2025 IEEE International Conference on Robotics and Automation (ICRA)
May 19-23, 2025. Atlanta, USA

Efficient Submap-based Autonomous MAV Exploration using Visual-Inertial SLAM Configurable for LiDARs or Depth Cameras

Sotiris Papatheodorou[1,2,3,4,*], Simon Boche[1,*], Sebastián Barbas Laina[1,3], Stefan Leutenegger[1,2,3,4]

***Abstract*— Autonomous exploration of unknown space is an essential component for the deployment of mobile robots in the real world. Safe navigation is crucial for all robotics applications and requires accurate and consistent maps of the robot's surroundings. To achieve full autonomy and allow deployment in a wide variety of environments, the robot must rely on on-board state estimation which is prone to drift over time. We propose a Micro Aerial Vehicle (MAV) exploration framework based on local submaps to allow retaining global consistency by applying loop-closure corrections to the relative submap poses. To enable large-scale exploration we efficiently compute global, environment-wide frontiers from the local submap frontiers and use a sampling-based next-best-view exploration planner. Our method seamlessly supports using either a LiDAR sensor or a depth camera, making it suitable for different kinds of MAV platforms. We perform comparative evaluations in simulation against a state-of-the-art submap-based exploration framework to showcase the efficiency and reconstruction quality of our approach. Finally, we demonstrate the applicability of our method to real-world MAVs, one equipped with a LiDAR and the other with a depth camera.**

I. Introduction

Autonomous exploration of unknown environments has been an active area of research in mobile robotics and remains a challenge. It is essential for potential real-world robot applications, such as industrial or construction site inspection. MAVs are a popular choice of platform due to their versatility, agility and ability to reach areas inaccessible to humans or ground robots. The objective of exploration is commonly formulated as discovering and mapping an environment as fast as possible [1], [2], [3]. Ideally, to achieve full autonomy, the robot incrementally builds an accurate and complete map of its environment online, solely based on sensor inputs, that contains all information required for navigating through and localizing in it. State-of-the-art approaches typically formulate this problem as identifying the next-best-pose that maximizes some utility and finding a collision-free path to the goal pose.

Simultaneous Localization and Mapping (SLAM) methods fusing complementary sensors are widely used for on-board pose estimation. Visual-Inertial SLAM (VI-SLAM), combining visual and inertial measurements, offers high short-term accuracy but suffers from the accumulation of drift [4], [5], [6]. Lately, driven by the availability of low-priced Light Detection and Ranging (LiDAR) sensors, LiDAR-Visual-Inertial SLAM (LVI-SLAM) has been a topic of extensive research [7], [8], [9] in the prospect of reducing drift.

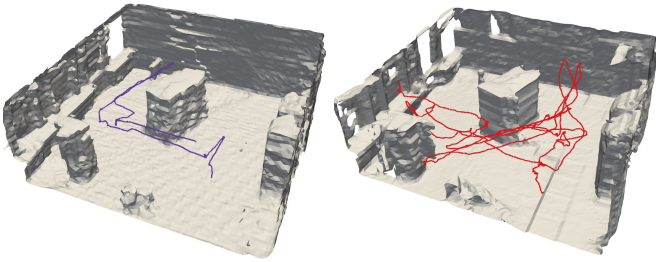

Fig. 1: MAV trajectory and live-reconstructed mesh at 10 cm resolution for the real-world experiments using a LiDAR (left) and a depth camera (right).

In large-scale scenarios, the inevitable drift can lead to significant deterioration of the map accuracy. One potential mitigation is using local submaps, based on the assumption that drift is negligible within a small region. To maintain global consistency, the relative poses between individually rigid submaps can be updated upon loop closures [10] or incorporated into the state estimation problem [9], [11].

Keeping track of frontiers, the boundaries between observed and unobserved space, becomes challenging when using submaps, as a large number of them may be involved. Recent works have proposed hierarchical exploration methods, distinguishing between local and global exploration planning [12], [13]. In our work, we show that a more complex hierarchical approach is not necessary for fast and efficient exploration.

We propose the following contributions:

- A fast and lightweight 3D exploration framework leveraging SLAM and submapping to account for odometry drift using loop closures.
- An efficient global frontier update method, allowing for accurate and complete large-scale exploration, without having to distinguish between local and global planning.
- The proposed system seamlessly supports MAVs equipped with either LiDAR or depth camera sensors out-of-the-box.
- Quantitative and qualitative evaluation in simulation and in the real world, showing faster exploration, reduced resource usage, and more consistent and complete reconstructions compared to a state-of-the-art method.

This work was supported by the Technical University of Munich, MIRMI, the TUM Innovation Network CoConstruct, Leica Geosystems AG. and the EU Horizon projects DigiForest (101070405) and AUTOASSESS (101120732).

[1]Smart Robotics Lab, School of Computation, Information and Technology, Technical University of Munich. E-mail addresses: {sotiris.papatheodorou, simon.boche, sebastian.barbas, stefan.leutenegger}@tum.de

[2]Smart Robotics Lab, Department of Computing, Imperial College London. E-mail addresses: {s.papatheodorou18, s.leutenegger}@ic.ac.uk

[3]Munich Institute of Robotics and Machine Intelligence (MIRMI).

[4]Munich Center for Machine Learning (MCML).

[*] Equal contribution.

2025 IEEE International Conference on Robotics and Automation (ICRA) | 979-8-3315-4139-2/25/$31.00 ©2025 IEEE | DOI: 10.1109/ICRA55743.2025.11128702

Authorized licensed use limited to: ShanghaiTech University. Downloaded on September 11,2026 at 02:55:24 UTC from IEEE Xplore. Restrictions apply.

