Large-scale Indoor Mapping with Failure Detection and Recovery in SLAM

Sharmin Rahman, Robert DiPietro, Dharanish Kedarisetti and Vinod Kulathumani

***Abstract*— This paper addresses the failure detection and recovery problem in visual-inertial based Simultaneous Localization and Mapping (SLAM) systems for large-scale indoor environments. Camera and Inertial Measurement Unit (IMU) are popular choices for SLAM in many robotics tasks (e.g., navigation) due to their complementary sensing capabilities and low cost. However, vision has inherent challenges even in well-lit scenes, including motion blur, lack of features, or even accidental camera blockage. These failures can cause drifts to accumulate over time and can severely impact the scalability of existing solutions to large areas. To address these issues, we propose an automatic map generation service with (i) a failure detection method based on visual feature tracking quality using a health tracker which identifies and discards faulty measurements and (ii) a continuous session merging approach in SLAM. Taken together, this allows us to handle erroneous data without any manual intervention, and allows us to scale to extremely large spaces. The proposed system has been validated on benchmark datasets. Also, experimental results on multiple custom large-scale grocery stores, each between 1700 m^2 to 3700 m^2, and duration 60 to 80 minutes, are presented. Our approach shows lowest error in all large-scale SLAM cases when compared with state-of-the-art visual-inertial SLAM packages, which often produce highly erroneous trajectories or lose track. Additionally, we provide dense 3D reconstruction with the presence of a depth camera by simply registering the point cloud from RGB-D image with respect to the SLAM generated trajectory – and the quality of the reconstruction illustrates the efficacy of our proposed method.**

I. INTRODUCTION

Mapping large-scale indoor environments such as warehouses, logistics centers, and smart shopping stores has immense significance in robotics, particularly in autonomous navigation where robots require detailed and accurate understanding of the surroundings to navigate through aisles in vast areas to perform tasks, e.g., deliver, retrieve, or stock products. By leveraging an accurate and up-to-date mapping layout, they can move around more efficiently and safely by optimizing their paths and avoiding obstacles to perform tasks reliably. However, usually large-scale mapping requires multiple expensive LiDARs, in combination with other sensors (e.g., high-resolution cameras), such as in NavVis [1] and LiDAR-SLAM [2], and incurs substantial computational costs for map generation and maintaining accuracy over time. In recent years, low-cost visual-inertial odometry (VIO) and SLAM approaches – e.g., OpenVINS [3] and ORB-SLAM3 [4] – have shown high-precision performance with accurate *scale* in indoor and outdoor environments. These methods handle small-scale benchmarks well, but do not handle large-scale and visually challenging scenes well [5], [6], [7]. In this paper, we explore opportunities within visual-inertial SLAM to automatically detect failure cases and gracefully recover to obtain a robust and accurate map in real-time. Fig. 1 shows the dense construction by registering the point clouds from the depth camera with our method. Note that depth images are used only for dense reconstruction.

S. Rahman, R. DiPietro, D. Kedarisetti, and V. Kulathumani are with the Amazon Web Services (AWS) applications, Amazon, MA, USA, `rahmansn, robdptro, dkedari, vkulathu@amazon.com`

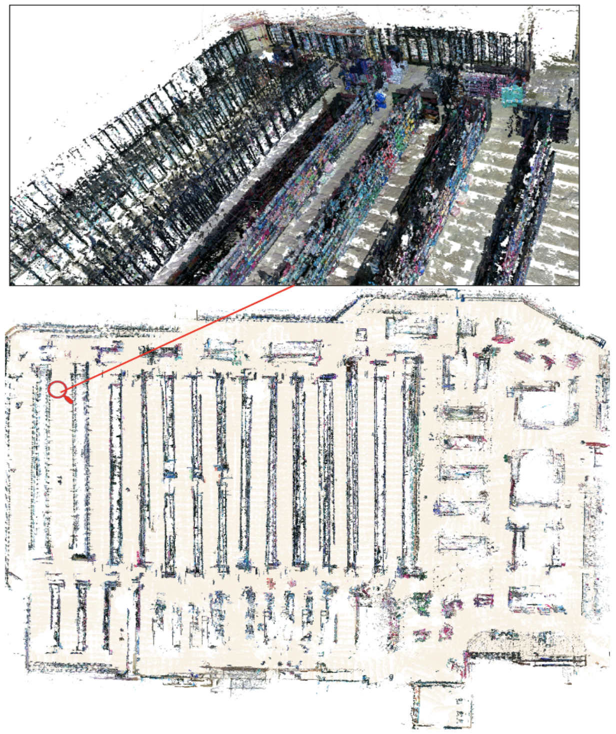

Fig. 1. Demonstration of our proposed SLAM method in 3D reconstruction (when Depth camera is available) in a large-scale grocery store covering an area of approx. 3700 m^2. Note, depth images are only used for dense reconstruction and not used for SLAM. *Top:* zoomed in 3D dense pointcloud pointed by the magnifying glass. *Bottom:* Top-down layout of the whole store.

In comparison to outdoor environments, indoor environments are less prone to severe appearance change (e.g., day vs night time and season change) and less affected by weather conditions (e.g., haziness, rain, and snow). However, indoor environments have their own challenges, including textureless regions, motion blur, flashed images by bright sunlight through glass windows and walls, glare, and dynamic objects in the scene – see Fig. 2. Exploiting geometric indoor structures, e.g., edges, lines, and planes [8], [9], [10], [11], can help with some of these cases, however, failures due to complete absence of any types of features are unavoidable. Sensor failures can also take place in long-duration collects

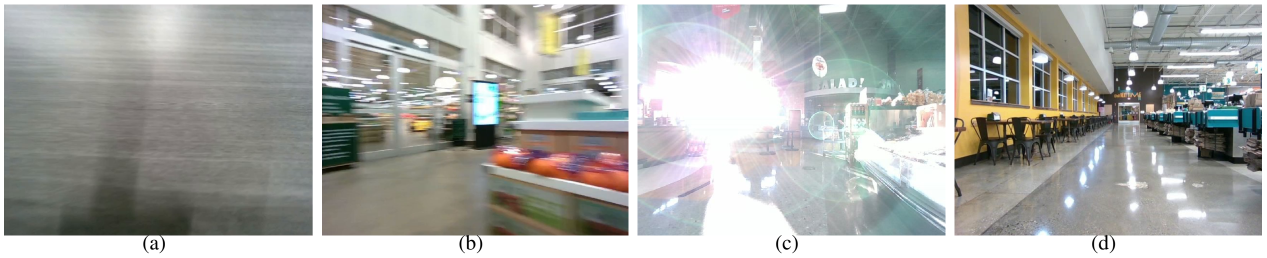

Fig. 2. Representative images from our datasets: (a) nearly textureless area, (b) severe motion blur, (c) bright sunlight, and (d) moving light glare.

