MIMOSA: A Multi-Modal SLAM Framework for Resilient Autonomy against Sensor Degradation

Nikhil Khedekar, Mihir Kulkarni, and Kostas Alexis

***Abstract*— This paper presents a framework for Multi-Modal SLAM (MIMOSA) that utilizes a nonlinear factor graph as the underlying representation to provide loosely-coupled fusion of any number of sensing modalities. Tailored to the goal of enabling resilient robotic autonomy in GPS-denied and perceptually-degraded environments, MIMOSA currently contains modules for pointcloud registration, fusion of multiple odometry estimates relying on visible-light and thermal vision, as well as inertial measurement propagation. A flexible back-end utilizes the estimates from various modalities as relative transformation factors. The method is designed to be robust to degeneracy through the maintenance and tracking of modality-specific health metrics, while also being inherently tolerant to sensor failure. We detail this framework alongside our implementation for handling high-rate asynchronous sensor measurements and evaluate its performance on data from autonomous subterranean robotic exploration missions using legged and aerial robots.**

I. INTRODUCTION

Autonomous robotic systems are being increasingly utilized in a wide variety of applications, such as inspection or transportation [1, 2], and allow for important tasks to be undertaken efficiently, with reduced costs and maximized safety for humans. Towards the ubiquitous deployment of robotic systems across application domains and environments, a critical challenge that needs to be reliably addressed is that of resilient Simultaneous Localization And Mapping (SLAM) facilitating "any-time" and "any-place" autonomy, including GPS-denied and possibly sensor-degraded conditions. Responding to the challenge, the research community has contributed pioneering works exploiting LiDAR-based odometry and mapping [3, 4], visual-inertial systems [5, 6], thermal vision-based solutions [7, 8], and other techniques. However, SLAM frameworks that rely on a single exteroceptive modality (e.g., LiDAR or vision) are susceptible to sensor degradation specific to the sensor type. For example, LiDAR-based methods can face geometric degeneracy and become ill-conditioned in self-similar environments [9], while vision-based techniques can drift and fail in poor-illumination and low-texture settings, and both LiDAR and visual systems can get severely degraded in obscurant-filled conditions (e.g., dust, smoke). A possible solution is the design of multi-modal SLAM frameworks combining complementary sensor data, including our previous work on CompSLAM [10], and community contributions [10–19].

This material was supported by the Research Council of Norway under project REDHUS (grant number 317773).

The authors are with the Autonomous Robots Lab, Norwegian University of Science and Technology (NTNU), O. S. Bragstads Plass 2D, 7034, Trondheim, Norway nikhil.v.khedekar@ntnu.no

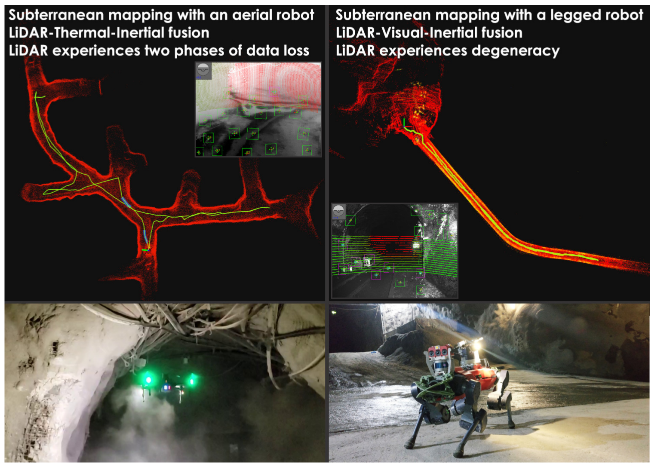
Subterranean mapping with an aerial robot
LiDAR-Thermal-Inertial fusion
LiDAR experiences two phases of data loss
Subterranean mapping with a legged robot
LiDAR-Visual-Inertial fusion
LiDAR experiences degeneracy

Fig. 1. Indicative results on multi-modal localization and mapping in subterranean environments using aerial and legged robots.

Motivated by the experience of the the severe conditions of perceptual degradation encountered in the DARPA Subterranean Challenge [20], where ground and aerial robots were tasked to explore and map diverse underground environments such as mines and caves, in this work we contribute MIMOSA, a new and versatile MultI-MOdal SlAm framework that aims to facilitate resilient long-term localization and mapping within GPS-denied environments that may present diverse sensor degradation challenges including but not limited to a) geometric self-similarity, b) presence of obscurants such as dust and smoke, c) poor or complete lack of illumination, and d) low-texture across large-scale complex geometries. The design goals of MIMOSA are to present a) robust performance against conditions of sensor degradation, b) redundancy against degeneracy or failure of any of the exteroceptive sensors, c) resourcefulness in its ability to produce a reliable odometry and map estimate by adapting the architecture of its solution, and d) offer a flexible framework for integration of diverse sensor updates.