II. Related Work

A. Multi-Sensor SLAM

Performing autonomous exploration in environments without prior infrastructure requires estimating the robot's state. VI-SLAM achieves this given camera images and measurements from an Inertial Measurement Unit (IMU), which are especially vital for MAVs, platforms capable of highly dynamic movements, which might cause purely visual approaches to fail due to losing track. VI-SLAM systems are often classified into filter-based, e.g. [14], [15], and optimization-based approaches [4], [5], [6], [16]. The latter tend to yield higher accuracy and usually formulate pose estimation as a factor graph optimization problem, minimizing residuals of IMU integration errors and visual reprojection errors. To reduce the drift of VI-SLAM systems caused by the integration of noisy IMU measurements, fusing LiDAR-based residuals has been extensively investigated. Many works adopted the early geometric LiDAR-based feature residuals (edge and plane) formulated in LOAM [17], e.g. [7]. However, direct or Iterative Closest Point (ICP) - based variants have also proven effective [8], [18]. In our work, we use our LVI-SLAM system as presented in [9].

B. Volumetric Mapping

An important design decision in autonomous exploration systems is the map representation used. Dense 3D maps, usable for downstream tasks, can be obtained from volumetric mapping. The pioneering work KinectFusion [19] represents the map as a Truncated Signed Distance Field (TSDF). Unfortunately, TSDF-based maps are unable to explicitly represent *free* space, making them unsuitable for navigation tasks. Using Euclidean Signed Distance Fields (ESDF) instead is one way to solve this issue, as e.g. in *voxblox* [20]. Another line of work uses occupancy mapping, representing the map as a discrete grid of occupancy probabilities. A large number of works follow *Octomap* [21] in using an octree data structure as the underlying map representation. Several extensions, such as *UFOMap* [22] and *supereight2* [23], have been proposed since then. Both approaches explicitly store *unknown* space and support adaptive-resolution; mapping the 3D space only up to the required detail. In our work, occupancy submaps are created using *supereight2* [23].

C. Autonomous Exploration

Related works in robotic exploration are commonly categorized into sampling-based and frontier-based methods.

The concept of frontiers was introduced in [1]. During the incremental reconstruction of an environment, frontiers are defined as the boundaries between known *free* and *unknown* space and indicate areas that potentially lead to a large gain of information when observed. This concept found adoption in a large number of subsequent works, e.g. [24], [25].

Sampling-based approaches on the other hand sample a number of candidate viewpoints or paths and select the next goal based on the computation of a utility function, following the pioneering work of [2]. Candidate next views are sampled in free space and the next best view is selected based on the unknown volume potentially observed from the candidate over the path length. The efficiency of these methods can be improved by informed sampling, e.g. in the proximity of frontiers [3], [26], [27], or other utility functions [28], [29].

To deal with large-scale environments, recent approaches have proposed hierarchical or multi-stage designs that distinguish between local and global planning. In [30], local planning is handled by a Rapidly-exploring Random Tree (RRT) that determines the next best view. The global planner is modelled as a Gaussian Process that stores the gain of non-executed viewpoints. *GBPlanner* [13] also follows this two-stage pattern and consists of a local RRT*-based planner and a global graph-based planner. The global graph guides the robot towards frontiers, when the local planner reaches a dead-end. Furthermore, [13] imposes additional safety constraints on the selected path.

Regardless of the design, accurate live pose estimates are required, typically obtained from SLAM. Several approaches have been proposed to overcome the deterioration of the map quality due to drift. [31] addresses this issue by computationally expensive de- and re-integration of measurements upon visual loop closures. Furthermore, exploration is actively steered towards potential loop closure candidates. Another line of works applies submapping strategies. The global map is represented as a collection of submaps which can be rearranged upon loop closures or pose-graph optimisation, as presented in [9], [11], [32], [33].

A successful combination of submapping and hierarchical exploration has been presented in *GLocal* [12]. Submaps based on *Voxgraph* [11] are periodically created and updated in a pose-graph optimization which allows handling loop closures. In *GLocal*, the MAV first explores its surroundings using a local, RRT*-based planner. Once the utility of nearby regions becomes low, the global planner guides the robot towards global frontiers. Traversability graphs from previously completed submaps allow for efficient global path planning.

Just like *GLocal*, our system builds upon the concept of submaps. In contrast to the aforementioned approaches, we do not need to make a distinction between local and global planning. Enabled by efficient updates of the local and global frontiers, a unified global planner can be applied, favoring nearby candidate next poses through the utility function design.

III. Problem Statement

In this work, we aim to build an accurate, complete and globally consistent volumetric representation of the environment using a fully autonomous MAV equipped with either a LiDAR or depth camera. To enable scaling to large environments, we represent the environment with submaps.

A. Environment Model

The static environment is modeled as a bounded volume $V \subset \mathbb{R}^3$ where each point $\mathbf{v} \in V$ has an associated occupancy probability $P_o(\mathbf{v})$. The occupancy of all points $\mathbf{v} \in V$ is initially *unknown*, defined as $P_o(\mathbf{v}) = 0.5$. Due to the environment's geometry as well as the MAV and sensor characteristics, there can be points $V_{\text{unob}} \subset V$ that are unobservable.

Authorized licensed use limited to: ShanghaiTech University. Downloaded on September 11,2026 at 02:55:24 UTC from IEEE Xplore. Restrictions apply.