and provide out-of-order sensor measurements. As pose estimation solely by IMU is not reliable for even one second, the only way to tackle the above situations is to detect them and resume tracking once visual scene quality is restored. In [12], we introduced SLAM health tracking by combining several simple metrics to provide live feedback on tracking quality. In this paper, we provide more efficient and robust metrics for determining track loss, including feature covisibility between frames, divergence based on velocity, and well conditioned VIO initialization. When tracking quality drops, a failure recovery mechanism takes place by creating a new VIO session where the last session ended and then whenever an overlap is found (by re-visiting a place) with any previous sessions, the new session gets merged by adjusting their poses and keypoints. Thus, we prevent any manual pre-processing without any extra time.

In this paper, we augment health tracking and continuous session merging into SVIn2 [6] to address the drifts and loss of localization in large-scale tracking. In addition, we extend it with multi-session capability for long-term updates. To validate the proposed approach, we assess performance on Hilti-Oxford [13] benchmark dataset and three large-scale grocery store datasets in a diverse set of conditions and compare it to other state-of-the-art opensource SLAM methods. In the absence of ground truth trajectories custom datasets, we used COLMAP [14] – a well known opensource Structure from Motion (SfM) library – as a baseline for comparing the trajectories from different methods. Our contributions are

- a reformulated and improved health tracker with more relevant and more efficient metrics
- automatic session merging with health-tracker feedback
- multi-session capability for long-term map updates
- demonstration of the proposed approach in multiple large-scale grocery stores, $1700\ \mathrm{m}^2$ to $3700\ \mathrm{m}^2$, where we observe robust localization and dense map reconstruction even where other state-of-the-art methods fail.

II. RELATED WORK

In recent years, visual-inertial fusion based navigation has gained much popularity as a means of precise localization both in indoor and outdoor environments, such as in augmented and virtual reality (AR/VR) [15], [16] and advanced driver assistance systems (ADAS) [17]. The state-of-the-art VIO and VI-SLAM approaches mostly fall into two categories – extended Kalman filtering (EKF) based and sliding-window optimization based. Multi-State Constraint Kalman Filter (MSCKF) [18] is the earliest VIO which captures the geometric constraints imposed by observing the same visual feature from multiple camera poses and exhibits linear complexity in the number of features. This work was later extended in several ways, e.g., to stereo vision [19]; square-root inverse sliding window filter (SR-ISWF) [20] for speeding up MSCKF while maintaining same level of accuracy; observability analysis by exploiting the environmental structure or the carrier (e.g., wheel) [21], [8], [22]; OpenVINS [3] based on sliding window EKF. Meanwhile, optimization-based approaches – such as OKVIS [23], ORB-SLAM3 [4], and VINS-Fusion [24] – solve a non-linear least-squares (NNLS) problem over a window of recent visual and inertial measurements and the old states are marginalized out to bound optimization complexity. Please refer to [25] for a comprehensive review and [26], [7] for comparison of the existing approaches in the literature.

While visual-inertial systems show significant performance improvement over vision-only, they can still fail due to common failure modes in real-world scenarios. The state-of-the-art SLAM algorithms are evaluated in [5] against a variety of scenarios (including fast motion, illumination variation, blur, and dynamic scenes) and their performances suggest the necessity of failure detection and recovery mechanism. Most of the methods [27], [28], [4] devise loop-closure and relocalization as a failure recovery mechanism. Loop-closure is an effective mechanism to guard drift accumulation over time in sliding window and marginalization-based state estimation – however, when the front-end completely fails to track, the re-initialization can be time consuming and depends on whether the system specific criteria are met or not. Indeed, after a complete track loss, most methods are unable to recover or miss a significant potion of trajectory during the recovery period, especially true for real-time large-scale datasets.

Our prior work SM/VIO [12] proposed a heath monitoring method for identifying failure modes to switch between VIO and model-based estimator. It estimates VIO divergence with relatively simple metrics, e.g., time to detect new keyframe, number of 3D tracked points, spread of keypoints throughout an image etc. However, those metrics might not be sufficient to reflect the true state of the estimator. In this work, we explain the necessity to reformulate them and propose improved metrics for health tracking. In addition, we propose an automatic session merging mechanism for recovery to enable robust, automated SLAM in extremely large indoor environments.

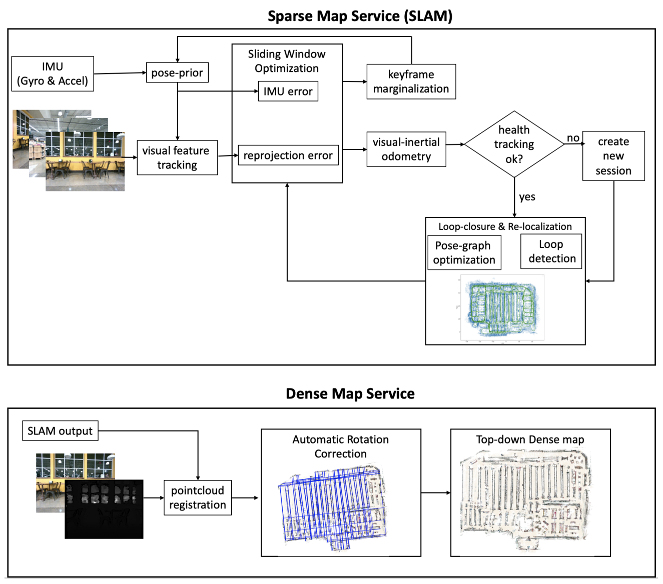

Fig. 3. Block diagram of the proposed system: a visual-inertial SLAM service with failure detection and recovery (top) and a dense map generation service in the presence of depth images (bottom).

