The 2010 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 18-22, 2010, Taipei, Taiwan

Evaluation of Pose Only SLAM

Gibson Hu, Shoudong Huang and Gamini Dissanayake

Abstract— **In recent SLAM (simultaneous localization and mapping) literature, Pose Only optimization methods have become increasingly popular. This is greatly supported by the fact that these algorithms are computationally more efficient, as they focus more on the robots trajectory rather than dealing with a complex map. Implementation simplicity allows these to handle both 2D and 3D environments with ease. This paper presents a detailed evaluation on the reliability and accuracy of Pose Only SLAM, and aims at providing a definitive answer to whether optimizing poses is more advantages than optimizing features. Focus is centered around TORO, a Tree based network optimization algorithm, which has gained increased recognition within the robotics community. We compare this with Least Squares, which is often considered one of the best Maximum Likelihood method available. Results are based on both simulated and real 2D environments, and presented in a way where our conclusions can be substantiated.**

I. INTRODUCTION

One of the main focuses on current SLAM research is the development of solutions to improve SLAM efficiency without compromising the accuracy. SLAM itself can be considered as an optimization problem, solved by combining the information gained from sensor observations and robot odometry. Researchers have typically resorted to simplifying data and fitting it around point feature based solutions which aims to compute optimal locations of both features and robot poses [1].

Recently, Pose Only SLAM approaches have gained popularity [2] [3]. A Pose Only implementation typically divides SLAM into two separate phases, one for the identification of relative pose constraints and the second for the optimization of robot poses. During the first phase, the consistency of information use must be monitored and information reuse must be avoided. For the second phase, the major focus is on efficiency and accuracy, that is, how to get a good quality solution quickly.

A popular Pose Only SLAM algorithms is Tree-based Network Optimizer or TORO. It has been evaluated to be much faster than most standard maximum likelihood approaches and stated to work well in many different applications [3].

When it is difficult or impossible to extract features from the sensor data, Pose Only SLAM seems to be a good choice for optimizing the robot poses and locating the robot in an unknown environment. However, if there are good quality features that can be extracted from the environment, how much accuracy or consistency is compromised for the efficiency in TORO or Pose Only SLAM?

In this paper, we assume the point feature based SLAM set up and ask the above question. We want to know whether it is necessary to compute the optimal locations of features at all if accurate feature positions can be gained from Pose Only SLAM. In other words, how accurate are the implementations of Pose Only SLAM when compared with the Full Least Squares solution. Also how does TORO's result differ from a Least Squares based optimization where information use is maximized?

The paper is structured as follows. Section II explains the three SLAM techniques we used to conduct our experiments. Section III explains our approach to maintain consistency when obtaining Pose Only constraints. Section IV describes our evaluation methods. Section V presents experimental results and Section VI discusses related work. Lastly, Section VII draws conclusions on our findings.

II. THREE SLAM ALGORITHMS

A fair comparison to a feature based SLAM solution can be made by providing a Least Squares benchmark. Unlike methods such as Extended Kalman Filtering (EKF), the Least Squares solution keeps all the robot poses and features in its state vector to avoid any information loss.

The computational efficiency can be improved by means of map joining or exploiting the sparseness of its information matrix [6]. The Least Squares result is arguably one of the most accurate estimations one can achieve.

The basic principle behind Least Squares is the minimization of the error function

$$(Z - F(X))^T P^{-1} (Z - F(X)) \tag{1}$$

where X is the state vector, Z is the measurement information and P is the covariance of the measurement. The problem itself is not always linear therefore the state vector should be solved iteratively by

$$(J^T P^{-1} J) \cdot X_{k+1} = J^T \cdot P^{-1} (Z - F(X_k) + J \cdot X_k) \tag{2}$$

where J is the Jacobian.

Often the algorithm can be made more robust by using a Levenberg Marquardt implementation where a damping factor is introduced to improve the convergence.

In this paper we use Least Squares in two ways.

