Active SLAM for Mobile Robots with Area Coverage and Obstacle Avoidance

Yongbo Chen, Shoudong Huang and Robert Fitch

Abstract—**We present an active simultaneous localization and mapping (SLAM) framework for a mobile robot to obtain a collision-free trajectory with good performance in SLAM uncertainty reduction and in an area coverage task. Based on a model predictive control (MPC) framework, these two tasks are solved by the introduction of a control switching mechanism. For SLAM uncertainty reduction, graph topology is used to approximate the original problem as a constrained non-linear least-squares problem. A convex half-space representation is applied to relax non-convex spatial constraints that represent obstacle avoidance. Using convex relaxation, the problem is solved by a convex optimization method and a rounding procedure based on singular value decomposition (SVD). The area coverage task is addressed with a sequential quadratic programming (SQP) method. A submap joining approach, called Linear SLAM, is used to address the associated challenges of avoiding local minima, minimizing control switching, and potentially high computational cost. Finally, various simulations and experiments using an aerial robot are presented that verify the effectiveness of the proposed method, showing that our method produces a more accurate SLAM result and is more computationally efficient compared with multiple existing methods.**

Index Terms—**Active simultaneous localization and mapping (SLAM), model predictive control (MPC), submap joining, graph topology, convex optimization, eigenvalue bound.**

I. Introduction

In active SLAM, a robot needs to solve a decision-making problem where a collision-free trajectory is selected both to improve estimation accuracy and to perform other tasks such as coverage. This problem thus involves both SLAM and path planning components, and is considered to be one of the most challenging problems in mobile robotics [1]. There are many important applications of active SLAM in complex unknown environments where global navigation infrastructure is not available, such as indoors, in forests and orchards, and in other places where GPS signals can be denied.

Different from traditional path planning, active SLAM involves estimating the location of the robot as it moves and can consider environments that are fully or partially unknown. The estimation problem must be considered during the planning process as an integral part. Even though techniques for SLAM are maturing [2], [3], a good trajectory is needed to make sure the SLAM process does not fail.

Well-known active SLAM frameworks, including the MPC framework [4], [5] and the partially observable Markov decision process (POMDP) formalism [6], aim to choose the optimal future trajectory via metrics that strike a balance between visiting new areas for exploration and revisiting known spaces to introduce loop-closure. This entails three basic issues: (i) how to generate a set of available actions, (ii) how to evaluate the performance of the candidate actions, and (iii) how to select an optimal action.

Two main approaches to generate the set of available actions (trajectories) are dynamic-based methods and geometry-based methods. In dynamic-based methods, future candidate trajectories are obtained using robot models or potential information fields. The popular MPC and POMDP frameworks belong to this class. Geometry-based methods are inspired by path planning algorithms such as RRT* [7], [8] and D* [9]. However it is difficult to meet the dynamic constraints for practical systems in complex environments, which limits the practicability of these methods. Therefore, dynamic-based methods are more popular than the geometry-based methods.

A common way to evaluate individual actions in terms of estimation accuracy is based on the Theory of Optimal Experimental Design (TOED) [10] and includes the A-, D-, and E-opt criteria for computing the predicted Fisher information matrix (FIM). In [11], the authors show that the monotonicity of these metrics during the exploration phase is decided by the uncertainty representation of rotations. The *D-opt* optimality criterion, which maximizes the log-determinant of the FIM, is commonly used.

Since future observations are not available at planning time, certain assumptions need to be applied, such as zero-innovation measurement [4], in order to obtain future information matrices. In [12], the predicted measurements are treated as random variables and an expectation-maximization (EM) method is used to obtain the information matrix. Recently, planning in a conservative sparse information space is presented to improve the run-time performance of active SLAM [13].

Choosing an optimal full trajectory is a non-convex non-linear optimal control problem with multiple constraints. Direct methods are a common solution approach, where the problem is formulated in the discrete optimization domain. The size of the search space is reduced, but the solution is sub-optimal; an important ex-

Yongbo Chen, Shoudong Huang and Robert Fitch are with Faculty of Engineering and Information Technology, University of Technology, Sydney, Ultimo, NSW, 2007 Australia e-mail: Yongbo.Chen@student.uts.edu.au.

ample is frontier-based exploration [14]. Other methods aim to search for locally optimal policies in continuous belief space [15]. Finding optimal (or even sub-optimal) solutions for active SLAM remains challenging.

In this paper, we present a control mechanism that can switch between two modes. The first mode addresses the uncertainty minimization using an MPC method whose objective function is the *D-opt* optimality criterion. Based on graph topology, the *D-opt* MPC problem is solved by a convex relaxation method and the CVX toolbox. The second mode addresses area coverage with uncertainty using an SQP method. In order to improve computational efficiency in both of these modes, Linear SLAM [16] is applied. Finally, simulations and experiments with an aerial robot are presented to verify the running-time efficiency of this framework. Results also show that our method can generate more accurate SLAM results as compared with multiple current methods, while covering the area and satisfying all the constraints.

The main contributions of this work include:

- A novel formulation of the coverage problem that considers robot pose uncertainty.
- A method to exploit graph topology to simplify the objective function and maintain good performance.
- A method that applies convex relaxation and optimization to achieve good planning results.
- The use of submap planning and estimation, fast objective function computation, and fast covariance recovery to improve planning efficiency.

This paper is an extended and revised version of our recent conference paper [17]. Here, we expand the scope of the problem to allow the presence of obstacles, and present several important technical improvements, new analysis, and additional experimental evaluation. Instead of generating the entire covariance matrix, a fast covariance recovery method is used to improve run-time performance. A new method to compute the objective function is used that improves computational performance by exploiting the sparseness of the FIM. A novel theorem about the relationship between the eigenvalues of the FIM of the submaps and those of the joined global map is proved and applied. The spectrahedral description of conv $SO(2)$, which can be extended to the 3D case, is used to replace the original relaxation method, which was only suitable for the 2D case. The rounding method, which is only suitable for trigonometric functions, is replaced by an SVD-based method. Finally, we include new simulations, new comparisons with existing methods and an additional experiment.

II. Models and problem statement

A. Kinematic model and sensor model

Assuming constant altitude and zero pitch/roll angle in a 2D environment, the kinematic equations of an unmanned aerial vehicle (UAV)[1] moving at constant velocity v are:

$$\begin{aligned} \widehat{\boldsymbol{v}} &= \boldsymbol{R}_k^\top(\boldsymbol{x}_{k+1}^v - \boldsymbol{x}_k^v) + (\delta x_k \ \delta y_k)^\top, \\ \omega_k &= (\theta_{k+1} - \theta_k)/\Delta t + \delta\omega_k \ , \end{aligned} \tag{1}$$

where $\widehat{\boldsymbol{v}} = (v\Delta t \ \ 0)^\top$, Δt is the discrete time interval, $(\boldsymbol{x}_k^{v\top}, \theta_k) = (x_k, y_k, \theta_k)$ is the UAV pose at the k-th step, ω_k is the angular velocity at the k-th step, $u_k = \omega_k$ is the control input of our active SLAM problem and is limited by $|\omega_k| \le C_u$ to meet the turning radius constraint, δx_k, δy_k and $\delta\omega_k$ are noise terms of the coordinates and angular velocity, $\boldsymbol{R}_k = \begin{pmatrix} \cos(\theta_k) & -\sin(\theta_k) \\ \sin(\theta_k) & \cos(\theta_k) \end{pmatrix} \in SO(2)$ is the rotation matrix of the k-th pose, and $SO(n)$, $n = 2, 3, \cdots$ represents the special orthogonal group.

We assume that there are a number of natural features located on the ground, and that a downward looking sensor such as a camera is installed at the bottom of the UAV. On the 2D plane, this sensor can be regarded as an omnidirectional sensor with range R_s. A relative position sensor model is used,

$$\boldsymbol{z}_k^{fi} = \boldsymbol{R}_k^\top(\boldsymbol{x}^{fi} - \boldsymbol{x}_k^v) + (w_x^k \ w_y^k)^\top, \tag{2}$$

where $\boldsymbol{z}_k^{fi}$ is the observed value of the i-th feature at the k-th step, $\boldsymbol{x}^{fi}$ is the coordinate of the i-th feature, and w_x^k, w_y^k are the sensor noises in the x and y axes.

B. Problem statement

We assume, in a partly unknown space, that there are many point features with unknown locations which can be observed when they are within the UAV sensor range. The boundaries of the area of interest to be covered and of any no-fly zones are known beforehand. An example scene is shown in Fig. 1. The active SLAM problem is to select a series of control inputs within a given time period to generate a safe collision-free trajectory and cover as much area as possible, and also to minimize the estimation uncertainty. The exact allowed uncertainty level of the mapped features and the time limitation of the task are defined based on the given scenario.

