Building Hybrid Omnidirectional Visual-Lidar Map for Visual-Only Localization

Jingyang Huang[1], Hao Wei[1], Changze Li[2], Tong Qin[*2], Fei Gao[1], and Ming Yang[2]

***Abstract*— Recently, there has been growing interest in using low-cost sensor combinations, such as cameras and IMUs, to achieve accurate localization within pre-built pointcloud maps. In this paper, we propose a novel hybrid visual-Lidar mapping and visual-only re-localization framework, specifically designed for UAVs with limited computational resources operating in challenging environments. Keyframes function as a bridge in our system, associating images with pointcloud to facilitate efficient and accurate pose estimation. Besides, our system creates omnidirectional keyframes at the mapping stage, enabling effective re-localization from any orientation, which enhance the robustness and practicability of our system. Experiments show that the proposed algorithm achieves high localization accuracy on pre-built maps and is capable of running in real-time on UAVs for autonomous navigation tasks. The source code will be made publicly available soon.**

I. Introduction

Visual localization is the task of estimating the 6-DoF camera pose of an image relative to a reference scene. It is vital for applications like robotics, autonomous driving, and 3D reconstruction, where precise positioning is essential for navigation or interaction with the environment. Typically, visual localization requires consecutive camera frames or independent images to build a scene-specific 3D map using Structure-from-Motion (SfM) or Simultaneous Localization and Mapping (SLAM) methods. This map is then used for real-time camera pose estimation, a process known as visual re-localization [1]. However, traditional visual mapping approaches often suffer from depth inaccuracies due to the inherent limitations of monocular or stereo vision [2]. While SLAM-based methods continuously update the map and refine localization, they still struggle with texture-sparse environments and scale ambiguity [3]. On the other hand, SfM methods typically rely on feature matching across multiple views, making them computationally expensive and unsuitable for real-time applications [4]. To address these limitations and enhance visual localization performance, researchers have explored the integration of cross-modal data, such as Lidar pointclouds [5], which provide more accurate depth information and greater robustness in challenging environments.

[1]Jingyang Huang, Hao Wei, and Fei Gao are with Zhejiang University, Hangzhou, China. {huangjy3, isweihao, fgaoaa}@zju.edu.cn

[2]Changze Li, Tong Qin, and Ming Yang are with the Global Institute of Future Technology, Shanghai Jiao Tong University, Shanghai, China. {changze, qintong, mingyang}@sjtu.edu.cn.

*is the corresponding author.

This work was supported by the Natural Science Foundation of Shanghai (Grant No. 24ZR1435600).

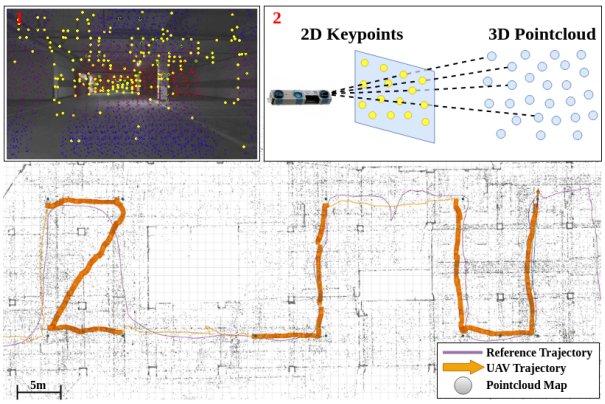


Fig. 1: The hybrid map is constructed in real-time using a UAV equipped with four pinhole cameras and a Lidar sensor (top-left). In this process, keyframes are utlized to associate 2D keypoints with 3D pointcloud for pose estimation (top-right). This map is later used by another UAV for visual-only re-localization and autonomous navigation tasks (bottom).

Cross-modal visual re-localization has recently gained significant attention [6], with research primarily focusing on localization utilizing visual data within a pre-built Lidar map. Existing methods can be broadly categorized into two approaches. The first approach leverages geometric alignment between Lidar and camera data [7], such as registering camera-generated pointclouds to Lidar scans using Iterative Closest Point (ICP) [5]. The second approach employs learning-based techniques to establish implicit associations between Lidar and camera data [8]. Despite their potential, these methods often require heavy computation and generalize poorly across environments, hindering deployment on resource-limited platforms like low-cost UAVs.

To address these challenges, this paper proposes a hybrid visual-Lidar keyframe mapping and re-localization framework. During the mapping stage, Lidar scans and quad-view pinhole camera images are fused to construct an omnidirectional keyframe-based representation. In the re-localization stage, candidate keyframes are first retrieved, followed by the establishment of 2D-2D feature correspondences. If a sufficient number of inliers is identified, 2D-3D associations are further established between the image keypoints and the Lidar pointcloud, leveraging keyframes as a bridge. The final pose estimation is then refined using PnP-RANSAC and pose graph optimization. Notably, both mapping and re-localization are designed for real-time execution on embedded platforms, facilitating efficient and scalable deployment in practical robotic applications, such as navigating along a given path, as shown in Fig. 1.

