To appear in *Advanced Robotics*
Vol. 00, No. 00, January 2013, 1–22

FULL PAPER

A Pose Pruning Driven Solution to Pose Feature GraphSLAM

Y. Wang[a], R. Xiong[a*] and S. Huang[b]

[a]*Zhejiang University, Hangzhou, P. R. China*; [b]*University of Technology, Sydney, Australia*

(*Received 00 Month 201X; accepted 00 Month 201X*)

To build consistent feature based map for the environment, GraphSLAM forms the graph using the collected information, with poses of robot and features being nodes while the odometry and observations being binary edges (edge links to two nodes). As the number of kept nodes grows unboundedly while robot moves, this method will become intractable for long duration operation. In this paper, we propose a pose pruning driven solution for pose feature SLAM by relating the size of graph to the size of map instead of the length of trajectory. It consists of two steps: (1) An online pose pruning algorithm that can select a pose to be pruned based on the contribution of the pose. Different from conventional methods considering the spatial distance between poses, the contribution is based on the feature observations of poses, taking mapping into consideration. (2) An edge generation algorithm that can build new consistent binary edges from n-nary edge (edge links to n nodes) induced by marginalizing the pruned pose. The type of new edges remains invariant (i.e. they are either odometry or pose to feature observations), so no extra change is required to be made on the GraphSLAM optimizer, making the proposed solution modular. In the experiment, we first employ this system on simulation datasets to show how it works. Then the large scale datasets: DLR, Victoria Park and CityTrees10000 are used to evaluate its performance.

Keywords: pose feature SLAM, pose pruning, Kullback-Leibler divergence, edge generation, consistency

1. Introduction

Maps are needed when the robot executes various high level tasks including navigation, manipulation and rescue. Simultaneous localization and mapping (SLAM) is a technique that can generate a map of an unknown environment from the information collected through the onboard sensors. A popular method, called GraphSLAM, is to represent the information as a graph model, which is followed by a GraphSLAM optimizer to output a global consistent map. In this paper, we talk about pose feature SLAM, in which the poses and features are graph nodes, while the odometries and observations are binary edges.

The advantage of GraphSLAM is the usage of its sparse graph structure, but the size of its graph is related to the length of the trajectory. It means that GraphSLAM cannot be applied to the situation where the robot operates for long time even in a small environment. The filtering based SLAM is efficient in such situation since it only keeps current pose and the map, which relates the complexity to the size of the environment. But filter based SLAM breaks the sparse structure and may produce inconsistent estimate [1]. So a desired solution is to develop GraphSLAM with size of its graph being related to the size of the map.

An example is shown in Fig. 1. This is the result of the proposed solution using the DLR dataset. Note that some repeated trajectory is removed and the densely consecutive poses are

[*]Corresponding author. Email: rxiong@iipc.zju.edu.cn

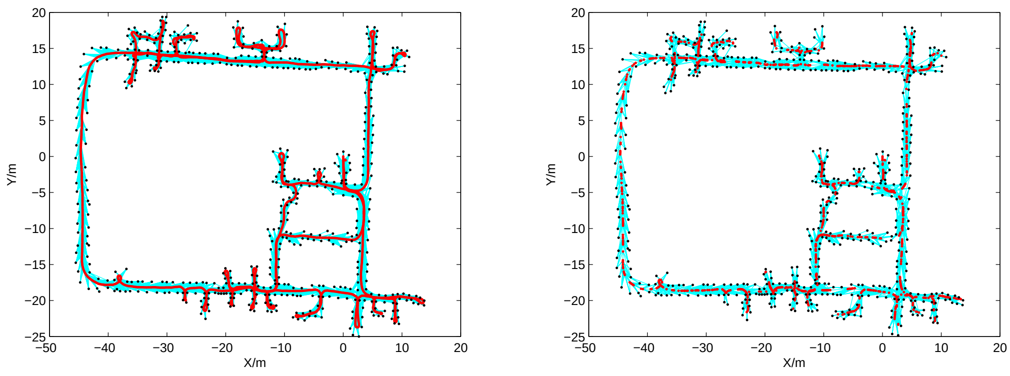
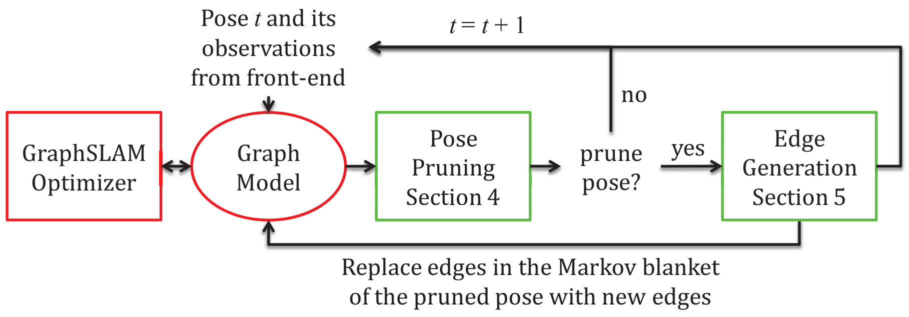


Figure 1. The SLAM result on DLR dataset using Full SLAM (left) and proposed solution (right). The number of poses is reduced from 3299 to 1076 and the number of edges is reduced from 17482 to 7197. The average difference of feature position is 0.3m. The time for whole graph optimization is reduced from 149.5s to 16.1s (in Matlab on PC with i7 CPU and 2G RAM).



Pose t and its observations from front-end
$t = t + 1$
no
GraphSLAM Optimizer
Graph Model
Pose Pruning Section 4
prune pose?
yes
Edge Generation Section 5
Replace edges in the Markov blanket of the pruned pose with new edges

Figure 2. The schematic of the proposed two-step solution and other modules. The modules in green blocks are algorithms proposed in this paper added to the SLAM system. The modules in red blocks are the original algorithm in the SLAM system.

pruned, decreasing the redundancy. The poses now are adaptively distributed according to the features, which is the result of relating the graph size to the size of the map. Besides, the graph size and optimization time are both reduced significantly with a slight difference introduced in the final estimation results. From the perspective of frame to frame localization or loop closure, processing can be more efficient as the graph is compact.

In the solution, to relate the size of graph to that of the map, we proposed a pose pruning algorithm that prunes poses based on a measure derived from the observations. After pruning, the marginalization of this pose breaks the sparse structure. This issue is solved by employing edge generation algorithm that can generate sparsely linked binary edges from a n-nary edge. The architecture of the solution is shown in Fig. 2. The two key components are presented as follows.

- **Pose pruning**: This online algorithm can select poses to be pruned from the graph based on its contribution, which is derived from the Kullbach-Leibler divergence (KLD) between the map generated using all poses and that generated using all poses except this pose. From this perspective, the feature observation is taken into consideration. This is the difference from existing methods considering spatial distance between poses. Intuitively, an observation on a common feature which has been observed by many poses makes limited innovation, leading to the redundancy of the graph. The idea is to keep the poses that observe more uncommon features. As a result, the size of the graph can be related to the size of the mapping area, because more poses in this area observes only common features and hence are pruned.
- **Edge generation**: This algorithm can generate a sparse set of new binary edges compati-

ble to the GraphSLAM optimizer. When a pose is pruned, a new n-nary edge (n is the size of the Markov blanket of the pruned pose) is generated. This edge is dense. Conventional ways sparsify it by approximating it using a new n-nary edge with constrained sparse structure [2] or a set of new type binary edges (may neither be observation nor odometry) [3]. These edges are no longer compatible to the original GraphSLAM optimizer. Using these methods requires extra replacement with their corresponding GraphSLAM optimizer. In this paper, the new binary edges are either observations or odometries. This compatibility is desired when the GraphSLAM optimizer is installed already. Besides, the new edges can still be re-linearized.

This paper extends [4] to a completed solution for pose feature GraphSLAM, which leads to a substantially improved accuracy in the final result by incorporating more information in the graph edges as shown in the experiments. In addition, more comprehensive experiments as well as comparison with other solutions are performed to test the performance of the proposed solution.

The remainder of the paper is organized as follows: In Section 2, the related works on pose pruning and edge generation are reviewed and discussed. In Section 3, a brief introduction to GraphSLAM as well as some notations in this paper are given. The two components of the proposed solution, pose pruning algorithm and edge generation algorithm, are introduced in Section 4 and 5. In Section 6, the experimental results using simulation and real world datasets are presented. The conclusion is given in Section 7.

2. Related works

The early solution to SLAM was based on extended Kalman filter [5], which keeps the current pose and all features in the state of filter. Its dense covariance matrix makes it inefficient when the dimension of the state becomes higher. By exploiting the approximately sparse structure of the inverse of covariance matrix, information matrix, the sparse extended information filter (SEIF) [6], was proposed to solve SLAM more efficiently by approximating the information matrix by a sparse matrix. However, this method brings extra inconsistency by ignoring near zero elements in information matrix. In [7], this cause was avoided by the proposed exactly sparse extended information filter (ESEIF), but there is still the inconsistency issue caused by varying the linearization points as EKF has [8], which cannot be ignored, especially for robot operating for long duration.

FastSLAM algorithm [9] is another solution to stochastic SLAM based on a Rao-Blackwellised particle filter (PF). It uses a set of samples, in which each sample records a robot trajectory, to approximate the posterior distribution in SLAM, instead of the single modal Gaussian distribution used in EKF/EIF. The performance of FastSLAM highly depends on the number of samples, which needs to be very large (exponential to the dimension) in order to accurately express the posterior distribution. As shown in [10], FastSLAM in its current form cannot produce consistent estimates in the long-term.