One example application for this problem is an agriculture UAV that performs tasks in a complex greenhouse environment, as shown in Fig. 2. Many agriculture tasks, like irrigation, pest control, and fertilization, involve the coverage task and are performed in an indoor environment without global location information from GPS. In order to avoid the large drift and task failure that can be caused by uncertainty growth in the SLAM result, active SLAM is advantageous. A rough floor plan of the greenhouse is in general available in this task and provides prior information about obstacles. The mapped features may be defined as normal visual features from images (such as ORB features and SIFT features) and any detected objects (such as fruit and plants).

III. MPC framework

An overview of the solution framework for our active SLAM problem is described in this section.

[1]Our method is presented for mobile robots. The UAV is just one example and is used as our experimental platform in Sec. VI.

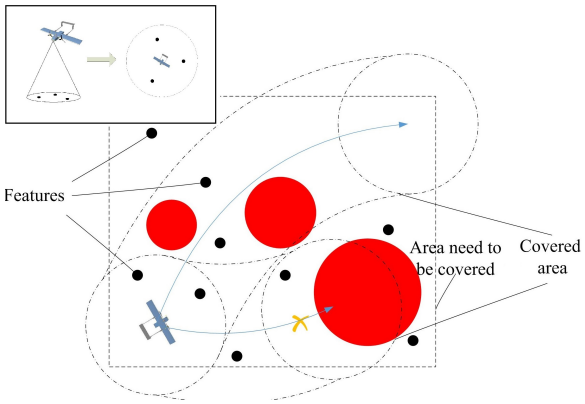

Fig. 1. SLAM task and coverage task

Fig. 2. Agriculture UAV in a greenhouse [18]

A. Motion planning for uncertainty minimization in SLAM solution

Using an MPC framework similar to [12], the objective function of the uncertainty minimization task is based on the generalized belief at the L-th planning step:

$$\begin{aligned} f_a(u_{k:k+L-1}) &= \hat{f_a}(gb(\boldsymbol{X}_{k+L})) = -\log\left(\det(\boldsymbol{\mathcal{I}}_{k+L}(\boldsymbol{X}_{k+L}^{opt}))\right) \\ \boldsymbol{X}_{k+L}^{opt} &= \underset{\boldsymbol{X}_{k+L}}{\operatorname{argmin}} \ -\log(p(\boldsymbol{X}_{k+L}|\boldsymbol{Z}_{1:k}, u_{0:k+L-1}, \boldsymbol{Z}_{k+1:k+L}), \end{aligned} \tag{3}$$

where $p(\star|\bullet)$ means the probability of $\star$ under the condition $\bullet$. Control inputs $u_0, \ \cdots, u_{k+L-1}$ are denoted by $u_{0:k+L-1}$, $\boldsymbol{X}_{k+L}$ is the state vector including the $k+L-1$ poses and the coordinates of the features, $gb(\boldsymbol{X}_{k+L})$ is the Gaussian belief of $\boldsymbol{X}_{k+L}$, $gb(\boldsymbol{X}_{k+L}) \approx \mathcal{N}(\boldsymbol{X}_{k+L}^{opt}, \boldsymbol{\mathcal{I}}_{k+L}(\boldsymbol{X}_{k+L}^{opt})^{-1})$, $\boldsymbol{\mathcal{I}}_{k+L}(\boldsymbol{X}_{k+L}^{opt})$ denotes the FIM corresponding to state $\boldsymbol{X}_{k+L}^{opt}$. Measurements $\boldsymbol{Z}_{1:k}$ are actual measurement from the sensors and $\boldsymbol{Z}_{k+1:k+L}$ are predicted measurements based on prior information.

Using the *D-opt* optimality criterion, optimization problem (3) in an obstacle environment can be simplified to the following problem (4) based on Bayes' theorem, zero-innovation assumption, and the Markov property:

$$\begin{aligned} \min_{u_{k:k+L-1}} \quad & f_a(u_{k:k+L-1}) = -\log\left(\det(\boldsymbol{\mathcal{I}}_{k+L}(\boldsymbol{X}_{k+L}^{opt}))\right) \\ & \boldsymbol{X}_{k+L}^{opt} = [\ \boldsymbol{X}_k^{opt\top} \quad \boldsymbol{x}_{k+1}^{opt\top} \quad \cdots \quad \boldsymbol{x}_{k+L}^{opt\top}\ ]^\top \\ & |u_{k+i}| \le C_u, \quad i = 0, \cdots, L-1 \\ & \{\boldsymbol{x}_k^{opt}, \cdots, \boldsymbol{x}_{k+L-1}^{opt}\} \notin \mathit{Obstacle}, \end{aligned} \tag{4}$$

where *Obstacle* is the set of no-fly zones, $\boldsymbol{x}_{k+i}^{opt} = f_v(\boldsymbol{x}_{k+i-1}^{opt}, u_{k+i-1})$ satisfies the motion equation without uncertainty shown in (1) and $\boldsymbol{x}_{k+i}^{opt}$ is the predicted pose at step $k+i$. For further details, please refer to the supplementary material [20].

B. Coverage task under uncertainty in MPC framework

When the UAV trajectory is perfectly known, the area covered by the on-board sensor can be obtained exactly. However, we can only obtain an estimated pose with uncertainty based on the SLAM result. In this paper, we consider the worst case (99% confidence ellipse) of the robot pose, which leads to a smaller believable coverage area. The specific equations used to compute the covered area based on uncertain pose are given in [17].

C. No-fly zone scale expansion under uncertainty

This work allows for the presence of circular no-fly zones in the area to be covered. Before performing the active SLAM task, positions and sizes of these zones are known. However, without knowing the actual global UAV pose, the scale of the no-fly zone needs to be enlarged in order to avoid collision. This enlargement is performed based on the uncertainty of the UAV pose; its operation is similar to that described in [17].

D. Solution framework using fast covariance recovery

Considering the highly-nonlinear and non-analytical functions involved, we apply the SQP approach to obtain a sub-optimal trajectory for the coverage problem. Then, for the uncertainty minimization problem, we apply the method presented in Sec. IV. Because of the significant difference of the physical meanings in the two objective functions, we use a switching mechanism [17]:

$$u_r = \begin{cases} u_c, & \mathit{Index}_1 \le C_1^{index}, \\ c_c u_c + c_a u_a, & \mathit{Index}_1 \in (C_1^{index}, C_2^{index}), \\ u_a, & \mathit{Index}_1 \ge C_2^{index}, \end{cases} \tag{5}$$

where $\mathit{Index}_1 = \sum_{i=1}^{N_f}(\lambda_i^{fx} + \lambda_i^{fy})$, u_c and u_a are respectively the first control inputs of the coverage task and the *D-opt* MPC solution, c_a and c_c are weights. C_1^{index} and C_2^{index} are two switching indices, λ_i^{fx} and λ_i^{fy} are the eigenvalues of the covariance matrices of the i-th features at the x and y axes, and N_f is the number of detected features.

We use the eigenvalues λ_i^{fx} and λ_i^{fy}, which are decided by the block diagonal-elements of the covariance matrix $\boldsymbol{C}$, as the index of uncertainty. The covariance matrix is dense and thus difficult to be stored and updated, and this motivates our use of the FIM. It is well known that the FIM $\boldsymbol{\mathcal{I}}(\boldsymbol{X})$ and the covariance matrix $\boldsymbol{C}$ obey $\boldsymbol{C} = \boldsymbol{\mathcal{I}}(\boldsymbol{X})^{-1}$. However, computing the inverse of a large matrix is very time consuming. Fortunately, we only need to consider the block diagonal elements corresponding to the i-th features. Using the recursive formula [19], the specific elements from the covariance matrix can be efficiently calculated by using the square root information matrix $\boldsymbol{H}$ ($\boldsymbol{\mathcal{I}}(\boldsymbol{X}) = \boldsymbol{H}^\top\boldsymbol{H}$):

$$\begin{aligned} \boldsymbol{C}_{ii} &= \frac{1}{\boldsymbol{H}_{ii}}\left(\frac{1}{\boldsymbol{H}_{ii}} - \sum\nolimits_{k=i+1, \boldsymbol{H}_{ik}\neq 0}^{n} \boldsymbol{H}_{ik}\boldsymbol{C}_{ki}\right), \\ \boldsymbol{C}_{ij} &= \frac{-1}{\boldsymbol{H}_{ii}}\left(\sum\nolimits_{k=i+1, \boldsymbol{H}_{ik}\neq 0}^{j} \boldsymbol{H}_{ik}\boldsymbol{C}_{kj} + \sum\nolimits_{k=j+1, \boldsymbol{H}_{ik}\neq 0}^{n} \boldsymbol{H}_{ik}\boldsymbol{C}_{jk}\right), \end{aligned} \tag{6}$$

where $\star_{ij}$ means the (i, j)-th element of the matrix $\star$. The operations, which are similar to the fast recovery method [19], use the approximate minimum degree permutation (AMDP) and a hash table as shown in Algorithm 1.