III. PROPOSED METHOD

Fig. 3 presents an overview of the proposed approach: (i) a sparse mapping service which combines IMU and camera measurements into a multi-session SLAM system with failure detection and recovery and (ii) a dense mapping service (when depth camera is available) combining SLAM results with depth images to create a 3D dense reconstruction of the environment. Our SLAM method uses Sonar-Visual-Inertial navigation (SVIn2) [6] as backbone, which is a tightly-coupled keyframe-based SLAM system and fuses vision (mono or multi camera), acoustic range (optional), inertial, and water pressure (optional) measurements in a non-linear optimization framework for small to large scale challenging underwater environments. Though developed for underwater, it is adaptable for other domains including indoor and outdoor, by choosing a subset of different sensor configurations with a minimal requirement of a camera and an IMU. The cost function for reprojection error and IMU error follows the formulation of OKVIS [23], a state-of-the-art sliding window optimization based visual-inertial system. For more details on the error terms of other sensors (sonar and water pressure), please refer to [29], [6]. Additionally, to reduce drift, especially for long trajectories, SVIn2 supports bag-of-words (BoW) based loop-closing and relocalization. Our proposed method extends SVIn2's capability with 1) a SLAM health tracker, 2) a continuous session merging mechanism for graceful failure recovery, and 3) multi-session capability to combine data over different periods of time.

A. Notation

To describe our camera rig, we define the following coordinate frames: C_i refers to the i-th Camera, D refers to the depth camera if available, I refers to the IMU, and G refers to the global coordinate frame. A homogeneous transformation matrix ${}_X\mathbf{T}_Y = [{}_X\mathbf{R}_Y | {}_X\mathbf{p}_Y]$ transforms a pose – i.e., rotation ${}_X\mathbf{R}_Y$ (with equivalent quaternion representation ${}_X\mathbf{q}_Y$) and position ${}_X\mathbf{p}_Y$ – from coordinate frame Y to X. A dense point cloud, i.e., a set of 3D points, $\{\mathbf{p}_0, \mathbf{p}_1, ...\}$, in coordinate frame X is represented as ${}_X\mathbf{p}_{\text{dense}}$. Our proposed system estimating the robot state $\mathbf{x}_R$ is defined as:

$$\mathbf{x}_R \quad = \quad [{}_G\mathbf{p}_I^T, {}_G\,\mathbf{q}_I^T, {}_G\,\mathbf{v}_I^T, \mathbf{b}_g{}^T, \mathbf{b}_a{}^T]^T \tag{1}$$

which contains the position vector ${}_G\mathbf{p}_I$, the orientation as quaternion ${}_G\mathbf{q}_I$, and the linear velocity ${}_G\mathbf{v}_I$ with respect to the global coordinate G. The gyroscopes and accelerometers bias are expressed as $\mathbf{b}_g$ and $\mathbf{b}_a$ respectively.

B. Health Tracker Metrics

Previous metrics and their limitations. VIO health tracking was introduced in our previous work SM/VIO [12], with relatively simple metrics, specifically: i) time elapsed before a new keyframe is detected, ii) number of 3D tracked keypoints, iii) number of detected features per image quadrant, and iv) ratio of newly created keypoints to total keypoints. These are important parameters representing visual feature tracking quality. However, their ability to estimate real tracking quality is limited. For example, in criterion (i), if the camera is in a no-motion state, then a new keyframe will never be created, and failure will be incorrectly reported. For (ii), using 10 to 20 tracked keypoints has been empirically found to be a good tracking indicator, but cases arise in which a few high-quality features can be found even in seemingly featureless areas, e.g., corners on a blank wall. When combined with a pose prior from the IMU, even a few features can reliably estimate accurate pose. Criterion (iii) ensures an approximately uniform features within the image, but in challenging scenarios, many features may not be present and yet still be sufficient for tracking.

Current metrics. In our multi-session SLAM framework, it is important to determine the true state of the VIO estimator. Otherwise, either too many small sessions which may not have any overlapping regions between them to merge will be created or will have incorrect trajectories when it fails. We borrow two metrics from the above – the number of 3D tracked keypoints with a much lower threshold (5) and the ratio of newly detected features – and we propose more reliable and accurate metrics for health tracker:

1) Covisibility of features between frames. A good feature should be observable from multiple frames and can be tracked for long duration. It ensures the 3D point created by triangulating features from these frames has less uncertainty. For example, the corner of a wall or features at the end a long passage are good features to track. On the other hand, bad features – e.g., repetitive patterns on a fabric, or glares from a light – disappear soon after detection. We set this threshold to be 5 (i.e., we need at least 5 strong features at every moment).
2) Divergence based on robot velocity. This is a threshold based on carrier's velocity, e.g., for a drone, its maximum allowable velocity. When VIO starts diverging, it shows high pose error and falsely indicates large jump

in motion between frame-to-frame. As such, it works as an effective failure indicator.

3) well conditioned VIO initialization. As the features are not triangulated before initialization, there are no 3D points. We ensure the VIO estimator initializes only when there is strong feature covisiblity between frames, and otherwise waits until good visual conditions appear. The initialization also requires translational motion. Otherwise if the estimator initializes at bad condition (e.g, camera is facing a blank wall), it will fail immediately due to lack of reliable features.

C. *Continuous session merging in SLAM*

When a tracking failure is detected by the health tracker, a new VIO session is created alongside the older session. Our multi-session SLAM scheme depends on visual similarities to join maps collected in subsequent sessions. For visual place recognition, a bag of binary words [30] database is maintained which contains the descriptors of the keypoints detected in each keyframe during local front-end tracking. A global pose-graph optimization (PGO) is performed using of all the poses in different sessions, which helps achieve global consistency. Subsequently, a local re-projection error based optimization is performed to ensure tightly-coupled fusion between the local tracking module and global pose-graph, which is critical for large-scale mapping. In the following we describe each of these components in details.