To deliver on that promise, MIMOSA contributes a new multi-modal fusion architecture combined with carefully selected and modified individual modality-specific SLAM and odometry frameworks. Specifically, MIMOSA

- offers a versatile optimization backend that exploits nonlinear factor graphs that allow to flexibly incorporate odometry and mapping constraints coming from any number of sensors. Currently, MIMOSA allows to fuse estimates from any number of LiDARs, visual and thermal cameras, alongside one IMU. The Georgia Tech Smoothing and Mapping (GTSAM) library [21] offers factor graph optimization, while the LiDAR Odometry And Mapping (LOAM) method [3], the RObust Visual Inertial Odometry (ROVIO) [5] and the RObust Thermal Inertial Odometry (ROTIO) [22] represent the modality-specific "building blocks" of the current implementation.

- exploits extensions in LOAM, ROVIO, and ROTIO to detect situations where LiDAR registration gets degenerate, or when camera-IMU methods present large pose uncertainty growth and accordingly adjusts its solution architecture. When a modality-specific estimate is detected to be unreliable, its updates are temporarily detached from the factor graph, while if the problem persists for long, then, when the degenerate estimation sub-system gets locally healthy again it first gets re-initialized before its constraints are added back to the solution. Exploiting cross modality information, the method also uses LiDAR data to initialize depth for visual and thermal landmarks.
- prevents the emergence of out of order measurement updates and swiftly handles updates for which the varying processing time of the sensor-specific estimation solutions can lead to a situation that the processing for an earlier measurement gets finished after the processing of a subsequent measurement.
- realizes smoothers and allows for the solution to exploit either Incremental Fixed Lag smoothing or incremental Smoothing And Mapping [23].
- provides adaptive update rates exploiting fast inertial measurement readings and thus its estimates can be used directly for fast feedback control loops.
- is extensively evaluated within deployments in underground environments.

Building upon these contributions, MIMOSA, which will be open-sourced, aims to represent a flexible and resilient framework for multi-modal SLAM and associated research.

The remainder of this paper is organized as follows: Section II presents related work, followed by proposed approach in Section III. Evaluation studies are detailed in Section IV, followed by conclusions drawn in Section V.

II. RELATED WORK

Responding to the observation that SLAM methods relying on a single exteroceptive modality are susceptible to the degradation of its sensor data, the research community has investigated a set of schemes exploiting multi-modal fusion. A significant component of this work has related to the fusion of LiDAR, visible-light cameras and IMUs [10–19]. LVI-SAM [11] is a tightly-coupled LiDAR-visual-inertial odometry method that utilizes smoothing and mapping techniques on a factor graph and a visual-inertial and a LiDAR-inertial system. Working towards a tightly-coupled approach, LVI-SAM leverages LiDAR-inertial estimation to facilitate initialization for the visual-inertial system. VILENS [12] also functions in a tightly-coupled fashion combining LiDAR, visual, and inertial sensor data taking advantage of 3D line and planar primitives and a passive synchronization of LiDAR-camera data, while performing backend optimization through a factor graph with fixed lag smoothing. LIC-Fusion2.0 [13] is a LiDAR-visual-inertial framework relying on a sliding-window methodology with online spatio-temporal calibration. The work in [14] proposed R3LIVE which also relies on a LiDAR-inertial and a visual-inertial odometry framework, while delivering a real-time RGB-colored point cloud. An earlier and significantly different development, the work in [15] does not make use of a co-optimizing backend (being filter- or factor graph-based) but employs a sequential, multi-layer pipeline that starts with IMU motion prediction, performs visual-inertial motion estimation and later LiDAR scan matching-based refinement of the motion estimates and registered maps. The Super Odometry [16] method employed an IMU-centric paradigm where visual-inertial and LiDAR-inertial odometry offer priors to constrain the IMU biases and receive motion predictions from the IMU. Focusing on large-scale metro vehicle localization, MetroLoc [17] also considers an IMU-centric scheme where LiDAR-inertial and visual-inertial odometry estimates constrain the IMU biases. Maintaining a loosely-coupled strategy and the coarse-to-fine approach in [15], our previous contribution on CompSLAM [10] exploits both visual-inertial and thermal-inertial odometry estimation scheme, as well as LiDAR Odometry and Mapping steps, while each of the individual estimation subsystems is health-checked with the goal that –as long as sensor degradation does not affect all visual, thermal and LiDAR data simultaneously– the overall method remains reliable. Focusing on fusing ultra-wideband range measurements with LiDAR, visual, and IMU data, the contribution in VIRAL-SLAM [18] offers a tightly-coupled framework with a two-staged optimization approach involving both local sliding window optimization and a global factor graph. Aiming towards a general framework for multi-sensor fusion for tasks such as SLAM, the FUSE software package [24] is extendable to different sensors but building on top of Google Ceres, it lacks essential incremental solving methods such as [23]. In this context, the proposed MIMOSA system is both a flexible and feature-rich framework for multi-modal SLAM and a solution offering an optimized, degeneracy-aware, solution for combined fusion of any number of LiDARs, visual and thermal cameras, as well as an IMU.

