On-line 3D active pose-graph SLAM based on key poses using graph topology and sub-maps

Yongbo Chen[1,2], Shoudong Huang[1], Robert Fitch[1], Liang Zhao[1], Huan Yu[2] and Di Yang[2]

Abstract— **In this paper, we present an on-line active pose-graph simultaneous localization and mapping (SLAM) framework for robots in three-dimensional (3D) environments using graph topology and sub-maps. This framework aims to find the best trajectory for loop-closure by re-visiting old poses based on the T-optimality and D-optimality metrics of the Fisher information matrix (FIM) in pose-graph SLAM. In order to reduce computational complexity, graph topologies are introduced, including weighted node degree (T-optimality metric) and weighted tree-connectivity (D-optimality metric), to choose a candidate trajectory and several key poses. With the help of the key poses, a sampling-based path planning method and a continuous-time trajectory optimization method are combined hierarchically and applied in the whole framework. So as to further improve the real-time capability of the method, the sub-map joining method is used in the estimation and planning process for large-scale active SLAM problems. In simulations and experiments, we validate our approach by comparing against existing methods, and we demonstrate the on-line planning part using a quad-rotor unmanned aerial vehicle (UAV).**

I. INTRODUCTION

In an unknown environment, active SLAM is to choose good trajectories for improving the SLAM result and performing the given tasks, such as coverage. In this paper, we present a 3D on-line active pose-graph SLAM framework.

Popular frameworks in this area, including model predictive control (MPC) [1] and partially observably Markov decision process (POMDP) [2], try to choose the best future trajectory via some indexes. Most of these indexes focus on the uncertainty level [3] and the data association award [4] of the future SLAM results. Recently, the Theory of Optimal Experimental Design (TOED) [5], including A-, D-, E- and T-optimality metrics, is widely applied in active SLAM and similar areas, such as active perception [5] and sensor selection [6]. A comparison of these criteria is presented in [7]. Recent work [8] shows that the monotonicity property of different metrics is greatly affected by the representation of uncertainty and the orientation of the robot pose.

Active SLAM can be viewed as a highly non-convex optimal control problem with a large search space. To generate candidate paths efficiently is also very important. The frontier-based exploration method [9] leads active SLAM into a discrete optimization domain by choosing a small subset of future way-points. Traditional shortest path planning algorithms, such as A* [10], RRT* [11] and D* [12], are also used in active SLAM to generate candidate paths.

In generating candidate actions based on these TOED metrics, active SLAM becomes computationally expensive. Therefore, most of the work in this area tries to reduce the computational complexity of the objective function of this problem. In the earlier literature, based on the Extended Kalman Filter (EKF) and the Extended Information Filter (EIF), the MPC framework is applied to limit the size of the planning problem by limiting its horizon [1]. In [13], based on the D-optimality metric, the authors develop a computationally efficient approach for decision making under uncertainty by applying the matrix determinant lemma and reusing calculation among all candidates. A conservative diagonal information matrix is introduced to reduce the computational complexity of the D-optimality objective function of the candidate actions in [14].

This paper presents an on-line active pose-graph SLAM framework using graph topology and sub-maps. First, by using RRT-style expansions, many random candidate paths are generated. Based on the weighted node degree and the weighted tree-connectivity, several key poses belonging to the best path are selected. Based on these key poses, considering the robot dynamic model, a continuous-time trajectory optimization problem is formulated and solved by the quasi-Newton method. In this process, sub-map planning and estimation are used to improve real-time capabilities.

The main contributions of this work are as follows:

- Hierarchical evaluation with a fast but general-performance method using weighted node degree for all candidate paths, and a slow but fruitful method using weighted tree-connectivity for a selected set of paths.
- Application of sub-map planning and estimation, to improve computational efficiency.
- Application of real-time sampling-based planning and continuous-time optimization in active SLAM.
- Introduction of key poses to build a hierarchical framework.

II. PROBLEM DESCRIPTION

Assume that the robot is performing some tasks in an unknown environment such as coverage, exploration or search. Our active pose-graph SLAM framework is used to reduce pose uncertainty and further helps to finish the original tasks. The active SLAM problem considered in this paper is to mostly retain the main originally designed path while finding good additional loop-closure trajectories between the current

[1]Yongbo Chen, Shoudong Huang, Robert Fitch and Liang Zhao are with Faculty of Engineering and Information Technology, University of Technology Sydney, Ultimo, NSW, 2007 Australia Yongbo.Chen@student.uts.edu.au

[2]Yongbo Chen, Huan Yu and Di Yang are with Beijing Institute of Technology (BIT), Beijing, 100081 China

pose and the previous poses to reduce the uncertainty of pose-graph SLAM subject to resource limitations.

For example, suppose the robot is performing a coverage task in an unknown space. When the pose uncertainty in pose-graph SLAM is large, in moving from the current pose, the robot needs to revisit a previous pose and finally return back to perform the original task. Fig. 1 shows a system using our method. Compared to the MPC framework [15], this method can be easily applied in different tasks, and does not need to run in every step.