The contributions of this paper are summarized as follows:

- We proposed a visual re-localization framework that leverages a hybrid visual-Lidar map, which can achieve accurate and robust re-localization in challenging environments. In the hybrid map, image features are associated with accurate depth information from Lidar scans.
- We proposed the rotation-free re-localization mechanism by employing four pinhole cameras which can simultaneously generate multi-directional keyframes. Therefore, the re-localization is not constrained by viewpoints.
- We have conducted extensive experiments with real UAVs, demonstrating that our algorithm can run in real-time on the computation-limited onboard computer. Using cameras, we achieve precise localization that is comparable to Lidar. The results of the experiment show the robustness and accuracy of our algorithm.

II. LITERATURE REVIEW

A. Cross Modal Localization

Accurate and efficient visual re-localization on UAV platforms requires balancing localization precision and sensor constraints. Lidar-based mapping offers high geometric accuracy and robustness to illumination changes, making it a reliable choice for localization. However, Lidar sensors are often expensive and heavy, posing challenges for deployment on low-cost, small-scale UAV platforms. In contrast, vision-based re-localization is more lightweight but generally falls short of Lidar in accuracy and robustness. Cross-modal localization aims to bridge this gap by leveraging Lidar for mapping while using a visual camera for re-localization within the pre-built Lidar map [5, 6]. This approach maintains localization accuracy while reducing sensor requirements during the re-localization phase, making it a promising direction for UAV applications.

From a classification perspective, existing cross-modal localization methods can be categorized into **projection-based** and **3D structure-based** approaches.

Projection-based methods [9, 10] project the 3D Lidar map onto a 2D plane, generating depth or intensity projection images. During re-localization, the current camera image is matched against these precomputed projection images to estimate the camera pose. While computationally efficient, this method discards 3D structural information during projection, leading to potential accuracy losses in localization. **3D structure-based methods** [7, 11] directly utilize geometric structures in the 3D Lidar map. These methods establish correspondences between visual features and 3D points or lines in the map to compute the camera pose. By preserving the full 3D spatial structure, these approaches achieve higher localization accuracy but typically require more computational resources.

From a methodological perspective, cross-modal localization can be further divided into **rule-based** and **learning-based** approaches.

Rule-based methods [5, 7, 12] rely on handcrafted feature descriptors and geometric constraints to establish correspondences between visual images and Lidar maps. These methods offer strong interpretability and do not require extensive training data but often struggle with robustness and generalization in varying environments. **Learning-based methods** [9, 13] leverage artificial neural network to learn cross-modal feature representations and matching strategies from data. These methods improve generalization across different scenes but typically require large-scale training datasets and substantial computational resources, making them less practical for real-time applications on resource-constrained UAV platforms.

B. Visual Place Recognition

Visual Place Recognition (VPR) [14] plays a crucial role in visual re-localization by identifying the camera's location within a previously constructed feature-based map. These feature maps are typically generated through SLAM [15, 16] or SfM [4, 12], consisting of visual landmarks with associated high-dimensional descriptors. The re-localization process begins by retrieving the most relevant reference images from a map database based on a query image and the camera pose is estimated using Perspective-n-Point (PnP) later [17].

Feature-based VPR methods face two key limitations: (1) high memory requirements for storing dense descriptors in large-scale environments, and (2) sensitivity to illumination changes that reduces localization accuracy. Recent learning-based approaches improve robustness through enhanced feature extraction or global descriptors [9], but require extensive training data and computational resources, limiting real-time deployment in constrained systems.

III. METHODOLOGY

An overview of our method is shown in Fig. 2. Our algorithm operates in two sequential stages. During the mapping stage, the visual-Lidar mapping module associates data from the Lidar sensor and four omnidirectional pinhole cameras to generate a hybrid visual-Lidar map. In the localization stage, the visual-only re-localization module first loads keyframes offline from the map. When the system is running online, the loop closure module uses Direct Bag-of-Words (DBoW) [18] to detect the most similar keyframe in the map and matches it with the current image. Since the keypoints in the keyframes database contain the 3D positions under the map coordinate system, we establish a 2D-3D correspondence between the image and pointcloud by matching the 2D keypoints between image and keyframes. We then solve the initial pose estimation using PnP with RANSAC. However, the pose obtained directly from PnP has a certain deviation. We further optimize the pose estimation result through a pose graph, and the final global odometry is fed to the planner to perform navigation tasks. In the subsequent sections, we will delve into the specifics of each module.