This work is supported by the ARC Linkage Grant (LP0884112), the ARC Centre of Excellence program (funded by the Australian Research Council (ARC) and the NSW State Government), the Pempek Systems Pty Ltd and the University of Technology, Sydney, Australia. Gibson Hu, Shoudong Huang and Gamini Dissanayake are with Faculty of Engineering and Information Technology, University of Technology Sydney, PO Box 123 Broadway NSW 2007, Australia {ghu,sdhuang,gdissa}@eng.uts.edu.au

A. Full Least Squares SLAM (F-LS)

Assuming the measurement Z_{pf} contains both robot odometries and observations. A F-LS solution optimizes the whole state vector X_{pf} in one go using this information.

The state vector X_{pf} in this algorithm thus contains all the robot poses and all the feature positions. That is

$$\begin{aligned} X_{pf} &= (R_1, R_2, \cdots, R_m, L_1, L_2, \cdots, L_n) \\ &= (x_1^r, y_1^r, \phi_1^r, \cdots, x_1^l, y_1^l, \cdots) \end{aligned}$$

where R_i is the global robot pose and L_i is global feature (landmark) position.

The measurement vector Z_{pf} contains all the available odometry and observation information

$$Z_{pf} = (R_{(01)}, O_{(0,1)}, O_{(0,2)}, \cdots R_{(12)}, O_{(1,1)}, O_{(1,2)} \cdots).$$

Here $R_{(01)}$, $R_{(12)}$ are the odometry (constraint between 2 adjacent poses) and $O_{(i,j)}$ is the observations made from pose i to landmark j.

A major issue associated with the F-LS approach, is that when the environment becomes complex with too many features the algorithm can becomes inefficient with a high computation cost. In terms of the accuracy, the F-LS can be considered as a benchmark for testing other SLAM algorithms.

B. Pose Only Least Squares (PO-LS)

If we can somehow transfer the original pose-to-feature observation information into relative pose constraint information, then we can apply the Pose Only SLAM techniques. When the relative pose constraints information is given, it can be argued that a Least Squares implementations will provide the best achievable solutions for optimizing only poses.

In this case, the state vector contains robot poses only and is expressed by

$$X_{po} = (R_1, R_2, \cdots, R_m).$$

The measurement information available is now

$$Z_{po} = (R_{(0,1)}, R_{(0,3)}, R_{(0,4)}, \cdots, R_{(1,3)}, R_{(1,4)} \cdots)$$

where $R_{(i,j)}$ is the relative pose from pose i to pose j.

C. TORO

The input to TORO is also Z_{po}. TORO is an efficient Pose Only SLAM algorithm combining the ideas of Grisetti *et al.* [3] and research done by Olson *et al.* [13], who was one of the first to introduce Stochastic gradient decent (SGD) to graph based approaches. The results have given rise to faster processing times as compared to traditional least squares approaches.

The SGD equation is governed by

$$X_{po}^{t+1} = X_{po}^t + \lambda \cdot K_{po} J_{po}^T P_{po}^{-1} (Z_{po} - F_{po}(X_{po}^t))$$

Here the state vector X_{po}^t contains only poses. J_{po} is the Jacobian of the error function, and $(Z_{po} - F(X_{po}^t))$ is the residual and K_{po} is a pre-conditioning matrix computed from the Hessian matrix.

The method argues that by selecting more important constraints to use, optimization can still produce a near accurate solution in most cases [10]. The major drawback of TORO which is not present in a F-LS solution, is the inability to handle non spherical covariances. Thus TORO can only be used in Pose Only SLAM instead of feature based SLAM.

III. OBTAINING RELATIVE POSE CONSTRAINTS

Before applying the Pose Only SLAM approach, we find getting information Z_{po} from Z_{pf} becomes a critical step. The process needs to be carefully performed such that information can be extracted with limited or no information loss or information reuse. For example, we can not simply use the observations made from robot pose i many times to obtain the relative pose constraints between pose i and other poses.

In this paper, we propose two different methods to obtain the constraint without information reuse. Both methods are based on the following idea: under Gaussian noise assumption, a single observation $L_{i,j}$ with covariance $P_{i,j}$ is equivalent to k observations each with covariance $k \times P_{i,j}$ in terms of information content. While these methods are not completely novel approaches, they are satisfactory in addressing our Pose Only SLAM evaluations.