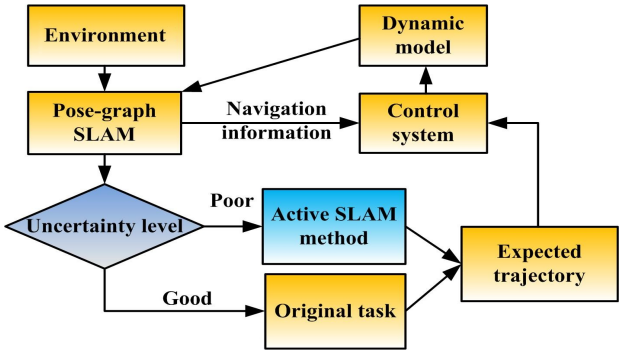


Fig. 1. Role of our active SLAM method in the robot system

III. Pose-graph SLAM

A. Graph Preliminaries

The pose-graph SLAM problem can be represented as a weighted weakly-connected directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \omega)$, where $\mathcal{V} = \{0, 1, 2, \cdots, n_p\}$, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ and $|\mathcal{E}| = m$. Each node $P_i \in \mathcal{V}$ denotes a robot pose, and each edge $e_k = (i_k, j_k) \in \mathcal{E}$ represents the k-th relative measurement between two robot poses P_i and P_j. Without loss of generality, the first node P_0 is anchored for the SLAM problem.

The node degree of pose P_i is denoted as d_i, which means the number of measurements connected by the i-th node. The set of poses connected with pose P_i is denoted as V_i. The reduced weighted Laplacian matrix corresponding to the pose-graph $\mathcal{G}$ is defined as $\boldsymbol{L} \triangleq \boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}^\top$, where $\boldsymbol{A} \in \{-1, 0, 1\}^{n_p \times m}$ means the reduced incidence matrix after anchoring P_0 to the origin, and $\boldsymbol{\Sigma} = \text{diag}\{\omega(e_1), \omega(e_2), \cdots, \omega(e_m)\}$ is a diagonal matrix whose diagonal elements are the weights of the graph edges.

B. Synchronization on $\mathbb{R}^n \times SO(n)$

2D/3D pose-graph SLAM, belonging to synchronization problem on $\mathbb{R}^n \times SO(n)$, $n = 2,\ 3$, consists of estimating the values of a set of n_p unknown poses $P_1, \cdots, P_{n_p} \in \mathbb{R}^n \times SO(n)$ given m noisy relative rotations $\boldsymbol{R}_j\boldsymbol{R}_i^\top$ and relative coordinate transformations $\boldsymbol{R}_i^\top(\boldsymbol{x}_j - \boldsymbol{x}_i)$. We assume that the noisy measurement follows [16]:

$$\begin{aligned} \boldsymbol{p}_{ij} &= \boldsymbol{R}_i^\top(\boldsymbol{x}_j - \boldsymbol{x}_i) + \boldsymbol{y}_{ij}, \quad \boldsymbol{y}_{ij} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma}_{ij}) \\ \boldsymbol{H}_{ij} &= \boldsymbol{Z}_{ij}\boldsymbol{R}_j\boldsymbol{R}_i^\top, \qquad \boldsymbol{Z}_{ij} \sim Lang(\boldsymbol{I}_{n\times n}, \kappa_{ij}), \end{aligned} \tag{1}$$

where $\mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma}_{ij})$, $\boldsymbol{\Sigma}_{ij} = \delta_{ij}^2\boldsymbol{I}_{n\times n}$ means the random vector following the isotropic Gaussian distribution, $Lang(\boldsymbol{I}_{n\times n}, \kappa_{ij})$ means the isotropic Langevin distribution with mean $\boldsymbol{I}_{n\times n}$ and concentration $\kappa_{ij} \geq 0$.

Given a set of noisy measurements $\boldsymbol{p}_{ij}$ and $\boldsymbol{H}_{ij}$, the pose-graph SLAM problem is to obtain a maximum-likelihood estimate for the poses $P_i = (\boldsymbol{x}_i, \boldsymbol{R}_i) \in \{\mathbb{R}^n \times SO(n)\}^{n_p}$:

$$\max_{P_i} \sum_{e_k \in \mathcal{E}} \kappa_{ij} tr(\boldsymbol{H}_{ij}\boldsymbol{R}_i\boldsymbol{R}_j^\top) - \frac{\delta_{ij}^{-2}}{2}\|\boldsymbol{p}_{ij} - \boldsymbol{R}_i^\top(\boldsymbol{x}_j - \boldsymbol{x}_i)\|^2. \tag{2}$$

For this pose-graph SLAM problem, the SE-sync method [16] is applied to solve it. This method is one of the state-of-the-art algorithms, which shows the outstanding computational efficiency and certifiably globally optimal property. After the computational enhancements, it is as fast as some highly optimized libraries, like g2o [17].

C. FIM and optimality design metrics

The FIM $\boldsymbol{\mathcal{I}}$ of the pose-graph SLAM result is a tool to assess the quality of the measurement network. Based on $\boldsymbol{\mathcal{I}} = \boldsymbol{J}^\top\boldsymbol{\Sigma}\boldsymbol{J}$, in which $\boldsymbol{J}$ is the Jacobian matrix for (1), the FIM $\boldsymbol{\mathcal{I}}_{2D}$ for 2D SLAM with block-isotropic Gaussian noise is shown in [18]:

$$\boldsymbol{\mathcal{I}}_{2D} = \left[\begin{array}{c|c} \boldsymbol{L}_w^{\mathbb{R}^2} & \triangle_w^\top \\ \hline \triangle_w & \boldsymbol{L}_w^{SO(2)} + \text{diag}\{\psi_1, \cdots, \psi_{n_p}\} \end{array}\right], \tag{3}$$

where $\boldsymbol{L}_w^{\mathbb{R}^2}$ and $\boldsymbol{L}_w^{SO(2)}$ are the reduced weighted Laplacian matrices after using the Kronecker product operation, $\psi_i = \sum_{j\in V_i^+} \delta_{ij}^{-2}\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2, i = 1, \cdots, n_p$, the specific structure of $\triangle_w$ is shown in [18]. A similar result for 3D SLAM is shown in [19].