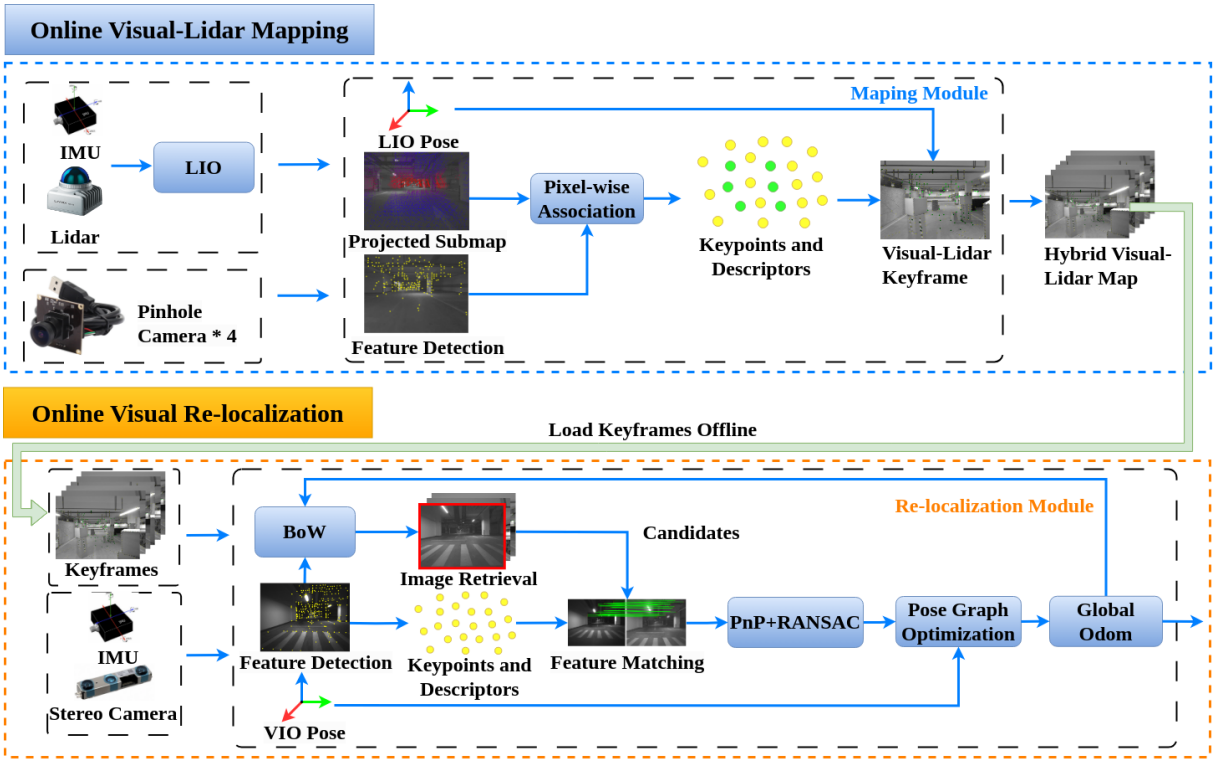


Fig. 2: System pipeline of the proposed method: In the mapping module, a hybrid visual-Lidar map consisting of omnidirectional keyframes is generated. The re-localization module then loads the pre-built map and accurately computes the pose in the world coordinate system, using keyframes to link image keypoints with the Lidar pointcloud.

A. *Visual-Lidar Mapping*

In the Visual-Lidar Mapping stage, the proposed method would combine data from Lidar and cameras. We propose a method that uses the keyframe F_i^M to store the pixel-wise association between pointcloud and image pixels. Here, M indicates that the keyframe belongs to the hybrid visual-Lidar map $\mathcal{F}^{\mathcal{M}}$, and i represents its index within the map. To prevent keyframes holding excessive amounts of duplicate information in nearby locations, a new keyframe is only generated when the UAV has traveled a predetermined distance γ_d. In our system, we generate four keyframes in four directions simultaneously, as shown in Fig. 3. Each of their poses can be calculated using extrinsic between cameras. All keyframes are then saved together to construct a hybrid visual-Lidar map.

1) Submap Projection: We select FAST-LIO2 [19] among various Lidar-based SLAM methods due to its low computational cost and ability to run in real-time on edge computing platforms. To collect the Lidar scans, we use a Livox Mid-360 sensor, which employs a non-repetitive scanning scheme. Therefore, it is necessary to accumulate raw pointcloud over time to generate a denser submap.

First, we transform the submap from the world coordinate to the local coordinate system using FAST-LIO2 pose $\mathcal{T}^S$. Next, we transform the local submap $\mathcal{P}^S = \{p_n^s\}$ into the camera coordinate system using the extrinsic calibration matrix ${}^C T_L$. Finally, we project the submap points onto the image plane to obtain the 2D pixel coordinates $\mathcal{X}^S = \{x_n^s\}$, using the calibrated camera intrinsic matrix K. Here $\cdot$ is the point-wise operation, and π is the projection function. The submap projection result is shown in Fig. 4(a).

$$\mathcal{X}^S = K \cdot {}^C T_L \cdot (\mathcal{T}^S \cdot \mathcal{P}^S) = \pi(\mathcal{T}^S \cdot \mathcal{P}^S) \tag{1}$$

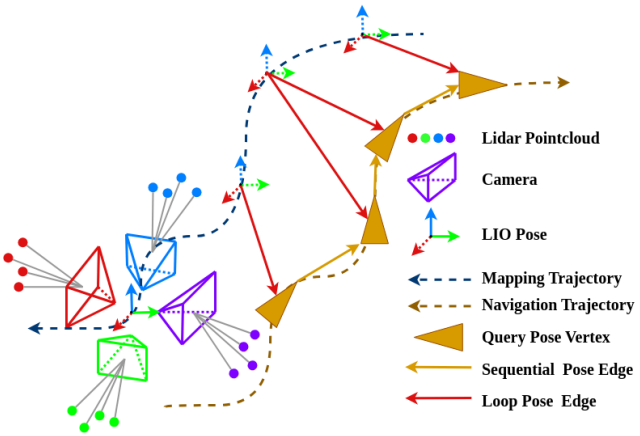


Fig. 3: The illustration of omnidirectional visual-Lidar mapping and visual-only localization based on pre-built map. Query poses would be refined with an Pose Graph Optimization (PGO) utilizing LIO poses.