Algorithm 1 Fast covariance recovery for features

Input: Fisher information matrix $\mathcal{I}(\boldsymbol{X})$
Output: The covariance block corresponding to the features
1: **Step 1**: Create $\boldsymbol{F}' = \mathcal{I}(\boldsymbol{X})$, separate its column and row corresponding to the features and poses, and get $\boldsymbol{F}'_{reduce}$ matrix for poses and $\boldsymbol{F}^*_{reduce}$ matrix for features;
2: **Step 2**: $\boldsymbol{p}_{order} = AMDP(\boldsymbol{F}'_{reduce})$;
3: **Step 3**: Obtain $\boldsymbol{F}$ by combining $\boldsymbol{F}'_{reduce}(\boldsymbol{p}_{order}, \boldsymbol{p}_{order})$ with $\boldsymbol{F}^*_{reduce}$ matrix ▷ It is noted that $\boldsymbol{F}^*_{reduce}$ matrix locate at the end of the combined matrix;
4: **Step 4**: $\boldsymbol{H} = SC(\boldsymbol{F})$ ▷ Sparse Cholesky decomposition;
5: **Step 5**: Covariance recovery by (6) using the hash table;

The eigenvalues resulting from the above algorithm are the same as those resulting from computing the inverse of the entire FIM, but the computational complexity is much lower.

IV. Solution to the uncertainty minimization problem

In this section, we present our solution to the *D-opt* MPC problem for improving SLAM accuracy. Our solution exploits the graph structure of 2D feature-based SLAM and applies convex relaxation.

A. 2D feature-based SLAM

Assumption 1: Isotropic Noise. We assume that noise satisfies: $(\delta x_k \ \ \delta y_k)^\top \sim \mathcal{N}\left(\boldsymbol{0}, \sigma_v(k)^2 \boldsymbol{I}^{2\times 2}\right)$, $(w_x^k \ \ w_y^k)^\top \sim \mathcal{N}\left(\boldsymbol{0}, \sigma_f(k)^2 \boldsymbol{I}^{2\times 2}\right)$, $\delta\omega_k \sim \mathcal{N}\left(0, \sigma_\theta(k)^2\right)$.

The state vector is rewritten as $\boldsymbol{X} = (\boldsymbol{p}^\top \ \boldsymbol{\theta}^\top)^\top$ where $\boldsymbol{p}$ represents the coordinates of the UAV and the features, and $\boldsymbol{\theta}$ represents the orientation angles of the UAV. The measurement vector corresponding to translation and rotation is $\boldsymbol{z} = (\boldsymbol{z}_p^\top \ \boldsymbol{z}_\theta^\top)^\top$. Then, the measurement model, $\boldsymbol{z} = h(\boldsymbol{X}) + \boldsymbol{\delta}$, can be expressed as:

$$\begin{bmatrix} \boldsymbol{z}_p \\ \boldsymbol{z}_\theta \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}^\top (\boldsymbol{A}_g \otimes \boldsymbol{I}^{2\times 2})^\top & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{A}_p^\top \end{bmatrix} \begin{bmatrix} \boldsymbol{p} \\ \boldsymbol{\theta} \end{bmatrix} + \boldsymbol{\delta}, \tag{7}$$

where $\boldsymbol{A}_p$ and $\boldsymbol{A}_g$ respectively are the reduced incident matrices corresponding to the pose graph and the whole graph, $\otimes$ is the Kronecker product, $\boldsymbol{R} \triangleq \mathrm{diag}(\boldsymbol{R}_1, \boldsymbol{R}_2, ..., \boldsymbol{R}_m)$, m is the number of measurements, $\boldsymbol{R}_i$ is the UAV rotation matrix corresponding to the i-th observation, $\boldsymbol{\delta} \sim \mathcal{N}(\boldsymbol{0}, \ \boldsymbol{\Sigma})$, $\boldsymbol{\Sigma} = \mathrm{diag}(\boldsymbol{\Sigma}_p \otimes \boldsymbol{I}^{2\times 2}, \boldsymbol{\Sigma}_\theta)$, where $\boldsymbol{\Sigma}_p = \mathrm{diag}(\sigma_v(1)^2, \cdots, \sigma_v(k)^2, \sigma_f(1)^2, \cdots, \sigma_f(m-k)^2)$, and $\boldsymbol{\Sigma}_\theta = \mathrm{diag}(\sigma_\theta(1)^2, \cdots, \sigma_\theta(k)^2)$.

The feature-based SLAM problem estimates UAV positions $\boldsymbol{p}$ and orientation angles $\boldsymbol{\theta}$ by:

$$\min_{\boldsymbol{X}} \|\boldsymbol{z} - h(\boldsymbol{X})\|^2_{\boldsymbol{\Sigma}^{-1}}, \tag{8}$$

where $\|\star\|^2_\bullet = \star^\top \bullet \star$, $\star$ is a vector, and $\bullet$ is a positive definite matrix. Its iteration update equation using the Gauss-Newton (GN) method is:

$$\boldsymbol{X}_{k+1} = \boldsymbol{X}_k + (\boldsymbol{J}(\boldsymbol{X}_k)^\top \boldsymbol{\Sigma}^{-1} \boldsymbol{J}(\boldsymbol{X}_k))^{-1} \boldsymbol{J}(\boldsymbol{X}_k)^\top \boldsymbol{\Sigma}^{-1} (\boldsymbol{z} - h(\boldsymbol{X}_k)), \tag{9}$$

where $\boldsymbol{J}(\boldsymbol{X}) \triangleq \frac{\partial h(\boldsymbol{X})}{\partial \boldsymbol{X}}$ is the Jacobian matrix of the measurement model, and $\boldsymbol{X}_{k+1}$ and $\boldsymbol{X}_k$ are the $k+1$-th and k-th generation state vectors, updated until convergence.

Based on $\mathcal{I}(\boldsymbol{X}) = \boldsymbol{J}(\boldsymbol{X})^\top \boldsymbol{\Sigma}^{-1} \boldsymbol{J}(\boldsymbol{X})$, using the measurement model (7), we can get the FIM:

$$\mathcal{I}(\boldsymbol{X}) = \begin{bmatrix} \boldsymbol{L}^g_{w_p} \otimes \boldsymbol{I}^{2\times 2} & \boldsymbol{A}^g_{w_p} \otimes \boldsymbol{I}^{2\times 2} \boldsymbol{\Gamma} \triangle_{w_p} \\ (\boldsymbol{A}^g_{w_p} \otimes \boldsymbol{I}^{2\times 2} \boldsymbol{\Gamma} \triangle_{w_p})^\top & \triangle^\top_{w_p} \triangle_{w_p} + \boldsymbol{L}^p_{w_\theta} \end{bmatrix}, \tag{10}$$

where $\boldsymbol{L}^g_{w_p} = \boldsymbol{A}_g \boldsymbol{\Sigma}_p^{-1} \boldsymbol{A}_g^\top$ and $\boldsymbol{L}^p_{w_\theta} = \boldsymbol{A}_p \boldsymbol{\Sigma}_\theta^{-1} \boldsymbol{A}_p^\top$ are respectively the reduced weighted Laplacian matrices of the entire graph and the pose graph. Matrices $\boldsymbol{A}^g_{w_p} = \boldsymbol{A}_g \boldsymbol{\Sigma}_p^{-\frac{1}{2}}$ and $\boldsymbol{A}^p_{w_\theta} = \boldsymbol{A}_p \boldsymbol{\Sigma}_\theta^{-\frac{1}{2}}$ are the reduced weighted incidence matrices, $\boldsymbol{\Gamma}$ is given by $\boldsymbol{\Gamma} \triangleq \boldsymbol{I}^{m\times m} \otimes \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$, $\triangle_{w_p} = \triangle \boldsymbol{\Sigma}_p^{-\frac{1}{2}}$, and $\triangle \in \mathbb{R}^{2m\times n}$ is shown in [20].

Then, similar to [21], we can obtain the lower bounds of the log-determinant function of the FIM for 2D feature-based SLAM (a proof is provided in [20]):

$$\log(\det(\mathcal{I}(\boldsymbol{X}))) \geq 2\log(\det(\boldsymbol{L}^g_{w_p})) + \log(\det(\boldsymbol{L}^p_{w_\theta})). \tag{11}$$

B. Discussion of the D-opt objective function

In most cases, the D-optimality criterion approaches its lower bound. In order to verify this result numerically, an experimental evaluation is provided in the supplemental material [20]. Based on [21], $\det(\boldsymbol{L}^g_{w_p})$ and $\det(\boldsymbol{L}^p_{w_\theta})$ are equal to the weighted number of the spanning trees of the entire graph and the pose graph, respectively. Thus, the values of these determinants are directly related to the number of the cycles in the two graphs. Hence, we only need to consider $\det(\boldsymbol{L}^p_{w_\theta})$ because the pose graph is acyclic in typical feature-based SLAM.