A. Method 1

Only using adjacent poses, to calculated SLAM, is a popular techniques used by many researcher. However there will be some information loss present in these methods. The reason for using Method 1 in this paper, is to validate how much information may be lost when Pose Only SLAM is applied to this type of approach.

In Method 1 observations are only ever used once or twice, which implies that the relationships are built up only between adjacent poses. e.g. $R_{(0,1)}, R_{(1,2)}, R_{(2,3)}, ..., R_{(n,0)}$.

Algorithm 1 Adjacent Relative Pose Extraction Method

```
1: Associate features between only adjacent poses.
2: Check if observations are used once or twice.
3: Loop for all poses
4: Double the covariance of all observation which are used
   twice.
5: Obtain relative pose constraint using least squares
6: End.
```

Odometry information are used in the calculation of all relative poses except for constraint $R_{(n,0)}$. For computing the relative pose constraint, a F-LS SLAM is first performed using the observation and odometry information within the two steps, then the features are marginalized out from the state vector to get the relative pose.

B. Method 2

In Method 2 we aim to maximize the information usage by trying to build as many relative pose relationships as possible. Information reuse is avoided in an offline perspective and by, $k \times P_{i,j}$, where k is the frequency observation $O_{(i,j)}$ is used and $P_{i,j}$ is the corresponding measurement covariance. The full potential of Pose Only SLAM optimization can be exploited without any information reuse.

Algorithm 2 Multiple Relative Pose Information Extraction

```
1: Obtain observations from all poses.
2: Choose pose pairs with at least 2 common features to find constraints.
3: Include odometry information for only adjacent poses.
4: Multiply covariance of observation by observation frequency.
5: Obtain relative pose constraints by least squares.
6: Loop for all poses and build up constraints vector $Z_{po}$ with new covariance $P_{po}$.
7: End
```

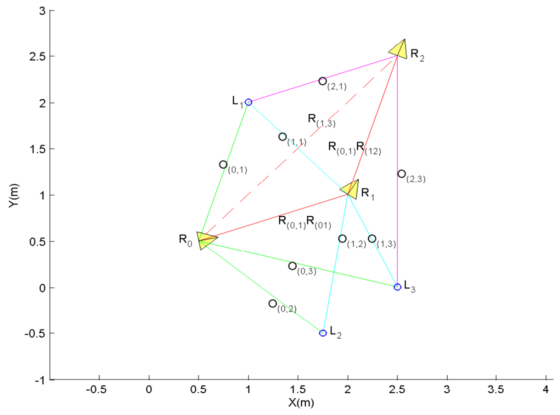

Fig. 1. Method 2

Fig.1 shows an example of information usage in Method 2. Here we assume there are only three robot poses and three landmarks. When computing relative pose $R_{(0,1)}$, odometry $R_{(01)}$ and observations $O_{(0,1)}$, $O_{(0,2)}$,$O_{(0,3)}$, $O_{(1,1)}$, $O_{(1,2)}$, $O_{(1,3)}$ are used; when computing relative pose $R_{(1,2)}$, we use $R_{(12)}$, $O_{(1,1)}$, $O_{(1,3)}$,$O_{(2,1)}$, $O_{(2,3)}$; when computing relative pose $R_{(0,2)}$, we use $O_{(0,1)}$, $O_{(0,3)}$,$O_{(2,1)}$, $O_{(2,3)}$.

Given that $O_{(0,2)}$ and $O_{(1,2)}$ are only every used once in this simulation, their covariance remain the same. The rest of the observations are used exactly twice and need to be multiplied by 2. Covariance multiplication does not apply to odometry information however, since it is only ever used once. To confidently obtain non adjacent constraints from least squares optimization without divergence, an adequate amount of features must be associated. For example, to calculated $R_{(0,2)}$ at least 2 common features must be observed from R_0 and R_2.

IV. QUANTIFICATION OF ESTIMATION RESULTS

We now have three different approaches to solve the SLAM problem.