2) Pixel-Wise Point Association: After projecting the submap onto the camera's pixel plane, we can associate Lidar submap points $\mathcal{P}^S$ with image feature points $\mathcal{X}^I = \{x_n^I\}$ at the pixel level. We use SuperPoint (SP) [20] to detect feature points and extract descriptors. Due to inevitable calibration errors, we consider projected points $\mathcal{X}^S$ within a 2-pixel

radius around a feature point x_n^I to be associable. However, it is not appropriate to roughly weight the depths of all neighboring projected points for a given feature point, as feature extractors tend to focus on corner points, which can lead to significant depth variation among nearby projected points. To address this, we apply an outlier rejection method, retaining only projected points with similar depths. We then perform weighted averaging of the depth values and assign the resulting depth to the feature point. Take the case in Fig. 4(b) for example, the purple and red boxes with number are projected submap points and the number is their depth, the green and yellow boxes in Character are keypoints. In this case x_A, x_B and x_C are assigned with Lidar depth while x_D is not. And x_C rejects a outlier with 20m depth which is too big for others.

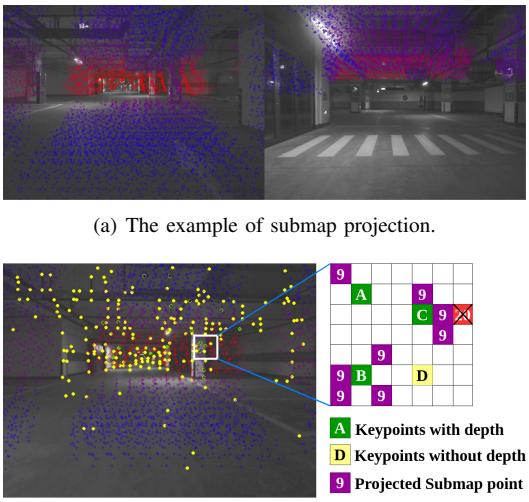
Following these two procedures, keypoints with descriptors and depth are obtained. As we employ DBoW for image retrieval in our localization procedure, more feature points contribute to a more unique representation of each picture, which is advantageous for image retrieval. For this reason, we preserve both keypoints in keyframe.

In addition to image keypoints $\mathcal{X}_i$, Lidar points $\mathcal{P}_i$ and descriptors $\mathcal{D}_i$, we also add a 6-DoF pose $\mathcal{T}_i \in SE(3)$ to each keyframe F_i^M . Thus, one keyframe F_i is consisted of four components: $\mathcal{X}_i$, $\mathcal{P}_i$, $\mathcal{D}_i$ and $\mathcal{T}_i$. Specifically, $\mathcal{X} = \{x_n\}$ represents the 2D pixel coordinates of image keypoint, $\mathcal{P} = \{p_n\}$ represents the 3D point coordinates of Lidar point and $\mathcal{D} = \{d_n\}$ represents the set of descriptors corresponding to the keypoints. The 6-DoF pose $\mathcal{T}_i$ is later used in image retrieval and pose estimation. By storing each keyframe F_i^M generated during the mapping process, we are able to construct a hybrid visual-Lidar map $\mathcal{F}^{\mathcal{M}} = \{F_i^M\}$, which is subsequently used in the visual re-localization stage.

B. Visual Re-localization

Similar to conventional visual re-localization, we utilize the visual information observed by the camera to recover the current pose of the system in the world coordinate frame. Our process is similar to the deep learning-based framework HLoc (Hierarchical Localization) [21] and rule-based framework VINS-Mono [3]. It mainly consists of the following key steps: feature detection and extraction, image retrieval, feature matching, pose estimation, and pose optimization.

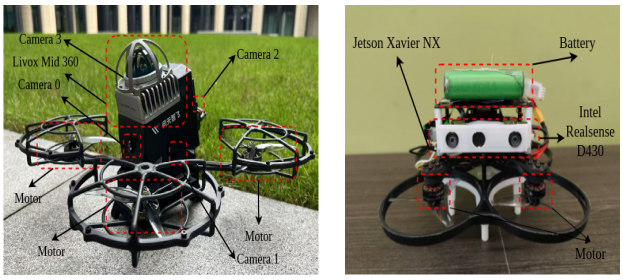
1) Feature Detection and Mathching: In our task, several pinhole cameras are used for mapping, while a RealSense stereo camera is employed for localization. To address this, we adopted the widely used SP detector for feature detection and descriptor extraction, and utilized the SuperGlue (SG) [22] matcher for subsequent descriptor matching. The query keyframe F_i^Q is generated in this way. Deep learning-based algorithms such as SP and SG have demonstrated outstanding performance in recent years, particularly when matching images captured by different devices under varying lighting conditions, making them well-suited for this challenging scenario.