Because we lack information from non-observed areas, the only opportunity to improve measurement probability is when the UAV observes mapped features within its sensor range R_s. This distance $\|\boldsymbol{x}^{opt}_{k+i} - \boldsymbol{x}^{opt}_{fj}\|$ is thus minimized, where $\boldsymbol{x}^{opt}_{fj}$ is the estimated coordinate of the observed feature j. Although every new measurement of an observed feature will introduce at least one new cycle, the benefit obtained by re-observing features diminishes; for frequently observed features, adding a new measurement will not contribute substantially to a decrease in estimate uncertainty. We introduce weight c_j to modulate the value of observing any given feature.

Hence, we use a new least-squares formulation to replace the original objective function $-\log(\det(\mathcal{I}_{k+L}(\boldsymbol{X}^{opt}_{k+L})))$. The main purpose of the new formulation is to encourage the robot to move closer to the most important features to add measurements with larger information value. This new formulation is:

$$\sum\nolimits_{i=1}^{L} \sum\nolimits_{j=1}^{N_f} c_j \|\boldsymbol{x}^{opt}_{k+i} - \boldsymbol{x}^{opt}_{fj}\|^2, \tag{12}$$

where $c_j = \frac{\log(\det(\widehat{\boldsymbol{L}}^g_{w_p}(j))) - \log(\det(\boldsymbol{L}^g_{w_p}))}{\sum_{j=1}^{N_f}(\log(\det(\widehat{\boldsymbol{L}}^g_{w_p}(j))) - \log(\det(\boldsymbol{L}^g_{w_p})))}$. $\widehat{\boldsymbol{L}}^g_{w_p}(j) = \boldsymbol{A}_g(j) \boldsymbol{\Sigma}^*_p(j)^{-1} \boldsymbol{A}_g(j)^\top$, where $\boldsymbol{\Sigma}^*_p(j) = \mathrm{diag}(\sigma_v(1)^2,$

$\cdots, \sigma_v(k)^2, \sigma_f(1)^2, \cdots, \sigma_f(m-k)^2, \sigma_f^{j^2}) \otimes \boldsymbol{I}^{2\times 2}$ and $\boldsymbol{A}_g(j) = (\boldsymbol{A}_g, \boldsymbol{a}_j)$ is the new weighted Laplacian matrix of the entire graph of $\boldsymbol{X}_{k+L}^{opt}$ created by adding a virtual measurement between the j-th mapped feature and the predicted future pose $\boldsymbol{x}_{k+1}^{opt}$, with vector $\boldsymbol{a}_j$ a column vector that has only two non-zero elements (-1 is located at the k-th element, and 1 is located at the $k+j$-th element).

The coefficient c_j is defined to simulate a future one-step measurement and quantify the expected information increment of the different features. For a large-scale SLAM problem, even though the weighted Laplacian matrix $\widehat{\boldsymbol{L}}_{w_p}^{g}(j)$ has fewer dimensions and a sparser structure than the FIM, to compute its log-determinant is still time-consuming. We present a fast algorithm using the "order re-use" strategy and the sparse Cholesky decomposition to compute c_j, as shown in Algorithm 2.

Algorithm 2 Compute Coefficient c_j

Input: Two weighted Laplacian matrices $\widehat{\boldsymbol{L}}_{w_p}^{g}(j)$ and $\boldsymbol{L}_{w_p}^{g}$.
Output: Coefficient c_j

1: $P \leftarrow AMDP(\boldsymbol{L}_{w_p}^{g})$
2: $P' \leftarrow [P; size(P)+1]$ ▷ e.g., Reuse order for $\widehat{L}_{w_p}^{g}(j)$
3: $C_1 \leftarrow SC\left(\boldsymbol{L}_{w_p}^{g}(P,P)\right)$, $C_2 \leftarrow SC\left(\widehat{\boldsymbol{L}}_{w_p}^{g}(j)(P',P')\right)$
4: $\log(\det(\boldsymbol{L}_{w_p}^{g})) \leftarrow 2\cdot 3\cdot \sum_i \log(C_1)_{i,i}$
5: $\log(\det(\widehat{\boldsymbol{L}}_{w_p}^{g}(j))) \leftarrow 2\cdot 3\cdot \sum_i \log(C_2)_{i,i}$
6: **return** $c_j \leftarrow \frac{\log(\det(\widehat{\boldsymbol{L}}_{w_p}^{g}(j)))-\log(\det(\boldsymbol{L}_{w_p}^{g}))}{\sum_{j=1}^{m}(\log(\det(\widehat{\boldsymbol{L}}_{w_p}^{g}(j)))-\log(\det(\boldsymbol{L}_{w_p}^{g})))}$

Algorithm 2 can compute c_j much faster than computing the log-determinant function of the dense matrix.

C. Convex relaxation and rounding

In this section, we discuss a spectrahedral description method and a no-fly zone half-space representation to produce a convex problem. The original uncertainty minimization problem can be rewritten as:

$$\begin{aligned} \min_{u_{k:k+L-1}} \quad & \text{Eq.(12)} \\ \text{s.t.} \quad & \boldsymbol{x}_{k+i+1}^{opt} = \boldsymbol{x}_{k+i}^{opt} + \boldsymbol{R}_{k+i}\widehat{\boldsymbol{v}} \\ & \theta_{k+i+1}^{opt} = \theta_{k+i}^{opt} + u_{k+i}\Delta T \\ & |u_{k+i}| \le C_u, i = 0, ..., L-1 \\ & \{\boldsymbol{x}_k^{opt}, \cdots, \boldsymbol{x}_{k+L-1}^{opt}\} \notin Obstacle. \end{aligned} \tag{13}$$

This problem is not convex because of the rotation matrix $\boldsymbol{R}_{k+i} \in SO(2)$ and the collision-free constraints. In order to use a convex optimization method, we need a spectrahedral description of conv $SO(2)$ and introduce a no-fly zone half-space representation.

1) Convex relaxation of rotation matrix:

Theorem 1 (Spectrahedral description [22]): The convex hull of $SO(n)$ is a spectrahedron for all $n \in \mathbb{N}$.

$$\begin{aligned} \text{conv } SO(n) = \{\boldsymbol{X} \in \mathbb{R}^{n\times n} : \begin{bmatrix} \boldsymbol{0} & \boldsymbol{X} \\ \boldsymbol{X}^\top & \boldsymbol{0} \end{bmatrix} \\ \preceq \boldsymbol{I}^{2n\times 2n}, \sum_{i,j=1}^{n} \boldsymbol{A}_{ij}[\hat{\boldsymbol{R}}\boldsymbol{X}]_{ij} \preceq (n-2)\boldsymbol{I}^{2^{n-1}\times 2^{n-1}}\}, \end{aligned} \tag{14}$$

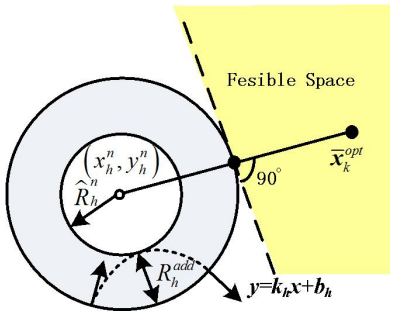

Fig. 3. Illustration of a convex half-space representation method

where $\boldsymbol{A}_{ij}$ is a collection of symmetric $2^{n-1}\times 2^{n-1}$ matrices, and $\hat{\boldsymbol{R}} = \text{diag}(1, 1, \ldots, 1, -1)^{n\times n}$. In the special case where $n = 2$, its representation can be simplified to:

$$\text{conv } SO(2) = \left\{ \begin{bmatrix} c & -s \\ s & c \end{bmatrix} \in \mathbb{R}^{2\times 2} : \begin{bmatrix} 1+c & s \\ s & 1-c \end{bmatrix} \succeq 0 \right\}. \tag{15}$$

Based on Theorem 1, let $c = u_{i,1}^*$ and $s = u_{i,2}^*$. The first constraint can then be written as:

$$\begin{aligned} x_{k+i+1}^{opt} &= x_{k+i}^{opt} + vu_{i,1}^*\Delta t, \\ y_{k+i+1}^{opt} &= y_{k+i}^{opt} + vu_{i,2}^*\Delta t, \end{aligned} \quad \begin{bmatrix} 1+u_{i,1}^* & u_{i,2}^* \\ u_{i,2}^* & 1-u_{i,1}^* \end{bmatrix} \succeq 0. \tag{16}$$

The constraints $\theta_{k+i+1}^{opt} = \theta_{k+i}^{opt} + u_{k+i}\Delta t$ and $|u_{k+i}| \le C_u$ can be written in a unified form:

$$|\theta_{k+i+1}^{opt} - \theta_{k+i}^{opt}| \le C_u\Delta t, \ i = 0, ..., L-1. \tag{17}$$

Based on the mean value theorem, we apply new constraints to relax the constraint (17):

$$|u_{i+1,1}^* - u_{i,1}^*| \le C_u\Delta t, \ |u_{i+1,2}^* - u_{i,2}^*| \le C_u\Delta t. \tag{18}$$

This convex relaxation method based on the spectrahedral description is not limited to the 2D case. It can be easily extended for 3D trajectory planning.

