

3D I-SLSJF: A Consistent Sparse Local Submap Joining Algorithm for Building Large-Scale 3D Map

Gibson Hu, Shoudong Huang and Gamini Dissanayake

Abstract—This paper presents an efficient and reliable algorithm for autonomous robots to build large-scale three dimensional maps by combining small local submaps. The algorithm is a generalization of our recent work on two dimensional map joining algorithm — Iterated Sparse Local Submap Joining Filter (I-SLSJF). The 3D local submap joining problem is formulated as a least squares optimization problem and solved by Extended Information Filter (EIF) together with smoothing and iterations. The resulting information matrix is exactly sparse and this makes the algorithm efficient. The smoothing and iteration steps improve the accuracy and consistency of the estimate. The consistency and efficiency of 3D I-SLSJF is demonstrated by comparing the algorithm with some existing algorithms using computer simulations.

I. INTRODUCTION

For an autonomous robot to navigate in an unknown environment, it is necessary to build up the knowledge of the map of the environment and be aware of the location of the robot itself within the map. Simultaneous localization and mapping (SLAM) is the process of building a map of an environment while concurrently generating an estimate for the location of the robot (also called “robot pose”). SLAM has been an active research topic for the past decade due to its numerous applications [1].

Recently a number of research has been focused on three dimensional SLAM problems. For 3D SLAM using laser or vision sensors, some approaches are “trajectory SLAM” where scan/image matching based techniques are used to obtain the relative poses between frames and then nonlinear optimization is applied to optimize the robot trajectory [4][5][20][15]. The problem of this kind of approaches is that the map model and the map uncertainty is not very well presented, and the map is not updated directly using the observation information.

For 3D feature based SLAM, most of the existing algorithms are using Extended Kalman Filter (EKF) or Extended Information Filter (EIF) related techniques [8][19]. The algorithms can be made more efficient by applying the local map strategies [12][18]. In local submap joining, a sequence of small sized local submaps are built by a SLAM algorithm (e.g. EKF SLAM [7] or maximal likelihood (ML) approach¹)

This work is supported in part by the ARC Centre of Excellence programme, funded by the Australian Research Council (ARC) and the New South Wales State Government.

Gibson Hu, Shoudong Huang and Gamini Dissanayake are with Faculty of Engineering and Information Technology, The University of Technology, Sydney, Australia. Email: {gibson.hu, sdhuang, gdissa}@eng.uts.edu.au

¹In this paper, the ML approach means to find the maximal likelihood of all the robot poses and all the observed feature positions using all the information available. It is also called Smoothing and Mapping (SAM) [6].

and then combined into a large-scale global map. However, the consistency of the map estimate using EKF/EIF based approach needs to be carefully analyzed [2][11].

Recently, some efficient graph optimization based approach [6][10] are developed using the sparse graph representations of the SLAM problem. Optimization based approach can provide more consistent estimate as compared with the filter based approach. However, a major issue of these approaches is that the state dimension is very high because all the robot poses are included [6][10][14][17].

Very recently, we proposed the Iterated Sparse Local Submap Joining Filter (I-SLSJF) algorithm [13]. In I-SLSJF, the map joining problem was formulated as a least squares problem and was solved by using multiple iterations at each map fusion step. Two dimensional large-scale simulation and experimental results show that the estimation results from I-SLSJF is more consistent as compared with that of EKF SLAM [7] or Sparse Local Submap Joining Filter (SLSJF) [12].

In this paper, the 2D I-SLSJF algorithm [13] is generalized to 3D to provide an efficient and consistent algorithm for large-scale 3D map building. In order to improve the quality of the global map, we propose to use ML to build the local maps and then use 3D I-SLSJF to join the local maps together.

The paper is organized as follows. The process of 3D local map building is stated in Section II. The idea and steps of the 3D I-SLSJF algorithm is described in Section III. Simulation results are provided in Section IV. Finally, Section V concludes the paper.

II. THE 3D LOCAL MAP BUILDING

Consistent local maps are needed in I-SLSJF such that consistent global map can be obtained. In order to improve the quality of the local maps, the ML approach instead of EKF is used to build the 3D local maps.