III. PROPOSED APPROACH

The architecture, key design choices and implementation details of the proposed multi-modal SLAM framework, named MIMOSA, are presented in this section.

A. Notation

We denote the homogeneous transformation that transforms a point P_A in frame A to a point P_B in frame B as $T_{BA} \in \mathbb{R}^{4\times4}$. We use W to denote the static world frame, and B_i, I_i, L_i, O_i for the dynamic body frame, IMU frame, LiDAR frame and odometry frame respectively at time t_i. All frames are assumed to be in the right-handed coordinate system and the static transformations between B_i, I_i, L_i, O_i are known a-priori through calibration. An input pointcloud or an odometry estimate is termed as a "measurement" and is denoted by M_i. A measurement that specifically represents a pointcloud is denoted by M_i^L and one that specifically represents an odometry estimate is denoted by M_i^O. A state $\mathbf{x}_i$ at timestamp t_i in the factor graph is represented as

$$\mathbf{x}_i = [\mathbf{R}_i, \mathbf{p}_i, \mathbf{v}_i, \mathbf{b}_i] \tag{1}$$

where $\mathbf{R}_i \in SO(3)$ and $\mathbf{p}_i \in \mathbb{R}^3$ are the rotation matrix and the position vector of the body frame in the world frame, $\mathbf{v}_i \in \mathbb{R}^3$ is the velocity of the body frame and $\mathbf{b}_i = [\mathbf{b}_{a,i}, \mathbf{b}_{\omega,i}]$ represents the stacked biases of the accelerometer and gyroscope of the IMU. We use $\mathbf{x}_i^L$ and $\mathbf{x}_i^O$ to represent states that are created at the timestamps of pointclouds and odometry estimates respectively.

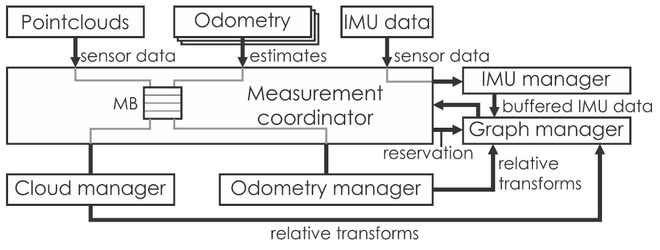

Fig. 2. Block diagram showing the various components of MIMOSA. The directed connections represent the flow of information between various modules that contribute to the handling of measurements from various sensing modalities.

B. Architecture

A block diagram overview of the MIMOSA framework is shown in Figure 2 along with an illustration of the maintained factor graph in Figure 3. The key functional blocks of MIMOSA are explained below:

Measurement Coordinator: The measurement coordinator provides the external interface of MIMOSA and contains the high level logic pertaining to the processing of input data such as pointclouds or odometry estimates, termed as "measurements" $\mathcal{M}_i$. On arrival, measurements are stored in the measurement buffer for the purpose of correcting for delayed measurements as explained in Section III-C. The oldest measurement from the buffer is extracted and a state is reserved for it in the factor graph as detailed in Section III-D after which it is passed to a dedicated thread for processing. It is noted that though IMU samples are obtained by the Measurement Coordinator, they are directly passed to the IMU manager instead of going through the measurement buffer.

Graph Manager: The graph manager is responsible for the storage and interfacing of the underlying nonlinear factor graph and the smoothers that operate on it, both of which are provided by the GTSAM library. Our implementation supports both full and sliding window smoothing through incremental Smoothing And Mapping (iSAM2) [23] and Incremental Fixed Lag smoothing (IFL). The Graph Manager also handles the preintegration of sets of IMU measurements queried from the IMU Manager. The overall objective function optimized by the factor graph is represented by Equation 2 in case of using iSAM2 and Equation 3 in case of using IFL.