A. input Z_{pf}, [F-LS], output X_{pf}.

B. input Z_{pf}, [Method 1 or 2], transfer to Z_{po}, [PO-LS], output X_{po}.

C. input Z_{pf}, [Method 1 or 2], transfer to Z_{po}, [TORO], output X_{po}.

How can one compare the results obtained? What are the measures that should be compared?

In this paper, the comparisons will focus on estimation consistency and accuracy. Estimation consistency is a crucial requirement for any algorithm. Roughly speaking, an estimation algorithm is consistent if the uncertainty predicted by the algorithm accurately represents the actual estimation error.

Four tests are conducted to evaluate our three algorithms.

A. *2σ bound check on consistency*

When the ground truth is available, one simple way to evaluate the consistency is by comparing the actual estimation error with its 2σ bound. This can be done for a single simulation run.

B. *NEES, Consistency check on robot pose estimate*

Another more accurate way of quantify the consistency is to run the simulation a few times (each time with difference random noise seed) and then compute the average normalized estimation error squared (NEES). Commonly known as a (χ^2) Chi Square test with n degree of freedom.

An average NEES can only be done where ground truth is available. This test allows us to see the exact consistency of each of the three SLAM algorithms, TORO, PO-LS, and F-LS. Before we can do this test however, poses and respective covariances must be extracted from the F-LS result.

$$\hat{X}_{pf} \rightarrow \hat{X}_{po}, \hat{P}_{pf} \rightarrow \hat{P}_{po}$$

Performing NEES on TORO gives rise to another issue. TORO itself does not return an information matrix hence no $\hat{P}_{po}$ is obtained directly from the algorithm. We resolve this by using the resultant $\hat{X}_{po}$ estimate from TORO to calculate its Jacobian. The information matrix $\hat{P}_{po}^{-1}$ can then be derived easily.

The NEES equation is

$$(X_{po}^{true} - \hat{X}_{po})^T \hat{P}_{po}^{-1} (X_{po}^{true} - \hat{X}_{po})$$

where X_{po}^{true} is ground truth robot poses. $\hat{X}_{po}$ is estimated robot poses obtained from the algorithms, and $\hat{P}_{po}$ is the covariance matrix of the estimate.

The comparison can now be made against 95% probability region of the χ^2 distribution. Also known as the Gate.

C. *χ^2_{po}, Accuracy of Pose Only methods*

Where ground truth is not available, an χ^2_{po} on relative pose constraint error can be performed when evaluating the accuracy of TORO when compared with PO-LS. The aim is to examine how much information is lost through approximations used in TORO.

To test the χ^2_{po} using constraint information, the following equation is used.

$$\chi^2_{po} = (Z_{po} - F_{po}(X_{po}))^T P_{po}^{-1} (Z_{po} - F_{po}(X_{po}))$$

Here Z_{po} is the relative pose constraint, and P_{po} is the corresponding covariance matrix, $F(X_{po})$ is the function relating the poses X_{po} to the constraints Z_{po}.

D. χ^2_{pf}, Accuracy comparison to feature based SLAM

Where ground truth is not available, one way to compare all three algorithms (TORO, PO-LS and F-LS) is by comparing their outputs with the initial input data Z_{pf}. To justify an accurate comparison all state vectors must be equal. Our issue that needs addressing is the fact that Pose Only SLAM does not produce landmark location estimates.

To overcome this, we can simply take the robot poses obtained from TORO (or PO-LS), fix their values and find the corresponding feature that best fit the data Z_{pf}, to obtain an estimate of $\hat{X}_{pf}$ [6]. After $\hat{X}_{pf}$ is obtained, the equation for computing the χ^2 error is

$$\chi^2_{pf} = (Z_{pf} - F_{pf}(\hat{X}_{pf}))^T P_{pf}^{-1}(Z_{pf} - F_{pf}(\hat{X}_{pf}))$$

In this test Z_{pf} and P_{pf} are the initial data and covariance values used for F-LS, $\hat{X}_{pf}$ is the estimate generated by each of the three SLAM algorithms now containing both landmark locations and robot poses.