(a) The example of submap projection.

(b) The illustration of pixel-wise association.

Fig. 4: In (a), the points from the submap are projected onto the pixel planes, where the deeper the projected point, the redder the corresponding pixel becomes. In (b), the projected points are then associated with nearby keypoints to assign depth values to the keypoints. During this process, some outliers would be rejected according to depth value.



(a) Drone1 (b) Drone2

Fig. 5: Drone1 features a LiDAR, four monocular cameras, a RealSense sensor and an IMU, while Drone2 retains only an IMU and a RealSense sensor. They specialize in mapping and navigation tasks respectively.

2) Image Retrieval: We use the classic DBoW to construct global features for both the query and reference images. However, due to the presence of some repeated patterns, a significant number of false positives may be retrieved. To mitigate this, we impose restrictions on the DBoW image retrieval process to ensure the search is confined to the appropriate range. Given that the depth values of feature points are obtained from Lidar, we consider the corrected poses to be highly accurate. Therefore, we introduce an additional image retrieval strategy: if no loop closure has occurred, the search is performed across all images; however, once a loop closure is detected, the search is restricted to keyframes with nearby positions.

And it should be noted that image retrieval is not orientation-restricted, as we construct the map with keyframes covering almost all directions. This ensures that loop closure can occur in any orientation, making our system robust to varying camera poses.

3) Pose Estimation: After DBoW retrieves the candidate keyframes $\mathcal{F}^{\mathcal{C}} = \{F_j^C\}$, the system uses the SuperGlue matcher to match the keypoints and descriptors in the query keyframe F_i^Q with those in the candidate keyframes $\mathcal{F}^{\mathcal{C}}$. The 2D-2D correspondences between $\mathcal{X}_i^Q$ and $\mathcal{X}_j^C$ are then transformed into 2D-3D matches between $\mathcal{X}_i^Q$ and $\mathcal{P}_j^C$ for pose estimation. When a sufficient number of 2D-3D matches are established, the system employs PnP with RANSAC to estimate the pose.

The PnP problem aims to minimize the reprojection error of 3D points $\mathcal{P}^R$ and their 2D pixel pairs $\mathcal{X}_i^Q$ by optimizing the camera pose $\mathcal{T}_{\mathcal{C}}^{\mathcal{Q}}$ as follows. While RANSAC is used to reject outliers and enhance the robustness of the PnP solution. Among all candidate keyframes $\mathcal{F}^{\mathcal{C}}$, the result from the keyframe F_R^C with the maximum number of inliers $\mathcal{I}_m$ is preserved.

$$\mathcal{T}_{\mathcal{C}}^{\mathcal{Q}^*} = \arg\min_{\mathcal{T}_{\mathcal{C}}^{\mathcal{Q}}} \sum_{n \in \mathcal{I}_m} \rho\left(||\pi(\mathcal{T}_{\mathcal{C}}^{\mathcal{Q}} p_n^R) - x_n^Q||^2\right) \tag{2}$$

Throughout this process, keyframes serve as a bridge, linking the 2D keypoints in the query image to the 3D pointcloud in the map. In this way, we obtain the estimation of camera pose $\mathcal{T}_{\mathcal{C}}^{\mathcal{Q}^*}$ within the pre-built map.

4) Pose Graph Optimization: In UAV navigation, high-frequency odometry is essential for stable controller input. A common approach is to use the pose from PnP to estimate cumulative drift and correct VIO accordingly. However, directly computing drift error from PnP poses can introduce abrupt error jumps, potentially causing controller instability or failure. To address this, we incorporate omni-directional pose graph optimization as depicted in Fig. 3. Everytime a new query keyframe F_i^Q come, it is push into a sliding keyframe window $\mathcal{F}^Q$. We would next get their connected loop-closure keyframes $\mathcal{F}^R$ if existing. All keyframes in $\mathcal{F}^Q$ and $\mathcal{F}^R$ are added into the pose graph $\mathcal{S}$ and serve as a vertex, and it connects with other vertexes through loop closure edge or sequential edge. The loop closure edges connects PnP pose of vertex F_i^Q and reference keyframe pose of vertex F_j^R, while sequential edges connect two sequential VIO pose vertex F_i^Q and F_j^Q. We construct a 4-DOF pose graph optimization and define the residual of the edge between vertex i and j minimally as :

$$\mathbf{r}_{i,j}\left(\mathbf{t}_i, \psi_i, \mathbf{t}_j, \psi_j\right) = \begin{bmatrix} \mathbf{R}\left(\hat{\phi}_i, \hat{\theta}_i, \psi_i\right)^{-1}\left(\mathbf{t}_j - \mathbf{t}_i\right) - \hat{\mathbf{t}}_{ij}^i \\ \psi_j - \psi_i - \hat{\psi}_{ij} \end{bmatrix} \tag{3}$$

Here, $\mathbf{t} \in \mathbb{R}^3$ represents the translation of the transformation $\mathcal{T}$ in the world coordinate, while the rotation matrix $\mathbf{R} \in SO(3)$ represents its rotational component. The angles ϕ, θ, and ψ correspond to the Euler angles for roll, pitch, and yaw, respectively.

$$\begin{aligned} \hat{\mathbf{t}}_{ij}^i &= \hat{\mathbf{R}}_i^{-1}\left(\hat{\mathbf{t}}_j - \hat{\mathbf{t}}_i\right) \\ \hat{\psi}_{ij} &= \hat{\psi}_j - \hat{\psi}_i \end{aligned} \tag{4}$$

The whole graph of sequential edges and loop closure edges are optimized by minimizing the following cost function:

$$\min_{\mathbf{t},\psi} \sum_{(i,j)\in\mathcal{S}} \left\|\mathbf{r}_{i,j}\right\|^2 \tag{5}$$

Pose graph optimization produces a more smooth corrected pose, which is fed into the planner to control the UAV's navigation tasks.