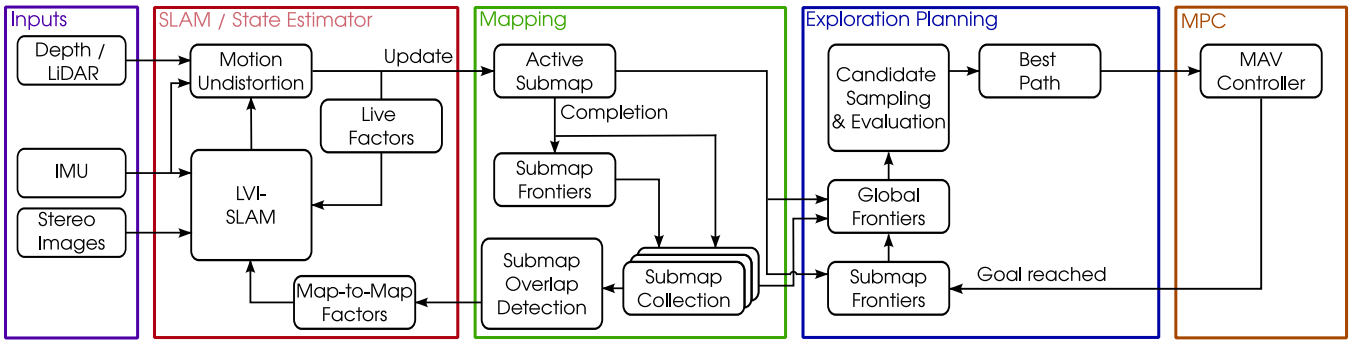

Fig. 2: Overview of the system's main modules: (i) a VI-SLAM system, optionally with LiDAR, that estimates the MAV's state based on sensor inputs and submaps, (ii) an occupancy mapping backend dividing the global volume into spatially bounded submaps and keeping track of a submap collection, (iii) an exploration planning module computing the next best path based on global frontiers of all submaps and (iv) a linear MPC executing the planned paths.

Thus exploration only considers the observable part of the environment $V_{\text{obs}} = V \setminus V_{\text{unob}}$. The goal of exploration is to create a collection of submaps $\mathscr{M} = \{M_i \subseteq V, i \in \{1 \dots n_m\}\}$ so that $\bigcup_{i=1}^{n_m} M_i = V_{\text{obs}}$, while updating the occupancy probability of all $\mathbf{v} \in V_{\text{obs}}$ to either *free* or *occupied*.

B. MAV Model

The MAV state $\mathbf{x}$ consists of the position $\mathbf{r} \in V$, orientation $\mathbf{q} \in SO(3)$, and linear velocity $\upsilon \in \mathbb{R}^3$. For exploration and path planning we consider a portion of the MAV's full state, $\hat{\mathbf{x}} = [\mathbf{r}^T \psi]^T \in V \times [-\pi, \pi)$, where ψ is its yaw angle with respect to the world frame $\underrightarrow{\mathscr{F}_W}$. We also assume the MAV to be enclosed in a sphere of radius $R \in \mathbb{R}^+$ centered at $\mathbf{r}$, and to have a maximum linear velocity $\upsilon_{\max} \in \mathbb{R}^+$ and a maximum yaw rate $\omega_{\max} \in \mathbb{R}^+$. The MAV is equipped with a LiDAR sensor or a depth camera with horizontal and vertical field of view $\alpha_h \in (0, 2\pi]$ and $\alpha_v \in (0, \pi]$ respectively, and a measurement range $[d_{\min}, d_{\max}] \subset \mathbb{R}^+$.

IV. System Overview

Our system combines VI-SLAM, dense 3D occupancy mapping, exploration planning and an MAV Model Predictive Controller (MPC). To deal with the accumulated VI-SLAM drift and achieve locally consistent and accurate maps, we employ spatially bounded submaps. Using SLAM-estimated poses, LiDAR measurements or depth images are integrated into the currently active submap. Upon completion, submaps and their local frontiers are saved in a submap collection. Global frontiers are only updated as needed, using the sumbap-local frontiers. This allows using a single, global exploration planner, making it unnecessary to distinguish between local and global planning, as other methods do. The global frontiers are used in a sampling-based method to determine the next best path from an information gain perspective, which is then tracked by the MPC. Our method does not require a GPU, allowing its deployment on small and resource-constrained platforms. In the following we describe the individual system modules, shown in Figure 2.

A. State Estimator

We use OKVIS2 [4] as our state estimator, a state-of-the-art sparse, optimization-based VI-SLAM system. It receives stereo grayscale image pairs and inertial measurements and produces state estimates at IMU rate which include the MAV state $\mathbf{x}$ and IMU biases. In the case of MAVs equipped with LiDARs, we also integrate our previous work [9], which formulates residuals based on occupancy maps and their gradients to optimize for consistency between (a) incoming measurements and previous submaps, and (b) overlapping submaps. The reader is referred to [9] for details.

B. Occupancy Mapping

We use *supereight2* [23] for volumetric occupancy maps using octrees. *Supereight2* explicitly represents *free* space and propagates minimum and maximum occupancy values to the octree root, allowing safe and efficient path planning. The original *supereight2* only allows integrating depth images, either from a depth camera or by projecting a structured LiDAR scan. In order to support dynamically moving LiDAR sensors, we adopt the integration scheme from [9].