V. Experiments

A. Comparing Method 1 and Method 2 for relative pose extraction

Firstly, we aim to identify the respective accuracies of the two relative pose extraction methods. The data for this experiment comes from simulation number 3 with noise seed (a). See Fig.3 and Table I. We use PO-LS in optimization and check the performance based on their 2σ bound and $(X_{po}^{true} - \hat{X}_{po})$ error values.

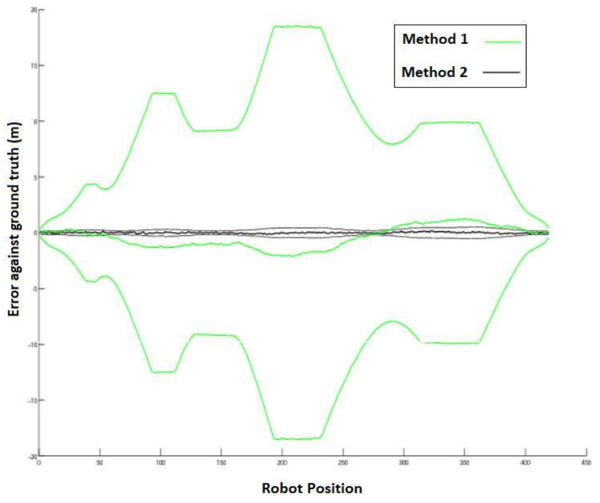

Fig. 2. PO-LS, Method 1 (black) and Method 2(green) 2σ Bound Comparison

When looking at Fig.2 we can see that although both the two estimate are consistent (staying within their 2σ bound), the uncertainty of the estimate obtained from Method 1 is much greater then that of Method 2, with some 2σ values reaching 18 meters. Clearly using Method 2 is more meaningful for evaluating Pose Only SLAM. Thus the method described in Section III(B) is used in all the rest of the simulation results.

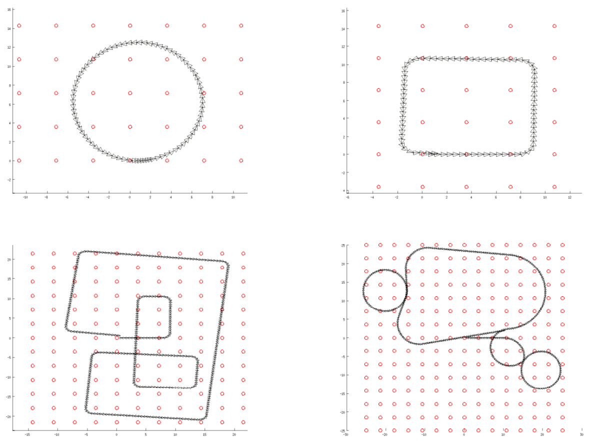

Fig. 3. Simulated Trajectories Top Left:1, Right:2, Bottom Left:3, Right:4

B. Simulation set up

The simulation set up follows the procedure outlined below.

- Simulate a trajectory, obtain odometry and observation information.
- Get constraints from Method 2.
- Apply PO-LS and TORO optimizer.
- Conduct a F-LS on initial data.
- Evaluate results by testing NEES and χ^2 .

Four trajectories shown in Fig. 3 were used in the evaluation. The environment consists of 225 point features all uniformly distributed. Fig. 3 shows the four scenarios, two with 82 Poses and two with 420 Poses. The more complex trajectories consist of several points where loop closure occurs.

For each step the robot moves 0.5 meters and rotates at a predefined angle, then observes any features within its sensor range (5m with 180 degree field of view). For each trajectory we test for three different noise level: (a) regular environment where the sensor and odometry noise is low; (b) changes in terrain resulting in higher odometry noise; (c) environmental effect resulting in higher sensor noise. The noise values are distributed using a Gaussian model, with the standard deviation values as described in Table I. Simulations for each noise type were repeated 10 times. The mean and standard deviation of corresponding NEES and χ^2 values are listed in Tables II,III.