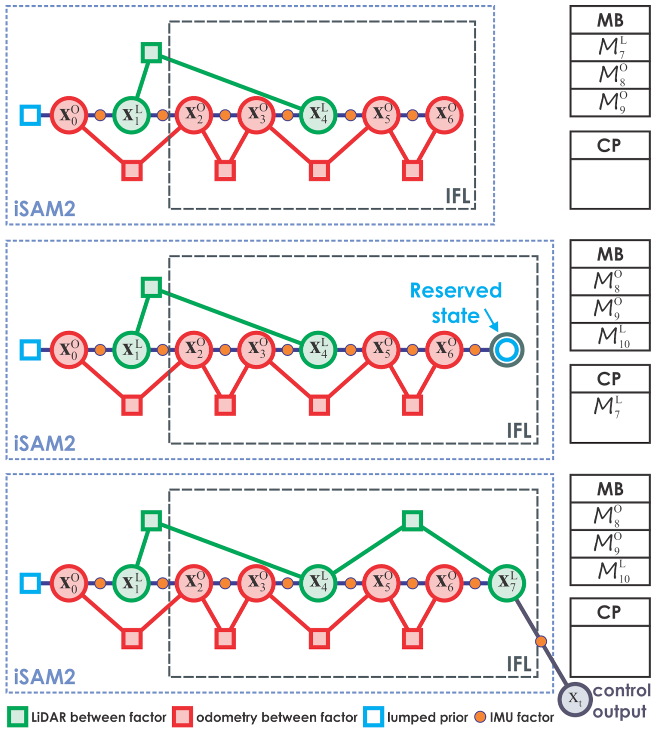

Fig. 3. This figure illustrates the maintained factor graph at different instances in the normal operation of MIMOSA. In each of the sub figures, the contents of the dashed boxes represent the variables and factors that are within the operating horizon of the smoother with the black box representing the Incremental Fixed Lag Smoother (IFL) and the blue box representing incremental Smoothing And Mapping (iSAM2). The boxes of MB and CP represent the contents of the Measurement Buffer and the measurement being processed by the Cloud Manager respectively. Top: The graph when $\mathcal{M}_6^O$ has been fully processed and $\mathcal{M}_7^L$ is about to start processing. Middle: A new state is reserved for the timestamp of $\mathcal{M}_7^L$, while it is processed by the Cloud Manager. Bottom: $\mathcal{M}_7^L$ has finished processing in the Cloud Manager and a relative transformation factor has been added between $(\mathbf{x}_4^L, \mathbf{x}_7^L)$. Also visualized here is $\mathbf{x}_t$, the state generated by propagating the IMU from the latest state in the graph $\mathbf{x}_7^L$ to the timestamp t. This state is never a part of any smoothing and hence exists outside the dashed boxes.

$$X^*_{iSAM2} = \arg\min_X \|\mathbf{r}_0\|^2_{\Sigma_0} + \sum_i \left(\|\mathbf{r}_{\mathcal{I}_{\Delta i}}\|^2_{\Sigma_{\mathcal{I}_{\Delta i}}} + \|\mathbf{r}_{\mathcal{L}_{\Delta i}}\|^2_{\Xi_i} + \sum_n \left(\|\mathbf{r}_{\mathcal{O}_{n,\Delta i}}\|^2_{\Sigma_{\mathcal{O}_{n,\Delta i}}} \right) \right) \tag{2}$$

$$X^*_{IFL} = \arg\min_X E_m + \sum_i \left(\|\mathbf{r}_{\mathcal{I}_{\Delta i}}\|^2_{\Sigma_{\mathcal{I}_{\Delta i}}} + \|\mathbf{r}_{\mathcal{L}_{\Delta i}}\|^2_{\Xi_i} + \sum_n \left(\|\mathbf{r}_{\mathcal{O}_{n,\Delta i}}\|^2_{\Sigma_{\mathcal{O}_{n,\Delta i}}} \right) \right) \tag{3}$$

where the terms $\mathbf{r}_0$, $\mathbf{r}_{\mathcal{I}_{\Delta i}}$, $\mathbf{r}_{\mathcal{L}_{\Delta i}}$ and $\mathbf{r}_{\mathcal{O}_{n,\Delta i}}$ represent the residuals of the prior factor, the preintegrated IMU factors and the relative transformation factors introduced by the Cloud Manager and the n^{th} Odometry source and the terms Σ_0, $\Sigma_{\mathcal{I}_{\Delta i}}$, Ξ_i and $\Sigma_{\mathcal{O}_{n,\Delta i}}$ represent their associated covariances respectively. i in both the equations provides indexing throughout the smoothing window of the respective smoother. Additionally in case Equation 3, E_m represents the marginalization prior.

IMU Manager: The IMU manager maintains a sliding window buffer of the IMU measurements and provides utility functions for accessing IMU data within queried timestamps including linear interpolation and extrapolation to the Graph and measurement managers.