The D-optimality and T-optimality design metrics are two important metrics directly using the FIM without the inverse operation. The D-optimality and T-optimality metrics are, respectively, to maximize the log-determinant function $\log(\det(\boldsymbol{\mathcal{I}}_{\boldsymbol{nD}}))$ and the trace function $\text{trace}(\boldsymbol{\mathcal{I}}_{nD})$.

IV. Relation between design metrics and graph topology

In this section, the D- and T-optimality design metrics are shown to have a strong relationship with the graph structures, including weighted node degree and weighted tree-connectivity, of the SLAM measurement network.

A. T-optimality metric and weighted node degree

Based on (3), for both 2D and 3D situation, we can get the T-optimality design metric of the FIM:

$$\begin{aligned} \text{trace}(\boldsymbol{\mathcal{I}}_{nD}) &= \text{trace}(\boldsymbol{L}_w^{\mathbb{R}^n}) \\ +\text{trace}(\boldsymbol{L}_w^{SO(n)}) &+ \sum_{i=1}^{n_p}\sum_{j\in V_i^+} \delta_{ij}^{-2}\|\boldsymbol{x}_j - \boldsymbol{x}_i\|^2, \end{aligned} \tag{4}$$

The specific derivation is shown in [19].

For many real-world pose-graph SLAM problems, compared with $\text{trace}(\boldsymbol{L}_w^{\mathbb{R}^n})$ and $\text{trace}(\boldsymbol{L}_w^{SO(n)})$, the last term $\sum_{i=1}^{n_p}\sum_{j\in V_i^+} \delta_{ij}^{-2}\|\boldsymbol{x}_j - \boldsymbol{x}_i\|^2$ is relatively small, so we have:

$$\begin{aligned} \text{trace}(\boldsymbol{\mathcal{I}}_{nD}) \approx& \text{trace}(\boldsymbol{L}_w^{\mathbb{R}^n}) + \text{trace}(\boldsymbol{L}_w^{SO(n)}) \\ =& \sum_{i=1}^{n_p}\sum_{j\in V_i} \left(\omega_{ij} + n\delta_{ij}^{-2}\right), \end{aligned} \tag{5}$$

where $\omega_{ij} = 2\kappa_{ij}$ (for 2D) is the weight of the rotation graph edge and its 3D value is shown in [19], $\sum_{j\in V_i}\left(\omega_{ij} + n\delta_{ij}^{-2}\right)$ is the weighted value of the graph edge connected with the i-th node, named weighted node degree. It is easy to find that maximizing the T-optimality metric is almost equivalent to maximizing the weighted node degree. The weighted node degree is a computationally efficient metric with a similar effect for the T-optimality-based active SLAM problem.

B. D-optimality metric and weighted tree-connectivity

In [18], the lower and upper bounds of the D-optimality metric of the FIM are shown as:

$$\begin{aligned} &\mathcal{L} \leq \log(\det(\boldsymbol{\mathcal{I}}_{2D})) \leq \mathcal{U} \\ &\mathcal{L} = \log(\det(\boldsymbol{L}_w^{\mathbb{R}^2})) + \log(\det(\boldsymbol{L}_w^{SO(2)})) \\ &\mathcal{U} = \log(\det(\boldsymbol{L}_w^{\mathbb{R}^2})) + \sum_{i=1}^{n}\log(\lambda_i(\boldsymbol{L}_w^{SO(2)}) + \lambda_\infty), \end{aligned} \tag{6}$$

where $\lambda_\infty = \max_{i=1,2,\cdots,n_p} \sum_{j\in V_i^+} \delta_{ij}^{-2}\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2$ means the biggest eigenvalue of diag$\{\psi_1, \cdots, \psi_{n_p}\}$ for the 2D case. A similar result for the 3D case is shown in [19].

We assume that the SLAM result is 'near-optimal'. For many real-world datasets, compared with the eigenvalues of the rotation group $\lambda_i(\boldsymbol{L}_w^{SO(2)})$, the term λ_∞ is relatively small, so we have:

$$\mathcal{U} \approx \mathcal{L} \Rightarrow \log(\det(\boldsymbol{\mathcal{I}}_{nD})) \approx \mathcal{L}. \tag{7}$$

It is easy to find that optimizing the D-optimality metric is almost equivalent to optimizing the lower bound of the FIM. The lower bound $\mathcal{L}$ is known as the weighted tree-connectivity of the SLAM measurement network [18].

The computational complexity of the weighted tree-connectivity is much smaller than the original metric due to its lower dimension and sparser structure. Based on Algorithm 1 in [19], it can be computed much faster than the log-determinant function of the dense matrix.