The DLR-Spatial-Cognition data was selected as our real data to be evaluated. This data set is available at *https://svn.openslam.org/data/svn/2d-i-slsjf*. Data was collected with a robot equipped with a camera, moving around in a building scattered with artificial landmarks (white/black circles) placed on the ground. The images acquired from the camera has been preprocessed and the relative position of the observed landmarks with respect to the observation point, are provided. This data contains both odometry and landmarks with good uncertainty measurements. Preprocessing of data has been performed with known data associations. There is a total of 3296 poses, 539 landmarks results in 14163 observation. Method 2 was used to obtain constraints in order to compute relative pose. As ground truth is not known only the χ^2 tests can be performed.

TABLE I
STANDARD DEVIATION FOR GENERATING GAUSSIAN NOISE SEEDS

Noise Type	Odometry			Observation	
	dx(m)	dy(m)	dθ(rad)	dx(m)	dy(m)
a	0.1	0.1	0.05	0.1	0.1
b	0.2	0.2	0.2	0.1	0.1
c	0.1	0.1	0.05	0.3	0.3

C. Results

As seen in Table II NEES values for PO-LS and F-LS stay very constant in every noise situation. Also PO-LS out performs F-LS in the optimization of robot positions when compared against ground truth. TORO however shows a lot of inconsistency, high mean and standard deviation. Especially in the case of noise (b), where sensor noise is high. When larger trajectories are used, TORO's accuracy drops dramatically, in all cases surpassing the gate values.

χ^2_{po} results from Table II confirms PO-LS to be a much more robust technique. Relative poses are optimized with far better accuracies than that of TORO.

A very interesting observation can be made when looking at the outcome of χ^2_{pf}, shown in Table III. When using PO-LS there is high indication that only a small amount of information is lost from the original data. This is evident when we compare the PO-LS values with F-LS values. There is only a slight increase in PO-LS error which leads us to believe that optimization of landmark is not such a big component when it comes to SLAM.

Finally, results from the DLR data set, Table IV, supports our claim with its result reflecting those of the simulations. TORO still shows very high inconsistency. Looking at Fig. 4, rotational error appears to be the contributing factor.

VI. DISCUSSION AND RELATED WORK

When Method 2 is applied the approximation effects of TORO are definitely noticed. The NEES test justifies Pose Only SLAM to be quiet effective in optimizing error, staying below the gate value during the majority of tests. When we do a full comparison with the F-LS, indicated in χ^2_{pf}, we can see little information is lost in PO-LS.

Nowadays more and more SLAM algorithms are being developed. Evaluation of different SLAM algorithms is becoming an important issue and has attracted more attention in the past few years. For example, Burgard et. al [5] and Kummerle et.al [12] provided an objective benchmark for comparing different trajectory based SLAM algorithms. The metric used is the relative position of poses for comparing the accuracy of the trajectories obtained from different SLAM algorithms, which allows us to compare SLAM approaches that use different estimation techniques or different sensor modalities since all computations are made based on the corrected trajectory of the robot. In [6], some performance metrics for comparing the consistency, accuracy and efficiency of different point-feature based SLAM algorithms are proposed. Moreover, a number of research groups [7][8] have collected large-scale experimental data with accurate ground truth such that different SLAM algorithms can be evaluated using real data.

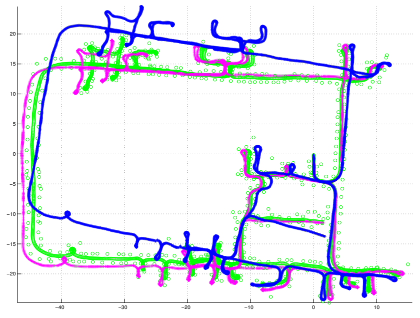
Fig. 4. DLR results (Green: F-LS Pink: PO-LS Blue: TORO)

One important issue in SLAM or any other information fusion techniques is information reuse. One way to deal with information reuse is first use whatever information available to get the estimate, then using Covariance intersection (CI) (see [14] and [15]), that facilitates combining two correlated pieces of information, when the extent of correlation itself is unknown is used to fuse these two estimates. Another way to separate the observations made from a particular pose into two parts, one part is used to compute the relative pose with respect to the previous pose and the other part is used to compute the relative pose to the next pose [9]. However, both there methods cause some information loss.