Odometry Manager: The Odometry Manager processes a pair of sequential odometry measurements $\mathcal{M}_1^o, \mathcal{M}_2^o$ with timestamps t_1 and t_2 to generate and add a relative transformation factor between the states $\mathbf{x}_1^O$ and $\mathbf{x}_2^O$. This factor is generated as the transformation between the body frames at the respective timestamps as

$$T_{B_1B_2} = T_{B_1O_1} T_{WO_1}^{-1} T_{WO_2} T_{B_2O_2}^{-1} \tag{4}$$

The Odometry Manager also performs its own bookkeeping for the states it adds to the graph. To support multiple odometry sources, the variables required for the creation of the factor and the bookkeeping are instantiated as vectors with the number of sources being a parameter set at initialization time. Our experiments involve the usage of ROVIO and ROTIO as the odometry sources as the utilized platforms are equipped with visual and thermal cameras respectively. However, the Odometry Manager only requires an odometry update with an associated covariance and $T_{B_iO_i}$ transform thereby allowing the framework to integrate odometry from other sensors (e.g. RADAR) also. Note that though our current implementation assumes $T_{B_iO_i}$ to be a static transform, future work could include this as a variable in the factor graph to be (re-)calibrated online.

Cloud Manager: The Cloud Manager implements the feature extraction, scan-to-scan matching and scan-to-submap matching modules of LOAM with additions for bookkeeping and dealing with degeneracy and sensor failure as detailed in Sections III-E and III-F respectively. The feature extraction module selects points from the pointcloud using a curvature metric and classifies them as surface and corner features. The scan-to-scan and scan-to-submap matching modules both solve a pointcloud registration problem to register the extracted features from the current pointcloud to the features from the previous pointcloud and the accumulated map respectively. The result of the scan-to-submap matching module is a low rate (here 5Hz) high fidelity odometry estimate that is used to generate the relative transform that is added as a factor

$$T_{B_1B_2} = T_{B_1L_1} T_{WL_1}^{-1} T_{WL_2} T_{B_2L_2}^{-1} \tag{5}$$

The noise parameters for the created factor are heuristically tuned scaled versions of the covariance of the matching given by Ξ, which is defined as

$$\Xi = \sigma^2 (\mathbf{A}^T \mathbf{A})^{-1} \tag{6}$$

where σ^2 is the sum of squared residuals from the calculated correspondences and $\mathbf{A}$ represents the Jacobian for the nonlinear optimization.

Considering this overall architecture, the following subsections describe some of the salient points of our approach.

C. Delayed Measurements

Ideally, a measurement $\mathcal{M}_1$, created at timestamp t_1 and received by the measurement coordinator at timestamp t_1^c and a measurement $\mathcal{M}_2$, created at timestamp t_2 and received by the measurement coordinator at timestamp t_2^c, such that $t_1 < t_2$, would respect $t_1^c < t_2^c$. This, however, may not occur in practice due to different kinds of delays (processing, propagation, random etc.). The measurement buffer serves to rectify this by buffering measurements $\mathcal{M}_i$ sorted by their creation timestamps t_i at insertion, and allowing only the oldest measurement to be extracted for processing, thereby ensuring that the measurements are always processed according to their creation timestamps t_i.

D. Reservation

The time required for processing a measurement to create a factor may be different for the different modalities in use. Considering MIMOSA's two measurement managers - the Odometry Manager and the Cloud Manager - the creation of the relative constraint by the Odometry Manager is extremely fast in comparison with the Cloud Manager. Assuming the latest state in the graph has a timestamp of t_0, if a pointcloud with the timestamp t_1 were to arrive at the timestamp t_1^c and finish processing at time t_1^p, there could be an odometry measurement generated at timestamp t_2 arriving at timestamp t_2^c which finishes processing at time t_2^p such that $t_1 < t_2$ and $t_1^c < t_2^c < t_2^p < t_1^p$. If the update to the graph were to happen after a measurement has been processed, the state $\mathbf{x}_2$ would be created and added to the graph before the state $\mathbf{x}_1$. The addition of $\mathbf{x}_1$ would then require the preintegrated IMU factor between the states $(\mathbf{x}_0, \mathbf{x}_2)$ to be replaced with a new state $\mathbf{x}_1$ and two new preintegrated factors between $(\mathbf{x}_0, \mathbf{x}_1)$ and $(\mathbf{x}_1, \mathbf{x}_2)$. Instead, we avoid this issue altogether by creating a new state $\mathbf{x}_{i+1}$ for the measurement $\mathcal{M}_{i+1}$ when it is extracted from the measurement buffer and add it along with a preintegrated IMU factor between the latest state in the graph, $\mathbf{x}_i$, and itself. Once the corresponding measurement manager processes this new measurement, it adds the generated relative factor to the states $\mathbf{x}_{i+1}$ and $\mathbf{x}_{i-k}$, where $\mathbf{x}_{i-k}$ is the state created for the last measurement of the same type as $\mathbf{x}_{i+1}$.