C. Comparison among the four metrics

The weighted node degree and the weighted tree-connectivity metrics have similar performance and better computational efficiency than T- and D-optimality, respectively. For two new metrics, from the computational complexity point of view, the weighted node degree is cheaper than the weighted tree-connectivity. However, from the effectiveness point of view, we have found that the weighted tree-connectivity can identify the uncertainty levels of two graphs with similar weighted node degree [19]. Thus, weighted tree-connectivity is a relatively expensive metric with good-performance.

Based on the above discussion, we have a basic idea for determining the loop-closure trajectory. Weighted node degree is a good metric to be used in large-scale search for rough candidate actions, and weighted tree-connectivity is suitable for sophisticated search within a small elite group. This idea leads to the hierarchical evaluation strategy listed in the first dot point of our contributions.

V. On-line active SLAM framework

A. RRT-connect and weighted node degree for initial search

First, N_c+1 potential old loop-closure poses is uniformly obtained by dividing the path into several small segments: $S = \{P_1, P_{r(i'/N_c)}, P_{2r(i'/N_c)}, ..., P_{N_c r(i'/N_c)}\}$, where $r(\star)$ means the maximal integer smaller than $\star$. For every potential old loop-closure pose in S, seeing it as the target and the last pose $P_{i'}$ as start point, we can generate N_{RRT} candidate paths based on the RRT-connect method [20]. So we have $N_{RRT}(N_c+1)$ candidate paths.

Then, for every candidate path, because the noise level is in general proportional to the number of points which are visible in both two poses, we can count its weighted node degree by the weighted number of the common features.

Finally, the best paths are chosen by sorting all the virtual weighted node degrees of the candidate paths. They make up the best elite group with N_e paths.

B. Tree-connectivity for elite search and key poses selection

For this small elite group, every path is evaluated by the weighted tree-connectivity metric.

Then, the best path is obtained by choosing the one with the largest weighted tree-connectivity. In this best path, all the poses, which are the nodes in RRT-connect, are divided into N_k sub-sets uniquely. In the k-th sub-set, the poses are sorted by their weighted node degrees and the best pose is defined as the k-th key pose $\widehat{P}_k$. The benefit of the application of the sub-set is to avoid the situation where key poses are located at adjoining positions, which reduces the effectiveness of the key poses in a smooth path.

C. Fast trajectory planning

Based on the above method, N_k key poses are found to guarantee the performance of the selected result. In order to reduce the length of the random sampling path and assign the smooth continuous velocity along the path, we apply a continuous-time trajectory planning method [21] to pass the key poses and connect the start point and the target.

In every dimension (x, y and z), the robot trajectory is represented as a high-degree polynomial spline with N-th order, whose variable is time t:

$$f_\mu(t) = \boldsymbol{A}^\top \boldsymbol{T}(t),\ \mu = x, y, z, \tag{8}$$

where $\boldsymbol{A} = [a_0, a_1, ..., a_N]^\top$ and $\boldsymbol{T}(t) = \left[t^0, t^1, ..., t^N\right]^\top$.

The coordinates of the key poses and the end constraints are set as the fixed derivatives $\boldsymbol{d_F}$ [21]. The remaining free derivatives $\boldsymbol{d_P}$, which are the optimized variables, are the free waypoints in the spline. The mapping between the polynomial coefficients $\boldsymbol{A}$ and the derivatives meets:

$$\boldsymbol{A} = \widehat{\boldsymbol{T}}^{-1}\boldsymbol{M}\left[\boldsymbol{d}_{\boldsymbol{F}}^\top\ \boldsymbol{d}_{\boldsymbol{P}}^\top\right]^\top, \tag{9}$$

where $\widehat{\boldsymbol{T}} = [\boldsymbol{T}(t_0)\ \boldsymbol{T}(t_1)\ \cdots\ \boldsymbol{T}(t_N)]^\top$ is a mapping matrix from polynomial coefficients to the end-derivatives, $t_0, \cdots, t_N$ are the reach time corresponding to the derivatives, $\boldsymbol{M}$ is the re-ordering matrix.

The objective function of the trajectory planning problem is to avoid obstacles and to minimize the snap:

$$J = \omega_1 J_1 + \omega_2 J_2$$
$$J_1 = \int_{t_0}^{t_N} \sum_{\mu=x,y,z} \frac{d^k f_\mu(t)}{dt^k} dt, J_2 = \int_{t_0}^{t_N} c(\boldsymbol{f}) \left\| \frac{d\boldsymbol{f}}{dt} \right\| dt, \tag{10}$$

where $\boldsymbol{f} = [f_x(t), f_y(t), f_z(t)]$, $c(\boldsymbol{f})$ is the potential cost to avoid an obstacle. Its computation is introduced in Section V-E. In our simulation, the coefficient k is set as 4. Using (12), the objective function in (13) can be derived as:

$$J_1 = [\boldsymbol{d_F}\ \boldsymbol{d_P}]\, \boldsymbol{M}^\top \widehat{\boldsymbol{T}}^{-\top} \boldsymbol{Q} \widehat{\boldsymbol{T}}^{-1} \boldsymbol{M} \left[\boldsymbol{d_F}^\top\ \boldsymbol{d_P}^\top\right]^\top, \tag{11}$$

where $\boldsymbol{Q}$ is the Hessian matrix of J_1.

Then this trajectory optimization problem is solved based on the quasi-Newton method with known gradient. The specific gradient equation is shown in [21].