To deal with these difficulties, further efforts were made on finding more efficient and consistent solution based on Gaussian posterior distribution. GraphSLAM, which has all poses and features in its state to form a very sparse information matrix, is found to be able to be solved very efficiently by modern solvers. Besides, re-linearization in GraphSLAM leads to a consistent solution to SLAM [11–14].

When it comes to GraphSLAM for large scale problem, the large number of poses becomes an issue and the key is how to reduce the number of poses while keeping the sparsity. Sparse local submap joining filter (SLSJF) [15] and iterated D-SLAM map joining [16], applied the map joining idea to reduce the number of poses while keep the global information matrix sparse. The sparsity of the information matrix depends on the size of the local submaps since the local map information matrix is dense. Similar to submap techniques, reducing the number of poses was achieved by marginalizing poses not spatially far from its previous one in [17]. Another perspective was to make use of the sparsity in an incremental way, only partial state is updated at each step [18, 19]. However, these techniques constantly added new pose into the graph, failing to prevent the graph from ever growing when robot is traveling in the same environment for a long duration.

In [20], a reduced pose graph was proposed which extended the spatial thresholding from consecutive poses to all poses using a grid. This mechanism makes the total number of poses constrained by the mapping area. However, lack of a more accurate measure of the pose disables the previously kept poses to be pruned anymore even when a new pose contains more information, which can be dealt with using our method proposed in this paper, since the measure is derived from observation information.

In [21], a pose is measured by the information gain of adding it into the information matrix. Their method was developed for pose graph based SLAM. Combined with this geometric derived result, visual information was used in [22]. The richness of the texture in the image captured at each pose was evaluated to determine whether the new pose was informative.

In [23], a pose pruning algorithm by measuring the information gain of a laser scan was proposed. The map used in their work was occupancy grid map with each grid having a discrete state. In our paper, the observation is based on features, which is in continuous distribution, making the derivation of the pose contribution quite different.

The main work after marginalization is to capture the information and avoid the fully correlated structure at the same time. The early method was developed on selectively updating of features such as in [7]. More recently, in [24], an optimization method was proposed to generate new n-nary edge that has minimum KLD to replace the original n-nary edge formed by the Markov blanket. The sparsity pattern of the new edge can be designed. This method performed better than filter based approach [7], giving a new insight to deal with the sparsification. In [2], the l_1 cost was added to the cost function in [24], enforcing an automatic sparsity pattern. In these methods, the generated n-nary edges prevent the re-linearization in the context of GraphSLAM.

In [23], to generate binary edges, Chow-Liu tree was applied to approximate the n-nary edge. Each new binary edge was computed using pairwise composition, which was also employed in [25]. But it was shown that the composition was unable to capture all the correlations in n-nary edge in [3]. In [3, 26], a new type of binary edge called sparse generic linear constraint (GLC) was proposed. This method was shown to be reasonable according to our theoretic analysis. But their method is not modular, which results in an extra replacement of the GraphSLAM optimizer to support their GLC type. In [27], a modular edge generation algorithm was proposed, but its estimation maybe overconfident because no explicit consistency constraint was added. These two ideas are analyzed and combined in our framework to develop a modular solution which is not overconfident for pose feature SLAM.

A method using the similar idea of pose pruning and edge generation specifically for pose feature SLAM was proposed in [28]. Their method periodically pruned a sequence of poses to reduce the size of the graph. Then only one new edge was generated by marginalizing all poses in the sequence except the first and the last pose from this local graph. The periodical pruning cannot relate the size of the graph to the size of the map. Besides, some features may be discarded during their marginalization, thus missing potential observations by the poses in the future. This method will be compared with ours and discussed in detail in the experiments.

3. Preliminary

Before the presentation of the proposed algorithms, we first give a brief introduction to GraphSLAM. This method represents all information in a graph model. Nodes encode poses $X = \big(\dots x_i \dots\big)$ and features $L = \big(\dots l_i \dots\big)$. Part of edges encode observations $Z = \big(\dots z_{ij} \dots\big)$

with each observation defined as

$$z_{ij} = f_z(l_i, x_j) + w_{ij} \tag{1}$$

where w_{ij} is in Gaussian distribution $N(0, \Sigma_{z_{ij}})$. The covariance of Z can be denoted as Σ_Z, which is in a diagonal form with $\Sigma_{z_{ij}}$ being diagonal blocks. The other edges encode odometries $U = \big(\dots u_{i,i-1} \dots\big)$ as

$$u_{i,i-1} = f_u(x_i, x_{i-1}) + v_{i,i-1} \tag{2}$$

where $v_{i,i-1}$ is in Gaussian distribution $N(0, \Sigma_{u_{i,i-1}})$, leading the covariance of U to be Σ_U with $\Sigma_{u_{i,i-1}}$ being diagonal blocks. f_z and f_u are concatenated as column vectors F_Z and F_U. An optimal solution is obtained by minimizing the following cost function

$$C_{Z,U}(L, X) = \|Z - F_Z(L, X)\|_{\Sigma_Z} + \|U - F_U(X)\|_{\Sigma_U} \tag{3}$$

where $\|e\|_\Sigma = e^T \Sigma^{-1} e$. The optimal solution is the posterior distribution $p(X, L|Z, U)$ in Gaussian distribution $N(\mu, \Omega^{-1})$. The solution can be obtained through nonlinear least squares Graph-SLAM optimizer.

Denote the pruned poses as X_p, the kept poses as X_s, we have

$$\mu = \begin{pmatrix} \mu_{X_p} \\ \mu_{X_s} \\ \mu_L \end{pmatrix}, \quad \Omega = \begin{pmatrix} \Omega_{X_pX_p} & \Omega_{X_pX_s} & \Omega_{X_pL} \\ \Omega_{X_sX_p} & \Omega_{X_sX_s} & \Omega_{X_sL} \\ \Omega_{LX_p} & \Omega_{LX_s} & \Omega_{LL} \end{pmatrix} \tag{4}$$

When poses are pruned, they are marginalized from the Gaussian distribution as

$$\mu_s = \begin{pmatrix} \mu_{X_s} \\ \mu_L \end{pmatrix}, \qquad \Omega_s = A - BC^{-1}B^T, \tag{5}$$

$$A \triangleq \begin{pmatrix} \Omega_{X_sX_s} & \Omega_{X_sL} \\ \Omega_{LX_s} & \Omega_{LL} \end{pmatrix}, \; B \triangleq \begin{pmatrix} \Omega_{X_sX_p} \\ \Omega_{LX_p} \end{pmatrix}, \; C \triangleq \Omega_{X_pX_p} \tag{6}$$

The information matrix is symmetric. This result is a dense n-nary edge, which is generated after pruning a pose, breaking the sparse structure. In this paper, the proposed two-step solution includes

(1) Find a pose X_p to be pruned that can reduce the redundant information saved in the graph.
(2) Find a set of binary edges that approximates the n-nary edge in Ω_s to obtain a sparse structure.

4. Pose pruning

The first component of the solution is a pose pruning algorithm that can measure the contribution of each pose with respect to the map and then prune the pose with low contribution. In the pose feature SLAM, the map is defined as a set of features. The contribution is derived from the KLD between the feature distribution generated using observations Z linking to poses X, and Z_s linking to X_s after pruning poses X_p, where Z_s is the set of observations linking to poses in X_s. The KLD provides a way to measure the contribution of X_p. Based on the theoretic analysis, the contribution of each pose can be computed online.

4.1 *Measure of a map*

The mapping process gives a feature distribution as $p(L|X=\mu_X, Z)$. For convenience, we denote it as $q(L)$. Then the feature distribution estimated using X_s and Z_s is denoted as $q_s(L)$. The KLD between the two map is stated as

$$KL_p = \int q(L)\ln\frac{q(L)}{q_s(L)}dL \tag{7}$$

The pruning problem can be regarded as finding X_s that can minimize the KLD. Expand (7) as

$$KL_p = \int q(L)\ln q(L)dL - \int q(L)\ln q_s(L)dL \tag{8}$$

We ignore the first term as it is a constant number and then denote the second term as Q_s, now we want to select X_s to maximize Q_s

$$Q_s \triangleq \int q(L)\ln q_s(L)dL \tag{9}$$

With conditional independence, we have

$$\begin{aligned} Q_s =& \int \prod_j q(l_j)\ln\prod_i q_s(l_i)dL \\ =& \sum_i \int \prod_j q(l_j)\ln q_s(l_i)dL \\ =& \sum_i \int q(l_i)\ln q_s(l_i)dl_i \end{aligned} \tag{10}$$

One can see that only $\ln q_s(l_i)$ is related to variable X_s, which is the log probability density of the ith feature estimated through optimizing L using Z_s at X_s. It is possible that all observations of a feature is pruned due to pose pruning, in this case the feature is modeled as a uniform prior. Now the complete model of $\ln q_s(l_i)$ is

$$\ln q_s(l_i) = \begin{cases} \ln\gamma_i - \frac{1}{2}\|l_i - \mu_{s,l_i}\|_{\Sigma_{s,l_i}} & l_i \in \Theta \\ m & l_i \in \neg\Theta \end{cases} \tag{11}$$

where μ_{s,l_i} and Σ_{s,l_i} are the mean and covariance of the Gaussian distribution, γ_i is

$$\gamma_i = \frac{1}{(2\pi)^{n/2}|\Sigma_{s,l_i}|^{1/2}} \tag{12}$$

a normalizer of the probability density, Θ is the set of observed features with observations in Z_s, $\neg\Theta$ is the set of features whose observations are not in Z_s ($L = \Theta + \neg\Theta$) and m is a constant negative number indicating the uniform prior.