E. Degeneracy and Method Failure

We use the method initially presented in [9] for the detection of degeneracy in the underlying optimization problem in the Cloud Manager. Degeneracy often appears in cases of geometric self-similarity. When degeneracy is detected, the estimated relative transformation is not used to add a factor since this is likely erroneous. Additionally, the cloud manager also maintains a history of degeneracy detections for the estimation of an additional health metric defined as the fraction of degeneracy detections over the length of the window. This metric is then thresholded to gauge a boolean health status. In case of being detected as unhealthy, the reservation of states is disabled for pointcloud measurements and no more factors derived from them are added to the graph. Though not being added in the graph, the pointclouds are still processed by the Cloud Manager so as to be able to detect when the status changes back to healthy, at which point the erroneous map in the scan-to-submap module is removed and reset. After processing some pointclouds allowing for the degeneracy detection history to be filled and ensuring the health metric to be zero, the reservation of states is re-enabled for pointcloud measurements and relative transform factors once again start to be added by the Cloud Manager. A similar method is implemented for the Odometry Manager using the D-Optimality (D_{opt}) metric [25] for degeneracy detection which is given as

$$D_{\text{opt}} = \exp(\log(\det(\Sigma)^{1/l})) \tag{7}$$

where Σ is the covariance matrix with dimensions $l \times l$ of the pose in the odometry estimate.

F. Sensor Failure and Recovery

Apart from degeneracy or failure at the method-level, sensors may experience failures leading to temporary data loss as well. Furthermore, as we incorporate odometry estimates, it is possible for the odometry method to crash and restart after some delay. As the formulation of our factor graph inherently relies only on the available modalities at every time instant, assuming that not all modalities experience this kind of failure simultaneously, our method is robust to failures of sensors and odometry methods by design.

G. Adaptive output rates for fast feedback control loops

The minimum output rate of MIMOSA is the maximum of individual measurement inputs in steady-state. In case of LOAM and ROVIO/ROTIO, this could be (in case of one of our experimental data) $max(5, 20)$ Hz assuming the updates to be perfectly synchronized. However, a robot may require state estimates at a higher rate for the purpose of its onboard feedback control loop. Commonly, this higher rate is obtained by leveraging an Extended Kalman Filter [26, 27] to fuse the low frequency-high fidelity updates from the extrinsic sensor-based solution with the onboard IMU for a smooth high frequency update. The requirement of this additional method is removed altogether by the addition of a simple timer thread in our framework. The timer thread wakes up at a predefined rate, propagates the IMU measurements from the latest state in the factor graph to the required timestamp and publishes it. This has been illustrated in the bottom subfigure of Figure 3.

H. Extendability to New Modalities

Our proposed solution is capable of supporting the addition of multiple different kinds of sensing modalities simultaneously. Additional sensing modalities can be added seamlessly to the existing framework by creating a dedicated measurement manager per sensing modality. Asynchronous measurements from each sensor are indexed based on their timestamp and are independently processed. A new factor is created on processing each measurement, which is then added to the factor graph. The framework is implemented in C++ and uses the Robot Operating System (ROS) [28] for the underlying message passing.

I. Implementation and performance enhancements

In order to support a variety of sensors, the framework is designed to be modular with the Measurement Coordinator running on a central processing thread. On receiving sensor data, this thread reserves a state within the factor graph and invokes the measurement manager corresponding to the sensor measurement. A separate thread is spawned for this purpose, enabling the central processing thread to remain free in order to receive asynchronous measurements from other sensors. Each measurement manager is assigned a thread so that separate measurements can be processed in parallel. This allows for increased scope for processing a high rate of measurements that arrive before the previous ones are fully processed.

IV. EVALUATION STUDIES

To demonstrate the resiliency of MIMOSA, we conduct evaluations based on data collected through autonomous exploration missions in underground mines using legged and aerial robots. To facilitate comparison, we present our results against those of CompSLAM [10], our previous contribution successfully fielded in the DARPA Subterranean Challenge. The first scenario presents regions of geometric self similarity while the second simulates sensor failure.

A. Scenario 1: Geometrically self similar environment

This evaluation uses a dataset collected using the ANYmal C robot from ANYbotics [29] in an abandoned mine in Switzerland as part of the preparations of Team CERBERUS for the DARPA Subterranean Challenge. The robot, as pictured in the inset image in Figure 4, carries a Velodyne VLP-16 LiDAR with a horizontal and vertical Field Of View (FOV) of $[360, 30]^\circ$ and 16 channels along with an Alphasense development kit from Sevensense Robotics [30] consisting of seven monochrome 0.4 MP cameras and an IMU. The robot traverses a long, narrow tunnel and returns to the beginning as shown in Figure 4. The feature-poor walls and the self-similar geometry of the tunnel cause the LiDAR pose estimates to be degenerate, at least in certain directions. For CompSLAM this is detected through the associated ill-conditioning check and the method uses priors from ROVIO to set the estimated transform in the degenerate directions. However both because ROVIO drifts and because the directions that are not detected to be degenerate on LiDAR still experience error, the method presents significant errors in the map and drift in the localization estimate. This becomes evident as the robot returns to the starting point. This problematic behavior is not observed on MIMOSA. Its LiDAR Odometry And Mapping component, based on LOAM, also detects degeneracy but at this point MIMOSA drops the LiDAR-based measurements altogether for the entirety of the duration for which the buffer is unhealthy, and uses measurements from the remaining modalities, namely IMU and LiDAR range-assisted ROVIO. ROVIO here performs