D. Special amendment for directional sensor

The above method is suitable only for omni-directional sensors, because the trajectory passes through key poses without a direction constraint. In order to apply this framework with a directional sensor, like a camera, a velocity constraint is introduced in the trajectory planning problem.

For every key pose $\widehat{P}_k$, we need to set its speed $\boldsymbol{V}(\widehat{P}_k)$ based on the RRT-connect result and set its reaching time $T^*(t_k)$. Except the fixed derivative corresponding to the key pose, an additional fixed derivative $\widehat{P}_k^{add}$ needs to be introduced into the continuous-time trajectory planning problem and its corresponding time is $T^*(t_k) + \Delta t$:

$$\widehat{P}_k^{add} = \widehat{P}_k + \boldsymbol{V}(\widehat{P}_k)\Delta t, \tag{12}$$

where Δt is very small to ensure the velocity direction. This simple operation can be used in any pose with the velocity direction constraint. An example of the continuous-time trajectory planning problem with the velocity constraint is shown in Fig. 2.

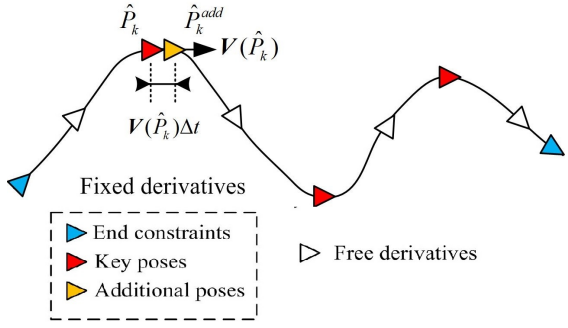


Fig. 2. An example of the continuous-time trajectory planning result with velocity constraint based on the fixed derivatives and free derivatives.

E. Map representation

Focusing on the pose-graph SLAM, we assume that the environment has been constructed on-line by some tools such as Octomap [21] or SDF tools [22]. In order to avoid obstacles in the following planning process, for a waypoint $\boldsymbol{s}$ in the planning space, we can first identify whether it is in collision with obstacles, and then get the closest point and the distance $d(\boldsymbol{s})$ between this waypoint and the obstacles. In this way, a potential field $c(\boldsymbol{s})$ using the Euclidean Singed Distance Field is built as described in [21]. In this process, in order to avoid the frequent computations of $c(\boldsymbol{s})$ and its gradient in the continuous-time trajectory optimization, they are computed in the mapping process by GPU or individual processor cores quickly before the continuous-time trajectory planning process and then be called using the hash table by the help of the grid representation of the planning space.

F. Whole framework summary

In this section, we summarize all the steps used in our framework and show how to apply it in performing the original task (Algorithm 1).

Algorithm 1: On-line active pose-graph SLAM method

Input: Robot parameters, original task
Output: Best active SLAM trajectory

while *Original task hasn't been finished* **do**
  if *Pose Uncertainty is accepted* ($\widehat{\boldsymbol{C}}_{i'i'} < Index_1$) **then**
    Keep performing the original task;
  else
    if *Robot resource is enough* **then**
      Planning 1:
      Step 1: Output all potential loop-closure poses with equal interval: S;
      Step 2: Generate multiple RRT-connect paths based on S and last pose $P_{i'}$, evaluate their weighted node degrees and choose elite group;
      Step 3: Evaluate weighted tree-connectivity, choose best path and select key poses;
      Step 4: Build continuous-time trajectory planning problem based on fixed key poses.
      end Planning 1
      repeat
        Control robot to follow continuous-time result;
        if *Reach target of Planning 1 result* **then**
          Planning 2:
          Generate multiple RRT-connect paths based on current pose and $P_{i'}$, then run rest operations in Step 2-4 again;
          end Planning 2
        end
      until *Go back to last pose* $P_{i'}$;
    else
      Keep performing original task;
    end
  end
end

VI. Sub-map planning and estimating

Even though the proposed method only needs to evaluate the weighted tree-connectivity of the small elite group, it is still expensive for a large-scale problem. So this paper uses the sub-map estimation and planning idea to improve accuracy of the local map and its real-time ability [23].

When the size of the problem is larger than a threshold ($n_p > Index_2$), a new sub-map is built. All the pose-graph structure before this pose is saved and a new pose-graph SLAM problem is built. The target of the global task is

transformed into local coordinates based on the coordinate frames of the sub-maps. The active SLAM method is applied based on the new pose graph without considering other sub-maps. It is noted that we divide the sub-maps by the measurements. There are some common poses in different sub-maps, which serve as loop-closures.

VII. Simulations and experiments

A. Simulation

For the simulation, the whole system including the dynamic model, control method, constructed map and active SLAM method, is implemented in MATLAB on a laptop PC with Intel Core i7-7700HQ @ 3.5 GHz and 8GB RAM.

In order to get the measurements following the noisy assumption shown in (1), based on the singular value decomposition, the noise-free relative rotation and translation are obtained from the 3D points observations, and then, the random noises are sampled by $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_{ij})$ and $Lang(\boldsymbol{I}_{n\times n}, \kappa_{ij})$. The indexes κ_{ij} and δ_{ij} are set to be proportional to the feature number N_v which are visible both from P_i and P_j [24].