Multi-session merging with geometric verification. For both multi-session mapping and loop closing, our method maintains a pose-graph to represent the connections between keyframes. More precisely, it is a weighted graph where a node represents a keyframe and an edge connects two keyframes if the weight defined by the ratio of common map points between them is greater than η, where $\eta = 0.75$ in experiments. As soon as a keyframe is marginalized out of the local optimization window, it is added to the pose-graph, *if* the health tracker indicates that it has good health. Our multi-session mechanism is designed to support both:

i) Long-term map update to incorporate any large change (e.g., structural change in layout) happened over a long period of time. A map is saved into disk as pose-graph with its all components – keyframe poses, 3D landmark locations, keyframe descriptors and weighted edge information. When the old map needs to be updated, the re-mapping data are collected only in the changed areas rather than over the whole environment. The previously saved map as pose-graph gets automatically merged with the new map of changed areas when some overlapping areas are found between them.

ii) Continuous session reset and merging to handle bad quality data efficiently by notifying a failure state. Though, similar strategy as *(i)* can be applied, i.e., in case of failure saving the current session map and loading subsequent sessions to merge them altogether – however, the save/load process is time and space consuming and has significant affects on large datasets. We address this problem by automatic session reset, i.e., when the SLAM health tracker reports a failure case, we reset the VIO estimator while keeping the pose-graph alive and unchanged. The estimator starts a new session and waits for initialization as mentioned in Section III-B with its coordinate frame origin being at the last healthy keyframe pose, ${}_G\mathbf{T}_{\text{old}}$. The new session trajectory, ${}_{\text{old}}\mathbf{T}_{\text{new}}$ and sparse map points, ${}_{\text{old}}\mathbf{p}_{\text{new}}$ from triangulated features are defined in the global frame G as:

$$\begin{aligned} {}_G\mathbf{T}_{\text{new}} &= {}_G\mathbf{T}_{\text{old}} \cdot_{\text{old}} \mathbf{T}_{\text{new}} \\ {}_G\mathbf{p}_{\text{new}} &= {}_G\mathbf{T}_{\text{old}} \cdot_{\text{old}} \mathbf{p}_{\text{new}} \end{aligned} \tag{2}$$

${}_G\mathbf{T}_{\text{new}}$ and ${}_G\mathbf{p}_{\text{new}}$ can be temporarily inconsistent, but get refined when loop-closures are found.

The session merging process is similar to the loop-closing and relocalization process of SVIn2 [6], here we describe how it works when there are multiple sessions. When a keyframe drops off from the local optimization window and its health status is 'good', it gets added into the pose-graph. For each new keyframe, this module searches for candidates with visual similarities in the BoW database. A query to the database only returns candidates which have a score greater than or equal to the neighboring keyframes of that node in the pose-graph and are outside the current marginalization window. If a match (or loop) is detected in any of the previous sessions, the candidate with the highest score is retained and 2D-to-2D descriptor correspondences between the newly added keyframe and loop candidate keyframe is established. Then a geometric validation is performed via a 3D-to-2D correspondences between the common landmarks between them with outlier rejection by PnP RANSAC. Finally, a global optimization aligns the poses between multiple sessions by 6-DoF pose-graph optimization. The error term between two keyframes i and j – one being the new keyframe and the other is either the connected neighboring keyframes or the loop candidate – is defined in the tangent space as: $\mathbf{e}_{\mathbf{x_T}}^{i,j} = \Delta\mathbf{T}_{ij}\hat{\mathbf{T}}_i\hat{\mathbf{T}}_j^{-1}$ where $\Delta\mathbf{T}_{ij} = \mathbf{T}_j\mathbf{T}_i^{-1}$ and $\hat{(\cdot)}$ denotes the estimated pose by the local VIO. With a Huber loss function ρ to down-weigh any incorrect loops, the cost function to minimize is:

$$\underset{\mathbf{x_T}}{\arg\min} \quad \sum_{i,j} \|\mathbf{e}_{\mathbf{x_T}}^{i,j}\|^2 + \sum_{(i,j)\in Loop} \rho(\|\mathbf{e}_{\mathbf{x_T}}^{i,j}\|^2) \tag{3}$$

Propagating drift correction to VIO estimator. An alignment between the optimized pose-graph and front-end VIO tracking module is performed by a local reprojection error optimization for the matched landmarks with loop candidate. It ensures that the tracking module follows the drift correction by the PGO for generating subsequent poses. The cost function to minimize:

$$\underset{\mathbf{x_T}}{\arg\min} \quad \sum_{i=1}^{n}\sum_{k=1}^{K}\sum_{j\in Loop(i,k)} \mathbf{e}_r^{i,j,k^T}\mathbf{P}_r^k\mathbf{e}_r^{i,j,k} \tag{4}$$

where $\mathbf{e}_r$ denotes the re-projection error in i^{th} camera with landmark index j observed in the k^{th} camera frame. $\mathbf{P}_r^k$ information matrix (weights) of visual landmarks. $\mathbf{e}_r$ follows standard formulation (i.e., the difference between a keypoint measurement and the corresponding landmark projection), please refer to [6] for details. Additionally, for a consistent sparse global map generation, we follow [31] (computationally linear with the number of landmarks) to update landmark positions after each session merging in real-time, without any computationally intensive bundle adjustment (BA).

D. Top-down dense map generation from depth images

If a Depth camera is available, the depth images are registered with respect to the SLAM poses to generate the dense 3D point cloud in the global coordinate. At time t, the local point cloud ${}_D\mathbf{p}_{\text{dense}}{}^t$ with their color (from the RGB image) and depth (from depth image) values are transformed into global point cloud, ${}_G\mathbf{p}_{\text{dense}}{}^t$ by the SLAM pose, $\mathbf{x_T} = {}_G\, \mathbf{T}_{C_i}{}^t$, i.e.,

$$ {}_G\mathbf{p}_{\text{dense}}{}^t \quad = \quad {}_G\mathbf{T}_{C_i}{}^t \cdot {}_{C_i}\,\mathbf{T}_D{}^t \cdot {}_D\,\mathbf{p}_{\text{dense}}{}^t \qquad (5) $$

At last, to create a up-right dense map, we use Hough line transform algorithm [32] to detect horizontal and vertical lines formed by the prominent fixtures in the image and calculate their slopes. Then all the poses and the whole point cloud are transformed according to the median of the slopes. The depth images are not used in the SLAM (i.e., no RGB-D SLAM), as firstly, it is computationally intensive for large-scale mapping in real-time and secondly, our preliminary experiments show depth images do not have much qualitative advancement over visual-inertial SLAM. Hence, we only use depth images for registration with respect to SLAM poses.