2) Non-convex no-fly zone constraints: The no-fly zone is a circular area, which is a convex region. Its complementary set in the whole space is a non-convex region. The half-space representation, firstly presented by [23], is a good way to approximate this non-convex feasible region. We use a similar idea and add an emergency mode to improve its robustness in the MPC framework.

As shown in Fig. 3, the no-fly zone is first enlarged by an emergency safety range R_h^{add}, and then a tangent line $y = k_h x + b_h$ is obtained according to the geometric relationship between the coordinate of the center point of the no-fly zone (x_h^n, y_h^n), the UAV position $\boldsymbol{x}_k^{opt}$, and the radius $R_h^{all} = R_h^{add} + \widehat{R}_h^n$. We use the following inequalities to limit the feasible space of the predicted future poses:

$$\begin{aligned} &(k_h x_{k+i}^{opt} + b_h - y_{k+i}^{opt})(k_h x_h^n + b_h - y_h^n) < 0, \\ &h = 1, 2, \cdots, N_o; \ i = 1, 2, \cdots, L, \end{aligned} \tag{19}$$

where N_o is the number of no-fly zones in the area.

Even if the UAV $\boldsymbol{x}_k^{opt}$ is located in free space, because of control limitations and the rounding operation (26),

future poses $\{\boldsymbol{x}_{k+1}^{opt}, \cdots, \boldsymbol{x}_{k+L}^{opt}\}$ may be fully or partly located on the other side of the tangent line $y = k_h x + b_h$. We introduce the emergency safety range R_h^{add} to avoid these special situations. When predicted future poses $\boldsymbol{x}_{k+i}^{opt}, i = 1, 2, \cdots, L$ fall within the emergency safety annulus (meaning the convex method is insoluble), the optimization method needs to be stopped and replaced by a sharp curve at the maximum control input C_u and a direction that steers the UAV to leave the emergency safety annulus as quickly as possible. The extreme case is when the UAV is oriented towards the center point of the no-fly zone (x_h^n, y_h^n) and its position lies on the boundary of the annulus. This case requires the maximum R_h^{add}, which can be computed by: $R_h^{add} = \sqrt{2\widehat{R}_h^n v/C_u + \widehat{R}_h^{n2}} - \widehat{R}_h^n$.

3) New convex problem: The convex problem for minimising uncertainty taking into account the no-fly zone constraints can now be defined as:

$$\min_{u_{k:k+L-1}} \text{Eq.(12)} \quad \text{s.t. Eq.(16), Eq.(18), Eq.(19).} \tag{20}$$

This problem can be solved using interior-point methods, which have already been integrated in many convex toolboxes such as the lsqlin function in MATLAB, CVX (used in this paper), and CVXGEN. Due to the problem's convexity, the solution obtained is globally optimal.

Because of the relaxation operations, the solution to (20) may not be suitable for the original problem. In this section, we show how to generate a feasible solution. Suppose the solution to the new convex problem (20) is: $\boldsymbol{v}_1^* = (\widehat{u}_{0,1}, \widehat{u}_{1,1}, ..., \widehat{u}_{L-1,1})^\top, \boldsymbol{v}_2^* = (\widehat{u}_{0,2}, \widehat{u}_{1,2}, ..., \widehat{u}_{L-1,2})^\top$.

Its corresponding solution in conv $SO(2)$ is:

$$\boldsymbol{R}_1^*, \cdots, \boldsymbol{R}_i^* = \begin{bmatrix} \widehat{u}_{i-1,1} & -\widehat{u}_{i-1,2} \\ \widehat{u}_{i-1,2} & \widehat{u}_{i-1,1} \end{bmatrix}, \cdots, \boldsymbol{R}_L^*. \tag{21}$$

The rounding procedure is to find the nearest valid rotation in $SO(2)$ to replace the solution shown in (21). The problem for conv $SO(n)$ has been defined [24]:

$$\begin{aligned} &\pi_R : \text{conv } SO(n) \rightarrow SO(n), \\ &\pi_R(\boldsymbol{R}_i^*) \in \min_{\widehat{\boldsymbol{R}}_i \in SO(n)} \|\widehat{\boldsymbol{R}}_i - \boldsymbol{R}_i^*\|_F, \end{aligned} \tag{22}$$

where $\| \bullet \|_F$ is the Frobenius norm. The solution to problem (22) is based on the SVD of $\boldsymbol{R}_i^*$: Suppose the SVD of $\boldsymbol{R}_i^*$ is $\boldsymbol{R}_i^* = \boldsymbol{U} * \boldsymbol{S} * \boldsymbol{V}^\top$, where $\boldsymbol{S}$ is a diagonal matrix, $\boldsymbol{U}$ and $\boldsymbol{V}$ are unitary matrices, then the solution to (22) is given by:

$$\widehat{\boldsymbol{R}}_i = \boldsymbol{U} * \boldsymbol{S}' * \boldsymbol{V}^\top, \ \boldsymbol{S}' = \begin{cases} \boldsymbol{I}^{n\times n} & if \ \det(\boldsymbol{U})\det(\boldsymbol{V}) = 1 \\ \text{diag}\{1, \cdots, 1, -1\}^{n\times n} & if \ \det(\boldsymbol{U})\det(\boldsymbol{V}) = -1. \end{cases} \tag{23}$$

For the $SO(2)$ case, there is an equivalent simple form:

$$\widehat{\boldsymbol{R}}_i = \begin{bmatrix} \frac{\widehat{u}_{i-1,1}}{\sqrt{\widehat{u}_{i-1,1}^2 + \widehat{u}_{i-1,2}^2}} & -\frac{\widehat{u}_{i-1,2}}{\sqrt{\widehat{u}_{i-1,1}^2 + \widehat{u}_{i-1,2}^2}} \\ \frac{\widehat{u}_{i-1,2}}{\sqrt{\widehat{u}_{i-1,1}^2 + \widehat{u}_{i-1,2}^2}} & \frac{\widehat{u}_{i-1,1}}{\sqrt{\widehat{u}_{i-1,1}^2 + \widehat{u}_{i-1,2}^2}} \end{bmatrix}. \tag{24}$$

Based on the relative difference of the two rotation matrices $\widehat{\boldsymbol{R}}_{i+1}$ and $\widehat{\boldsymbol{R}}_i$, we can obtain feasible control inputs by using the equation:

$$u_{k+i} = f_{round}\left(f_{mr}\left(\widehat{\boldsymbol{R}}_{i+1}^\top \widehat{\boldsymbol{R}}_i\right)\right), \tag{25}$$

where $f_{mr}(\star)$ means the minimal representation of $SO(n)$ corresponding to control inputs. For the 2D case, $f_{mr}\left(\widehat{\boldsymbol{R}}_{i+1}^\top \widehat{\boldsymbol{R}}_i\right) = \text{atan2}(\widehat{u}_{i+1,2}, \widehat{u}_{i+1,1}) - \text{atan2}(\widehat{u}_{i,2}, \widehat{u}_{i,1})$, and $f_{round}(\star)$ is a piecewise rounding function:

$$f_{round}(\star) = \begin{cases} -C_u, & \star/\Delta t < -C_u, \\ \star/\Delta t, & -C_u \le \star/\Delta t \le C_u, \\ C_u, & \star/\Delta t > C_u. \end{cases} \tag{26}$$

Thus, we arrive at a solution $(u_k, u_{k+1}, .., u_{k+L-1})^\top$.

V. Submap planning and full algorithm

There are several state-of-the-art techniques to obtain a SLAM solution for medium-to-large scenarios quickly, such as SLAM++ [19] and iSAM2 [3]. Submap joining [16] is not an essential tool for SLAM alone, but is useful for the planning process in active SLAM. An overview of Linear SLAM [16] is provided in the supplementary material [20]. If a global map is not required, we can perform active SLAM in each submap in isolation without joining them together. However, having a global map can be useful for improving obstacle avoidance and area coverage. We use Linear SLAM to obtain such a global map. In this section, we show the benefits of planning in the submap, and draw related conclusions that then allow us to use submaps in active SLAM.