Then substitute (11) into (10), leading to

$$Q_s = \sum_{l_i\in\Theta}(\ln\gamma_i - \frac{1}{2}E_{q(l_i)}\{\|l_i - \mu_{s,l_i}\|_{\Sigma_{s,l_i}}\}) - |m||\neg\Theta| \tag{13}$$

where $E\{\cdot\}$ is the expectation operator. It can be seen directly that decreasing the size of $\neg\Theta$ increases Q_s, meaning that the more features observed, the better.

As observations are regarded as unbiased in general, and poses are set to be the same value in both $q(l_i)$ and $q_s(l_i)$, the mean value of the two distribution are the same, leading to the conditional expectation in (13) computed as,

$$E_{q(l_i)}\{\|l_i - \mu_{s,l_i}\|_{\Sigma_{s,l_i}}\} = tr\{\Sigma_{s,l_i}^{-1}\Sigma_{l_i}\} \tag{14}$$

where Σ_{l_i} is the covariance of the Gaussian distribution $q(l_i)$. The first term in (13) is computed as

$$\sum_{l_i\in\Theta} \ln\gamma_i = \tilde{c}|\Theta| - \frac{1}{2}\sum_{l_i\in\Theta}\ln|\Sigma_{s,l_i}| \tag{15}$$

where $\tilde{c} = -\ln(2\pi)^{n/2}$.

Denote J_{Z_i,l_i} as the Jacobian $\frac{\partial(Z_i - F_{Z_i})}{\partial l_i}$, where Z_i includes all observations linking to feature l_i. Then we have

$$\Sigma_{s,l_i} = (J_{Z_i,l_i}^T \Sigma_{Z_i}^{-1} J_{Z_i,l_i})^{-1} \tag{16}$$

where Σ_{Z_i} is a diagonal block covariance matrix with each block being the covariance matrix of a observation linking to l_i. In pruning stage, the Σ_{Z_i} is set as identity matrix, indicating that each feature is regarded equally. In pose feature SLAM, an observation function f_z has the Jacobian equalling to R, which is a rotation matrix. As a result, we have

$$\Sigma_{s,l_i} = \frac{1}{O_{s,i}} I \tag{17}$$

$$|\Sigma_{s,l_i}| = \frac{1}{O_{s,i}^n} \tag{18}$$

where $O_{s,i}$ is the number of observations linking to l_i in the set Z_s, n is the dimension of l_i. Now substitute the result into (14) and (15) and then (13), we have

$$\tilde{Q}_s = \tilde{c}|\Theta| + \frac{n}{2}\sum_{l_i\in\Theta}(\ln O_{s,i} - \frac{O_{s,i}}{O_i}) - |m||\neg\Theta| \tag{19}$$

where O_i is the number of observations linking to l_i in the set Z. Note that the difference between $\tilde{Q}_s$ and Q_s is the noise ϵ_s, introduced by setting the covariance matrix being identity, leading to $Q_s = \tilde{Q}_s + \epsilon_s$. Before introducing the pruning algorithm, we make some comments on (19):

- The first and second term have positive effects on $\tilde{Q}_s$ while the third term, negative, indicating that the more features observed, the better selection of X_s. The second term is an increasing function, indicating that the more observation made by X_s, the better selection of X_s.
- The increasing trend of the second term becomes slower when $O_{s,i}$ increases, meaning that observations are encouraged to link to the uncommon features, which are less observed. Inversely, if a feature has been observed for many times, an observation on this feature is less important.
- The setting of identity matrix to Σ_{Z_i} means that we want the structure of the graph to be better, such as a smaller average node degree, which is shown in the experiment. The edge generation algorithms considers the original Σ_{Z_i} to deal with the uncertainty.

- During the analysis, μ_X is used to achieve (14). But its value does not need to be computed. It means that the important thing is whether the information is enough to compute μ_X instead of the value so that poses can be the same in both $q(l_i)$ and $q_s(l_i)$. When poses should be pruned, all poses are known at first, making μ_X computable. So in practice, we can avoid running the GraphSLAM optimizer at each step.

4.2 *Online pruning algorithm*

Recall (8), the first term equals to Q_s when $X_s = X$ as

$$\tilde{Q} = \tilde{c}|\Theta + \neg\Theta| + \frac{n}{2}\sum_{l_i \in \Theta + \neg\Theta}(\ln O_i - 1) \tag{20}$$

leading to

$$KL_p = Q - Q_s = \tilde{Q} - \tilde{Q}_s + \epsilon - \epsilon_s \tag{21}$$

Denote $\Delta Q \triangleq \tilde{Q} - \tilde{Q}_s$, an estimator of KLD which is easier to be computed. Since the selection of poses that minimize ΔQ is an NP-hard problem, a greedy pruning algorithm is used by setting $|X| - |X_s| = 1$, fixing the number of pruned pose at each time be 1.

The features observed by a pruned pose X_p can be classified as

- features that are also observed by other poses in X_s, denoted as Υ, $\Upsilon \subset \Theta$.
- features that are only observed by X_p, not observed by X_s, denoted as Ψ, $\Psi = \neg\Theta$.

Expand ΔQ, we have

$$\Delta Q = \alpha|\Psi| + \frac{n}{2}\sum_{l_i \in \Upsilon}(\ln\frac{O_i + 1}{O_i} - \frac{1}{O_i + 1}) \tag{22}$$

where α is a constant number merging all constant numbers related to $|\Psi|$. When a set of X_s is defined, a pruned pose X_p is also determined. From this perspective, the ΔQ becomes a measure of X_p's contribution, which can be extended to all poses. When a new pose comes, updating of ΔQ occurs in a limited number of poses, which are ones having the observation on the features observed by the new pose, reducing the computation time to constant. The algorithm is shown in Algorithm 1 where ξ is a threshold to contribution of a pruned pose.

In application, α is an algorithm parameter weighting features in Ψ, as loss of such special feature (only observed by one pose) can be either good or bad. If the number of features is low, it is better to tune α high since the features are important for future loop closure. However, when the number of features in a frame is high, such features are less informative.

5. Edge generation

When a pose is pruned, a dense n-nary edge occurs as the result of marginalization as shown in Section 3. The second component of the solution is to approximate this edge using a set of binary edges. In [2], the edge generation starts from the Markov blanket of the pruned pose shown in Fig. 3. In detail, the set of poses and features linking to X_p are regarded as X_{sm} and L_m. All edges only depend on these nodes are denoted as Z_m (observations) and U_m (odometries). Given these information, we can optimize this local graph. Each pose and feature is now expressed in the local coordinates, denoted as X_p^n, X_{sm}^n, L_m^n. Denote $X_m^n = \left(X_p^n, X_{sm}^n\right)$. From (3), the

Algorithm 1: Online Pose Pruning Algorithm

```
Data: New pose $x_t$, New observations $Z_t$, $\alpha$, $\xi$
Result: Pruned pose $X_p$
Set $X_p$ NULL
for each pose $x_i$ observing features in $\Upsilon$ of $x_t$ do
    Update $\Delta Q$ for $x_i$ (22)
end
Get pose $x_{min}$ with minimum $\Delta Q_{min}$
if $\Delta Q_{min} < \xi$ then
    Add $x_{min}$ to $X_p$
    for each pose $x_i$ observing features in $\Upsilon$ of $x_{min}$ do
        Update $\Delta Q$ for $x_i$ (22)
    end
end
```

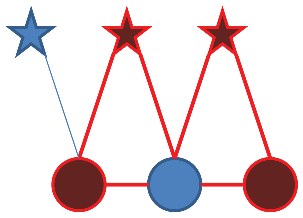


Figure 3. The nodes in the Markov blanket of the intermediate pose are shown in dark red. The edges included iin Z_m and U_m are shown in red.

optimization can then be expressed as

$$\min C_{Z_m,U_m}(L_m^n, X_m^n) \tag{23}$$

Refer to the notations defined in Section 3, the result is an Gaussian distribution $N(\mu_m^n, \Sigma_m^n)$. After marginalizing X_p, we have a dense n-nary edge as $N(\mu_{sm}^n, \Sigma_{sm}^n)$, where $\Sigma_{sm}^n = (\Omega_{sm}^n)^{-1}$. This is the n-nary edge we want to approximate.

Denote the set of new binary edges as $\tilde{Z}_m$ and $\tilde{U}_m$, we have

$$\begin{aligned} \tilde{Z}_m &= F_{\tilde{Z}_m}(L_m^n, X_{sm}^n) + w_m \\ \tilde{U}_m &= F_{\tilde{U}_m}(X_{sm}^n) + v_m \end{aligned} \tag{24}$$

where w_m and v_m are noises. Note that these are new edges, whose configuration of linking can be different from the original ones. But the type are still observations and odometries. The task of edge generation is to find the mean and covariance matrix of w_m and v_m so that minimizing

$$C_{\tilde{Z}_m,\tilde{U}_m}(L_m^n, X_{sm}^n) \tag{25}$$

can give the poses and features in Gaussian distribution $N(\tilde{\mu}_{sm}^n, \tilde{\Sigma}_{sm}^n)$ close to $N(\mu_{sm}^n, \Sigma_{sm}^n)$.