A. The observation and odometry information

The observation information describes the relative position of features with respect to the robot pose at which the observation is made. Suppose the robot pose is

$$X_r = (x_r, y_r, z_r, \alpha_r, \beta_r, \gamma_r), \quad (1)$$

where (x_r, y_r, z_r) denotes the 3D robot position and $(\alpha_r, \beta_r, \gamma_r)$ are the ZYX Euler angles [16] describing the orientation of the robot. Let the position of feature i be denoted as

$$X_i = (x_i, y_i, z_i), \quad (2)$$

then the relative position between the feature position X_i and the robot pose X_r is

$$h(X_i, X_r) = Rot(\alpha_r, \beta_r, \gamma_r)^T \left(\begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix} - \begin{pmatrix} x_r \\ y_r \\ z_r \end{pmatrix} \right) \quad (3)$$

where $Rot(\alpha, \beta, \gamma)$ is the rotation matrix.

Suppose the observation value is denoted as Z_i , then the observation equation can be written as

$$Z_i = h(X_i, X_r) + w_i \quad (4)$$

where w_i is the observation noise which is assumed to be Gaussian with zero-mean and covariance matrix P_{Z_i} .

The odometry information describes the relative pose of the two consecutive robot poses. Suppose the two poses are denoted as $X_r^1 = (x_r^1, y_r^1, z_r^1, \alpha_r^1, \beta_r^1, \gamma_r^1)$ and $X_r^2 = (x_r^2, y_r^2, z_r^2, \alpha_r^2, \beta_r^2, \gamma_r^2)$, then the relative pose

$$g(X_r^2, X_r^1) = (\delta x_r, \delta y_r, \delta z_r, \delta \alpha_r, \delta \beta_r, \delta \gamma_r)^T \quad (5)$$

can be obtained as follows. For $\delta x_r, \delta y_r, \delta z_r$,

$$\begin{pmatrix} \delta x_r \\ \delta y_r \\ \delta z_r \end{pmatrix} = Rot(\alpha_r^1, \beta_r^1, \gamma_r^1)^T \left(\begin{pmatrix} x_r^2 \\ y_r^2 \\ z_r^2 \end{pmatrix} - \begin{pmatrix} x_r^1 \\ y_r^1 \\ z_r^1 \end{pmatrix} \right) \quad (6)$$

For $\delta \alpha_r, \delta \beta_r, \delta \gamma_r$, using the relationship between the rotation matrices

$$Rot(\alpha_r^2, \beta_r^2, \gamma_r^2) = Rot(\delta \alpha_r, \delta \beta_r, \delta \gamma_r) Rot(\alpha_r^1, \beta_r^1, \gamma_r^1), \quad (7)$$

one gets (because the rotation matrix is orthogonal)

$$Rot(\delta \alpha_r, \delta \beta_r, \delta \gamma_r) = Rot(\alpha_r^2, \beta_r^2, \gamma_r^2) Rot(\alpha_r^1, \beta_r^1, \gamma_r^1)^T. \quad (8)$$

Now the ZYX Euler angles $\delta \alpha_r, \delta \beta_r, \delta \gamma_r$ can be obtained from the rotation matrix $Rot(\delta \alpha_r, \delta \beta_r, \delta \gamma_r)$ [16].

Suppose the odometry information is denoted as O_1^2 , then the odometry equation can be written as

$$O_1^2 = g(X_r^2, X_r^1) + w_1^2. \quad (9)$$

where w_1^2 is the odometry noise which is assumed to be Gaussian with zero-mean and covariance matrix $P_{O_1^2}$. The function g can be obtained by combining (6) and (8).

B. Build local map using ML

The ML approach finds the maximal likelihood of the robot poses and the observed feature positions using all the odometry and observation information available. And this process is performed after each step when new information arrives. This is arguably the best one can do for estimating the feature positions and the robot poses.

Due to the Gaussian assumption on the observation and odometry noises, the ML estimation problem is equivalent to a least squares formulation [6]. The least squares problem can be formulated using (4) and (9) similar to that in [6].

