SLAM Homework 4 - BA

Runze Yuan (yuanrz@shanghaitech.edu.cn)

November 11, 2021

1. Pipeline

Generate Poses

1. Generate 9 camera poses according to the arrangement, which is a list(length 9) of 6 dimentional vector **T**. Here we use axis-angle to represent rotation.
2. Generate 25 2d points in the reference view. Using the given intrinsic matrix and the 2m scale, we can reproject them into 3D space, which is 3x25 matrix **P**.
3. Transform the P in to 9 views using the camera poses and normalized them, which gives us 25 measurements under each view, which is a list(length 9) of 2x25 matrix. **M**

Add noise

1. perturb all the T_i with a gaussian noise $\mathcal{N}(0, 0.002)$
2. perturb all the P_i with a gaussian noise $\mathcal{N}(0, 0.02)$
3. project the M_i into image plane and perturb it with a gaussian noise $\mathcal{N}(0, 0.25)$. Then reproject it into the normalized plane.

Optimization

Given noisy poses, 3d points and 2d measurements, this section we implement BA to optimize our camera poses and 3d points.

The objective we optimized is the reprojection error of the 3d points. Note that it is the reprojection error in normalized plane rather than image plane, in which we can avoid intrinsic parameter in our residual.

Here the optimized variable is the change of the initial value.

The object function is as follow

$$\min_{\delta R, \delta t, \delta P} \sum_i ||M_i - \frac{\delta R R(P + \delta P - t - \delta t)[:2]}{\delta R R(P + \delta P - t - \delta t)[2]}$$

We use LM method to do the optimization problem, here we

1. Given a initial state which is the initial value of R,t,P. Create a change state Δ of length $9*6+25*3 = 129$ and initialized with 0, which is the initial value of the variable we would optimize.$\Delta = [\delta R, \delta t, \delta P]$

2. For each view and point, we project the point into the view. Calculate the error with respect to the corresponding measurement, which is e. Also, we calculate the 2x129 jacobian J. Sum the J^TJ and $-J^Te$ respectively. Here we choose to use numerical differentiation to calculate jacobian, the *delta* chosen is $10e-5$
3. solve the linear euqation $\sum J^TJ\Delta_i = \lambda + \sum -J^Te$. The initial $\lambda = 10e-3$, if the solving fails $\lambda = \lambda * 10$, otherwise, $\lambda = \lambda/10$
4. $\Delta += \Delta_i$
5. iterative until reach the max iteration

Usage

1. Dependency:
   (a) Python3, numpy, scipy (for computation)
   (b) matplotlib (for visualization)
2. Run: **python BA.py**

2. Evaluation

The performance is shown below

Table 1: performance table

	reproj_error_before reproj_error_after	noise_r_error opt_r_error	noise_t_error opt_t_error	noise_p_error opt_p_error	time
std_2d(0.25) std_3d(0.02) std_pose(0.002)	0.23725 **0.00944**	0.02741 **0.00628**	0.02443 **0.01589**	0.83088 **0.14956**	3.00922s
std_2d(1) std_3d(0.02) std_pose(0.002)	0.24468 **0.03545**	0.02713 **0.04572**	0.02031 **0.10263**	0.91976 **0.69121**	3.29432s
std_2d(1) std_3d(0.2) std_pose(0.002)	2.18324 **0.00892**	0.03124 **0.01027**	0.02632 **0.03417**	8.25131 **2.18096**	2.99967s
std_2d(1) std_3d(0.2) std_pose(0.2)	5.08768 **0.008837**	2.54156 **0.01039**	3.22236 **0.03222**	0.77398 **0.80116**	2.97716s

Here we tested four kinds of noise, which are small noise, large 2d noise, large 3d noise, large pose noise.

1. In the first situation, which is that the noise is relative small, we can see the the algorithm can somehow converge to a relative good result.
2. In the second situation, which is that the noise of measurement is large. We can see that, the optimized pose is even worse than the noisy one. And the reprojection error is relative larger than other one's. This makes sense since our measurement is inaccurate.