5.1 *Consistency analysis*

As original edges, we want w_m and v_m to have zero mean. This can be done by substitute μ_{sm}^n into (24), getting the values of $\tilde{Z}_m = F_{\tilde{Z}_m}(\mu_{L_m^n}, \mu_{X_{sm}^n})$ and $\tilde{U}_m = F_{\tilde{U}_m}(\mu_{X_{sm}^n})$. Besides, one can see that with these equations, $\tilde{\mu}_{sm}^n$ will be equal to μ_{sm}^n if the set of new edges has equal or higher dimensions than μ_{sm}^n, which is easy to achieve.

Now the KLD between the $N(\tilde{\mu}_{sm}^n, \tilde{\Sigma}_{sm}^n)$ and $N(\mu_{sm}^n, \Sigma_{sm}^n)$ is simplified as

$$KL_m = -\ln|\tilde{\Omega}_{sm}^n| + tr\{\tilde{\Omega}_{sm}^n \Sigma_{sm}^n\} \tag{26}$$

which only relates to the information matrices. Denote

$$\tilde{J}_m = \begin{pmatrix} \frac{\partial(\tilde{Z}_m - F_{\tilde{Z}_m})}{\partial L_m^n} & \frac{\partial(\tilde{Z}_m - F_{\tilde{Z}_m})}{\partial X_m^n} \\ \frac{\partial(\tilde{U}_m - F_{\tilde{U}_m})}{\partial L_m^n} & \frac{\partial(\tilde{U}_m - F_{\tilde{U}_m})}{\partial X_m^n} \end{pmatrix}, \quad \tilde{\Omega}_m = \begin{pmatrix} \Omega_{\tilde{Z}_m} & 0 \\ 0 & \Omega_{\tilde{U}_m} \end{pmatrix} \tag{27}$$

where $\Omega_{\tilde{Z}_m}$ and $\Omega_{\tilde{U}_m}$ are the information matrices of w_m and v_m respectively with diagonal block structure. Then (26) becomes

$$KL_m = -\ln|\tilde{J}_m^T \tilde{\Omega}_m \tilde{J}_m| + tr\{\tilde{J}_m^T \tilde{\Omega}_m \tilde{J}_m \Sigma_{sm}^n\} \tag{28}$$

Applying the cyclic permutation to the second trace term, we have

$$KL_m = -\ln|\tilde{J}_m^T \tilde{\Omega}_m \tilde{J}_m| + tr\{\tilde{\Omega}_m \tilde{J}_m \Sigma_{sm}^n \tilde{J}_m^T\} \tag{29}$$

As $\tilde{J}_m \Sigma_{sm}^n \tilde{J}_m^T$ is symmetric, leading to

$$KL_m = -\ln|\tilde{J}_m^T \tilde{\Omega}_m \tilde{J}_m| + tr\{\tilde{\Omega}_m [\tilde{J}_m \Sigma_{sm}^n \tilde{J}_m^T]_{bd}\} \tag{30}$$

where $[\cdot]_{bd}$ is to extract the diagonal block with the dimensions the same as the diagonal block structure of $\Omega_{\tilde{Z}_m}$. If $\tilde{J}_m$ is square, we set the gradient to be zero as

$$-\tilde{\Omega}_m^{-1} + [\tilde{J}_m \Sigma_{sm}^n \tilde{J}_m^T]_{bd} = 0 \tag{31}$$

giving us a closed form optimal solution

$$\tilde{\Omega}_m^{-1} = [\tilde{J}_m \Sigma_{sm}^n \tilde{J}_m^T]_{bd} = \tilde{\Sigma}_m \tag{32}$$

Note that the solution is the error propagation of (24), transforming the covariance to observations and odometries, whose correlations are ignored.

The consistency constraint requires

$$\tilde{J}_m^T (\tilde{J}_m \Sigma_{sm}^n \tilde{J}_m^T)^{-1} \tilde{J}_m - \tilde{J}_m^T \tilde{\Omega}_m \tilde{J}_m \geq 0 \tag{33}$$

Note that this constraint exist also for non-square form $\tilde{J}_m$. But now we first talk about square form. As $\tilde{J}$ is non-singular, (33) is equivalent to

$$\tilde{\Sigma}_m - \tilde{J}_m \Sigma_{sm}^n \tilde{J}_m^T \geq 0 \tag{34}$$

Denote $\tilde{\Sigma}_{m\Gamma}$ as the non diagonal block entries of $\tilde{J}_m \Sigma_{sm}^n \tilde{J}_m^T$, it has

$$tr\{-\tilde{\Sigma}_{m\Gamma}\} = \sum_i \lambda_i = 0 \tag{35}$$

where λ_i is the ith eigenvalue. The only possibility guaranteeing (34), i.e. $-\tilde{\Sigma}_{m\Gamma} \geq 0$ is that $\tilde{\Sigma}_{m\Gamma}$ is zero matrix, leading to

$$\tilde{J}_m \Sigma_{sm}^n \tilde{J}_m^T = [\tilde{J}_m \Sigma_{sm}^n \tilde{J}_m^T]_{bd} \tag{36}$$

which, however, is impossible as Σ^n_{sm} is dense. So the closed form estimation is overconfident.

Extend the solution (32) to the case of $\tilde{J}_m$ in any form, we have

$$rank(\tilde{\Sigma}_m) \geq rank(\Sigma^n_{sm}) = rank(\tilde{J}_m \Sigma^n_{sm} \tilde{J}^T_m) \tag{37}$$

Partition $\tilde{J}_m$ as

$$\tilde{J}_m = \begin{pmatrix} \tilde{J}_{m1} \\ \tilde{J}_{m2} \end{pmatrix} \tag{38}$$

where $\tilde{J}_{m1}$ is square. Call the observations generated by $\tilde{J}_{m1}$ as full observations and the observations generated by $\tilde{J}_{m2}$ as excessive observations. The excessive observations can be predicted by full observations. If we extract the diagonal block entries to form $\tilde{\Sigma}_m$, excess observations become independent which are actually not, making them the innovations, which causes serious inconsistency.

Based on the analysis above, the consistency cannot be guaranteed if we generate the edges from Σ^n_{sm}, which also includes other type of edges. For example, GLC [3] can be regarded as such a method by setting its Jacobian to $\tilde{J}_m$.

5.2 *Numerical solution*

To generate consistent edges, a matrix P can be added to $-\tilde{\Sigma}$, making

$$P - \tilde{\Sigma}_{m\Gamma} > 0, \quad P > 0 \tag{39}$$

where P has the same diagonal block structure as $\tilde{\Sigma}_m$. Then we have

$$\tilde{\Sigma}_m + P - \tilde{J}_m \Sigma^n_{sm} \tilde{J}^T_m \geq 0 \tag{40}$$

which means each edge should dilate its covariance matrix. In the other word, the information should be less confident to guarantee the consistency. Denote $\tilde{\Sigma}_m = Q\Lambda Q^T$, the eigen-decomposition. As the inverse of summation of matrices is hard, we change (40) as

$$QQ_pW^{-1}\Lambda Q^T_p Q^T - \tilde{J}_m \Sigma^n_{sm} \tilde{J}^T_m \geq 0, \quad 0 < W < 1 \tag{41}$$

where W is a diagonal matrix, Q_p is an orthogonal matrix with determinant equalling to 1. The degrees of freedom in (40) and (41) are the same. Now the covariance matrix is dilated and rotated. Substitute into (33)

$$\tilde{J}^T_m (\tilde{J}_m \Sigma^n_{sm} \tilde{J}^T_m)^{-1} \tilde{J}_m - DQ_pW\Lambda^{-1}Q^T_p D^T \geq 0, \quad 0 < W < 1 \tag{42}$$

where $D = \tilde{J}^T_m Q$. It means the information of each edge is reduced.

Now the optimization problem can be expressed as minimizing

$$\begin{aligned} KL_{mc} = -\ln |DQ_pW\Lambda^{-1}Q^T_p D^T| + tr\{DQ_pW\Lambda^{-1}Q^T_p D^T \Sigma^n_{sm}\} \\ s.t. \ (42) \ \ holds \end{aligned} \tag{43}$$

It is a MAXDET problem [29]. Searching for D and W has the same degrees of freedom to finding a new $\tilde{\Sigma}_m$ in [27], but with consistency constraint. The number of parameters in this problem is large. We can set $Q_p = I$. Then (43) becomes

$$\begin{aligned} KL_{mcr} = -\ln |DW\Lambda^{-1}D^T| + tr\{DW\Lambda^{-1}D^T\Sigma_{sm}^n\} \\ s.t.\ \ (42)\ \ holds \end{aligned} \tag{44}$$

This MAXDET problem is equivalent to [26] when the edges are sparse GLC types. But here the form $\tilde{J}_m$ is free.