In this paper, we dealt with this issue in a different way. We assume the Gaussian noise assumption and separate one single observation into different parts - each with reduced amount of information (enlarged covariance matrix). This provides us the information fusion results without information reuse and without information loss.

VII. CONCLUSION

After careful evaluation we can say that PO-LS is able to achieve results accurately without much information loss, provided that the relative pose information is extracted properly. Thus PO-LS can be regarded as a promising alternative for F-LS when the computational cost of F-LS becomes an issue.

It is evident that approximations involved in TORO does seem to affect its ability to fully optimize its solution in some scenarios, as compared with PO-LS. However without more

TABLE II

NEES AND χ^2_{po} TESTS, MEAN(STANDARD DEVIATION) VALUES FROM 10 SIMULATIONS

Trajectory Noise Type	NEES GATE	NEES PO-LS	NEES TORO	NEES F-LS	χ^2_{po} PO-LS	χ^2_{po} TORO
1(a)	280.36	152.76 (9.60)	413.73 (391.54)	190.74 (14.09)	33.96 (5.76)	301.73 (359.16)
1(b)	280.36	128.79 (19.26)	179.76 (74.7)	189.35 (16.90)	18.47 (3.13)	67.78 (48.66)
1(c)	280.36	188.33 (12.42)	2280.94 (354.64)	193.19 (11.30)	34.13 (5.74)	2145.99 (351.48)
2(a)	280.36	157.80 (12.79)	193.73 (18.49)	195.88 (10.51)	29.08 (3.29)	82.85 (12.82)
2(b)	280.36	125.33 (12.0)	144.53 (17.8)	192.56 (11.77)	16.62 (2.84)	39.08 (12.67)
2(c)	280.36	181.44 (12.42)	722.07 (110.32)	189.25 (12.0)	31.45 (4.64)	613.88 (111.5)
3(a)	1340.59	628.77 (14.62)	2561.88 (328.61)	957.89 (39.06)	697.97 (12.95)	2958.97 (328.76)
3(b)	1340.59	551.32 (24.26)	3022.718 (3288.33)	968.88 (49.23)	562.01 (14.88)	3120.39 (3328.61)
3(c)	1340.59	798.08 (42.39)	10688.30 (692.99)	965.68 (46.8)	736.19 (17.04)	11180.28 (701.09)
4(a)	1340.59	661.12 (14.62)	2273.29 (328.61)	987.11 (39.06)	641.43 (12.95)	2523.50 (328.76)
4(b)	1340.59	581.17 (41.02)	1475.20 (466.24)	975.03 (35.2)	392.25 (196.43)	1351.07 (727.41)
4(c)	1340.59	809.64 (42.39)	8851.06 (692.9)	980.33 (46.8)	674.88 (17.04)	9264.40 (701.09)

TABLE III

χ^2_{pf} TESTS, MEAN(STANDARD DEVIATION) VALUES FROM 10 SIMULATIONS

Trajectory Noise Type	PO-LS	TORO	F-LS
1(a)	539.13 (80.77)	1039.25 (669.99)	327.50 (16.79)
1(b)	612.41 (104.46)	1010.63 (471.12)	339.15 (21.05)
1(c)	362.40 (16.06)	2505.23 (358.56)	327.76 (14.11)
2(a)	501.19 (31.38)	569.44 (69.19)	339.05 (11.10)
2(b)	501.28 (26.05)	614.02 (96.61)	336.09 (16.3)
2(c)	398.41 (23.02)	1004.85 (120.62)	342.19 (17.87)
3(a)	3290.44 (90.83)	7146.97 (743.58)	3135.11 (46.89)
3(b)	3252.42 (49.24)	8666.17 (6520.58)	3137.72 (48.59)
3(c)	3264.81 (42.69)	15581.44 (1236.35)	3121.31 (30.23)
4(a)	3248.97 (90.83)	6826.11 (743.58)	2970.42 (46.89)
4(b)	3254.75 (56.74)	6550.57 (1258.82)	2652.96 (536.5)
4(c)	3187.42 (42.69)	13135.66 (1236.35)	2988.83 (30.23)

TABLE IV

χ^2 ERROR COMPARISON USING DLR DATA SET

χ^2_{po}	PO-LS	TORO	
	8750	1291644	
χ^2_{pf}	PO-LS	TORO	F-LS
	36232	1343662	27678

testing using larger data sets, the tradeoff between efficiency and accuracy involved in TORO is still undetermined.

