HW3 Report

Jinxi Xiao
2021533005
ShanghaiTech University

Abstract

This report contains my solutions to the given problems as well as some of my own understandings. In addition, the sections listed in this report does not one-to-one correspond to the problems, but they do overall cover the whole problem set. [1]

1. Deal With Scale Drift

1.1. What is the scale drift?

This situation often occurs in monocular slam, and we may be confused with the concept of scale invariance at the beginning. We say that the whole slam system is scale invariance is that we can multiple an arbitrary scale factor such that the created map is still correct and reasonable, and we can adopt the method of scale propagation to ensure that. However, due to the noises within the system, our measurements are not 100 percent correct, thus lead to wrong triangularizations and thus wrong scale factors. When all these errors get accumulated, scale drift may occur. In our assignment, the case is show in Figure 1. Different from what we have seen in lidar based pose graph optimization, we can observe that the trajectory is scale-inconsistent. And this is the main point for scale-aware pose graph optimization.

1.2. Methods

All methods are inherited from [1].

We instead make use of the group $Sim(3)$, which gives

$$\exp_{\mathrm{Sim(3)}}\begin{pmatrix}\omega\\ \sigma\\ v\end{pmatrix} = \begin{bmatrix} e^{\sigma}\exp_{\mathrm{So(3)}}(\omega) & Wv \\ 0 & 1\end{bmatrix} = \begin{bmatrix} sR & t \\ 0 & 1.\end{bmatrix}$$

So we given two states S_i and S_j along with the corresponding measurement $\Delta S_{i,j}$, we define the residual as

$$r_{i,j} = \log_{Sim(3)}(\Delta S_{i,j} \cdot S_i \cdot S_j^{-1})$$

[1] Codes are stored at my github repo

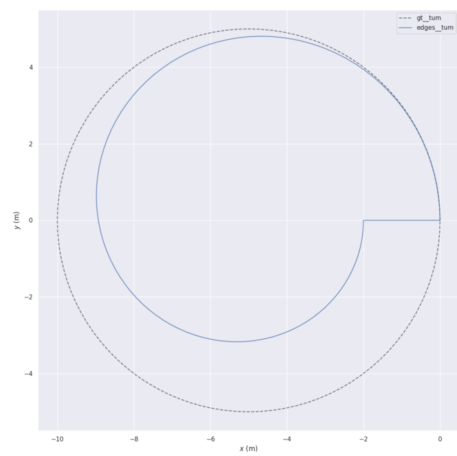

Figure 1. Trajectory of the first assignment. Dotted line is gt and the blue line is the measured trajectory.

and our goal is to minimize the error energy

$$\mathbf{e} = \sum_{i,j} r_{i,j}^T I_{7\times 7} r_{i,j}$$

1.3. Experiment and results

I adopt the library g2o to implement this work. More specifically, I use the levenberg marquardt algorithm and iterate for 10 iterations. In the end, the time consumption is around 4.5 to 5.0 seconds, and the visual results are shown in Figure 2.

2. Deal With Scale Jump

2.1. What is scale jump?

One big assumption made in scale drift is that the relative scale is always a measured. Gaussian-distributed variable. However, when the agent loses the local map and needs to reinitialize, a new scale factor is created, and it maybe inconsistent with the previous scale factors. As a result, a scale "jump" occurs.

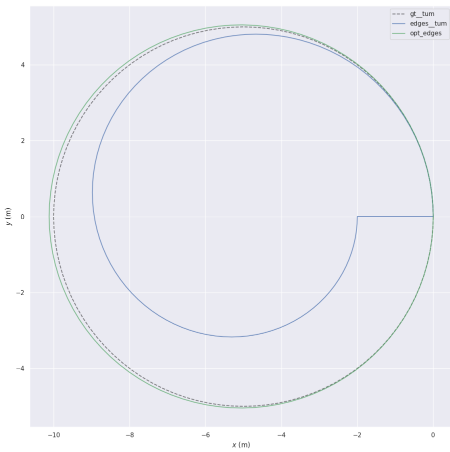

Figure 2. Trajectories after optimization. The green line is the optimized trajectory. The blue line is the origin trajectory and the dotted line is the gt.