Algorithm 2: Edge Generation Algorithm

```
Data: Pruned pose $X_p$, $ESM$, $CON$
Result: Edge set $\tilde{Z}_m$ and $\tilde{U}_m$ with information matrix $\tilde{\Omega}_m$
Set $X_{sm}$ and $L_m$ to be the Markov blanket of $X_p$
Get $N(\mu_m^n, \Sigma_m^n)$ by optimizing local graph model (23)
Get $N(\mu_{sm}^n, \Sigma_{sm}^n)$ by marginalizing $X_p$ (5)
Get $\tilde{Z}_m$ and $\tilde{U}_m$ by edge selection according to $ESM$
Assign value to $\tilde{Z}_m$ and $\tilde{U}_m$ using $\mu_{sm}^n$
Get $\tilde{\Omega}_m$ using error propagation (32)
if $CON == 1$ then
    Improve $\tilde{\Omega}_m$ by minimizing $KL_{mc}$ (43)
end
if $CON == 2$ then
    Improve $\tilde{\Omega}_m$ by minimizing $KL_{mcr}$ (44)
end
```

5.3 *Edge selection*

To determine the form of $\tilde{J}_m$, the linking configuration should be selected. We introduce three mechanisms as shown in Fig. 4. In pose feature SLAM, the Markov blanket of a pruned pose always contains the previous pose and the next pose, as well as the features observed by the pruned pose. After pruning, feature to feature edges are generated, breaking modularity since the GraphSLAM optimizer can not deal with such kind of edges. The options for the types of new edges can only be odometry and observation.

- **Diagonal**: As shown in (32), when $\tilde{J}_m$ is square, the solution minimizing KL_m has closed form. This solution holds the equality in (37), so its overconfidence is not serious. The square form means we can select one odometry for each pose and one observation for each feature, thus the Jacobian is block diagonal. These edges all link to the root pose.
- **Exact**: At each pose, there are some observations collected in real world. The Diagonal will generate observations for a pose on features actually not observed by this pose. These observations give geometry constraints for the graph, but are not regarded as the contribution of this pose in stage of pose pruning. The Exact ignores these edges and selects observations that the pose actually has in real world.
- **All**: Given the distribution of $N(\mu_{sm}^n, \Sigma_{sm}^n)$, all edges we can select include odometry between two poses and observations between each pose and each feature. All includes all possible odometry and observations which can be generated from the Markov blanket. We again claim that the feature to feature edge is forbidden because of the pose feature SLAM optimizer. Compared to the two mechanisms above, the number of edges using this mechanism will grow faster.

Now we can put mechanism and the further optimization together to generate a new set of edges, which is consistent. If numerical techniques (43) or (44) is not applied, we can get an

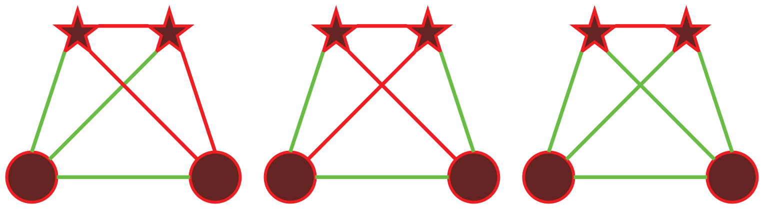


Figure 4. Follow the situation in Fig. 3. The graph model is the Markov blanket in the distribution $N(\mu^n_{sm}, \Sigma^n_{sm})$. The selected binary edges forming the linking configuration of $N(\tilde{\mu}^n_{sm}, \tilde{\Sigma}^n_{sm})$ are shown in green. From left to right the mechanism are Diagonal, Exact and All.

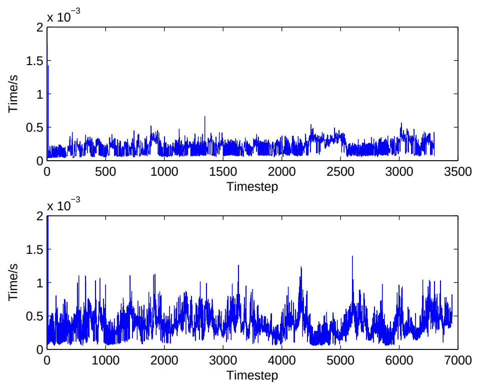


Figure 5. The time for pose pruning algorithm on DLR (top) and Victoria Park (bottom).

overconfident solution as (32). So there are multiple combinations to do edge generation. Denote the label of edge selection mechanism as ESM, the method of MAXDET optimization is label as CON. The whole process of edge generation is shown in Algorithm 2.

6. Experiments

The whole system is shown in Fig. 2. The GraphSLAM optimizer only supports odometries and observations. The optimizer is run periodically. The α and ξ in Algorithm 1 are set as 2 and 0.2 in all experiments. (43) or (44) is optimized using cvx [30]. The dataset used includes two simulation datasets, one generated by ourselves and the other is CityTree10000 [18], as well as two datasets collected from the real world, DLR [31] and Victoria Park [32]. The solution is compared with two systems proposed in [4] and [28], and also the standard Full SLAM system. In addition, different combinations are compared to support the theoretic results in Section 5.

6.1 *Pruning method comparison*

In this experiment, two methods are compared: First, the periodic method that keep a pose at a fixed time interval [28]. Second, the random selection method, which is not used in practice, but gives statistic result with multiple runs. Both two methods are tuned to give number of poses similar to the proposed algorithm. To evaluate the performance, (19) is used to show the contribution of the selected poses. As this is optimized in the pose pruning, two more measures are used: the average node degree of features [33] and the standard deviation of the histogram of the observations with respect to features. The results are in Tab. 1. We can find that our method gives significantly better results as compared to others. It tells that the proposed method shows smallest KLD to the original distribution and more stable graph structure. The smallest standard deviation indicates that the features are observed more uniformly. Besides, the proposed pose

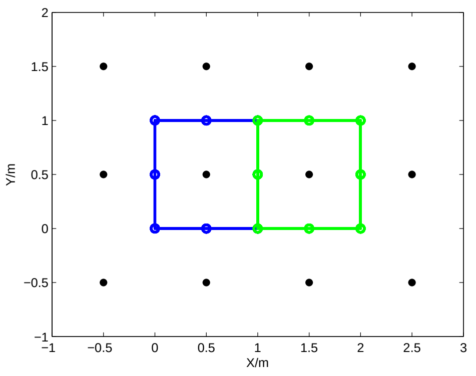


Figure 6. The simulation dataset with 12 features and 200 poses. Blue trajectory is for the first 100 poses and green one is for the second 100 poses. Black dots are for the features.

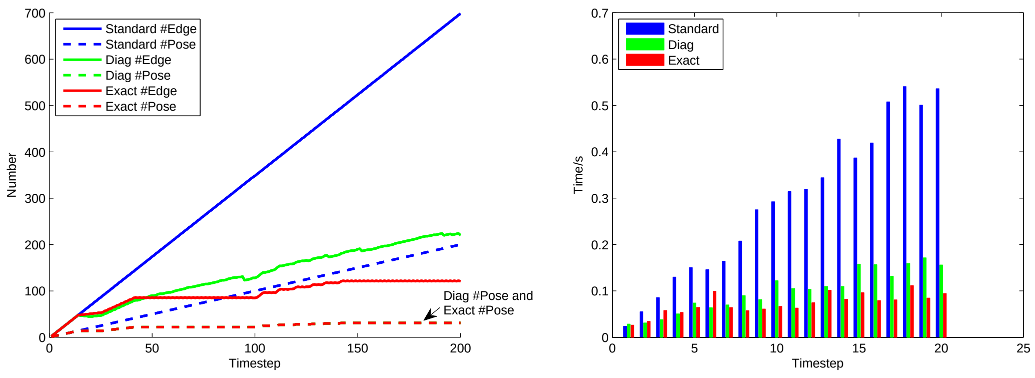


Figure 7. The evolution of number of poses and edges (left) and optimization time (right) on simulation dataset. The curves for the number of poses using Diag and Exact overlap.

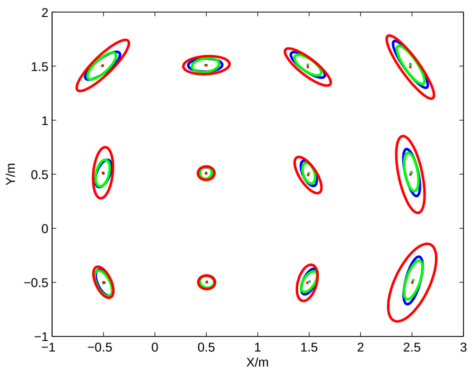
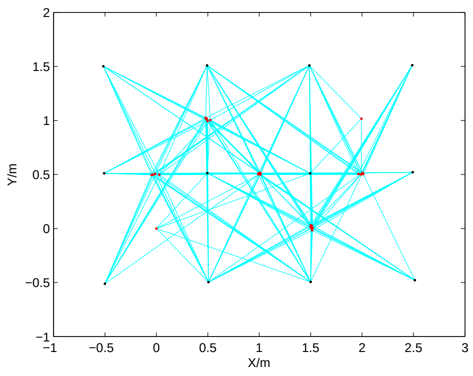


Figure 8. The covariance ellipses of each features with Full SLAM (blue), pruning using overconfident methods (green) and consistent methods (red).

pruning method has constant computational complexity when the mapping area is fixed, while both periodic and random cannot achieve this unless the total number of poses is known at the very beginning. The time of our method is shown in Fig. 5, one can see that it is bounded. It also shows that the measure (19) is related to existing effective measures, but with a probabilistic explanation.



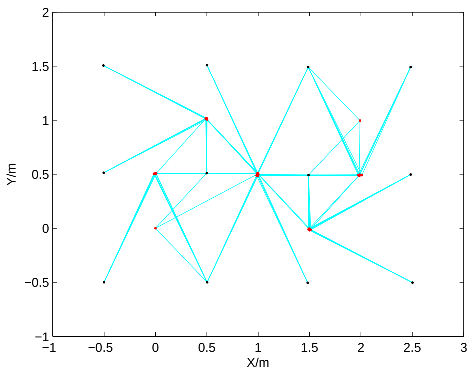


Figure 9. The result on the simulation dataset using Diagonal (left) and Exact (right).