Supereight2 forms the basis of our submapping framework, as in [9] and [34]. New submaps are generated based on the geometric overlap in the case of LiDAR sensors, as in [9], or based on visual keyframes in the case of depth cameras, as in [34]. In both cases submaps are anchored to OKVIS2 keyframe states and are transformed along with them on pose optimization, including loop closures. Once a new submap is created, the previous one is frozen, allowing only rigid transformations for the remainder of the mission.

C. Exploration Planning

We extend the exploration planner for monolithic maps we proposed in [3] to account for submaps. The key ideas of this planner remain unchanged and are re-stated here for convenience. Candidate next positions $\hat{\mathbf{r}}_j \in V, j \in \{1 \dots n_c\}, n_c \in \mathbb{N}^+$ are sampled close to frontiers, since they correspond to regions that will expand the map if observed. A path is planned from the current MAV position to each candidate and its estimated duration t_j is computed assuming the MAV flies at $\upsilon_{\max}$ and $\omega_{\max}$. A 360° map entropy raycast is performed from each candidate position $\hat{\mathbf{r}}_j$ producing a gain image. Given the sensor's horizontal field of view α_h, a sliding window optimization is performed on the gain image to determine the yaw angle ψ_j resulting in the highest potential

Authorized licensed use limited to: ShanghaiTech University. Downloaded on September 11,2026 at 02:55:24 UTC from IEEE Xplore. Restrictions apply.

gain g_j. The utility of each candidate j is computed as $u_j = g_j / t_j$, essentially maximizing information gain over time. The candidate with the highest utility becomes the next goal and the process is repeated once the MAV reaches it.

The exploration planner used in this work differs from [3] in two aspects. First, we compute global frontiers from the collection of submaps $\mathscr{M}$, presented in Section V as one of our core contributions. Second, the yaw optimization over the gain image is performed differently for sensors with $\alpha_h = 360°$, which [3] is not designed to handle. In this case the candidate yaw angle ψ_j can be chosen arbitrarily while the candidate gain g_j is just computed over the whole image. For simplicity, we compute ψ_j using a sliding window corresponding to a horizontal field of view smaller than 360°.

D. MAV Controller

We use a linear MPC based on [35] for MAV trajectory following, with modifications as described in [34]. In short, the MPC is modified to ensure correct trajectory tracking even in the case of major odometry changes, such as loop closures. This is achieved by anchoring trajectories to OKVIS2 keyframe states and elastically deforming them as the keyframe states are updated over time.

V. Global Frontier Computation

The exploration planner samples candidate next positions close to frontiers, thus, a set of global frontiers, considering all submaps and their overlaps, must be computed. Frontiers in one submap corresponding to fully observed regions in another submap are not global frontiers.

To achieve efficient global frontier computation, we (i) keep track of submap-local and global frontiers separately, and (ii) update both only as needed. The local frontiers of the currently active submap are updated before each global frontier computation. Local frontiers are also updated upon submap completion and then remain frozen for the remainder of the mission. In both cases, only the submap regions modified since the last local frontier computation are considered. Global frontiers are re-computed at each exploration planner iteration, considering all currently known local frontiers. Since global frontiers are only used for sampling candidate next views, this scheme ensures no unnecessary computations are performed. The following describes the local and global frontier computation in detail.

A. Local Frontier Computation

For each submap M_i we keep track of a set of local frontiers $\mathscr{F}_i^k$ at timestep k, by considering M_i in isolation from other submaps. Local frontiers are *free* voxels that (i) have at least one *unknown* face neighbor voxel or (ii) are at the submap boundary. The latter case is essential for correct global frontiers when using bounded-extent submaps.

Only the local frontiers $\mathscr{F}_i^k$ of the currently active submap are updated. This update is done in two steps. First, all preexisting local frontiers $\mathscr{F}_i^{k-1}$ are re-tested so that the out-of-date frontiers $\mathscr{F}_{i,\text{stale}}^{k-1}$ are detected. Then, the set of new frontiers $\mathscr{F}_{i,\text{new}}^k$ is computed by only testing voxels that

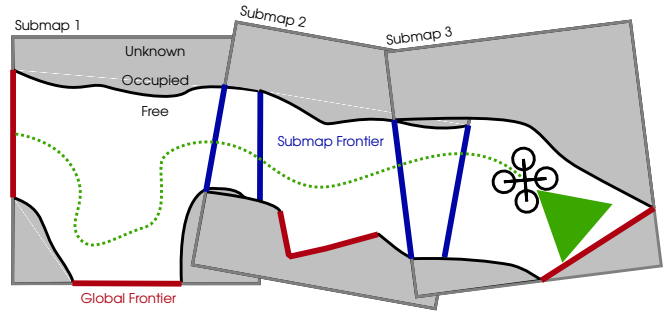

Fig. 3: Local-only (blue) and global (red) frontiers in a set of submaps. The local-only submap frontiers correspond to regions mapped in other submaps and are thus not considered to be global frontiers.

were modified since $\mathscr{F}_i^{k-1}$ was computed. Finally, the local frontiers at timestep k are computed as

$$\mathscr{F}_i^k = \left(\mathscr{F}_i^{k-1} \setminus \mathscr{F}_{i,\text{stale}}^{k-1}\right) \cup \mathscr{F}_{i,\text{new}}^k. \tag{1}$$

The detection of stale frontiers $\mathscr{F}_{i,\text{stale}}^{k-1}$ can be performed in parallel with the computation of the new frontiers $\mathscr{F}_{i,\text{new}}^k$.