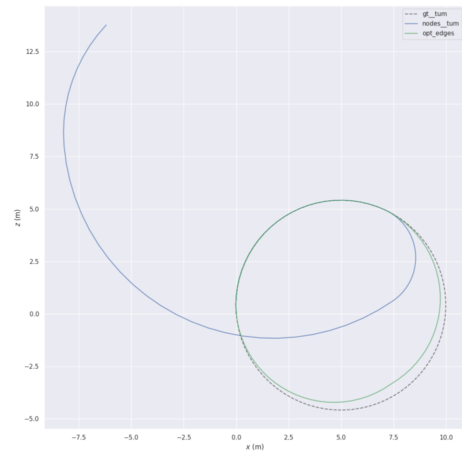

Figure 3. Trajectories for 3 jumps. The green line is the optimized trajectory. The blue line is the origin trajectory and the dotted line is the gt.

2.2. Methods

All methods are inherited from [2]. And I do not have anything more to mention.

2.3. Experiments and results

I adopt the library g2o to implement this work. More specifically, I use the levenberg marquardt algorithm and iterate for 20 iterations. In the end, the time consumption for 3 jumps is around 7.5 seconds, while for 4 jumps it would be around 11 seconds. Visual results are shown in Figure 3 for 3 jumps and Figure 4 for 4 jumps. For 3 jumps, we can see that the optimized result is very close to gt, while for 4 jumps the method does not work.

2.4. Analysis

To determine if we can obtain a scale consistent trajectory, we need to make sure that the solution x of $Ax = 0$ having one degree of freedom, where

$$A = \begin{bmatrix} \mathcal{I}_{3\times 3N}(s_1, e_1) & v_1 & & \\ \vdots & & \ddots & \\ \mathcal{I}_{3\times 3N}(s_B, e_B) & & & v_B \\ [I_{3\times 3}\ \mathbf{0}_{3\times 3(N-1)}] & \mathbf{0} & \cdots & \mathbf{0} \end{bmatrix} \in \mathbb{R}^{3(B+1)\times(3N+B)} \tag{1}$$

As for the case of 3 jumps, the matrix $A \in \mathbb{R}^{12\times 12}$ has a rank of 11, thus a global scale can be recovered. But for the case of 4 jumps, the construct of bar is similar to Figure 5. The matrix A is of shape $(15, 16)$ with rank 14, thus the degree of freedom of solution x is 2. Consequently, we can not recover a global scale.

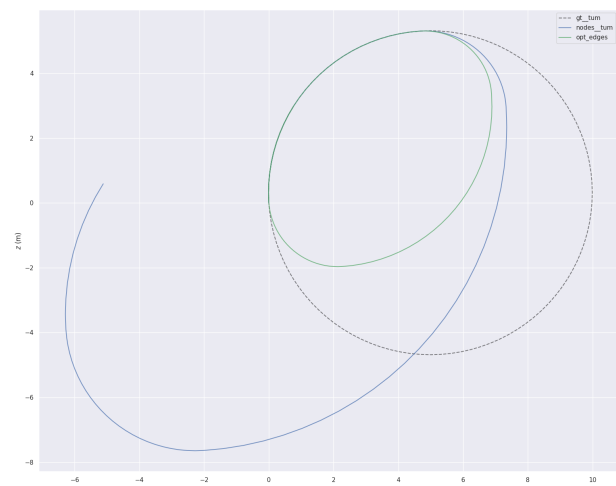

Figure 4. Trajectories for 4 jumps. The green line is the optimized trajectory. The blue line is the origin trajectory and the dotted line is the gt.

References

[1] H. Strasdat, J. Montiel, and A. J. Davison, “Scale drift-aware large scale monocular slam.,” in *Robotics: Science and Systems*, vol. 2, p. 5, 2010.

[2] R. Yuan, R. Cheng, L. Liu, T. Sun, and L. Kneipl, “Scale jump-aware pose graph relaxation for monocular slam with re-initializations,” in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3445–3452, 2023.

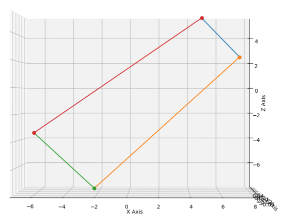

Figure 5. Construct of bar of 4 jumps case