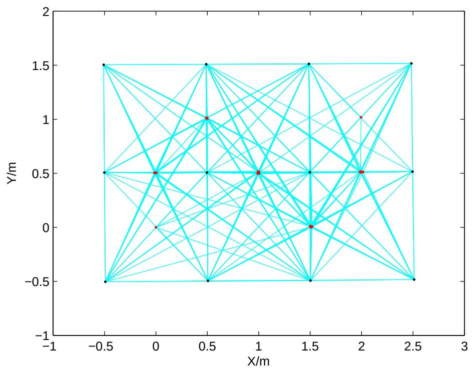
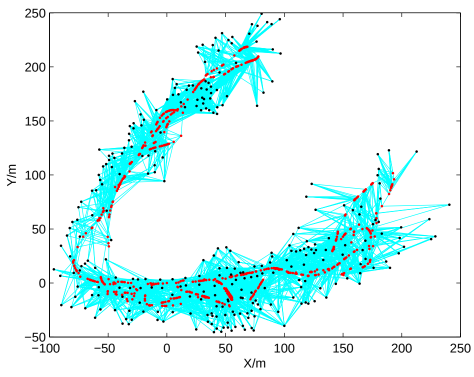


Figure 10. The result on the simulation dataset using Full SLAM with pruned poses being marginalized. The edges are fully linked and new type of edge with feature linking to feature occurs.



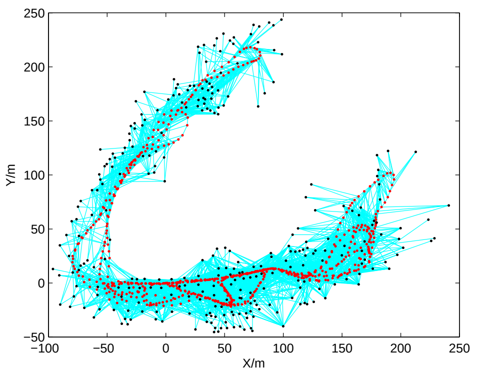


Figure 11. The SLAM result on Victoria Park dataset using Exact (left) and C-KLAM (right).

6.2 *An illustrative example*

We give an illustrative example to show how our solution retains the complexity adaptive to the size of the map in this experiment. The scenario is shown in Fig. 6. It has 12 features. The robot can observes features in the range of 1.2m. It runs in loop for the first 100 steps. At the 101st step, the robot enters a new area and runs in loop for another 100 steps. The period of running GraphSLAM optimizer is 10 steps.

Two combinations are employed: Diagonal and Exact with consistent constraint (44). We can see in Fig. 7 that the number of poses and edges kept in the system grows up first and

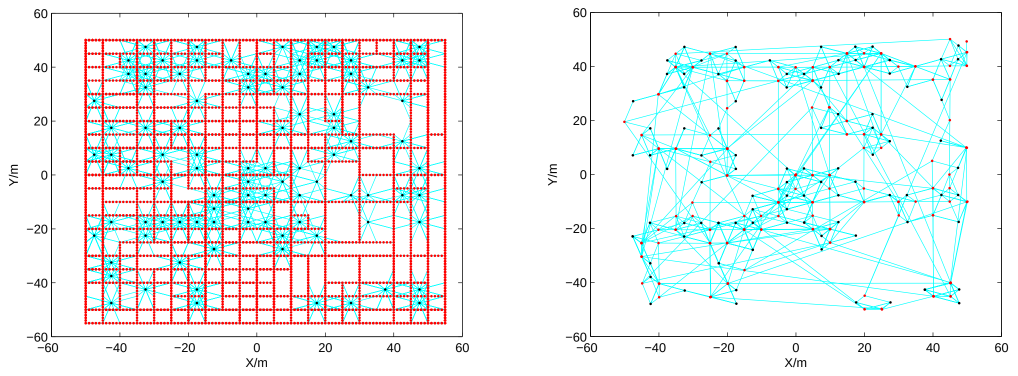


Figure 12. The SLAM result on CityTree10000 dataset using Full SLAM (left) and Diagonal (right).

Table 1. The comparison of pruning algorithms. AD, average node degree, SD, standard deviation, LF, the number of lost feature

Dataset	DLR					Victoria Park				
Measure	Q_s	AD	SD	#Pose	#LF	Q_s	AD	SD	#Pose	#LF
Full	117.3	25.8	15.9	3299	0	412.3	151.8	187.0	899	0
Periodic [28]	-131.6	8.6	5.3	1054	5	-42.6	15.2	19.0	690	10
Random	-142.9	9.07	5.9	1142.4	9.4	-78.9	15.4	19.6	695.8	22.2
Proposed	-18.1	11.2	4.3	1076	0	161.9	18.2	13.4	606	0

keeps constant later using Exact edge selection mechanism. At the 101st step, the number of poses grows again since the robot enters a new area but later becomes constant soon. This result supports that the proposed solution has complexity related to the size of map. Turn to the Diagonal, the constant trend occurs at last but slower. The two mechanisms follow the same pose pruning result, thus their curves overlap. The reason why Exact retains constant complexity sooner is that it only keeps the observations the pose really has. Both methods prevent the unboundedly growing trend in case of the Full GraphSLAM. The same trend also occurs in the evolution of optimization time as shown in Fig. 7. The covariance of each feature is shown in Fig. 8. We dilate the covariance 50 times for visualization. Note that the solution with consistency constraint gives conservative estimation, which is safer for some applications, such as navigation and obstacle avoidance. As shown in the theoretic analysis in Section 5, the closed form (32) is overconfident. In Fig. 9, the edge linking configuration is shown. One can see that a pose can virtually observe features using Diagonal. In Fig. 10, the dense n-nary edge obtained by marginalizing the information matrix of Full SLAM is drawn. Not only the number of edges is much higher as shown in Fig. 7. The features are also connected, lowering the efficiency and breaking the modularity.

6.3 *Solution comparison*

On DLR, Victoria Park and CityTree10000, the period of GraphSLAM optimizer is 200. The mean square error (RMSE) of features is applied as follows.

$$RMSE = \sqrt{\frac{1}{|L|}\sum_i \|l_i - \hat{l}_i\|^2} \tag{45}$$

Table 2. The evaluation of different methods. D., Diagonal, E., Exact, A., All, C., with consistent constraint, 1.,MAXDET optimizing (43), 2.,MAXDET optimizing (44), P, Pose, F, Feature, E, Edge

Dataset	DLR					Victoria Park					CityTree10000				
Summary	#P 3299 #F 549 #E 17482					#P 6899 #F 299 #E 52288					#P 10000 #F 100 #E 14442				
Measure	#P	#F	#E	RMSE	ER	#P	#F	#E	RMSE	ER	#P	#F	#E	RMSE	ER
D.	1076	549	6759	0.16	0.064	606	299	6808	0.52	0.025	228	100	1848	0.25	0.006
D.C.2	1076	549	6759	0.21	0.470	606	299	6808	0.72	0.151	228	100	1848	0.16	0.004
E.C.1	1076	549	7197	0.24	0.193	606	299	6038	0.92	0.101	228	100	1274	0.30	0.004
E.C.2	1076	549	7197	0.34	0.328	606	299	6038	0.71	0.108	228	100	1274	0.30	0.004
A.	1076	549	8264	0.45	1.888	606	299	10429	0.93	0.345	228	100	2962	0.93	0.148
OO [4]	1076	549	7197	1.06	1.069	606	299	6038	1.51	0.193	228	100	1274	1.49	0.024
C-KLAM [28]	1101	544	5815	0.48	0.990	698	289	5318	0.26	0.256	289	98	833	0.87	0.045

Besides, the error ratio (ER) [34] is also employed on features as

$$ER = \frac{\min C_{Z,U}(L_o = \hat{L_o}, L_u, X) - \min C_{Z,U}(L, X)}{|L|} \tag{46}$$

where $|L|$ is the number of features, l_i is the ith feature ground truth/estimation of Full SLAM while $\hat{l}_i$ is its estimation, L is a concatenation of l_i and $\hat{L}$ is a concatenation of $\hat{l}_i$, L_o and L_u is a partition of L, L_o is the set of kept features and L_u is the set of lost features. In DLR and Victoria Park, lack of ground truth leads to a replacement with Full SLAM results, which is also an estimation. But RMSE cannot consider the uncertainty in the estimate. In such situation, ER is a good measure to tell how worse the solution is compared to the Full SLAM result. For C-KLAM, L_u is not empty due to the lost of features.

There are two parameters ESM and CON in Algorithm 2. Different combinations are used, including Diagonal without/with consistency constraint optimizing (44) (D. & D.C.2), Exact with consistent constraint optimizing (43)/(44) (E.C.1 & E.C.2) and All (A.). The results are shown in Tab. 2. The DLR result using E.C.2 is shown in Fig. 1. The Victoria Park result using E.C.2 and C-KLAM is shown in Fig. 11. The CityTree10000 using D. is shown in Fig. 12.

The consistency is measured by the minimum eigenvalue of $\tilde{J}_m^T(\tilde{J}_m\Sigma_{sm}^n\tilde{J}_m^T)^{-1}\tilde{J}_m - \tilde{J}_m^T\tilde{\Omega}_m\tilde{J}_m$. If it is positive, the estimate is regarded consistent. If it is negative, the estimate is overconfident. The absolute value of the minimum eigenvalue can reflect to some extent how overconfident the estimate is. If the overconfidence is significant, the solution is inconsistent. The evolution of the minimum eigenvalue is shown in Fig. 13. Only a segment is shown for clear visualization. The overconfidence in estimation using A. is much serious than that using D.I.. The estimate are always consistent by applying consistency constraint.