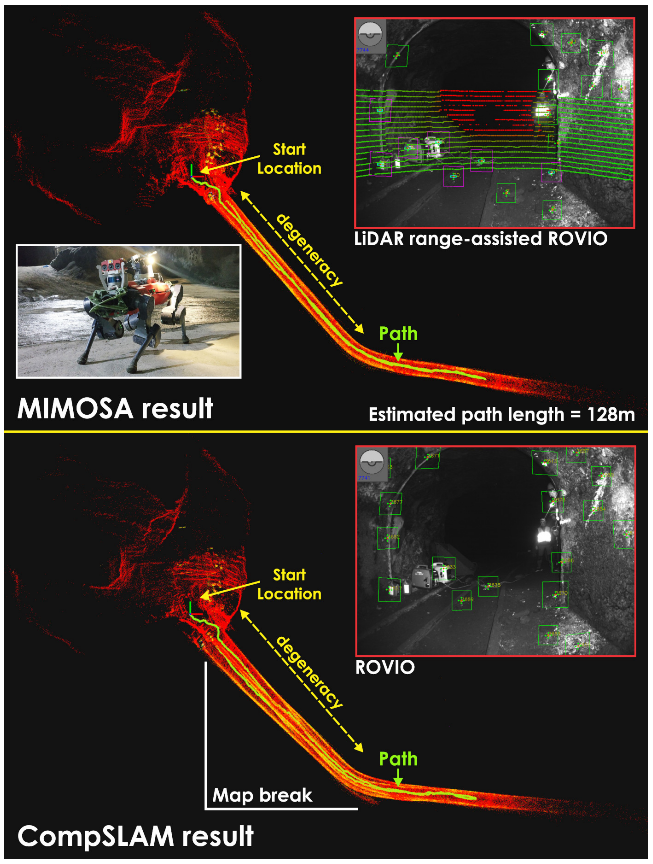

Fig. 4. This evaluation study shows the comparison between CompSLAM and MIMOSA in a geometrically self-similar, narrow tunnel. CompSLAM detects degeneracy and uses ROVIO to set the transforms. The method drifts as the robot returns to the starting point. MIMOSA also detects degeneracy but drops the LiDAR-based measurements when the buffer is unhealthy, and uses measurements from the IMU and LiDAR range-assisted ROVIO to provide a better estimate.

significantly better as it is assisted by LiDAR for landmark depth initialization. Once the LiDAR measurements become healthy again, the Cloud Manager is re-initialized, and the measurements from LiDAR are added between the states. This experiment demonstrates the capability of MIMOSA to selectively drop estimates from unhealthy modalities, and exploit cross-modality information (initializing visual feature depth from LiDAR) towards improved estimates.

B. Scenario 2: Simulation of sensor failure

To demonstrate the robustness of MIMOSA against sensor failure, we present results against CompSLAM on a dataset collected using an aerial robot as it is deployed in an underground mine in northern Nevada, USA during DARPA Subterranean Challenge preparations. The dataset comprises of thermal vision, LiDAR and IMU data. Specifically, the sensor-suite onboard the robot consists of a FLIR Tau2 LWIR camera running at 30 Hz, an Ouster OS-1 LiDAR with 64 channels and a $[360, 33.2]^{\circ}$ FOV along with a temperature-compensated VectorNav VN-100T IMU. The data is collected in flight, with the robot covering a path length of 268.3 m over a period of 464 s. To simulate sensor failure, the LiDAR data is removed for two intervals, and the performance of our method is compared against CompSLAM. Figure 5 shows the estimated trajectory of the robot using both methods, with the simulated sensor failure at times $t_1 = 59.5$ s and $t_2 = 78.5$ s for 5 s each. The estimated trajectory of the robot is shown in yellow, with blue highlights over the segments representing the loss of LiDAR data. As CompSLAM relies on LiDAR measurements to update the estimate, the loss of data causes it to suspend the update of the estimate. Once the sensor measurements are resumed, it is not able to fit the pointcloud accurately over the map, which causes an underestimation of the distance travelled by the robot. At the same time, MIMOSA uses the updates available from the IMU and ROTIO to provide a better estimate of the position of the robot.