IV. EXPERIMENTAL RESULTS

We validated our proposed method on a benchmark named Hilti-Oxford dataset [13] and custom collected large-scale datasets. All experiments were performed in a laptop with an Intel i7 CPU (8 cores) @ 3.0 GHz, 16 GB RAM and in real-time configuration with ROS Noetic. For quantitative evaluation and plotting trajectories, we used evo [33] – an opensource package for odometry and SLAM evaluation.

A. Validation on benchmark dataset

The Hilti-Oxford dataset provides a large range of real-world challenging scenarios for SLAM, e.g., featureless areas and varying illumination conditions, collected on construction sites and Sheldonian Theatre in Oxford. The sensor platform contains multiple cameras, LiDAR, and inertial sensors and millimeter level accurate 6 DOF or 3 DOF ground truth poses are provided for each sequence. The 2022 dataset includes 7 short *odometry* sequences each ranging from 1 minute to 6 minutes and a long *SLAM* sequence with 16 minutes of duration in the Sheldonian Theatre revisiting the ground hall multiple times for loop closures.

As our proposed system is visual-inertial, we only use a monocular camera and an IMU to evaluate its performance as well as other state-of-the-art SLAM systems, namely VINS-Fusion [24], OpenVINS [3], and ORB-SLAM3 [4]. As the original OpenVINS is a VIO estimator, for the Sheldonian SLAM sequence we additionally compared OpenVINS_w_LC (published by the authors) which adapts loop-closure from VINS-Fusion in a loosely coupled manner. Table I

TABLE I
The mean absolute translation error (RMSE) in meters with standard deviation over three runs using only monocular camera (cam0) and an IMU for each Hilti-Oxford sequence. The best performing algorithm is highlighted in **bold**, × indicates tracking failure, and n/a showing not applicable method.

	our_SLAM	VINS-Fusion [24]	OpenVINS [3]	OpenVINS_w_LC	ORB-SLAM3 [4]
Exp 04: Construction L1	**1.1994 +/- 0.04**	×	4.3561 +/- 0.40	n/a	×
Exp 05: Construction L2	0.7348 +/- 0.17	×	**0.1630 +/- 0.007**	n/a	×
Exp 06: Construction L3	0.4323 +/- 0.15	×	**0.1656 +/- 0.01**	n/a	×
Exp 10: Cupola 2	0.4152 +/- 0.03	×	**0.3004 +/- 0.03**	n/a	×
Exp 14: Basement 2	0.1981 +/- 0.006	×	**0.0892 +/- 0.012**	n/a	×
Exp 16: Attic	0.54821 +/- 0.06	×	**0.4807 +/- 0.12**	n/a	×
Exp 18: Corridor	0.4040 +/- 0.04	×	**0.2112 +/- 0.03**	n/a	×
Exp 23: Sheldonian SLAM	**0.5904 +/- 0.02**	×	1.2921 +/- 0.10	0.6474 +/- 0.32	×

shows the Root Mean Square Error (RMSE) of the Absolute Pose Error (APE) for the translation part after SE(3) Umeyama alignment with EVO [33]. For every sequence, each algorithm has been 3 times and their average with standard deviation is also reported. Our method and OpenVINS are able to complete all the sequences without losing track. OpenVINS shows the lowest RMSE in most of the short odometry sequences (i.e., top 7 rows – Exp 04 to Exp 18), its *zero velocity update* module helps to reduce uncertainty based on no-motion knowledge. However, the benefit of our proposed method becomes evident in the long SLAM sequence (i.e., bottom most row – Exp 23) where our method shows lowest RMSE and OpenVINS even with loop closure (OpenVINS_w_LC) shows higher error. ORB-SLAM3 fails in all sequences due to the featureless regions, it initiates multiple new maps after each tracking failure but eventually does not succeed in recovering. Similarly, VINS-Fusion fails in all sequences. Fig. 4 shows the successful trajectories for each method together with the ground truth for one of the Hilti-Oxford sequences.

B. Validation on custom large-scale dataset

The proposed system was evaluated in three large-scale, long-duration retail datasets to capture real-world challenges often overlooked in benchmark datasets. These datasets are ranked in order of increasing difficulty, i.e., Store 1 being the least difficult and Store 3 the most difficult. The experimental data were collected with an Intel Realsense D455 RGB-D

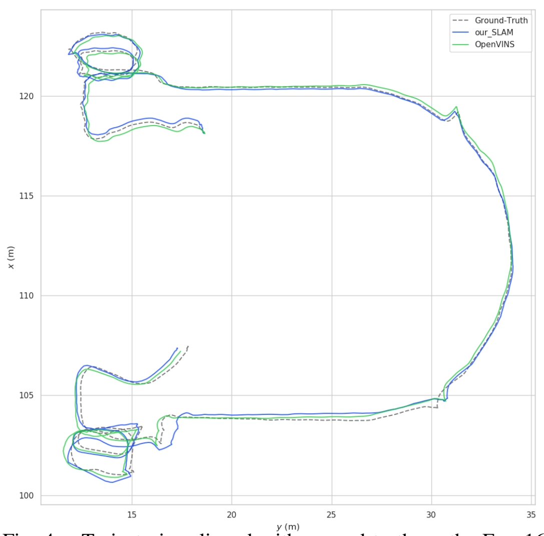

Fig. 4. Trajectories aligned with ground truth on the Exp 16 sequence of the Hilti-Oxford dataset [13].

camera. All the datasets display challenging visual conditions as shown in Fig. 2. In particular:

- Store 1 (difficulty level 1): covers an area of 2250 m^2 over a duration of 66.5 minutes. It revisits some places to facilitate loop-closure, especially at the end where most of the store is revisited with a fast zigzag patterned motion.
- Store 2 (difficulty level 2): the largest store covering an area of 3700 m^2 over a duration of 63.8 minutes. There are very few loop-closures, hence the frontend VIO estimates should have less drift. The scenes contain many instances of severe motion blur during fast turning and moving glares on floor.
- Store 3 (difficulty level 3): the longest dataset with a duration of 80 minutes and covers an area of 1700 m^2. There are abundance of textureless areas and many instances of very abrupt motions resulting acute motion blur in the entire duration of the dataset.