Since both the number of robot poses and the number of features involved in the local map are small, ML is computationally efficient in building good quality local maps.

C. Marginalize out previous robot poses

The required format of the local maps for 3D I-SLSJF is

$$(\hat{X}^L, I^L) \quad (10)$$

where \hat{X}^L (the superscript ‘L’ stands for the local map) is an estimate of the state vector

$$X^L = (X_r^L, X_1^L, \dots, X_n^L) \quad (11)$$

and I^L is the associated information matrix (the inverse of the covariance matrix). The state vector X^L contains the robot final pose X_r^L and all the local feature positions X_1^L, \dots, X_n^L .

Since all the robot poses are included in the state vector of ML, the previous robot poses need to be marginalized out such that the local map will have the format (10).

Since the information matrix of the whole state vector (denoted as I_{ML}) is available in the ML result. The information matrix corresponding to the state \hat{X}^L in (10) can be computed by the Schur Complement of the corresponding block in I_{ML} .

III. THE 3D I-SLSJF ALGORITHM

The input to the 3D I-SLSJF is a sequence of 3D local maps each with the format (10). It is assumed that the robot starts to build local map $k+1$ as soon as it finishes local map k . Therefore the robot end pose of local map k is the same as the robot start pose of local map $k+1$.

A. State vector of the global map

The coordinate frame of the global map is the same as that of the first local map. The state vector of the global map are created by fusing each local maps together in sequence.

After local maps 1 to k are fused into the global map, the global state vector is denoted as $X^G(k)$ (the superscript ‘G’ stands for the global map) and is given by

$$\begin{aligned} & X^G(k) \\ &= (X_1^G, \dots, X_{n_1}^G, X_{1e}^G, \\ & X_{n_1+1}^G, \dots, X_{n_1+n_2}^G, X_{2e}^G, \\ & \dots, \dots \\ & X_{n_1+\dots+n_{k-1}+1}^G, \dots, X_{n_1+\dots+n_{k-1}+n_k}^G, X_{ke}^G) \end{aligned} \quad (12)$$

where $X_{n_1+1}^G, \dots, X_{n_1+n_2}^G$ are the global positions of those features in local map 2 but not in local map 1. The subscript ‘e’ stands for robots ‘end pose’. When fusing local map $k+1$, only new features $X_{n_1+\dots+n_k+1}^G, \dots, X_{n_1+\dots+n_k+n_{k+1}}^G$ and the final pose $X_{(k+1)e}^G$, from the local map $k+1$, are added to (12) to form the new global state $X^G(k+1)$.

B. Local map fusion as a least squares problem

Suppose local map j is given by (\hat{X}_j^L, I_j^L) and suppose the features involved in local map j are $X_{j1}^G, \dots, X_{jn}^G$ in the global map, then the local map state estimate \hat{X}_j^L can be regarded as an observation of the true relative positions from the robot start pose $X_{(j-1)e}^G$ to the features $X_{j1}^G, \dots, X_{jn}^G$ and the robot end pose X_{je}^G . That is,

$$\hat{X}_j^L = H_j(X^G(k)) + W_j \quad (13)$$

where $H_j(X^G(k))$ is the vector of relative positions given by

$$H_j(X^G) = \begin{pmatrix} g(X_{je}^G, X_{(j-1)e}^G) \\ h(X_{j1}^G, X_{(j-1)e}^G) \\ \vdots \\ h(X_{jn}^G, X_{(j-1)e}^G) \end{pmatrix}$$

where h is given by (4) and g is given by (9), and W_j is the zero-mean Gaussian ‘‘observation noise’’ whose covariance matrix is $P_j^L = (I_j^L)^{-1}$ (when $j = 1$, $X_{(j-1)e}^G = (0, 0, 0, 0, 0, 0)$).

So the problem of fusing local maps 1 to k is to estimate the global state $X^G(k)$ using all the local map information (13) for $j = 1, \dots, k$. This problem can be formulated as a least squares problem:

$$\min_{X^G(k)} \sum_{j=1}^k (\hat{X}_j^L - H_j(X^G(k)))^T I_j^L (\hat{X}_j^L - H_j(X^G(k))). \quad (14)$$

C. Overall structure of 3D I-SLSJF

The details of 3D I-SLSJF is similar to that of 2D I-SLSJF [13]. The main steps of the algorithm are listed below. For the details of each step, please refer to [13] and [12].