TABLE I

Comparison of trajectory length and computational time

		Average planning time		Trajectory length	
Algorithm		Mean (s)	STD (s^2)	Mean (m)	STD (m^2)
Task 1	Non	-	-	25.44	-
	Ours	2.52	0.37	41.63	1.46
	Con-D	7.65	0.38	41.02	1.72
	Con-T	6.29	0.63	43.92	3.71
	RRT-D	8.45	0.53	41.16	0.78
	RRT-T	6.90	0.55	42.88	0.87
Task 2	Non	-	-	15.20	-
	Ours	1.45	0.56	28.35	4.21
	Con-D	6.00	0.79	27.09	3.29
	Con-T	5.18	0.89	27.17	1.94
	RRT-D	6.45	1.16	30.41	0.44
	RRT-T	5.81	0.53	29.69	0.70

1) A small-scale simulation: In this part, we present a simulation environment in a 7m$\times$7m$\times$1m space with several regular obstacles as shown in Fig. 4. The original task of the robot is to pass several way-points (Blue pentagrams) with a velocity 0.1m/s. The simulation time step Δt is set as 1s. The noise parameter δ_{ij}^{-2} and κ_{ij} for the control input are 10^5 and 10^4 in every step. The noise parameter δ_{ij}^{-2} and κ_{ij} for the odometry are $20N_v$ and $10^4 N_v$. $Index_1$, N_c and N_{RRT} are respectively set as 1×10^{-3}, 10 and 5. The sub-map idea is not used in this small-scale simulation. In other words, only one sub-map is used in each simulation. The maximal one-way loop-closure trajectory is limited below 4m. In order to avoid frequent loop-closure, in one sub-map, the active SLAM method is only allowed to be triggered twice ($N_s = 2$). Limiting the number of the triggered times and length of the loop-closure trajectory are our ways to handle the limitation of the robot resources. There are other ways to do that.

We compare our method with some other methods. The simulation of following the way points without active loop closure is named as non-active method (Non). The simulations, which use continuous-time trajectory planning method to directly re-visit the potential loop-closure poses without using RRT-connect method, based on the T- and D-optimality metrics are named as continuous-time with D/T-opt (Con-D/T). The simulations, using RRT-connect method, are named as RRT-connect with D/T-opt (RRT-D/T). Because of the randomness, for every task, based on different methods, we run the simulations 10 times, and then apply the cumulative distribution function (CDF) to evaluate the covariance of the robot position at every step [25]. The uncertainty comparison results are shown in Fig. 3. We also show the statistical results of the trajectory length and the average simulation time of Planning 1 and Planning 2 of one simulation in Table I (The data association of the common features for the predicted poses is included in the planning time). One of the trajectories and final estimated results for tasks 1 (398 Poses) and 2 (208 Poses) using our method is shown in Fig. 4.

Fig. 3 and Table I show that our method has the best performance in uncertainty reduction and planning speed. In other 10 simulations with 10 different paths, we find that the mean position covariances improve 37% on average by the additional 71% path compared with non-active method. In Fig. 4, we see that the active SLAM is triggered twice because of the weak-connectivity of the measurement network. Relative measurements are shown by the blue lines, and the red circles are the estimated robot poses at last step.

2) Large-scale simulation with multiple sub-maps: Even though the space is small, because of the low velocity and the large number of features, there are more than 10^4 relative pose measurements in tasks 1 and 2. It is hard to solve the active SLAM problem in real-time. In order to reduce the computational complexity, the sub-map idea is used. $Index_2$ is set as 150. Simulation results with 9 sub-maps and more than 10^5 relative measurements are shown in Fig. 6.

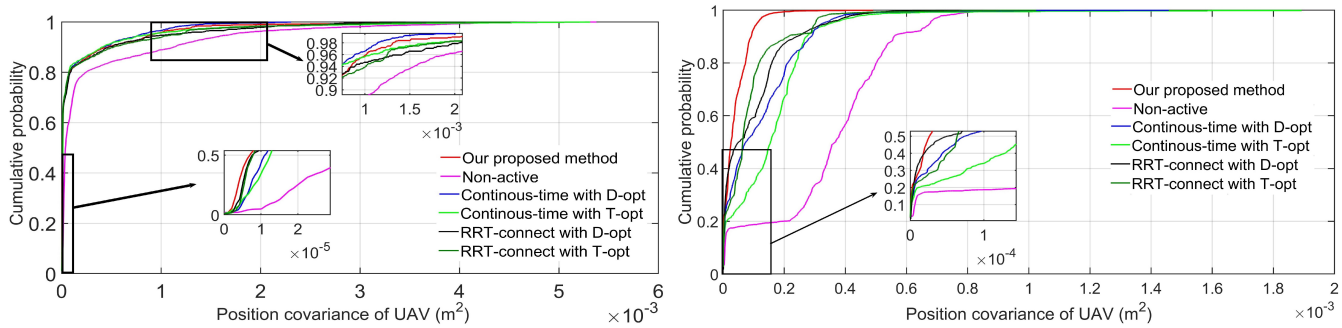
With the help of sub-maps, the planning and SLAM algorithms are limited to 2s and 1.5s, respectively, in every sub-map for the above simulations.

B. On-line experiments

For the on-line experiments, based on a real quad-rotor UAV platform, the planning method shown in Algorithm 1 is implemented on-line in C++ on a desktop PC with Intel(R) Core(TM) i7-4790K CPU @ 4.0GHz with 32 GB of DDR3 1600MHz RAM. The map is constructed off-line. The virtual relative measurements are obtained by the global localization system using the relative poses and adding expected noise. Pose-graph SLAM is performed on-line but not at every step. The final trajectory is shown in Fig. 5. The planning parts of the active SLAM are triggered twice, shown by the yellow lines. The experiments show that the planning part can be finished on-line in 0.2-0.4s (2-5Hz) with about 80 poses in the pose-graph.