1) Comparison with state-of-the-art SLAM systems: We evaluated state-of-the-art visual-inertial SLAM systems which support a global pose optimization module via loop-closure for fair comparison with our proposed method on large-scale datasets. Fig. 5 - Fig. 7 show the performances of VINS-Fusion [24], OpenVINS [3], OpenVINS_w_LC, and ORB-SLAM3 [4].

For Store 1 and 2, Fig. 5(b) and Fig. 6(b) show the generated trajectories from each method. Only our method and VINS-Fusion are able to maintain their tracks. Though VINS-Fusion tracks completely, there are large jumps in poses (especially towards the end) as the PGO module struggles to support computation for increasing number of keyframes (see Appendix Fig. 9 for a zoomed-in view). In Store 3 – Fig. 7(b), VINS-Fusion loses track at 63.25 minutes due to textureless areas, and failed to track for the remaining 16.75 minutes. OpenVINS, OpenVINS_w_LC, and ORB-SLAM3 tracked partially in all the datasets. Table II shows when each system lost track. OpenVINS, being an odometry method, shows drifts in poses that accumulate over time. With the loop-closure module in OpenVINS_w_LC, drifts are partially corrected, however, in each of the datasets, it fails within a few minutes. We believe it fails because of the high volume of loop data. ORB-SLAM3 also tracks for limited duration, even after several trials to re-initialize when it loses track. In addition, the local mapping stops when the global bundle adjustment is performed after every loop-closure, leading to track loss in these large datasets. In all of the datasets, our method worked for the entire duration by using health metrics and automatically recovering from failure. The dense maps in Fig. 5(a) - Fig. 7(a) validate the correctness of trajectories by the proposed method.

TABLE II

Deviation from (noisy, only partially-complete) COLMAP baseline for each algorithm, along with percentage of time successfully tracked (throughout all stores). Mean absolute translation error (RMSE) is in units of meters. The best performing algorithm is highlighted in **bold** and $\times$ indicates failure.

	our_SLAM	VINS-Fusion [24]	OpenVINS [3]	OpenVINS_w_LC	ORB-SLAM3 [4]
Complexity1: Store 1	**0.1407**	0.1941	0.4273	$\times$	$\times$
Tracking %	**100%**	**100%**	46.6%	5.6%	7.7%
Complexity2: Store 2	**0.7719**	0.8039	0.9873	$\times$	$\times$
Tracking %	**100%**	**100%**	70.5%	8.8%	17.2%
Complexity3: Store 3	**0.2981**	0.3192	$\times$	$\times$	$\times$
Tracking %	**100%**	69.1%	7.5%	7%	4.5%

2) Quantitative analysis with COLMAP as a comparative baseline: With the absence of ground truth in the indoor environments (e.g., GPS), we use COLMAP [14] (a multi-view stereo based Structure-from-Motion algorithm) for generating a comparative baseline trajectory to evaluate our system with the state-of-the-art methods. However COLMAP should not be considered ground truth: first, scale cannot be recovered with monocular bundle adjustment, and second, even after much manual intervention, it is only able to reconstruct small areas, and often with significant error. This is true despite running our experiments with multiple Nvidia Tesla V100 GPUs. We selected small segments with good visual conditions, taking 3 frames per second and enabled loop detection by vocabulary tree search. Still, it generates incomplete and incorrect trajectories, especially visible in Fig. 6(c) (when turning around). Table II shows the RMSE of the APE with respect to the translation part calculated by EVO [33] with Sim(3) Umeyama alignment for COLMAP. In each datasets, our method shows the lowest RMSE. Fig. 5(c) - Fig. 7(c) show the aligned partial trajectories with COLMAP.

C. Long-term map update example

Fig. 8 shows an old map being updated with new changes by collecting data only from the changed areas 4 months apart. The two maps get merged when there are overlapping

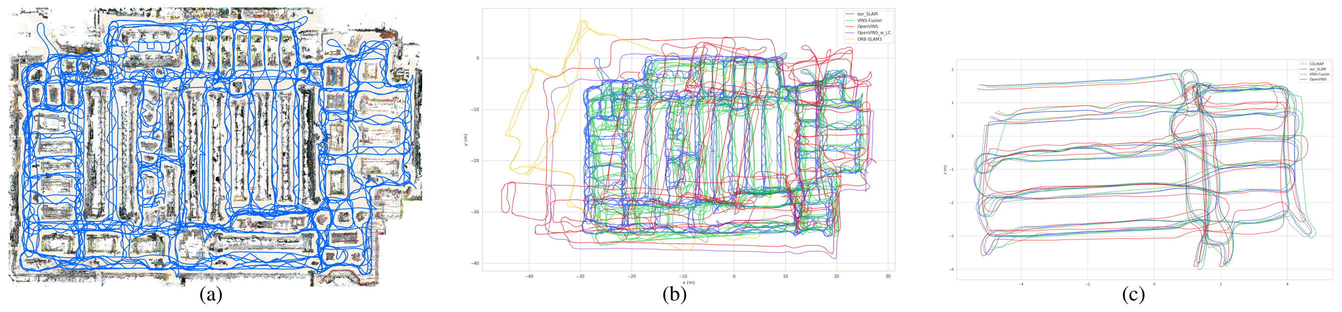

Fig. 5. Large store 1 (difficulty level: medium) dataset with an area of 2250 m^2 and 66.5 minutes duration. (a) Top-down dense map and trajectory produced by our SLAM. (b) Trajectories generated by the state-of-the-art VIO algorithms on the whole dataset. (c) sim3 (similarity transformation) aligned trajectories with COLMAP (comparative baseline) produced partial segment having 452.4 m of path length and 7.2 minutes of duration.