The overall structure of the algorithm is presented in Algorithm 1.

Algorithm 1 Overall structure of 3D I-SLSJF

- 1: Set local map 1 as the *global map*
 - 2: For $k = 1 : p - 1$ (p is the total number of local maps), fuse *local map* $k + 1$ into the *global map*
 - 3: End
-

The steps used in fusing local map $k + 1$ into the global map are listed in Algorithm 2.

Algorithm 2 Fuse local map $k + 1$ into global map

- 1: Data association
 - 2: Initialization using EIF
 - 3: Update using EIF
 - 4: Use least squares to do smoothing when necessary
-

D. Data Association

Data association here refers to finding the features in local map $k + 1$ that are already included in the global map and their corresponding indices in the global state vector. It can be performed using the same procedure as that in SLSJF [12] as described in Algorithm 3.

E. Smoothing using least squares

The steps for smoothing using the least squares method are listed in Algorithm 4.

Algorithm 3 Data association between local map $k + 1$ and the global map

Require: *global map* and *local map* $k + 1$

- 1: Determine the set of potentially overlapping local maps
 - 2: Find the set of potentially matched features
 - 3: Recover the covariance submatrix associated with X_{ke}^G and the potentially matched features
 - 4: Use statistical data association approach to find the match
-

Algorithm 4 Smoothing using least squares

- 1: Recompute the information matrix $I(k + 1)$ and the information vector $i(k + 1)$
 - 2: Compute the Cholesky Factorization of $I(k + 1)$
 - 3: Recover the global map state estimate $\hat{X}^G(k + 1)$
 - 4: Repeat the above process until $\hat{X}^G(k + 1)$ converges.
-

F. Efficiency and Consistency of I-SLSJF

Since each local map only involves some ‘‘nearby objects’’ — the features and the robot start/end poses involved in the local map, the information matrix in 3D I-SLSJF is exactly sparse [12][13].

As pointed out in [13], the main reason why I-SLSJF is more consistent as compared with SLSJF is because smoothing and iterations are used and the Jacobians are evaluated at the final estimate. This avoids the scenarios where the Jacobian with respect to the same feature/pose be evaluated at different estimate data, which is one of the major causes of inconsistency for EIF/EKF SLAM algorithms [11]. In fact, the state estimate obtained in I-SLSJF is the optimal solution of the least squares problem (14).

IV. SIMULATION RESULTS

In this section, simulations results are presented to illustrate the consistency and efficiency of the 3D I-SLSJF over EKF SLAM and ML (the simulation data is available online: <http://services.eng.uts.edu.au/~sdhuang/research.htm>).

A. Simulation results using a small data set

This relatively small data set is used to compare the consistency of three different algorithms — 3D I-SLSJF, EKF SLAM and ML.

The $100m \times 100m \times 15m$ simulation environment contains a total of 1125 features distributed uniformly into 5 layers. The robot starts from the center of this environment and moves in a square shaped trajectory at the same time pivoting in the z axis. The trajectory has a total of 100 steps. The robot is able to observe previous features on its path which allows for loop closure, as shown in Fig. 1(a).

We assume the robot can observe all features within a distance of $20m$ in front of it with a viewing angle ± 90 degrees. The observation data is generated by adding Gaussian noise on the relative 3D position between the robot pose and the observed features. Subsequently the odometry data is obtained by adding noise on the relative pose between two

TABLE I

NEES OF THE MAP ESTIMATES FROM DIFFERENT ALGORITHMS USING THE 100 STEPS DATA SETS.