B. Global Frontier Computation

The global frontiers $\mathscr{F}_g^k$ at timestep k are computed as

$$\mathscr{F}_g^k = \bigcup_{i=1}^{n_m} \mathscr{F}_{g,i}^k, \tag{2}$$

where $\mathscr{F}_{g,i}^k \subseteq \mathscr{F}_i$ are the local frontiers of M_i that are also global frontiers at timestep k, and $\mathscr{F}_i$ are the last-known frontiers of M_i. $\mathscr{F}_{g,i}^k$ is computed by testing each frontier in $\mathscr{F}_i$ against all other submaps

$$\mathscr{F}_{g,i}^k = \left\{ f_i \in \mathscr{F}_i,\ \bigwedge_{l \neq i} h(f_i, M_l) \right\}, \tag{3}$$

where the Boolean-valued function $h(f_i, M_l)$ indicates whether frontier f_i of M_i is also a frontier when considering M_l. $h(f_i, M_l)$ is *true* iff $f_i \notin M_l$ or f_i corresponds to an *unknown* region of M_l or f_i corresponds to a local frontier of M_l. The computation of $\mathscr{F}_{g,i}^k$ can be performed in parallel for each submap.

Figure 3 shows an example of local and global frontiers.

VI. Evaluation

A. Simulation Results

Our simulation setup consists of Gazebo [36] and the PX4 autopilot [37] simulated in software, mimicking the setup of the real-world MAV. The simulated MAV is based on the RMF-Owl [38], a $0.38 \times 0.38 \times 0.24$ m quadcopter equipped with a LiDAR sensor and an Intel RealSense D455 RGB-D camera. The evaluation is done in the $30 \times 15 \times 9$ m warehouse-like environment *Depot*, shown in Figure 4. All simulated experiments were conducted on a computer with an Intel Core i7-1165G7 CPU and 32 GB of RAM.

We compare our method with the state-of-the-art, submap-based, 3D exploration planner *GLocal* [12], in terms of exploration speed, environment reconstruction accuracy and completeness, as well as CPU and memory usage. *GLocal*

Authorized licensed use limited to: ShanghaiTech University. Downloaded on September 11,2026 at 02:55:24 UTC from IEEE Xplore. Restrictions apply.

requires a LiDAR sensor and performs submap alignment using *Voxgraph* [11], similarly to our approach based on [9]. Thus, we only evaluate LiDAR-based exploration in simulation. Both approaches receive poses from OKVIS2, including visual loop closures. Our approach includes submap alignment based on [9], whereas *GLocal* uses map-to-map alignment from [11]. We use the MAV controller described in Section IV-D for both pipelines. We made an effort to use the same parameters for both methods where applicable, the most important of which are listed in Table I.

Map resolution	0.1 m	LiDAR resolution	360×180
R	0.5 m	α_h, α_v	360°, 90°
$\upsilon_{\max}$	0.5 m/s	$d_{\min}$, $d_{\max}$	1 m, 10 m
$\omega_{\max}$	0.5 rad/s	n_c (n/a to *GLocal*)	20

TABLE I: Simulation experiment parameters.

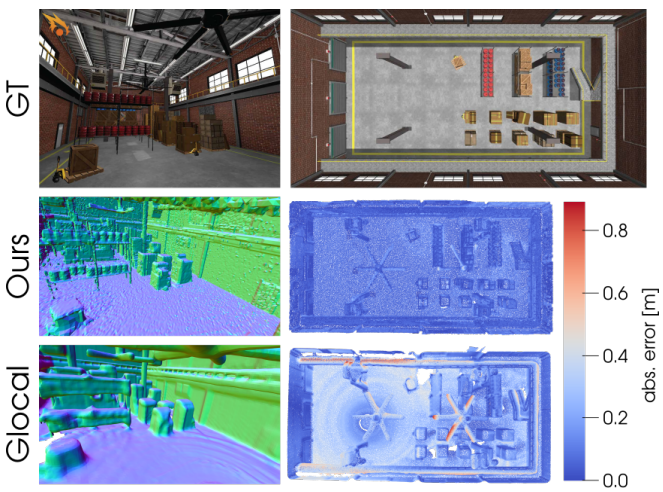

Fig. 4: Perspective and top-down visualisation of the *Depot* environment. Top: Ground-truth. Middle and Bottom: Final reconstruction (left) and mesh accuracy (right), using our approach and *GLocal* respectively.

1) Explored Volume: In order to avoid discrepancies in the explored volume due to different mapping frameworks, we record the LiDAR measurements and corresponding ground-truth MAV poses. We use them to construct a monolithic *supereight2* map, measuring the explored volume after each integration. For each method, we performed 10 runs using poses from OKVIS2 and another 10 runs using ground-truth poses as a baseline. The median, 10th and 90th percentiles of the percentage of the total volume explored by the two methods is shown in Figure 5. Our method outperforms *GLocal* in terms of exploration speed, consistency and final environment coverage. While our approach nearly always explores 100% of the volume within 300 s, *GLocal* sometimes fails to explore the whole volume. Using ground-truth poses results in less variance between runs, which can be attributed to the more consistent maps resulting in fewer frontiers and more free space for the MAV to navigate through.