3. In the third situation, which is that the noise of the 3d points is large, we can observe that the error on point is larger. Also we can see from the visualization that there exists a large gap along z-axis. This is because that the 3d points are all on a plane, which lacks the observation along z-axis. Then the result is not good.

4. In the last situation, which is that the noise of the pose is large. we can observe that result is relative good except for little error on 3d point. This indicates that given measurements and 3d points with a little noise, BA can give a relative good result on poses.

The visualization of the four situation is as below, note that the green one is ground truth, the red one is the noisy data and the blue one is the optimzied data.

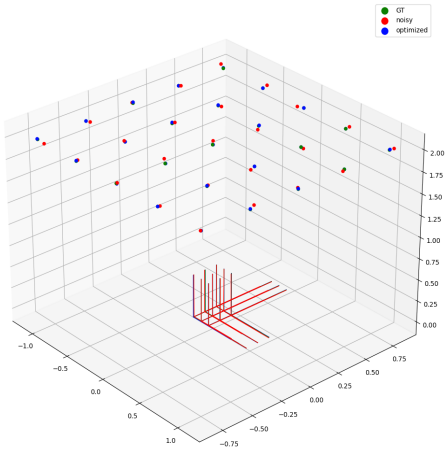

Figure 1: std_2d(0.25),std_3d(0.02),std_pose(0.002)

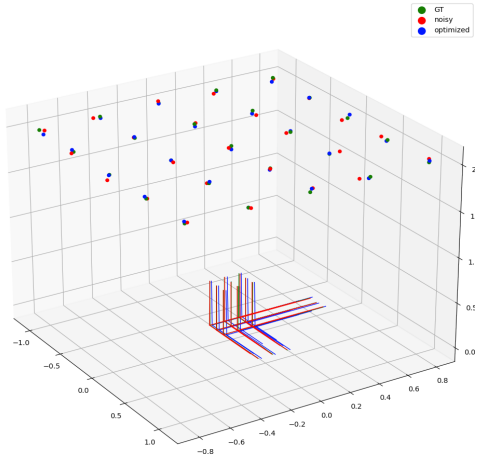

Figure 2: std_2d(1),std_3d(0.02),std_pose(0.002)

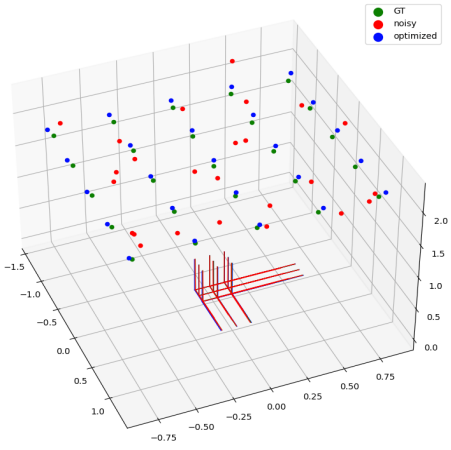

Figure 3: std_2d(0.25),std_3d(0.2),std_pose(0.002)

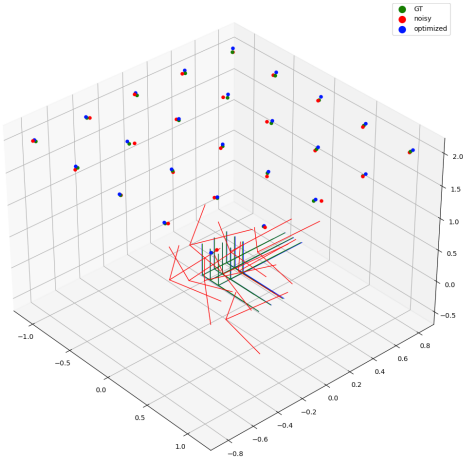

Figure 4: std_2d(1),std_3d(0.02),std_pose(0.2)

Currently, the algorithm is not robust if added much noise. The possible improvement should be trying more optimization method like Dog-leg. Also, the differentiation here is numerical differentiation. We should use analytical differentiation instead to speed up the algorithm. Last, here we directly solve the linear equation when computing the updating step, we should try other metho d like cholesky decomposition to speed up.