ODOMETRY NOISE: $\text{diag}(0.5, 0.5, 0.5, 0.1, 0.1, 0.1)$;

OBSERVATION NOISE: $\text{diag}(0.01, 0.01, 0.01)$

simulation run	ML	3D I-SLSJF	EKF	95% χ^2 gate
1	57.95	108.41	52567	277.14
2	54.03	138.74	341320	277.14
3	51.89	157.94	42100	277.14
4	55.12	231.84	32379	277.14
5	95.72	180.93	1097500	277.14
6	96.63	1009.58	479490	277.14
7	129.92	362.84	102980	277.14
8	111.63	571.53	412390	277.14
9	68.46	323.47	81293	277.14
10	77.84	189.63	43154	277.14

consecutive robot poses. The total number of observations made is 672 and the total number of features observed is 80.

The odometry and the observation data are used to build the map by three different algorithms: 3D I-SLSJF, EKF and ML. 3D I-SLSJF involves dividing the data up into 20 local maps and then fuse them using the 3D I-SLSJF algorithm.

Ten simulation data sets were generated each with the same parameters but different random seeds for the noises (Monte Carlo runs). The results from one of the data sets are shown in Fig. 1(b) to 1(d). Fig. 1(b) shows the 3D map obtained by EKF SLAM algorithm where the estimated feature positions (red circles) are significantly different from the true feature positions (blue crosses). Fig. 1(c) shows the map obtained by 3D I-SLSJF while Fig. 1(d) shows the map obtained by ML. The estimated feature positions of these two algorithms are very accurate.

To compare the consistency of the different mapping algorithms, the normalised estimation error squared (NEES) [3] of the map estimates from different algorithms are computed. The formula is

$$NEES = (\hat{x} - x_{true})^T I (\hat{x} - x_{true}) \quad (15)$$

where \hat{x} is the estimate of the map (the observed feature positions) and x_{true} is the corresponding true feature positions. I is the information matrix (the inverse of the covariance matrix) obtained from the algorithm.

Table I shows the NEES results of the different algorithms for the 10 runs and the 95% χ^2 (chi-square) gate. It is clear that the EKF result can not be accepted to be consistent while the results of 3D I-SLSJF and ML are both acceptable.

B. Simulations results using a large data set

The same simulation environment is used with $100m \times 100m \times 15m$ in size and 1125 features. This time, the robot starts from the center of the cuboid, first closing a smaller loop then a larger one shown in Fig. 2(a). The robot moves a total of 880 steps to complete its trajectory.

The odometry and observation data are generated in the same way as the 100 step data set. The total number of observation made is 5555 and the total number of observed features is 570.

Fig. 2(b) shows the 3D map obtained by ML. For 3D I-SLSJF, 44 small sized local maps are built by ML using the odometry and measurement information. Fig. 2(c) shows XY top view generated by fusing the 44 local maps using 3D I-SLSJF. It can be seen that the feature position estimates computed by 3D I-SLSJF methods are very close to the true feature positions. Fig. 2(d) shows the sparse information matrix from the 3D I-SLSJF result. There are 192866 non-zero elements and $1974^2 - 192866 = 3703810$ exactly zero elements. The sparseness of the information matrix makes the 3D I-SLSJF efficient.

The XY views of the map obtained by ML and the map obtained by EKF SLAM are shown in Fig. 2(e) and Fig. 2(f). It is clear that the map obtained by EKF is not accurate.

The number of poses involved in ML is significantly larger than that of 3D I-SLSJF (880 vs 44). Table II compares some key factors of EKF SLAM, ML and 3D I-SLSJF in terms of efficiency and consistency. The table shows that the results of the 3D I-SLSJF appears to be acceptable although more Monte Carlo runs are necessary to have a proper consistency check using average NEES [2].

V. CONCLUSION

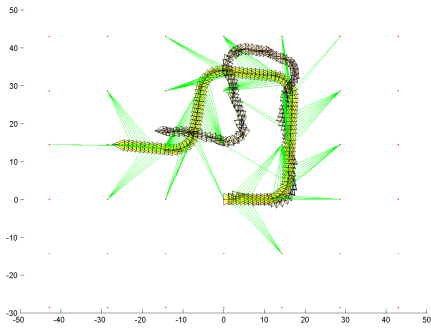
This paper extend the 2D I-SLSJF into 3D I-SLSJF to efficiently build consistent 3D point feature based maps. By treating each local map as an observation and including robot start/end poses in the global state vector, the map joining problem is formulated as a least squares problem. EIF combined with the linearized least squares approach are used in the map fusion step. As compared with filter based mapping algorithm, the consistency of the map is improved. As compared with the optimal ML approach, the computational cost is significantly reduced. Simulation results demonstrated the efficiency and consistency of the proposed map joining algorithm.