IV. Experiments

A. Dataset

The primary objective of our algorithm is to mitigate accumulated localization drift during long-distance navigation. We evaluated the proposed system on a self-collected dataset captured in a dimly lit underground environment characterized by sparse textures. The data used for mapping and evaluating localization accuracy includes pointcloud from Livox Mid-360, IMU, four RGB pinhole cameras (FOV $\approx 90°$) and a RealSense camera, as shown in Fig. 5. For navigation, only data from the RealSense stereo camera and IMU is utilized. We construct the prior 3D map leveraging a modified FAST-LIO2 [19]. All images are captured at a resolution of 640×480 and processed in grayscale. To quantitatively assess the performance of the proposed method, the dataset is annotated with ground truth poses obtained through map-based Lidar localization within the pre-constructed pointcloud map. The evaluation metrics include recall, precision, Root Mean Square Error (RMSE). Specifically, recall is defined as the ratio of correctly detected loop closures to the total number of detection attempts, while precision is determined by verifying whether the translational component of the estimated PnP pose deviates by less than 1.0 meters from the ground truth.

B. Mapping Performance

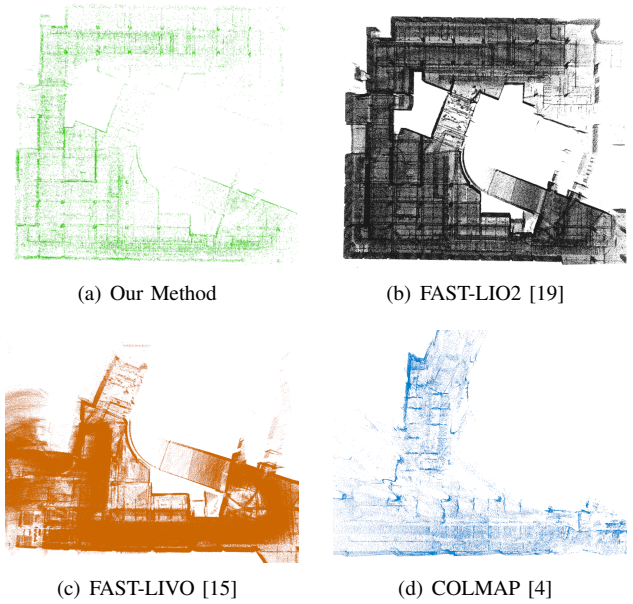
Our experimental evaluation compares the proposed framework with representative SLAM/SfM methods through qualitative visualization (Fig. 6), revealing fundamental trade-offs between geometric completeness, feature richness, map size and efficiency.

VF [16] achieve lightweight operation but lack explicit 3D structures, limiting re-localization accuracy. FAST-LIO2 [19] excel at geometric reconstruction yet fail to support vision-only navigation requiring feature associations. FAST-LIVO [15] combine both modalities but suffer from error propagation in texture-sparse scenes, leading to inflated map sizes and degraded real-time performance. SfM methods (e.g., COLMAP) extract numerous features but reconstruct sparse maps due to bundle adjustment limitations.

The proposed hybrid mapping framework tackles these challenges using three main ideas: organizing the map into smaller submaps, selectively linking visual features with pointcloud data, and adapting descriptors to the environment.

TABLE I: Comparison of Different Mapping Methods

Method	Mapping Approach	Map Format	Map Size (MB)	Online Execution Time (ms)
VINS-Fusion [16]	Visual SLAM	Visual Features	**2.7**	**34.3**
FAST-LIO2 [19]	Lidar SLAM	Dense Pointcloud	44.5	46.2
FAST-LIVO [15]	LIVO SLAM	Dense Pointcloud	338.3(drift)	274.7
COLMAP [4]	SfM	Dense Pointcloud+Visual Features	3.9(incomplete)	N/A(Offline)
Ours	Hybrid Association	Sparse Pointcloud+Visual Features	91.7	104.5



(a) Our Method (b) FAST-LIO2 [19]

(c) FAST-LIVO [15] (d) COLMAP [4]

Fig. 6: Maps generated by different methods on the UG-1 dataset. By selectively fusing visual features with pointcloud data, we achieve geometrically sparse but informative map representations (a).