2) Safety Evaluation: We evaluate the safety of the exploration planner by computing the minimum distance between the *Depot* ground-truth mesh and every point of the ground-truth MAV trajectory. Figure 6 shows a histogram

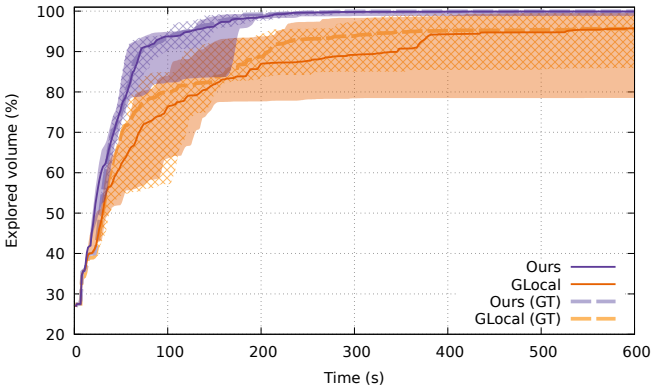

Fig. 5: Explored volume median, 10th and 90th percentiles for 10 runs in the *Depot* environment. Solid for SLAM, dashed/hatched for ground-truth runs.

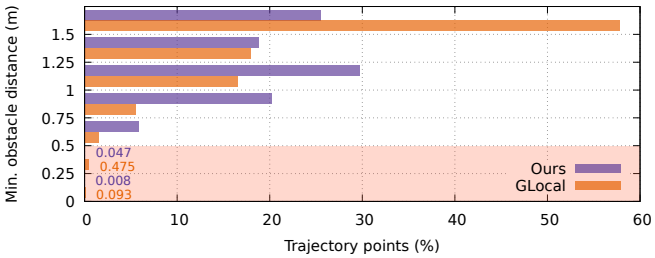

Fig. 6: Histogram of minimum distance of the MAV to obstacles across 10 simulated missions with SLAM poses and a safety radius of 0.5 m. The bin size is 0.25 m.

of these distances across all 10 missions. Our approach in general succeeds to navigate safely even in proximity to obstacles while *GLocal* violates the safety radius $R = 0.5$ m significantly more often, leading to a higher risk of collisions.

3) Reconstruction Quality: We also evaluate the quality of the final mesh reconstructions at a 0.1 m resolution obtained from a post-processing step. In our method, the final map reconstruction is computed using poses estimated after a final bundle adjustment to provide the best possible accuracy. We use the ground-truth *Depot* mesh to compute the root-mean-square-error (RMSE) of the reconstruction. The completeness is computed as the percentage of the reconstruction within 0.2 m or 0.4 m of the ground-truth. The mean RMSE and completeness of all 10 runs are presented in Table II. Our approach yields significantly better results in both metrics, showcasing it is possible to achieve safe path planning without sacrificing reconstruction quality. This is explained to some extent by the fact that the *voxblox* mapping framework used by *GLocal* artificially inflates obstacles for safer path planning. A visual comparison of the final reconstructions as well as their accuracy is presented in Figure 4.

4) Resource Usage: We also compare the computational resources required by our method and *GLocal* in Table III. Even though both methods compute frontiers only as needed, ours benefits from the more efficient map representation of *supereight2*. While we only compute the utility for a limited number of candidate next positions, in *GLocal* it is computed for each vertex of the tree used for local planning, requiring

Authorized licensed use limited to: ShanghaiTech University. Downloaded on September 11,2026 at 02:55:24 UTC from IEEE Xplore. Restrictions apply.

	RMSE (m) ↓	Completeness (%) ↑ within 0.2 m	within 0.4 m
Ours	**0.112**	**86.07**	**97.73**
GLocal	0.219	44.19	76.97
Ours (GT)	0.095	89.89	97.98
GLocal (GT)	0.213	45.29	78.20

TABLE II: Mesh reconstruction RMSE and completeness using both SLAM and ground-truth (GT) poses in the Depot environment.

significantly more time. The planning time includes utility computation, exploration planning, and path planning. The high planning time in *GLocal* is due to a dedicated thread re-planning at a high rate, accumulating a large amount of CPU time, even after ignoring trivial planning iterations requiring less than 200 ms. Finally, the smaller memory usage of our system is due to the efficiency of *supereight2* maps.

	Frontiers (s)	Utility (s)	Planning (s)	Memory (GB)
Ours	**16 ± 7**	**11 ± 6**	**134 ± 45**	**6.7 ± 1.0**
GLocal	52 ± 7	275 ± 84	421 ± 76	23.5 ± 1.8

TABLE III: Mean and standard deviation of per-mission frontier, utility, and path and exploration planning computation time and memory usage.

B. Real World Experiments

Real-world experiments were conducted to further showcase the effectiveness of the proposed exploration approach on resource-constrained platforms and demonstrate its applicability to both LiDAR sensors and depth cameras. The experiments were carried out in an $8.0 \times 8.1 \times 3.7$ m ≈ 240 m^3 room containing some tall obstacles.

Fig. 7: MAVs used for real-world experiments. Left: Leica BLK2Fly with LiDAR. Right: Custom MAV with Intel Realsense D455 RGB-D camera.

For LiDAR exploration, we use a Leica BLK2Fly MAV, equipped with 5 cameras, 5 IMUs and a single-beam, dual-axis spinning LiDAR sensor. For VI-SLAM, only the front, left, and bottom cameras and the bottom IMU are used. The LiDAR has an effective frequency of 5 Hz and a 360° field of view in both axes, although the MAV body occludes part of it, resulting in α_h and α_v both being less than 360°. On-board filtering of the LiDAR point clouds reduces the number of points to $\approx$ 100,000 to 200,000 points per second. As on-board resources are not made available for general use by the manufacturer, the sensor data is streamed via WiFi to a laptop where LVI-SLAM, mapping and exploration planning are running. The laptop used for off-board processing has an Intel Core i7-13850HX CPU and 32 GB of RAM.