VIII. Conclusions

This paper presents an on-line active pose-graph SLAM method for a robot operating in a 3D unknown environment. This method performs active SLAM by adding loop-closure



(a) CDF of the covariance in the robot position at every time step of task 1 (b) CDF of the covariance in the robot position at every time step of task 2

Fig. 3. CDF for task 1 and 2 based on 10 simulations (The high position covariance means worse performance. We would like a CDF that gets closed to y-axis and reaches 1 as quickly as possible. The red line (Our method) shows good performance (gets closed to y-axis with high ratio) in all lines)

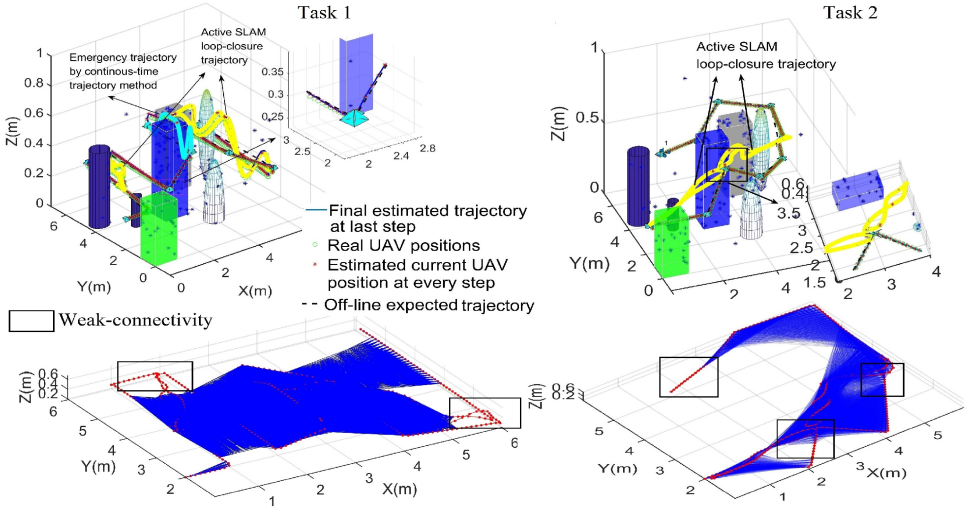


Fig. 4. Active SLAM trajectory, estimated pose graph results and relative measurements for task 1 and 2 in a small environment simulation

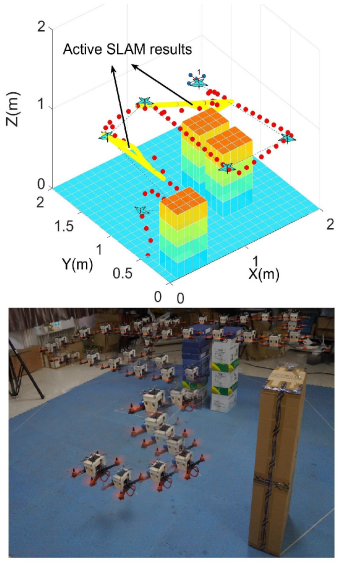


Fig. 5. On-line active SLAM (Composite trajectory)

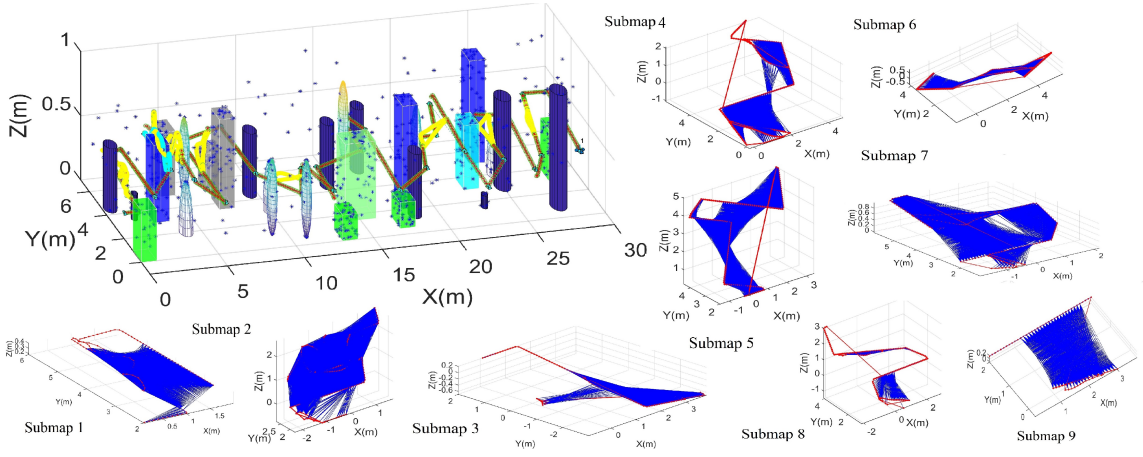


Fig. 6. Active SLAM, real and estimated trajectory results in 9 submaps

trajectories to reduce pose uncertainty. It uses the RRT-connect sampling-based path planning method and nonlinear continuous-time trajectory planning method based on key poses. In this process, the cheap metrics combining the weighted node degree and weighted tree-connectivity are used to choose the key poses and its corresponding best loop-closure trajectory. Simulations and experiment show that our method has good performance and can be run in real-time.