A number of techniques can be applied to further reduce the computational cost of the 3D I-SLSJF algorithm. One way is to use the “divide and conquer” idea [18] or tree presentation [9] instead of the sequential map joining currently being used. However, if the sequential map joining is not used, then the data association is a critical issue to be resolved. Another way is to use the graph optimization techniques [10] for the smoothing step of 3D I-SLSJF. In this case, the correlation among the local map features need to be taken into account.

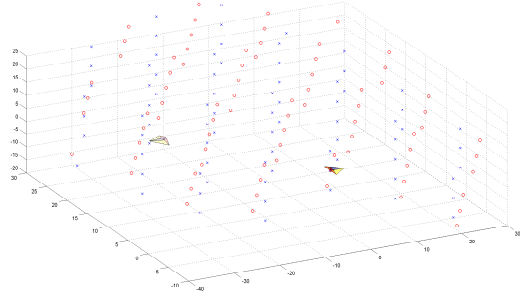
Currently we are in the process of testing the 3D I-SLSJF using large-scale experimental data. In the future, we aim to develop the robot path planning strategies such that it can perform the mapping tasks more effectively and efficiently.

REFERENCES

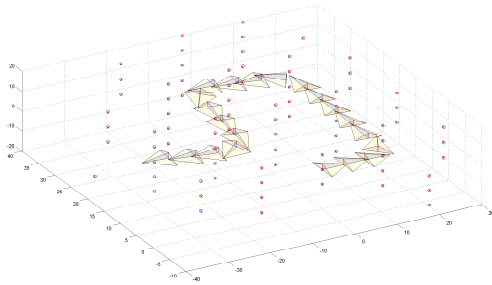
- [1] T. Bailey and H. Durrant-Whyte, “Simultaneous localization and mapping (SLAM): Part II”. *IEEE Robotics & Automation Magazine*, vol. 13, Issue 3, Sept. 2006, pp. 108-117.
- [2] T. Bailey, J. Nieto, J. Guivant, M. Stevens and E. Nebot. “Consistency of the EKF-SLAM algorithm”. In *Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3562-3568, Beijing, China, October 9-15, 2006.



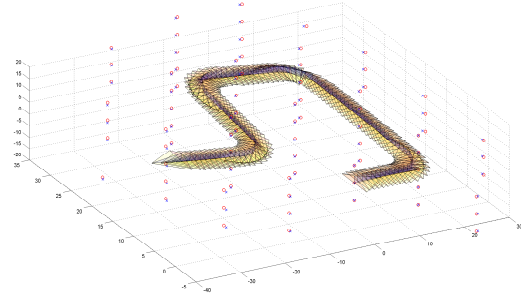
(a) The true robot trajectory (green lines indicate observations to features) and the trajectory computed from odometry information



(b) The 3D map obtained by EKF SLAM



(c) The 3D global map obtained by fusing 20 local maps by 3D I-SLSJF



(d) The map obtained by ML

Fig. 1. Simulation results using the 100 steps data set

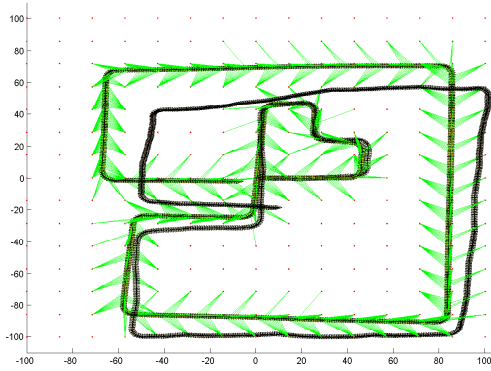
TABLE II

EFFICIENCY AND CONSISTENCY COMPARISON OF EKF SLAM, 3D I-SLSJF AND ML USING THE 880 STEP SIMULATION DATA.