For depth camera exploration we use a custom-built quadcopter based on the Holybro S500 frame and equipped with an Intel RealSense D455 stereo RGB-D camera and an NVIDIA Jetson Orin NX 16 GB on-board computer. All computations, including VI-SLAM, dense occupancy mapping, exploration planning and MPC are performed on-board.

Both MAV platforms are shown in Figure 7. The parameters used for the experiments are listed in Table IV.

	LiDAR	Depth camera
Map resolution	0.1 m	0.1m
$d_{\min}$, $d_{\max}$	0.25 m, 10 m	0.2 m, 4 m
R	0.6 m	0.6 m
$\upsilon_{\max}$	1.0 m/s	0.5 m/s
$\omega_{\max}$	-	0.785 rad/s

TABLE IV: Real-world experiment parameters.

We conducted one experiment using each MAV platform. The final mesh reconstructions and estimated MAV trajectories are shown in Figure 1 while Figure 8 shows the observed volume over time. As expected, due to its much larger field of view, the LiDAR-equipped MAV achieves a higher exploration speed and reaches an almost complete environment coverage within less than a minute. Nevertheless, even using a depth camera with a much more limited field of view, we can almost fully explore the room in an efficient manner.

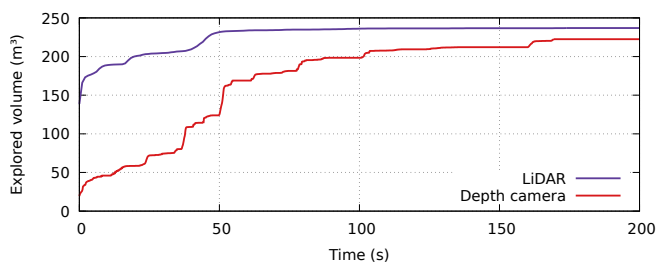

Fig. 8: Explored volume over time for the two real-world experiments.

VII. Conclusion

In this work we propose an efficient and lightweight submap-based autonomous exploration method that we demonstrate to accept both depth cameras and LiDARs as input modalities. Our method leverages submapping and VI-(LiDAR)-SLAM, in order to achieve accurate and consistent mapping despite odometry drift, thus ensuring safe operation. The efficient computation of global frontiers from the aggregated submaps allows us to apply only one unified global planning approach, rendering the distinction between local and global planning that state-of-the-art methods employ unnecessary. Compared to a state-of-the-art submap-based large-scale exploration framework, our method achieves faster exploration and more accurate environment reconstructions while being even more resource efficient. It is further deployed on two real-world MAV platforms, one using a depth camera, and the other a LiDAR.

In future work, we would like to integrate semantic information and Vision-Language Models (VLMs) to produce environment maps richer in information and more useful for downstream tasks. We would also like to investigate integrating trajectory planning taking the MAV dynamics into account as well as safe MAV control strategies.

Authorized licensed use limited to: ShanghaiTech University. Downloaded on September 11,2026 at 02:55:24 UTC from IEEE Xplore. Restrictions apply.

REFERENCES

[1] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *Proceedings 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation CIRA'97.'Towards New Computational Principles for Robotics and Automation'*. IEEE, 1997, pp. 146–151.

[2] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart, "Receding horizon "next-best-view" planner for 3D exploration," in *2016 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2016, pp. 1462–1468.

[3] A. Dai, S. Papatheodorou, N. Funk, D. Tzoumanikas, and S. Leutenegger, "Fast frontier-based information-driven autonomous exploration with an MAV," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 9570–9576.

[4] S. Leutenegger, "OKVIS2: Realtime scalable visual-inertial SLAM with loop closure," *arXiv preprint arXiv:2202.09199*, 2022.

[5] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós, "ORB-SLAM3: An accurate open-source library for visual, visual–inertial, and multimap SLAM," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021.

[6] V. Usenko, N. Demmel, D. Schubert, J. Stückler, and D. Cremers, "Visual-inertial mapping with non-linear factor recovery," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 422–429, 2019.

[7] T. Shan, B. Englot, C. Ratti, and D. Rus, "LVI-SAM: Tightly-coupled lidar-visual-inertial odometry via smoothing and mapping," in *2021 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2021, pp. 5692–5698.

[8] C. Zheng, Q. Zhu, W. Xu, X. Liu, Q. Guo, and F. Zhang, "FAST-LIVO: Fast and tightly-coupled sparse-direct LiDAR-inertial-visual odometry," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 4003–4009.

[9] S. Boche, S. Barbas Laina, and S. Leutenegger, "Tightly-coupled LiDAR-visual-inertial SLAM and large-scale volumetric occupancy mapping," *arXiv preprint arXiv:2403.02280*, 2024.

[10] Y. Wang, N. Funk, M. Ramezani, S. Papatheodorou, M. Popović, M. Camurri, S. Leutenegger, and M. Fallon, "Elastic and efficient LiDAR reconstruction for large-scale exploration tasks," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 5035–5041.

[11] V. Reijgwart, A. Millane, H. Oleynikova, R. Siegwart, C. Cadena, and J. Nieto, "Voxgraph: Globally consistent, volumetric mapping using signed distance function submaps," *IEEE Robotics and Automation Letters*, vol. 5, no. 1, pp. 227–234, 2019.

[12] L. Schmid, V. Reijgwart, L. Ott, J. Nieto, R. Siegwart, and C. Cadena, "A unified approach for autonomous volumetric exploration of large scale environments under severe odometry drift," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4504–4511, 2021.