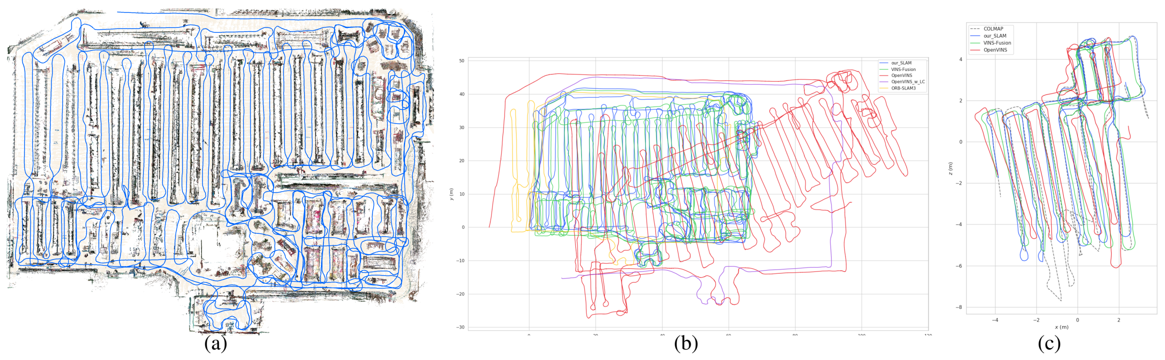

Fig. 6. Large store 2 (difficulty level: harder) with an area of 3700 m^2 and 63.8 minutes duration. (a) Top-down dense map and trajectory produced by our SLAM. (b) Trajectories generated by the state-of-the-art VIO algorithms on the whole dataset. (c) sim3 (similarity transformation) aligned trajectories with COLMAP (comparative baseline) produced partial segment having 486.5 m of path length and 12.1 minutes of duration.

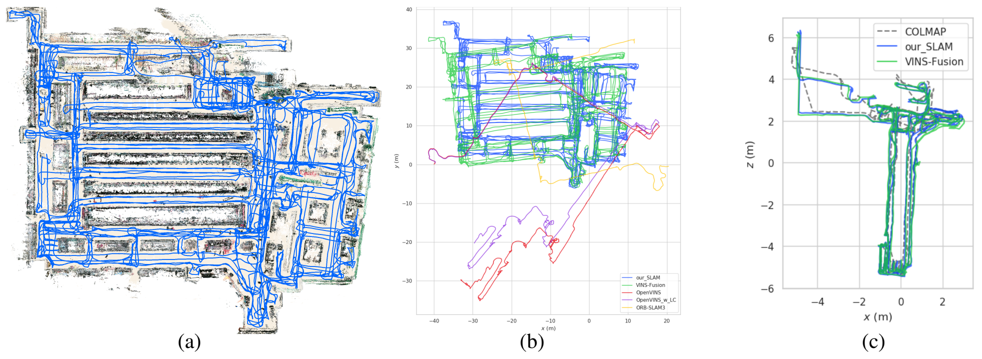

Fig. 7. Large store 3 (difficulty level: hardest) with an area of 1700 m^2 and 80 minutes duration. Note that the store is physically oblique on the right side (i.e., it is not a reconstruction error). (a) Top-down dense map and trajectory produced by our SLAM. (b) Trajectories generated by the state-of-the-art VIO algorithms on the whole dataset. (c) sim3 (similarity transformation) aligned trajectories with COLMAP (comparative baseline) produced partial segment having 558.9 m of path length and 12.4 minutes of duration.

areas (as little as fraction of a meter) between them. We verified the reconstruction accuracy by comparing the lengths (distance between two points) of reconstructed fixtures with manual measurements, resulting into an error $< 0.2\%$ (off by several cm).

V. CONCLUSIONS

In this paper, we propose methods for automatic failure detection, recovery, and session merging in visual-inertial SLAM in order to enable robust mapping in extremely large-scale indoor environments. By relying on covisibility of features across frames, consistency with estimated velocity, and well conditioned VIO initialization, we are able to generate high-quality maps in large indoor environments – up to 3700 m^2 in our experiments – without any manual intervention. We demonstrate that this approach reduces error significantly when compared with widely-used systems such as VINS-Fusion, OpenVINS, ORB-SLAM3, and COLMAP, which are often exhibit extreme distortion or even fail entirely, even after extensive manual tuning. We demonstrate the utility of our system by presenting dense reconstructions of multiple full-size grocery stores.

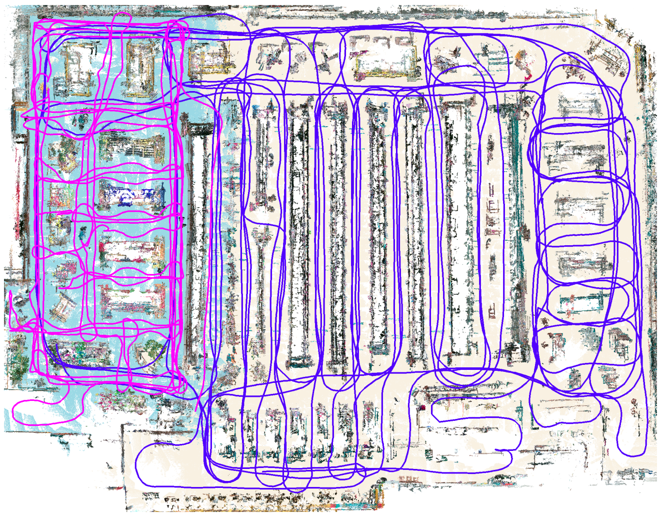

Fig. 8. Long-term map update demonstration with collects 4 months apart in a 2100 m^2 store. The light-blue background (with magenta trajectory) denotes the old map, and the light-brown background (with blue trajectory) denotes the new map.

Appendix

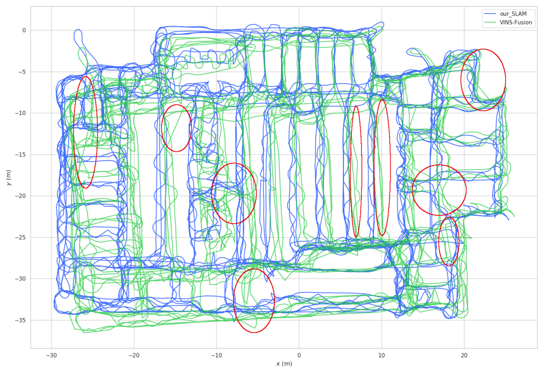

Fig. 9. Qualitative comparison between our SLAM and VINS-Fusion in large store 1. Some of the high jumps in VINS-Fusion's trajectory are highlighted in red ellipse.

References

[1] NavVis, "A wearable mobile 3D scanner," URL:https://www.navvis.com/.

[2] D. Lee, S. Ryu, S. Yeon, Y. Lee, D. Kim, C. Han, Y. Cabon, P. Weinzaepfel, N. Guérin, G. Csurka, *et al.*, "Large-scale localization datasets in crowded indoor spaces," in *Proc. CVPR*, 2021, pp. 3227–3236.