In the future, we are planning to use more large-scale data sets to compare the three algorithms in both 2D and 3D scenarios. Some performance comparison with other SLAM algorithms based on local map joining will also be very interesting.

REFERENCES

[1] F. Dellaert and M. Kaess, “Square root SAM: Simultaneous localization and mapping via square root information smoothing”. *International Journal of Robotics Research*, vol. 25, no. 12, December 2006, pp. 1181-1203.

[2] F. Lu and E. Milios. Globally consistent range scan alignment for environment mapping. *Autonomous Robots*, 4:333-349, 1997.

[3] Giorgio Grisetti, Cyrill Stachniss, Slawomir Grzonka, and Wolfram Burgard; ”A Tree Parameterization for Efficiently Computing Maximum Likelihood Maps using Gradient Decent.”, *Robotics: Science and Systems (RSS)*, 2007

[4] G. Grisetti, D. L. Rizzini, C. Stachniss, E. Olson and W. Burgard, “Online constraint network optimization for efficient maximum likelihood mapping”. *In Proceedings of 2008 IEEE International Conference on Robotics and Automation (ICRA)*, Pasadena, California, on May 19-23, 2008.

[5] W. Burgard, C. Stachniss, G. Grisetti, B. Steder, R. Kuemmerle, C. Dornhege, M. Ruhnke, A. Kleiner, J. D. Tardos, “A comparison of SLAM algorithms based on a graph of relations.” IROS, 2009

[6] S. Huang, Z. Wang, G. Dissanayake, and U. Frese, “Iterated D-SLAM map joining: evaluating its performance in terms of consistency, accuracy and efficiency.” *Autonomous Robots*, (2009) 27: 409-429

[7] J. L. Blanco, F. A. Moreno, J. Gonzalez, A collection of outdoor robotic datasets with centimeter-accuracy ground truth. *Autonomous Robots*, (2009) 27: 327-351.

[8] S. Ceriani, G. Fontana, A. Giusti, D. Marzorati, M. Matteucci, D. Migliore, D. Rizzi, D. G. Sorrenti, P. Taddei, Rawseeds ground truth collection systems for indoor self-localization and mapping. *Autonomous Robots*, (2009) 27: 353-371.

[9] Eustice, R. M., Singh, H., and Leonard, J. 2005. Exactly sparse delayed-state filters. In *Proc. IEEE International Conference on Robotics and Automation*, pp. 2428-2435.

[10] Giorgio Grisetti, Cyrill Stachniss, and Wolfram Burgard: Non-linear constraint network optimization for efficient map learning, *IEEE Transactions on Intelligent Transportation Systems,* Volume 10, Issue 3, Pages 428-439, 2009

[11] Grisetti Giorgio, Slawomir Grzonka, Cyrill Stachniss, Patrick Pfaff, and Wolfram Burgard: Efficient Estimation of Accurate Maximum Likelihood Maps in 3D., *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS),* 2007

[12] R. Kummerle, B. Steder, C. Dornhege, M. Ruhnke, G. Grisetti, C. Stachniss, A. Kleiner, On measuring the accuracy of SLAM algorithms, *Autonomous Robots,* (2009) 27: 387-407.

[13] E. Olson, J. Leonard, and S. Teller. Fast iterative optimization of pose graphs with poor initial estimates, *In Proc. of the IEEE Int Conf on Robotics and Automation (ICRA)*, pages 2262 -2269, 2006

[14] Chen, L., Arambel, P. O., and Mehra, R. K. 2002. Estimation under unknown correlation: covariance intersection revisited. *IEEE Transactions on Automatic Control* 47(11):1879-1882.

[15] Julier, S. J., and Uhlmann, J. K. 2001. Simultaneous localization and map building using split covariance intersection. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1257-1262.