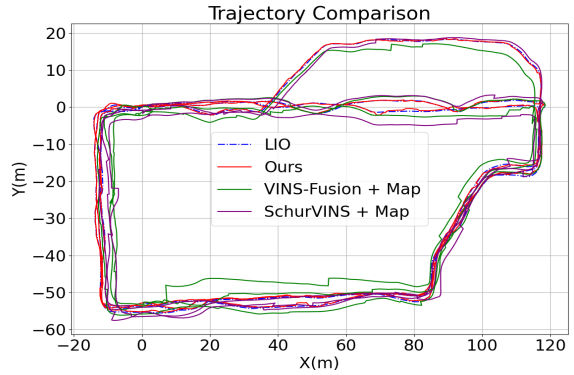


Fig. 7: The trajectory comparison of different methods on dataset UG-0. Our method achieves high localization accuracy in challenging environment.

By decoupling geometric mapping from feature tracking and connecting them via learned descriptors, we balance storage efficiency with information density. While SP descriptors increase map size, they enable reliable matching in texture-limited areas.

C. Long-Term Localization in Challenging Environments

1) Evaluation on self-collected dataset: We evaluate the localization performance on different sequences in our Underground dataset collected by Drone 1. The reference sequence is used to construct the Lidar map, other sequences are used for localization. Lidar trajectories from the reference sequence serve as the mapping prior, while those from other sequences provide the ground truth for localization.

TABLE II: Localization Comparison of Different Methods on Our Datasets

Scenes	UG-0			UG-1			UG-2		
	R	P	E	R	P	E	R	P	E
VF [16] with Map	0.11	0.56	3.88	0.18	0.62	3.20	0.10	0.49	4.15
SV [23] with Map	0.15	0.81	2.02	0.22	0.85	2.07	0.13	0.72	2.43
Ours w/o SPFDM	0.27	0.65	2.31	0.35	0.75	1.87	0.25	0.58	2.80
Ours w/o PA	0.24	0.57	3.86	0.28	0.63	4.10	0.20	0.51	4.40
Ours w/o PGO	0.79	0.83	0.56	0.85	0.90	0.40	0.78	0.76	0.70
Ours	**0.79**	**0.83**	**0.43**	**0.85**	**0.90**	**0.35**	**0.78**	**0.76**	**0.55**

[1] R means the Recall, P means the Precision and E denotes the RMSE[m].
[2] SPFDM represents SuperPoint Feature Detection and Matching, PA represents Pixel-wise Association, and PGO is Pose Graph Optimization.

TABLE III: Localization Comparison with Different Orientations on Our Datasets

Scenes	UG-0			UG-1			UG-2		
	R	P	E	R	P	E	R	P	E
90°	0.57	0.75	0.85	0.63	0.72	0.82	0.58	0.68	0.95
180°	0.77	0.54	0.54	0.82	0.88	0.32	0.81	0.84	0.45
0-360° change	0.64	0.75	0.94	0.72	0.80	0.60	0.68	0.76	0.70

[1] R , P , and E denote the same as in TABLE II.

We compared our methods with the following methods: VF with loop closure and map and SchurVINS (SV) [23] with with loop closure and map. HLoc is not compared [21] as the map generated by COLMAP is incomplete. As for learning-based systems like [8, 9], they were not included due to the lack of sufficient labeled data for supervised learning. Besides, we integrate FAST-LIO2 pose into the visual mapping of VF and SV and utilize the identical feature extraction and matching module in both VF and SV for fairness. The results are presented in TABLE II and Fig. 7. It show that our method achieved the highest recall, precision, and localization accuracy. Besides, the ablation analysis demonstrates that SP-based feature detection/matching and pixel-wise association dominate the localization performance improvements, while PGO contributes marginally. Furthermore, the proposed hierarchical and lightweight localization method maintains high accuracy while being computationally efficient. This indicates that our system is well-suited for automatic navigation tasks, which demand high localization accuracy with limited computational resources.

2) Evaluation on Orientation Change: Theoretically, the proposed algorithm is able to re-localization at almost any position and orientation within the map as four 90-degree-gapped keyframes were stored. To test the robustness to deviation, we maintained a consistent or varying angle deviation

TABLE IV: Localization Time Cost (ms)

Steps	Desktop	Xavier NX
Feature Detection	35.75	130.45
Submap Projection + Association	32.99	50.07
Image Retrieval	3.97	34.28
Feature Matching	31.64	164.32
PnP+RANSAC	10.29	24.34

during flight. We conducted comparisons on three datasets, with orientations of 90 degrees, 180 degrees, and varying between 0-360 degrees relative to the forward direction of mapping trajectory. The result is depicted in TABLE III. The experiments demonstrate that our algorithm can maintain accurate localization even with changes in orientation. In the 180-degree orientation experiment, results were more favorable. This is likely because this orientation resembles walking backwards, which causes keypoints in camera FoV to disappear more slowly, maintaining a higher recall rate and allowing for more frequent corrections. In contrast, for the 90-degree and 0-360-degree cases, the visual similarity between the query and reference frames is lower, leading to a reduced recall rate and less satisfactory results. Since the VF algorithm uses stereo triangulated points for mapping, its map is oriented in a single direction, making it unable to re-localize from other orientations within the map.