A. Importance of submap planning

In our active SLAM process, we perform planning and estimating in submaps as opposed to a global map. The benefits of using submap planning are discussed:

1) Control switching problem: Our control switching strategy (5) relies on the availability of suitable switching indices λ_i^{fx} and λ_i^{fy}. In a global map, which increases in size with the length of the robot trajectory, the uncertainties associated with new features will become large if pose accuracy deteriorates. It is difficult to find constant switching indices to assign control inputs. An alternative would be to find increasing functions instead of constants, but this is difficult and ill-suited. Planning in submaps limits the range of the eigenvalues of the covariance matrices because a new covariance matrix is built for each submap assuming zero uncertainty of its first pose. Then, it is straightforward to set the switching indices λ_i^{fx} and λ_i^{fy} based on the size of the submap.

2) Local minimum problem: From (20), we can see that the objective function will encourage the robot to move closer to a center point $\boldsymbol{p}_c$ determined by features $\boldsymbol{x}_{f1}^{opt}, \cdots, \boldsymbol{x}_{fm}^{opt}$ and weights $c_1, \cdots, c_m$. With an increasing number of previously observed features, the point $\boldsymbol{p}_c$ will approach a local minimum and the UAV would endlessly circle around this point. This problem will greatly slow down the active SLAM process, reduce its efficiency, and even cause the failure of the whole

task. Planning in submaps avoids such local minima by generating a new center point $\boldsymbol{p}_c$ for each submap.

3) Computational efficiency improvements: When used together with powerful SLAM tools, the submap strategy can help to further improve run-time performance with an acceptable level of information loss. For the planning components, submaps help to limit the number of variables and the number of no-fly zones in a given problem instance. In large-scale scenes, distant features and no-fly zones increase computational burden yet contribute little to plan quality. Submap planning limits problem size and thus helps to improve computational efficiency.

B. Bound on the eigenvalues of the global map's FIM

In this section, we discuss the relationship between the eigenvalues of the FIM of the local submaps and of the joined global map. We show how to use this relationship to quickly check the accuracy of the SLAM result without performing the map joining process.

We first establish the following conclusions:

Conclusion 1: The matrix $\boldsymbol{A_Z}^\top \boldsymbol{A_Z}$ is a diagonal matrix, where $\boldsymbol{A_Z}$ is a sparse coefficient matrix that gives the data association relationship between the state vectors of the local map and the global map.

Conclusion 2: The i-th smallest eigenvalue $\lambda_i(\boldsymbol{\mathcal{I}_Z})$ of the FIMs of submaps 1 and 2, and the i-th smallest eigenvalue $\lambda_i(\boldsymbol{\mathcal{I}_{all}})$ of the FIM of the global map satisfy:

$$\lambda_i(\boldsymbol{\mathcal{I}_Z})\widehat{\lambda}(\boldsymbol{A_Z}) \le \lambda_i(\boldsymbol{\mathcal{I}_{all}}) \le \lambda_{n-k+i}(\boldsymbol{\mathcal{I}_Z})\widetilde{\lambda}(\boldsymbol{A_Z}), \tag{27}$$

where $\widehat{\lambda}(\boldsymbol{A_Z})$ and $\widetilde{\lambda}(\boldsymbol{A_Z})$ are the minimum and maximum eigenvalues of matrix $\boldsymbol{A_Z^\top A_Z}$, the specific equations of $\boldsymbol{\mathcal{I}_Z}$ and $\boldsymbol{\mathcal{I}_{all}}$ and proofs for these conclusions are given in the supplementary material [20].

A key question in the application of submap planning is, after building several submaps, when is the best time to generate the global map using Linear SLAM? The joining process is time-consuming, so it is computationally advantageous to perform this process infrequently. When not joining the submaps, We need to estimate the global positions of the obstacles and the coverage area by computing a coordinate transform between subsequent submaps, and global uncertainty is increased by this linear superposition. The approximate estimated uncertainty may be much larger than the one after using Linear SLAM. It affects the safety ranges R_h^{add} and the performance of the coverage and obstacle avoidance tasks. Thus it is necessary to apply the joining process at suitable times. Conclusion 2 provides a way to check the SLAM accuracy and identify when to use Linear SLAM.

We can terminate the submap joining algorithm when the bounds in (27) meet the condition:

$$\frac{\sum_{i=1}^{k} \lambda_{n-k+i}(\boldsymbol{\mathcal{I}_Z})\widetilde{\lambda}(\boldsymbol{A_Z})}{\sum_{i=1}^{k} \lambda_i(\boldsymbol{\mathcal{I}_Z})\widehat{\lambda}(\boldsymbol{A_Z})} \le D, \tag{28}$$

where D is user-defined coefficient to specify the gap between the lower and upper bounds. The global map will then either obey these bounds tightly or else we can use Linear SLAM to generate the global map. This process can be implemented on an independent CPU core which is parallel to the main program and would not affect the run-time performance of active SLAM.

C. Challenges introduced by submap joining

In order to use submaps in estimation and planning, several challenges need to be addressed. The first is how to solve the no-fly zone avoidance problem and perform the coverage task in the submap. Obstacle avoidance and coverage task are based on the global coordinate of the UAV and the no-fly zones, so when the joining process is not applied, we need to estimate the UAV pose, its covariance matrix and the coordinates of the no-fly zones in every submap by the approximate coordinate transformation, which is shown in supplementary martial [20] and our conference paper [17].

The other challenge is that the features detected and mapped in other submaps will be regarded as new in the present submap. This may lead to the UAV continuously visiting these previously mapped features as if they were unknown. In order to address this challenge, we need to identify whether a given feature has been observed in a previous submap and whether its uncertainty level has already been reduced to an acceptable level. If so, this feature will be deleted in (12).

D. Full algorithm

We now collect all algorithmic components and present an integrated algorithm for our active SLAM problem. Pseudocode for the full method is shown in Algorithm 3. Two metrics in line 24 and 27 are user-defined based on the scenario, discussed in Sec. VI.

VI. Simulations and experiments

A. Active SLAM using proposed method

All simulations and experimental implementations are performed on an HP EliteDesk 800 G2 desktop with an Intel Core i5-6500 3.20 GHz processor. For the simulations, the UAV moves in a 100 m $\times$ 100 m square-shaped space with known bounds, 30 unknown static point features and 10 known static no-fly zones. We created ten such sets of features and no-fly zones, labelled Sets 1-10. The remaining parameters include: v = 1 m/s, R_s = 20 m, C_u = 0.3 rad/s, $\triangle t$ = 1s, $\sigma_v(k)$ = 0.1 m, $\sigma_\omega(k)$ = 0.1 rad and $\sigma_f(k)$ = 0.3 m, c_a = 0.85, c_c = 0.15, C_1^{index} = $0.06N_f$ m, C_2^{index} = $0.1N_f$ m, and the number of poses in each submap is 50. The maximal allowable number of simulation steps is limited to 3000. The stopping conditions are set such that more than 98% of the whole area is covered and all features are mapped.

The final result of the proposed method using Set 1 is presented in Fig. 4. The blue triangles, black points, red star points and yellow circles are respectively the real UAV poses (each 5 steps), the SLAM estimated result, the features and the no-fly zones.

In Fig. 4, this result shows that the UAV completes the coverage task with good estimation accuracy.

Algorithm 3 Active SLAM framework

Input: Environmental, vehicle and other parameters.
Output: Estimated poses and mapped features (SLAM results).

repeat
  //Get model and measurement
  Move the robot based on dynamic model with noise
  Get noisy measurements from sensor model
  //SLAM
  Perform data association
  Solve SLAM problem using GN method
  //Switching mechanism
  Obtain $Index_1$ using Algorithm 1
  if $Index_1 \geq C_2^{index}$ **then**
    //Uncertainty minimization task
    Build weighted Laplacian matrix $\widehat{\boldsymbol{L}}_{w_p}^{\mathcal{S}}$
    Mark mapped features from other submaps that have sufficient estimation accuracy
    Compute coefficient c_j using Algorithm 2
    Build and solve convex optimization problem (20)
  elseif $Index_1 \leq C_1^{index}$ **then**
    //Coverage task
    Build and solve coverage optimization (16) in [17]
  else
    //Both tasks
    Solve uncertainty minimization task using lines 11-15 and coverage task using line 17-18
  end if
  //Submap joining
  Decide whether to open a new submap
  Decide whether to use Linear SLAM
  $t \leftarrow t + \triangle t$
until Believable covered area is large enough

B. Comparisions

1) Effect of submap planning: Except for the submap size (Submap 10-100) and the datasets (Set 1-10), the parameter settings are the same as in Sec. VI-A. Results using different submap sizes are shown in Fig. 5.

In order to exclude the effect of differences in pose number, only the *D-opt* optimality criteria corresponding to the features $\log(\det(\boldsymbol{I}_{\boldsymbol{f}}(\boldsymbol{X})))$ is compared. All simulations using submaps complete the coverage task and detect all features. In Fig. 5, with increasing submap size, the SLAM process will be more time-consuming because of the larger numbers of estimated variables.