REFERENCES

[1] C. Leung, S. Huang, N. Kwok and G. Dissanayake, "Planning under uncertainty using model predictive control for information gathering," *Robotics and Autonomous Systems*, vol. 54, no. 11, 898-910, 2006.

[2] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains, *Articial Intelligence*, vol. 101, no. 1, pp. 99-134, 1998.

[3] V. Indelman, L. Carlone, and F. Dellaert, "Planning in the continuous domain: A generalized belief space approach for autonomous navigation in unknown environments," *The International Journal of Robotics*, vol. 34, no. 7, pp. 849-882, 2015.

[4] S. Pathak, A. Thomas, and V. Indelman, "A unified framework for data association aware robust belief space planning and perception," *The International Journal of Robotics Research*, vol. 37, no. 2-3, pp. 287-315, 2018.

[5] S. Wenhardt, B. Deutsch, E. Angelopoulou, and H. Niemann, "Active visual object reconstruction using D-, E-, and T-optimal next best views," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007, pp. 1-7.

[6] S. Liu, S.P. Chepuri, M. Fardad, E. Maazade, G. Leus, and P.K. Varshney, "Sensor selection for estimation with correlated measurement noise," *IEEE Transactions on Signal Processing*, vol. 64, no. 13, 2016, pp. 3509-3522.

[7] H. Carrillo, I. Reid, and J. A. Castellanos, "On the comparison of uncertainty criteria for active SLAM," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2012, pp. 2080-2087.

[8] M.L. Rodríguez-Arévalo, J. Neira, and J.A. Castellanos, "On the Importance of Uncertainty Representation in Active SLAM." *IEEE Transactions on Robotics*, vol. 34, no. 3, 2018, pp. 829-834.

[9] M. Keidar and G. A. Kaminka, "Efficient frontier detection for robot exploration," *The International Journal of Robotics Research*, vol. 33, no. 2, 2014, pp. 215-236.

[10] A. Kim and R.M. Eustice, "Active visual SLAM for robotic area coverage: Theory and experiment," *The International Journal of Robotics Research*, vol. 34, no. 4-5, 2015, pp. 457-475.

[11] J. Vallvé and J. A. Cetto, "Active pose SLAM with RRT*," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 1050-4729.

[12] I. Maurović, M. Seder, K. Lenac and I. Petrović, "Path Planning for Active SLAM Based on the D* Algorithm With Negative Edge Weights," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 99, no. 1, pp. 1-11, 2017.

[13] D. Kopitkov, and V. Indelman, "Computationally Efficient Belief Space Planning via Augmented Matrix Determinant Lemma and Reuse of Calculations," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, 2017, pp. 506-513.

[14] V. Indelman, "No Correlations Involved: Decision Making Under Uncertainty in a Conservative Sparse Information Space," *IEEE Robotics and Automation Letters*, vol. 1, no. 1, 2016, pp. 407-414.

[15] Y. Chen, S. Huang, and R. Fitch, "Efficient Active SLAM based on Submap Joining, Graph Topology and Convex Optimization," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5159-5166, 2018.

[16] D.M. Rosen, L. Carlone, A.S. Bandeira and J.J. Leonard, "SE-Sync: A certifiably correct algorithm for synchronization over the special Euclidean group," arXiv preprint arXiv:1612.07386, 2016.

[17] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g2o: A general framework for graph optimization," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3607-3613, 2011.

[18] K. Khosoussi, M. Giamou, G.S. Sukhatme, S. Huang, G. Dissanayake, and J.P. How, "Reliable graph topologies for SLAM," *The International Journal of Robotics Research*, 2018.

[19] Y. Chen, K. Khosoussi, S. Huang, L. Zhao, and G. Dissanayake, "Cramér-Rao bounds and optimal design metrics for pose-graph SLAM," 2018. https://github.com/cyb1212/A-submitted-paper/blob/master/main.pdf

[20] J.J. Kuffner and S.M. LaValle, "RRT-connect: An efficient approach to single-query path planning," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 995-1001, 2000.

[21] H. Oleynikova, M. Burri, Z. Taylor, J. Nieto, R. Siegwart, and E. Galceran, "Continuous-time trajectory optimization for online UAV replanning," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5332-5339, 2016.

[22] F. Gao, Y. Lin, and S. Shen, "Gradient-based online safe trajectory generation for quadrotor flight in complex environments," in *IEEE/RSJ International Conference on intelligent Robots and Systems (IROS)*, pp. 3681-3688, 2017.

[23] L. Zhao, S. Huang, and G. Dissanayake, "Linear SLAM: A linear solution to the feature-based and pose graph SLAM based on submap joining," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013, pp. 24-30.

[24] R. Mur-Artal and J.D. Tards, "Orb-slam2: An open-source slam system for monocular, stereo, and RGB-D cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255-1262, 2017.

[25] H. Carrillo, P. Dames, V. Kumar, and J.A. Castellanos, "Autonomous robotic exploration using a utility function based on Rényi's general theory of entropy," *Autonomous Robots*, vol. 42, no. 2, pp. 235-256, 2018.