[13] T. Dang, S. Khattak, F. Mascarich, and K. Alexis, "Explore locally, plan globally: A path planning framework for autonomous robotic exploration in subterranean environments," in *2019 19th International Conference on Advanced Robotics (ICAR)*. IEEE, 2019, pp. 9–16.

[14] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint Kalman filter for vision-aided inertial navigation," in *Proceedings 2007 IEEE international conference on robotics and automation*. IEEE, 2007, pp. 3565–3572.

[15] P. Geneva, K. Eckenhoff, W. Lee, Y. Yang, and G. Huang, "OpenVINS: A research platform for visual-inertial estimation," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 4666–4672.

[16] T. Qin, S. Cao, J. Pan, and S. Shen, "A general optimization-based framework for global pose estimation with multiple sensors," *arXiv preprint arXiv:1901.03642*, 2019.

[17] J. Zhang and S. Singh, "LOAM: Lidar odometry and mapping in real-time." in *Robotics: Science and systems*, vol. 2, no. 9. Berkeley, CA, 2014, pp. 1–9.

[18] I. Vizzo, T. Guadagnino, B. Mersch, L. Wiesmann, J. Behley, and C. Stachniss, "KISS-ICP: In defense of point-to-point ICP–simple, accurate, and robust registration if done the right way," *IEEE Robotics and Automation Letters*, vol. 8, no. 2, pp. 1029–1036, 2023.

[19] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon, "KinectFusion: Real-time dense surface mapping and tracking," in *2011 10th IEEE international symposium on mixed and augmented reality*. IEEE, 2011, pp. 127–136.

[20] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto, "Voxblox: Incremental 3D Euclidean signed distance fields for on-board MAV planning," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 1366–1373.

[21] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: An efficient probabilistic 3D mapping framework based on octrees," *Autonomous robots*, vol. 34, pp. 189–206, 2013.

[22] D. Duberg and P. Jensfelt, "UFOMap: An efficient probabilistic 3D mapping framework that embraces the unknown," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6411–6418, 2020.

[23] N. Funk, J. Tarrio, S. Papatheodorou, M. Popović, P. F. Alcantarilla, and S. Leutenegger, "Multi-resolution 3D mapping with explicit free space representation for fast and accurate mobile robot motion planning," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 3553–3560, 2021.

[24] T. Cieslewski, E. Kaufmann, and D. Scaramuzza, "Rapid exploration with multi-rotors: A frontier selection method for high speed flight," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 2135–2142.

[25] W. Gao, M. Booker, A. Adiwahono, M. Yuan, J. Wang, and Y. W. Yun, "An improved frontier-based approach for autonomous exploration," in *2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV)*. IEEE, 2018, pp. 292–297.

[26] D. Duberg and P. Jensfelt, "UFOExplorer: Fast and scalable sampling-based exploration with a graph-based planning structure," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2487–2494, 2022.

[27] P. Zhong, B. Chen, S. Lu, X. Meng, and Y. Liang, "Information-driven fast marching autonomous exploration with aerial robots," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 810–817, 2021.

[28] L. Schmid, M. Pantic, R. Khanna, L. Ott, R. Siegwart, and J. Nieto, "An efficient sampling-based method for online informative path planning in unknown environments," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1500–1507, 2020.

[29] A. Batinovic, A. Ivanovic, T. Petrovic, and S. Bogdan, "A shadowcasting-based next-best-view planner for autonomous 3D exploration," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2969–2976, 2022.

[30] M. Selin, M. Tiger, D. Duberg, F. Heintz, and P. Jensfelt, "Efficient autonomous exploration planning of large-scale 3-D environments," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1699–1706, 2019.

[31] Y. Zhang, B. Zhou, L. Wang, and S. Shen, "Exploration with global consistency using real-time re-integration and active loop closure," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 9682–9688.

[32] A. Millane, Z. Taylor, H. Oleynikova, J. Nieto, R. Siegwart, and C. Cadena, "C-blox: A scalable and consistent TSDF-based dense mapping approach," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 995–1002.

[33] Y. Wang, M. Ramezani, M. Mattamala, S. T. Digumarti, and M. Fallon, "Strategies for large scale elastic and semantic LiDAR reconstruction," *Robotics and Autonomous Systems*, vol. 155, p. 104185, 2022.

[34] S. Barbas Laina, S. Boche, S. Papatheodorou, D. Tzoumanikas, S. Schaefer, H. Chen, and S. Leutenegger, "Scalable outdoors autonomous drone flight with visual-inertial SLAM and dense submaps built without LiDAR," *arXiv preprint arXiv:2403.09596*, 2024.

[35] D. Tzoumanikas, W. Li, M. Grimm, K. Zhang, M. Kovac, and S. Leutenegger, "Fully autonomous micro air vehicle flight and landing on a moving target using visual–inertial estimation and model-predictive control," *Journal of Field Robotics*, vol. 36, no. 1, pp. 49–77, 2019.

[36] Open Robotics, "Gazebo Fortress", https://gazebosim.org.

[37] PX4 Autopilot, "PX4 Autopilot Software", https://px4.io.

[38] P. De Petris, H. Nguyen, M. Dharmadhikari, M. Kulkarni, N. Khedekar, F. Mascarich, and K. Alexis, "RMF-Owl: A collision-tolerant flying robot for autonomous subterranean exploration," in *2022 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2022, pp. 536–543.

Authorized licensed use limited to: ShanghaiTech University. Downloaded on September 11,2026 at 02:55:24 UTC from IEEE Xplore. Restrictions apply.