.