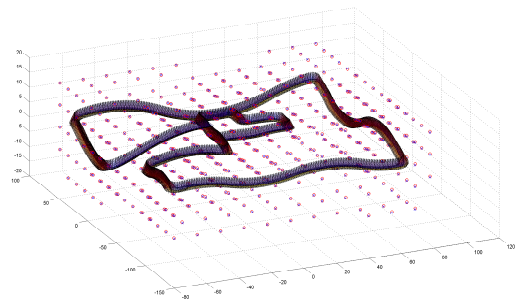
ODOMETRY NOISE: $diag(0.1, 0.1, 0.1, 0.01, 0.01, 0.01)$, OBSERVATION NOISE: $diag(0.01, 0.01, 0.01)$

Algorithm	state dimension	number of poses / features	non-zero elements	computation time	NEES	95% χ^2 gate
EKF SLAM	1716	1 / 570	2944656	2148 seconds	30571	1807
3D I-SLSJF	1974	44 / 570	192866	479 seconds	481	1807
ML	6990	880 / 570	255111	4303 seconds	451	1807

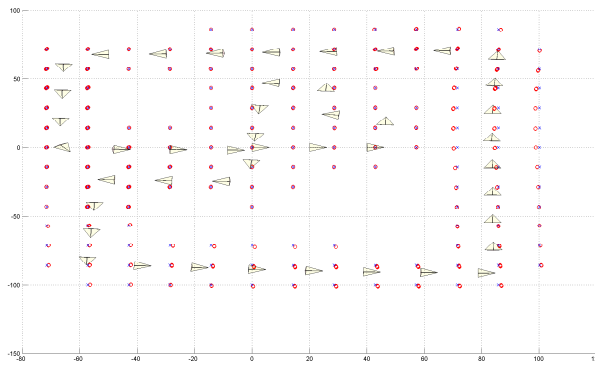
- [3] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation: Theory Algorithms and Software*. John Wiley & Sons. 2001. (Electronic Version)
- [4] D. Borrmann, J. Elseberg, K. Lingemann, A. Nüchter and J. Hertzberg. "Globally consistent 3D mapping with scan matching". *Journal of Robotics and Autonomous Systems (JRAS)*, Vol. 56, Issue 2, pp. 130-142, February 2008.
- [5] D. M. Cole and P. Newman, "Using laser range data for 3D SLAM in outdoor environments". *In Proceedings IEEE International Conference on Robotics and Automation*, Orlando, Florida, USA, 2006, pp. 1556-1563.
- [6] 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.
- [7] G. Dissanayake, P. Newman, S. Clark, H. Durrant-Whyte, and M. Csorba, "A solution to the simultaneous localization and map building (SLAM) problem," *IEEE Trans. on Robotics and Automation*, vol. 17, pp. 229-241, 2001.
- [8] L. P. Ellekilde, S. Huang, J. Valls Miro, G. Dissanayake, "Dense 3D map construction for indoor search and rescue", *Journal of Field Robotics*, 24(1/2), 71-89 (2007)
- [9] U. Frese, "Efficient 6-DOF SLAM with Treemap as a generic back-end," *In Proceedings of 2007 IEEE International Conference on Robotics and Automation (ICRA)*, Rome, Italy, 10-14 April 2007, pp. 1670-1677.
- [10] G. Grisetti, D. L. Rizzini, C. Stachniss, E. Olson and W. Burgard, "On-line constraint network optimization for efficient maximum likelihood mapping". *In Proceedings of 2008 IEEE International Conference on Robotics and Automation (ICRA)*, Pasadena, California, May 19-23, 2008, pp.1880-1885.
- [11] S. Huang and G. Dissanayake, "Convergence and consistency analysis for Extended Kalman Filter based SLAM". *IEEE Transactions on Robotics*, 2007, vol. 23, no. 5, 1036-1049.
- [12] S. Huang, Z. Wang and G. Dissanayake. "Sparse local submap joining filter for building large-scale maps". *IEEE Transactions on Robotics*, 2008, Vol. 24, No. 5, pp.1121-1130, October 2008.
- [13] S. Huang, Z. Wang, G. Dissanayake, and U. Frese, "Iterated SLSJF: A sparse local submap joining algorithm with improved consistency", *2008 Australasian Conference on Robotics and Automation*. Canberra, Australia, December 2008. Available online: <http://www.araa.asn.au/acra/acra2008/papers/pap102s1.pdf>
- [14] M. Kaess, A. Ranganathan, F. Dellaert, "iSAM: Fast incremental Smoothing and Mapping with efficient data association," *In Proceedings of 2007 IEEE International Conference on Robotics and Automation (ICRA)*, Rome, Italy, 10-14 April 2007, pp. 1670-1677.