To examine a non-submap simulation, we present an example trajectory result in Fig. 6. At the beginning, the switching indices are suitable. As the UAV moves further from its starting position, because of not using submaps, large uncertainty of new features such as A, B and C cannot trigger control switching based on (5). The UAV therefore continues to perform the uncertainty minimization task among the known features, which leads to no new features being detected. The UAV gets trapped in a local minimum position (indicated by grid-lines). Because the distances between the local minimum position and the marginal features (such as A, B, C and D) are larger than the sensor range, the UAV cannot add any new measurements for these features. Meanwhile, features located in the blue box have been observed many times and can no longer contribute to improving the objective function, and measurements of other features (A, B and C) are insufficient. The overall result is poor, supporting the use of submaps as an important part of our active SLAM solution.

2) Advantage of the proposed method: We compared our method with the SQP method and a greedy algorithm (60 discrete uniformly distributed candidate actions) using the original objective function [17], and a state-of-the-art algorithm [13] that consists of a greedy algorithm in conservative sparse information space. The initial inputs of the SQP method are the solutions of the greedy method. Results for Set 1 with varied look-ahead steps are presented in Fig. 7. Results using different methods with 15 look-ahead steps are shown in Fig. 9, 10 and 11.

Because of the non-convex property of the original objective function, we find that the planning time grows quickly with increasing look-ahead step L. Meanwhile, many local minima are observed using the other methods. Our solution is better in these respects. Thanks to its convexity, our formulation can be solved in polynomial time. In these results, all frameworks are implemented using our submap planning strategy, otherwise planning time will greatly increase with the number of poses.

We also compare our method, whose submap size is set as 50, with a simple lawnmower pattern approach using the same parameter settings. For Sets 1-10, our algorithm outperforms the lawnmower approach by, respectively, 3.85%, 5.52%, 4.23%, 5.08%, 2.29%, 2.34%, 9.75%, 1.53%, 3.66% and 8.67% (uncertainty level). The final result of the lawnmower pattern for Set 1 is shown in Fig. 12. This approach leads to relatively good exploration of the unknown environment, which means that it has a reasonable performance in general. However, in many situations, the features are distributed nonuniformly, which may greatly limit its performance.

C. Discussion of control behavior

1) Control switching: Fig. 8 shows the Set 1 result for our method at the 80-th/144-th step, including the UAV trajectory and the covered area. We see that the control input from the uncertainty minimization task makes the UAV tend to perform loop-closure, and the coverage task leads the UAV to explore the unknown area. At the 144-th step, the UAV needs to finish an emergency obstacle avoidance operation, shown as the peach trajectory. Based on the suitable switching indices C_1^{index} and C_2^{index}, we can switch the control inputs easily. In Fig. 4 (Set 1), the likelihood of using the emergency avoidance control, the coverage control, the uncertainty minimization or the combination control as the final acting control u_r are 5.36%, 26.08%, 21.86% and 46.70%. Our choice of switching indices allows these four different control inputs to be used appropriately.

2) Effect of no-fly zones: Compared with a collision-free environment, emergency operations are not controlled by an optimization process and will reduce the quality of the solution. Thus the number of emergency operations should not be too large. The proportions of control

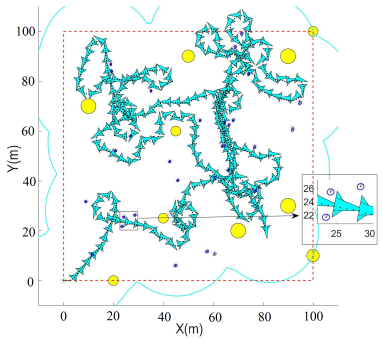

Fig. 4. Final results using Set 1 (The blue circles around the features are the 3σ covariance ellipses of coordinates.)

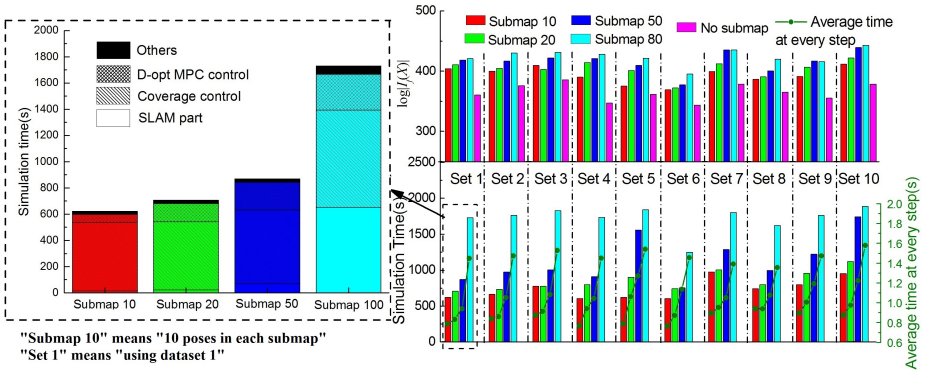

Fig. 5. Comparision results with different submap sizes (In 3000s, instances that do not use submaps fail to complete the coverage task and coverage percentages are all below 41.2%.)

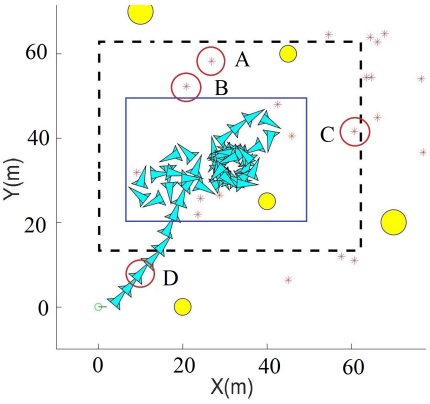

Fig. 6. Result without using submaps (The algorithm cannot finish the coverage task.)

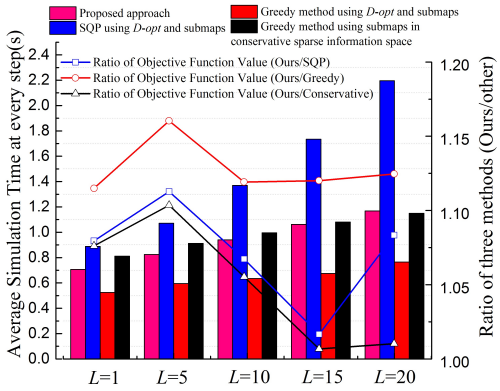

Fig. 7. Comparison results using multiple methods (Histogram and line graph shows the time and ratio respectively.)

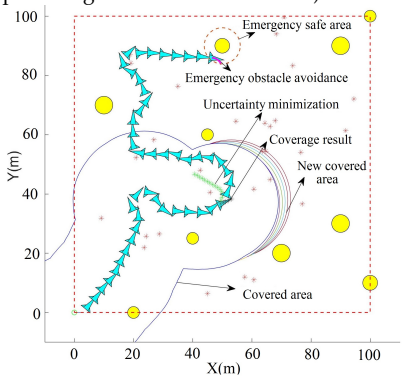

Fig. 8. Trajectory and optimization outputs (The green and black dotted lines respectively indicate the uncertainty minimization results and the coverage result.)

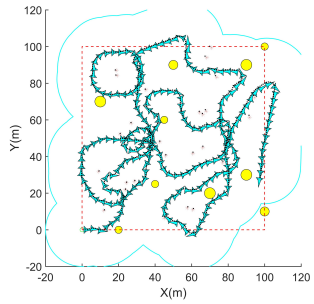

Fig. 9. Results using the SQP method

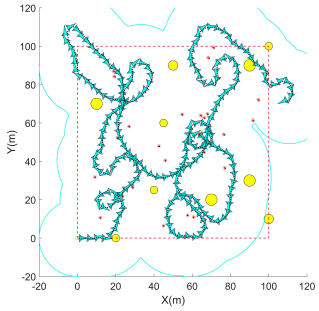

Fig. 10. Results using the greedy method

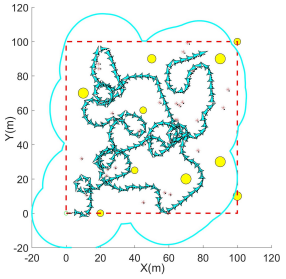

Fig. 11. The result of the greedy method in conservative sparse information space

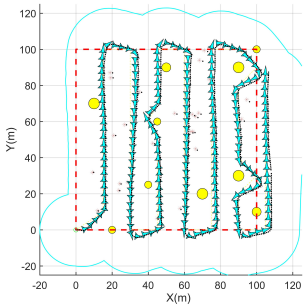

Fig. 12. Results using lawn-mower pattern

inputs that involve emergency operations for our 10 datasets are 5.36%, 3.35%, 3.85%, 2.11%, 4.127%, 1.61%, 5.61%, 4.99%, 4.41% and 6.84%. They are all small, which indicates that solution quality is minimally affected.