Overall speaking, Diagonal gives the small RMSE and ER as it optimizes KLD without any other constraints (26). A. gives the worst result, as the overconfidence is very serious and it is not the optimal solution to unconstrained KLD (26) when $\tilde{J}_m$ is non-square. Diagonal and Exact with enforcing consistency give similar performance slightly worse than Diagonal, but consistent (conservative). OO and C-KLAM give worse results since they both lose information. The details are discussed as follows,

- The advantage of C-KLAM is that it has better linearized points since it generates new edge from a sequence. The shortcoming is that the lost features may be observed in the future which can be informative, leading to serious information loss. The proposed solutions generate new edges by computing the linearized point in local and can be re-linearized furthermore, thus the advantage of C-KLAM over our methods is not obvious. The proposed solutions give better RMSE as well as ER for the datasets tested.
- E.C.1/2 and OO have the same linking configuration. However, OO reserves information only from its linked poses, i.e. odometries, leading to the serious loss of information since the observations are discarded. Besides, regarding Markov blanket as the local graph gives better local linearized points. This is why the proposed solution gives substantial better

Table 3. The non-zero elements ratio in the information matrix

Dataset	DLR	Victoria Park	CityTree10000
Full	0.76%	3.45%	7.54%
Diag	0.53%	1.58%	3.22%
Exact	0.56%	1.42%	2.33%
All	0.63%	2.32%	4.95%

results than OO.

- The consistency constrained estimation on DLR and Victoria Park gives slightly worse than unconstrained estimation but better result in CityTree10000. In first two datasets, the noise is relatively small, so the local linearized point is good, some overconfidence is suppressed by the advantage of unconstraint. But in CityTree10000, the bigger noise leads to poorer linearized points. Then the overconfident of poorer linearized point has predominant effect on the result as shown in [1].
- Though E.C.1/2 has a risk of information loss, it is controlled by the pose pruning algorithm. This is the reason why D.C.2 and E.C.1/2 give similar results. The number of edges using E.C.1/2 is smaller in most cases. Note Fig. 7, they can lead to a better convergence of the complexity since it does not generate observations which is actually absent. So E.C.1/2 gives clearer and more understandable result.
- Theoretically speaking, E.C.1 should give better result than E.C.2. But it is not the case in experiments. This reflects that more degrees of freedom have both bad and good effects. It becomes harder for the MAXDET optimizer to find the optimal solution when there are $O(n^2)$ parameters in (43) while $O(n)$ in (44). But if n is small, more degrees of freedom can optimize the cost function better, which is the case in DLR.

As a summary, if information loss is serious, no method can be applied to improve the result. If there is no information loss or the information loss is slight, the effect of poor local linearized points can be decreased by using the consistency constraint (43) or (44). If the local linearized points are good, such overconfidence can be ignored. If the overconfidence is led by the rank mismatch, the result is inconsistent as shown in [35]. Using the proposed pruning method, the information loss can be controlled and informative observations are kept automatically. Based on these analysis, D. and E.C.2 are good choices.

The evolution of the number of poses and edges are shown in Fig. 14, Fig. 15 and Fig. 16. Our solutions with the same ESM have the same evolution, so the ESM of the solutions are used as indicators in the figures. Since the same pose pruning algorithm is used, the curves for the number of poses overlap together except C-KLAM. The curves for the number of edges using Exact and OO overlap each other as both of them generate observations according to that the robot originally has. In all datasets, the proposed solution gives much slower growing trend than Full SLAM. Especially for Exact, the trend becomes constant soon. In the original datasets, CityTree10000 has the largest number of poses, followed by Victoria Park and then DLR. But in the final result after pruning, the order is inverse as shown in Tab. 2. This is in accordance to the order of numbers of features in three datasets. It gives evidence that the complexity of the proposed solution is related to the map instead of the length of the trajectory. For C-KLAM, as shown before, its number of poses/edges keeps growing because it is not adaptive to the map.

To see how the dense information matrix is sparsified from the perspective of non-zeros elements, the information matrix using proposed solution are compared with the information matrix marginalized from the full information matrix using Full SLAM. The non-zero elements ratio are shown in Tab. 3. These results show our solution can achieve good estimation, while also reduces the fill-in at the same time. C-KLAM has different size of information matrix as it loses some features, so its ratio is not presented.

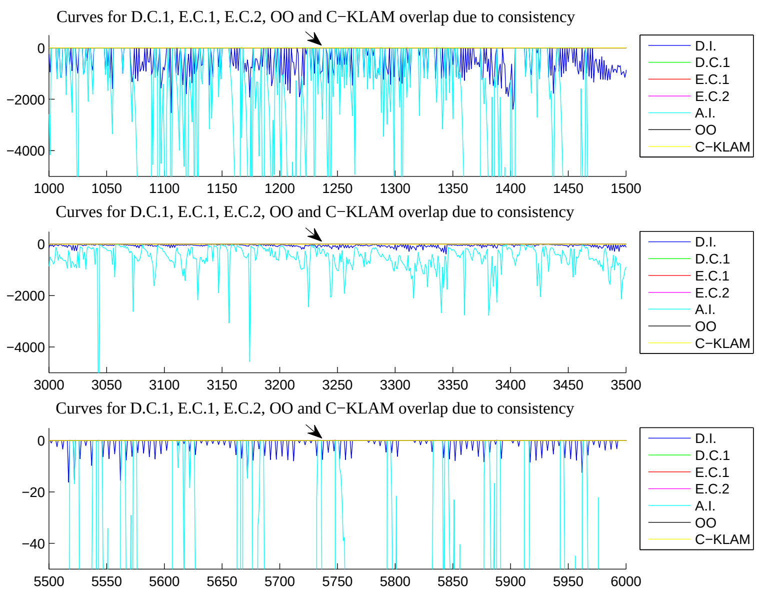


Figure 13. The evolution of minimum eigenvalue in a segment of time on DLR (top), Victoria Park (middle) and CityTree10000 (bottom).

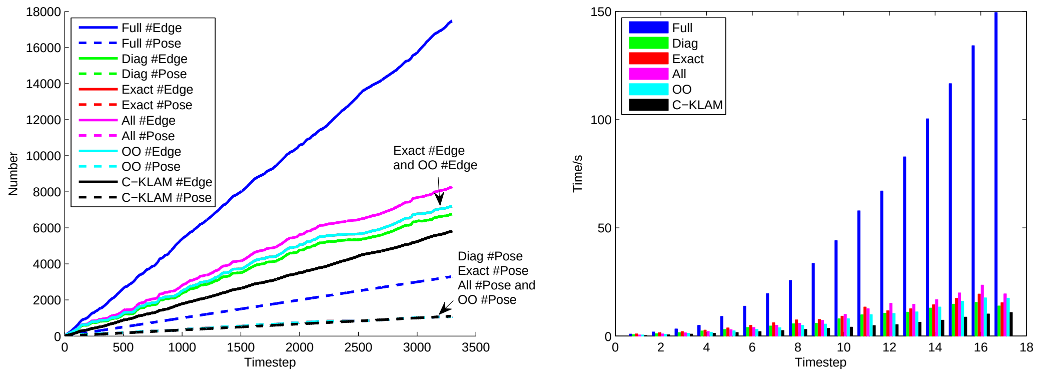


Figure 14. The evolution of number of poses and edges (left) and optimization time (right) on the DLR dataset. The curves for the number of poses overlap except C-KLAM. The curves for the number of edges using Exact and OO overlap.

7. Conclusion

In this paper a modular two-step solution is presented to enable the GraphSLAM has complexity related to the size of the map. It consists of an online pose pruning algorithm and an edge generation algorithm. By using the proposed solution, the pose that mostly observing common features which are observed many times can be pruned, relating the complexity of the graph to the size of the map instead of the length of the trajectory. To avoid the sparse structure being broken when a pose is pruned, an edge generation algorithm is conducted. The consistent constrained version is more conservative while the unconstrained version may be more accurate when linearized point is good. Based on the comparison in experiments, there are three factors having effects on the result, including linearized points, information loss and overconfidence. By a balanced consideration, unconstrained Diagonal and consistency constrained Exact using (44) will be good options for practical application with their top two ER performances on datasets tested (0.032 and 0.099 in average, respectively). Between the two methods, unconstrained Diagonal has lowest complexity with its closed form solution to edge generation while consistency

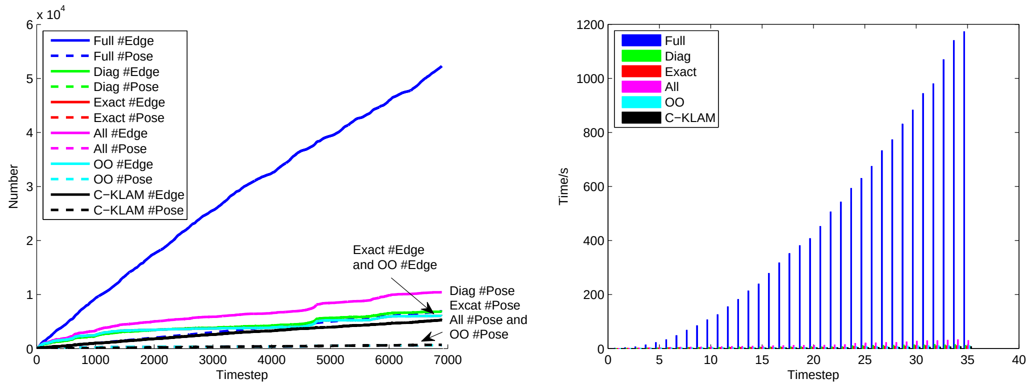


Figure 15. The evolution of number of poses and edges (left) and optimization time (right) on the Victoria Park dataset. The curves for the number of poses overlap except C-KLAM. The curves for the number of edges using Exact and OO overlap.