3) Localization Runtime Analysis: The localization runtime performance of the proposed system is measured in two different platforms: an Intel Core i9-13900H Desktop with GeForce RTX 4060 and a Nvidia Jetson Xavier NX (16G). The SuperPoint feature detector and SuperGlue feature matcher are speeded up using the TensorRT engine. The time cost result is presented in TABLE IV, in which the localization step is the sum of local feature extraction and tracking steps. The re-localization takes about 250 ms on the Xavier platform, which is sufficient for correcting VIO pose and is suitable for onboard applications.

V. Conclusion

In this paper, we present a hybrid visual-Lidar mapping system for visual re-localization, specifically designed for computation-limited platforms. By utilizing keyframes to associate visual features with Lidar data, our approach achieves high-precision re-localization while maintaining real-time performance, making it ideal for autonomous navigation tasks in challenging environments. Additionally, the use of omnidirectional keyframes allows the system to operate without orientation constraints. Future work will aim to further enhance scalability and robustness, ensuring seamless performance across a variety of environments.

References

[1] J. Miao, K. Jiang, T. Wen, Y. Wang, P. Jia, B. Wijaya, X. Zhao, Q. Cheng, Z. Xiao, J. Huang *et al.*, "A survey on monocular re-localization: From the perspective of scene map representation," *IEEE Transactions on Intelligent Vehicles*, 2024.

[2] K. Wang, F. Gao, and S. Shen, "Real-time scalable dense surfel mapping," in *2019 International conference on robotics and automation (ICRA)*. IEEE, 2019, pp. 6919–6925.

[3] T. Qin, P. Li, and S. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE transactions on robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

[4] J. L. Schönberger and J.-M. Frahm, "Structure-from-motion revisited," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[5] T. Caselitz, B. Steder, M. Ruhnke, and W. Burgard, "Monocular camera localization in 3d lidar maps," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 1926–1931.

[6] C. Zhang, H. Zhao, C. Wang, X. Tang, and M. Yang, "Cross-modal monocular localization in prior lidar maps utilizing semantic consistency," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 4004–4010.

[7] Y. Kim, J. Jeong, and A. Kim, "Stereo camera localization in 3d lidar maps. in 2018 ieee," in *RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1–9.

[8] S. S. Puligilla, M. Omama, H. Zaidi, U. S. Parihar, and M. Krishna, "Lip-loc: Lidar image pretraining for cross-modal localization," in *2024 IEEE/CVF Winter Conference on Applications of Computer Vision Workshops (WACVW)*. IEEE, 2024, pp. 939–948.

[9] J. Pan, X. Mu, T. Qin, C. Xu, and M. Yang, "2d-3d cross-modality network for end-to-end localization with probabilistic supervision," in *2024 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2024, pp. 906–912.

[10] Q. Shen, H. Zhao, W. Yan, C. Wang, T. Qin, and M. Yang, "Cross-modal visual relocalization in prior lidar maps utilizing intensity textures," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2024, pp. 10 650–10 655.

[11] H. Yu, W. Zhen, W. Yang, J. Zhang, and S. Scherer, "Monocular camera localization in prior lidar maps with 2d-3d line correspondences," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 4588–4594.

[12] J. He, H. Huang, S. Zhang, J. Jiao, C. Liu, and M. Liu, "Accurate prior-centric monocular positioning with offline lidar fusion," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 11 934–11 940.

[13] D. Cattaneo, M. Vaghi, S. Fontana, A. L. Ballardini, and D. G. Sorrenti, "Global visual localization in lidar-maps through shared 2d-3d embedding space," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 4365–4371.

[14] S. Lowry, N. Sünderhauf, P. Newman, J. J. Leonard, D. Cox, P. Corke, and M. J. Milford, "Visual place recognition: A survey," *ieee transactions on robotics*, vol. 32, no. 1, pp. 1–19, 2015.

[15] 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.

[16] T. Qin, J. Pan, S. Cao, and S. Shen, "A general optimization-based framework for local odometry estimation with multiple sensors," 2019.

[17] S. Carmichael, R. Agrawal, R. Vasudevan, and K. A. Skinner, "Spot: Point cloud based stereo visual place recognition for similar and opposing viewpoints," *arXiv preprint arXiv:2404.12339*, 2024.

[18] D. Gálvez-López and J. D. Tardós, "Bags of binary words for fast place recognition in image sequences," *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1188–1197, October 2012.

[19] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "Fast-lio2: Fast direct lidar-inertial odometry," *IEEE Transactions on Robotics*, vol. 38, no. 4, pp. 2053–2073, 2022.

[20] D. DeTone, T. Malisiewicz, and A. Rabinovich, "Superpoint: Self-supervised interest point detection and description," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2018, pp. 224–236.

[21] P.-E. Sarlin, C. Cadena, R. Siegwart, and M. Dymczyk, "From coarse to fine: Robust hierarchical localization at large scale," in *CVPR*, 2019.

[22] P.-E. Sarlin, D. DeTone, T. Malisiewicz, and A. Rabinovich, "Superglue: Learning feature matching with graph neural networks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 4938–4947.

[23] Y. Fan, T. Zhao, and G. Wang, "Schurvins: Schur complement-based lightweight visual inertial navigation system," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 17 964–17 973.