[3] P. Geneva, K. Eckenhoff, W. Lee, Y. Yang, and G. Huang, "Openvins: A research platform for visual-inertial estimation," in *Proc. ICRA*. IEEE, 2020, pp. 4666–4672.

[4] 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 Trans. Robot.*, vol. 37, no. 6, pp. 1874–1890, 2021.

[5] M. Bujanca, X. Shi, M. Spear, P. Zhao, B. Lennox, and M. Luján, "Robust slam systems: Are we there yet?" in *Proc. IROS*. IEEE, 2021, pp. 5320–5327.

[6] S. Rahman, A. Quattrini Li, and I. Rekleitis, "SVIn2: A multi-sensor fusion-based underwater slam system," *Int. J. Robot. Res.*, vol. 41, no. 11-12, pp. 1022–1042, 2022.

[7] B. Joshi, S. Rahman, M. Kalaitzakis, B. Cain, J. Johnson, M. Xanthidis, N. Karapetyan, A. Hernandez, A. Quattrini Li, N. Vitzilaios, and I. Rekleitis, "Experimental Comparison of Open Source Visual-Inertial-Based State Estimation Algorithms in the Underwater Domain," in *Proc. IROS*, 2019.

[8] H. Yu and A. I. Mourikis, "Vision-aided inertial navigation with line features and a rolling-shutter camera," in *Proc. IROS*. IEEE, 2015, pp. 892–899.

[9] Y. Yang and G. Huang, "Aided inertial navigation with geometric features: Observability analysis," in *Proc. ICRA*. IEEE, 2018, pp. 2334–2340.

[10] D. G. Kottas and S. I. Roumeliotis, "Exploiting urban scenes for vision-aided inertial navigation." in *Proc. RSS*, 2013.

[11] J. J. Tarrio and S. Pedre, "Realtime edge based visual inertial odometry for MAV teleoperation in indoor environments," *J. Intell. Robot. Syst.*, pp. 235–252, 2017.

[12] B. Joshi, H. Damron, S. Rahman, and I. Rekleitis, "SM/VIO: Robust underwater state estimation switching between model-based and visual inertial odometry," pp. 5192–5199, 2023.

[13] L. Zhang, M. Helmberger, L. F. T. Fu, D. Wisth, M. Camurri, D. Scaramuzza, and M. Fallon, "Hilti-oxford dataset: A millimeter-accurate benchmark for simultaneous localization and mapping," *IEEE Robot. Autom. Lett.*, vol. 8, no. 1, pp. 408–415, 2023.

[14] J. L. Schönberger, E. Zheng, M. Pollefeys, and J.-M. Frahm, "Pixelwise view selection for unstructured multi-view stereo," in *Proc. ECCV*, 2016.

[15] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM: A Versatile and Accurate Monocular SLAM System," *IEEE Trans. Robot.*, vol. 31, no. 5, pp. 1147–1163, 2015.

[16] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *IEEE and ACM Int. Symp. on Mixed and Augmented Reality*, 2007, pp. 225–234.

[17] J. Cheng, L. Zhang, Q. Chen, X. Hu, and J. Cai, "A review of visual slam methods for autonomous driving vehicles," *Engineering Applications of Artificial Intelligence*, vol. 114, p. 104992, 2022.

[18] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint Kalman filter for vision-aided inertial navigation," in *Proc. ICRA*. IEEE, 2007, pp. 3565–3572.

[19] K. Sun, K. Mohta, B. Pfrommer, M. Watterson, S. Liu, Y. Mulgaonkar, C. J. Taylor, and V. Kumar, "Robust stereo visual inertial odometry for fast autonomous flight," *IEEE Robot. Autom. Lett.*, vol. 3, no. 2, pp. 965–972, 2018.

[20] K. Wu, A. M. Ahmed, G. A. Georgiou, and S. I. Roumeliotis, "A square root inverse filter for efficient vision-aided inertial navigation on mobile devices." in *Proc. RSS*, vol. 2. Rome, Italy, 2015, p. 2.

[21] J. A. Hesch, D. G. Kottas, S. L. Bowman, and S. I. Roumeliotis, "Camera-imu-based localization: Observability analysis and consistency improvement," *Int. J. Robot. Res.*, pp. 182–201, 2014.

[22] K. J. Wu, C. X. Guo, G. Georgiou, and S. I. Roumeliotis, "VINS on wheels," in *Proc. ICRA*. IEEE, 2017, pp. 5155–5162.

[23] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual-inertial odometry using nonlinear optimization," *Int. J. Robot. Res.*, vol. 34, no. 3, pp. 314–334, 2015.

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

[25] G. Huang, "Visual-inertial navigation: A concise review," in *Proc. ICRA*. IEEE, 2019, pp. 9572–9582.

[26] J. Delmerico and D. Scaramuzza, "A benchmark comparison of monocular visual-inertial odometry algorithms for flying robots," in *Proc. ICRA*, 2018.

[27] B. Williams, G. Klein, and I. Reid, "Automatic relocalization and loop closing for real-time monocular slam," *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, no. 9, pp. 1699–1712, 2011.

[28] ——, "Real-time slam relocalisation," in *Proc. ICCV*. IEEE, 2007, pp. 1–8.

[29] S. Rahman, A. Quattrini Li, and I. Rekleitis, "Sonar Visual Inertial SLAM of Underwater Structures," in *Proc. ICRA*, 2018.

[30] D. Gálvez-López and J. D. Tardos, "Bags of binary words for fast place recognition in image sequences," *IEEE Trans. Robot.*, vol. 28, no. 5, pp. 1188–1197, 2012.

[31] B. Joshi, M. Xanthidis, S. Rahman, and I. Rekleitis, "High definition, inexpensive, underwater mapping," in *Proc. ICRA*. IEEE, 2022, pp. 1113–1121.

[32] R. O. Duda and P. E. Hart, "Use of the hough transformation to detect lines and curves in pictures," *Communications of the ACM*, vol. 15, no. 1, pp. 11–15, 1972.

[33] M. Grupp, "evo: Python package for the evaluation of odometry and slam." https://github.com/MichaelGrupp/evo, 2017.