D. Off-line experimental verification

In this section, we provide preliminary results with a real quad-rotor UAV to verify the practicability of the final trajectory in an indoor environment and also to evaluate the performance of the SLAM result. Because our MATLAB implementation is not suited for onboard computation, we perform an experiment where the UAV executes an active SLAM trajectory computed off-line.

Due to lack of the on-board sensors, we use a motion capture system to obtain virtual measurements in the local frame. The specific operations are as follows: Based on the motion capture system, we obtain the ground truth of the UAV poses and the global coordinate of the features. Then, we compute local measurements using the models (1) and (2) based on the poses and the features with added Gaussian noise when the features lie within the virtual sensor range (1 m). These noise terms, which are equal to the simulation data in active SLAM, are generated previously. After executing the entire off-line trajectory, we can generate all measurements similar to the simulation case but also including the real dynamic model. Using these measurements, we can solve the SLAM problem and obtain its *D-opt* metric. The ground truth and estimated result are plotted in Fig. 13. We also perform an experiment to follow the trajectory considering the coverage task only. Compared with it, the objective function $\log(\det(\boldsymbol{I}_f(\boldsymbol{X})))$ of our method has a 9.3% advantage with the similar coverage ratio (100%).

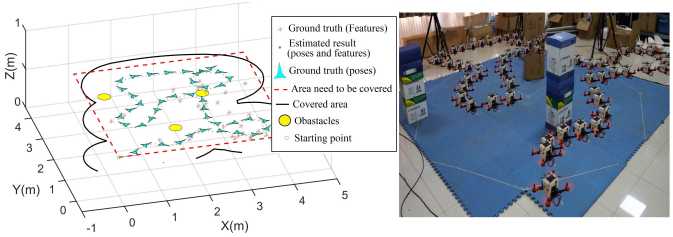

Fig. 13. Ground truth (blue vehicles and red stars), estimated result (green points) and UAV trajectory (In 170s, the UAV covered 98.12% area. Meanwhile, the SLAM method successfully estimates the features and poses. The right figure presents a composite photograph of the UAVs motion. Following the off-line trajectory, the flying experiment shows that the obtained trajectory is flyable for the quad-rotor UAV.)

VII. Conclusions and future work

We have presented a solution to an active SLAM problem combining uncertainty minimization and area coverage within an MPC framework. A submap joining method is applied to ensure the effectiveness of the proposed method and improve its run-time performance. A graph topology and convex optimization approach are used to solve the *D-opt* MPC problem and the SQP method is used to solve the coverage problem. We have verified our method in realistic simulation scenarios and in an off-line experiment. The results demonstrate that the proposed method can generate a collision-free trajectory to finish the coverage task with superior performance relative to multiple existing approaches.

This paper provides a promising framework to solve active SLAM problems where coverage is required and multiple constraints need to meet. In the future, we would like to extend the results to 3D and conduct real-time experiments to further evaluate its practicality.

References

[1] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Trans. on Robotics*, vol. 32, no. 6, pp. 1309-1332, 2016.

[2] 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," *Int. J. Robot. Res.*, vol. 38, no. 2-3, pp. 95-125, 2019.

[3] M. Kaess, H. Johannsson, R. Roberts, V. Ila and J. J. Leonard, F. Dellaert, "iSAM2: Incremental smoothing and mapping using the Bayes tree," *Int. J. Robot. Res.*, vol. 31, no. 2, pp. 216-235, 2012.

[4] S. Huang, N. M. Kwok, G. Dissanayake, and Q. P. Ha, "Multi-step look-ahead trajectory planning in SLAM: Possibility and necessity," in *Proc. IEEE Int. Conf. Robot. Autom.*, pp. 1091-1096, 2005.

[5] C. Shen, Y. Shi, and B. Buckham, "Integrated path planning and tracking control of an AUV: A unified receding horizon optimization approach," *IEEE/ASME Trans. on Mechatronics*, vol. 22, no. 3, pp.1163-1173, 2017.

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

[7] J. Vallvé and J. A. Cetto, "Active pose SLAM with RRT*," in *Proc. IEEE Int. Conf. Robot. Autom.*, pp. 1050-4729, 2015.

[8] L. Chen, Y. Shan, W. Tian, B. Li, and D. Cao. "A fast and efficient double-tree RRT*-like sampling-based planner applying on mobile robotic vehicles," *IEEE/ASME Trans. on Mechatronics*, 2018.

[9] I. Maurović, M. Seder, K. Lenac, and I. Petrovic, "Path planning for active SLAM based on the D* algorithm with negative edge weights," *IEEE Trans. on Systems, Man, and Cybernetics: Systems*, vol. 99, no. 1, pp. 1-11, 2017.

[10] A. Pazman, "Foundations of optimum experimental design," Springer, 1986.

[11] M. L. Rodríguez-Arévalo, J. Neira, and J.A. Castellanos, "On the importance of uncertainty representation in active SLAM," *IEEE Trans. on Robotics*, vol. 34, no. 3, pp. 829-834, 2018.

[12] V. Indelman, L. Carlone, and F. Dellaert, "Planning in the continuous domain: A generalized belief space approach for autonomous navigation in unknown environments," *Int. J. Robot. Res.*, vol. 34, no. 7, pp. 849-882, 2015.

[13] V. Indelman, "No correlations involved: Decision making under uncertainty in a conservative sparse information space," *IEEE Robot. and Autom. Letters*, vol. 1, no. 1, pp. 407-414, 2016.

[14] M. Keidar and G. Kaminka, "Efficient frontier detection for robot exploration," *Int. J. Robot. Res.*, vol. 33, no. 2, pp. 215-236, 2014.

[15] V. D. Berg, S. P. Jur, and R. Alterovitz, "Motion planning under uncertainty using iterative local optimization in belief space," *Int. J. Robot. Res.*, vol. 31, no. 11, pp. 1263-1278, 2012.

[16] L. Zhao and S. Huang, and G. Dissanayake, "Linear SLAM: Linearising the SLAM problems using submap joining," *Automatica*, vol. 100, pp. 231-246, 2019.

[17] Y. Chen, S. Huang, R. Fitch, and J. Yu, "Efficient active SLAM based on submap joining, graph topology and convex optimization," in *Proc. IEEE Int. Conf. Robot. Autom.*, pp. 5159-5166, 2018.

[18] W. Xu, http://www.cchzykf.com/tech/2018/1225/3220.html.

[19] V. Ila, L. Polok, M. Solony, and P. Svoboda, "SLAM++-A highly efficient and temporally scalable incremental SLAM framework," *Int. J. Robot. Res.*, vol. 36, no. 2, pp. 210-230, 2017.

[20] Y. Chen, S. Huang and R. Fitch, "Active SLAM for mobile robots with area coverage and obstacle avoidance," (Supplementary Material), 2018. https://github.com/cyb1212/SM-for-TMECH

[21] K. Khosoussi, S. Huang, and G. Dissanayake, "Novel insights into the impact of graph structure on SLAM," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, pp. 2707-2714, 2014.

[22] J. Saunderson, P. A. Parrilo, and A. S. Willsky, "Semidefinite descriptions of the convex hull of rotation matrices," *SIAM J. on Optimization*, vol. 25, no. 3, pp. 1314-1343, 2015.

[23] J. B. Mueller and R. Larsson, "Collision avoidance maneuver planning with robust optimization," in *Proc. Int. ESA Conf. Guid. Navi. Contr. Sys.*, 2008.

[24] D. M. Rosen, C. DuHadway and J. J Leonard, "A convex relaxation for approximate global optimization in simultaneous localization and mapping," in *Proc. IEEE Int. Conf. Robot. Autom.*, pp. 5822-5829, 2015.

Yongbo Chen received the bachelors in engineering from Beijing Institute of Technology (BIT), Beijing, China, in 2012.

He is now a PhD candidate with the Centre for Autonomous Systems, Faculty of Engineering and Information Technology, University of Technology Sydney, Sydney, Australia. His research interests include UAV motion/mission planning and mobile robots simultaneous localization and mapping (SLAM).

Shoudong Huang received the Ph.D. degree in automatic control from Northeastern University, Shenyang, China, in 1998.

He is currently an Associate Professor with the Centre for Autonomous Systems, Faculty of Engineering and Information Technology, University of Technology Sydney, Sydney, Australia. His research interests include mobile robots simultaneous localization and mapping (SLAM), robot path planning and control.

Robert Fitch received the Ph.D. degree in computer science from Dartmouth College, Hanover, NH, USA, in 2005.

He is currently an Associate Professor in the Centre for Autonomous Systems, University of Technology Sydney, Australia. His research interests include systems of outdoor robots and their application to key problems in agriculture and environmental monitoring.