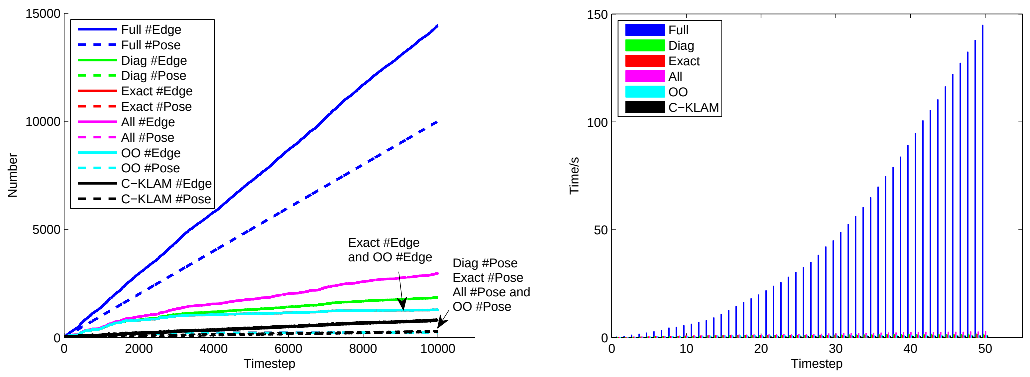


Figure 16. The evolution of number of poses and edges (left) and optimization time (right) on the CityTree10000 dataset. The curves for the number of poses overlap except C-KLAM. The curves for the number of edges using Exact and OO overlap.

constrained Exact has the lowest number of edges and most stable evolution of number of edges.

In the future, more work needs to be done to further improve the solution. Firstly, we will investigate more efficient MAXDET solver in order to make the solution more efficient. Secondly, more large scale datasets will be used to evaluate the performance. Thirdly, the solution will be extended to 3D environment for more applications.

Acknowledgement

This work is supported by the National Natural Science Foundation of China (61473258), the Program for Zhejiang Leading Team of S&T Innovation (2009R50014), and the Joint Centre for Robotics Research (JCRR) between Zhejiang University and the University of Technology, Sydney.

References

[1] Huang S, Dissanayake G. Convergence and consistency analysis for extended kalman filter based slam. Robotics, IEEE Transactions on. 2007;23(5):1036–1049.

[2] Huang G, Kaess M, Leonard JJ. Consistent sparsification for graph optimization. In: Mobile robots (ecmr), 2013 european conference on. IEEE. 2013. p. 150–157.

[3] Carlevaris-Bianco N, Eustice RM. Generic factor-based node marginalization and edge sparsification

for pose-graph slam. In: Robotics and automation (icra), 2013 ieee international conference on. IEEE. 2013. p. 5748–5755.

[4] Wang Y, Xiong R, Li Q, Huang S. Kullback-leibler divergence based graph pruning in robotic feature mapping. In: Mobile robots (ecmr), 2013 european conference on. IEEE. 2013. p. 32–37.

[5] Dissanayake MG, Newman P, Clark S, Durrant-Whyte HF, Csorba M. A solution to the simultaneous localization and map building (slam) problem. Robotics and Automation, IEEE Transactions on. 2001;17(3):229–241.

[6] Liu Y, Thrun S. Results for outdoor-slam using sparse extended information filters. In: Icra. 2003. p. 1227–1233.

[7] Walter MR, Eustice RM, Leonard JJ. Exactly sparse extended information filters for feature-based slam. The International Journal of Robotics Research. 2007;26(4):335–359.

[8] Huang S, Dissanayake G. Convergence and consistency analysis for extended kalman filter based slam. Robotics, IEEE Transactions on. 2007;23(5):1036–1049.

[9] Montemerlo M, Thrun S, Koller D, Wegbreit B, et al.. Fastslam: A factored solution to the simultaneous localization and mapping problem. In: Aaai/iaai. 2002. p. 593–598.

[10] Bailey T, Nieto J, Nebot E. Consistency of the fastslam algorithm. In: Robotics and automation, 2006. icra 2006. proceedings 2006 ieee international conference on. IEEE. 2006. p. 424–429.

[11] Thrun S, Montemerlo M. The graph slam algorithm with applications to large-scale mapping of urban structures. The International Journal of Robotics Research. 2006;25(5-6):403–429.

[12] Dellaert F, Kaess M. Square root sam: Simultaneous localization and mapping via square root information smoothing. The International Journal of Robotics Research. 2006;25(12):1181–1203.

[13] Newman P, Sibley G, Smith M, Cummins M, Harrison A, Mei C, Posner I, Shade R, Schroeter D, Murphy L, et al.. Navigating, recognizing and describing urban spaces with vision and lasers. The International Journal of Robotics Research. 2009;28(11-12):1406–1433.

[14] Grisetti G, Kummerle R, Stachniss C, Burgard W. A tutorial on graph-based slam. Intelligent Transportation Systems Magazine, IEEE. 2010;2(4):31–43.

[15] Huang S, Wang Z, Dissanayake G. Sparse local submap joining filter for building large-scale maps. Robotics, IEEE Transactions on. 2008;24(5):1121–1130.

[16] Huang S, Wang Z, Dissanayake G, Frese U. Iterated d-slam map joining: evaluating its performance in terms of consistency, accuracy and efficiency. Autonomous Robots. 2009;27(4):409–429.

[17] Konolige K, Agrawal M. Frameslam: From bundle adjustment to real-time visual mapping. Robotics, IEEE Transactions on. 2008;24(5):1066–1077.

[18] Kaess M, Ranganathan A, Dellaert F. isam: Incremental smoothing and mapping. Robotics, IEEE Transactions on. 2008;24(6):1365–1378.

[19] Kaess M, Johannsson H, Roberts R, Ila V, Leonard JJ, Dellaert F. isam2: Incremental smoothing and mapping using the bayes tree. The International Journal of Robotics Research. 2012;31(2):216–235.

[20] Johannsson H, Kaess M, Fallon M, Leonard JJ. Temporally scalable visual slam using a reduced pose graph. In: Robotics and automation (icra), 2013 ieee international conference on. IEEE. 2013. p. 54–61.

[21] Ila V, Porta JM, Andrade-Cetto J. Information-based compact pose slam. Robotics, IEEE Transactions on. 2010;26(1):78–93.

[22] Kim A, Eustice RM. Combined visually and geometrically informative link hypothesis for pose-graph visual slam using bag-of-words. In: Intelligent robots and systems (iros), 2011 ieee/rsj international conference on. IEEE. 2011. p. 1647–1654.

[23] Kretzschmar H, Stachniss C. Information-theoretic compression of pose graphs for laser-based slam. The International Journal of Robotics Research. 2012;31(11):1219–1230.

[24] Vial J, Durrant-Whyte H, Bailey T. Conservative sparsification for efficient and consistent approximate estimation. In: Intelligent robots and systems (iros), 2011 ieee/rsj international conference on. IEEE. 2011. p. 886–893.

[25] Eade E, Fong P, Munich ME. Monocular graph slam with complexity reduction. In: Intelligent robots and systems (iros), 2010 ieee/rsj international conference on. IEEE. 2010. p. 3017–3024.

[26] Carlevaris-Bianco N, Eustice RM. Conservative edge sparsification for graph slam node removal. In: Robotics and automation (icra), 2014 ieee international conference on. IEEE. 2014.

[27] Mazuran M, Tipaldi GD, Spinello L, Burgard W. Nonlinear graph sparsification for slam. In: Robotics: Science and systems. Berkley. 2014.

[28] Nerurkar E, Wu K, Roumeliotis S. C-klam: Constrained keyframe-based localization and mapping. In: Robotics and automation (icra), 2014 ieee international conference on. IEEE. 2014.

[29] Vandenberghe L, Boyd S, Wu SP. Determinant maximization with linear matrix inequality constraints. SIAM journal on matrix analysis and applications. 1998;19(2):499–533.
[30] Grant M, Boyd S, Ye Y. Cvx: Matlab software for disciplined convex programming. 2008.
[31] Kurlbaum J, Frese U. A benchmark data set for data association. Univ Bremen, Bremen, Germany, SFB/TR. 2009;8:017–02.
[32] Guivant JE, Nebot EM. Optimization of the simultaneous localization and map-building algorithm for real-time implementation. Robotics and Automation, IEEE Transactions on. 2001;17(3):242–257.
[33] Olson E, Kaess M. Evaluating the performance of map optimization algorithms. In: Rss workshop on good experimental methodology in robotics. Vol. 15. 2009.
[34] Huang S, Wang Z, Dissanayake G, Frese U. Iterated d-slam map joining: evaluating its performance in terms of consistency, accuracy and efficiency. Autonomous Robots. 2009;27(4):409–429.
[35] Huang GP, Mourikis AI, Roumeliotis SI. Observability-based rules for designing consistent ekf slam estimators. The International Journal of Robotics Research. 2010;29(5):502–528.