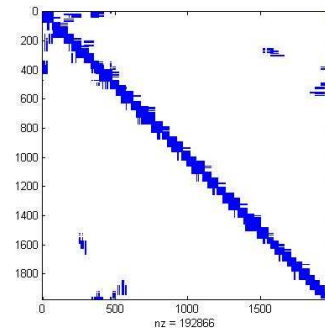
(a) The true robot trajectory (green lines indicate observations to features) and the trajectory computed from odometry information, XY-view



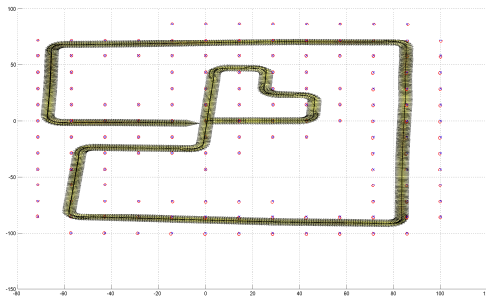
(b) The 3D ML results



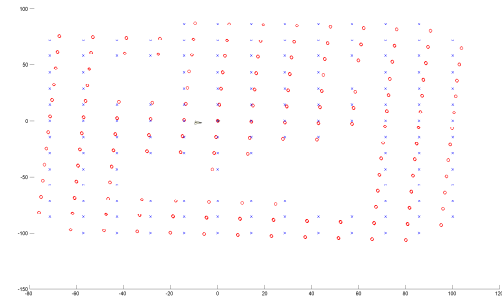
(c) XY-view of the 3D global map obtained by 3D I-SLSJF



(d) The sparse information matrix obtained by 3D I-SLSJF (192866 non-zero elements and 3703810 zero elements)



(e) XY-view of the 3D map obtained by ML, the true feature position are overlapped with the estimated feature position



(f) The map obtained by EKF, XY-view

Fig. 2. Simulation results using the 880 steps data set

- [15] K. Konolige, M. Agrawal, "FrameSLAM: From bundle adjustment to real-time visual mapping". *IEEE Transactions on Robotics*, 24(5): 1066-1077, 2008.
- [16] J. B. Kuipers, *Quaternions and Rotation Sequences: A Primer with Applications to Orbits, Aerospace, and Virtual Reality*. Princeton University Press, Princeton, New Jersey, 1998.
- [17] K. Ni, D. Steedly, and F. Dellaert, "Tectonic SAM: Exact, out-of-core, submap-based SLAM," *In Proceedings of 2007 IEEE International Conference on Robotics and Automation (ICRA)*, Rome, Italy, 10-14 April 2007, pp. 1678-1685.
- [18] L. M. Paz, P. Pinis, J. D. Tardos and J. Neira, "Large Scale 6DOF SLAM with a stereo camera in hand", *IEEE Transactions on Robotics*, Volume 24, No. 5, pp.946-957, October 2008.
- [19] O. Stasse, A. J. Davison, R. Sellaouti and K. Yokoi, "Real-time 3D SLAM for Humanoid Robot considering Pattern Generator Information", *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 348-355. Beijing, China, October 9-15, 2006.
- [20] O. Wulf, A. Nüchter, J. Hertzberg, and B. Wagner. "Benchmarking urban six-Degree-of-Freedom simultaneous localization and mapping". *Journal of Field Robotics*, Vol. 25, Issue 3, pp. 148-163, March, 2008.