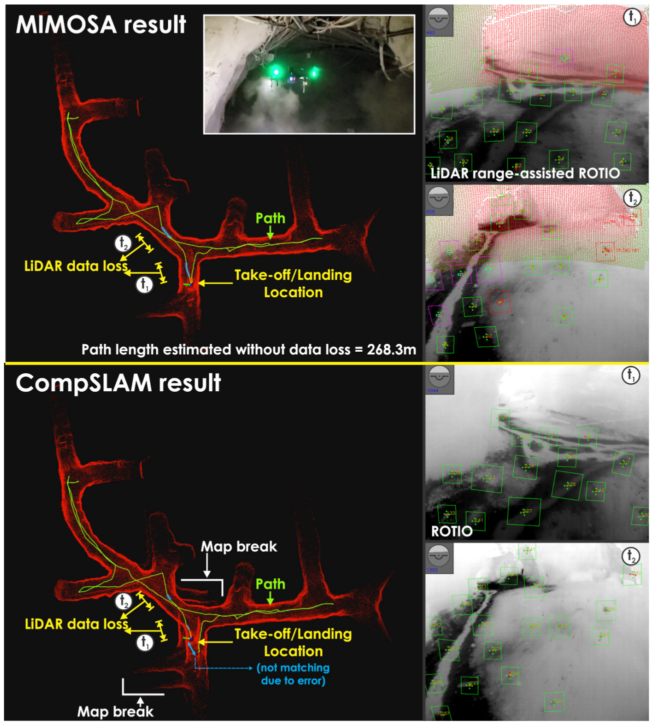
MIMOSA result
LiDAR range-assisted ROTIO
Path
LiDAR data loss
Take-off/Landing Location
Path length estimated without data loss = 268.3m
CompSLAM result
ROTIO
Map break
Path
LiDAR data loss
Take-off/Landing Location
(not matching due to error)
Map break

Fig. 5. The performance of CompSLAM and MIMOSA is compared to determine the effect of sensor failure. The LiDAR data is removed for two intervals of 5 s each at times t_1, t_2. The resulting trajectory of the robot is plotted alongside the registered pointcloud. It is noted that the presented path length is without any data loss although the odometry and mapping results shown include phases where LiDAR data is missing.

V. CONCLUSIONS

This paper presented MIMOSA, a new framework for multi-modal SLAM with a current focus on the complementary fusion of LiDAR range data, visible-light and thermal camera frames, as well as inertial measurement cues. The method presents a loosely-coupled architecture through a versatile optimization backend that allows to incorporate odometry and mapping constraints from any number of sensors, while certain cross-modality information is also exploited such as acquiring depth to visual or thermal features from LiDAR. MIMOSA prevents the emergence of out of order updates, realizes both incremental fixed lag smoothing and incremental smoothing and mapping, while also offering high-update output for feedback control purposes. Extensive evaluations in challenging data from subterranean robotic exploration missions using legged and aerial robots allow to demonstrate the resilience of the approach, its robust performance and key features such as the ability to handle geometric degeneracy and the loss of data from certain sensors.

ACKNOWLEDGMENT

We would like to thank our CERBERUS partners from the Robotic Systems Lab of ETH Zurich for sharing the data from the ANYmal C robot at the Seemühle mine.

REFERENCES

[1] M. Tranzatto, F. Mascarich, L. Bernreiter, C. Godinho, M. Camurri, S. M. K. Khattak, T. Dang, V. Reijgwart, J. Loeje, D. Wisth *et al.*, "Cerberus: Autonomous legged and aerial robotic exploration in the tunnel and urban circuits of the darpa subterranean challenge," *Field Robotics*, 2021.

[2] J. Levinson, J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J. Z. Kolter, D. Langer, O. Pink, V. Pratt *et al.*, "Towards fully autonomous driving: Systems and algorithms," in *2011 IEEE intelligent vehicles symposium (IV)*. IEEE, 2011, pp. 163–168.

[3] 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.

[4] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, "Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 5135–5142.

[5] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart, "Robust visual inertial odometry using a direct ekf-based approach," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*. IEEE, 2015, pp. 298–304.

[6] C. Forster, M. Pizzoli, and D. Scaramuzza, "Svo: Fast semi-direct monocular visual odometry," in *2014 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2014, pp. 15–22.

[7] S. Khattak, C. Papachristos, and K. Alexis, "Keyframe-based direct thermal–inertial odometry," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2019.

[8] S. Zhao, P. Wang, H. Zhang, Z. Fang, and S. Scherer, "Tp-tio: A robust thermal-inertial odometry with deep thermalpoint," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 4505–4512.

[9] J. Zhang, M. Kaess, and S. Singh, "On degeneracy of optimization-based state estimation problems," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 809–816.

[10] S. Khattak, H. Nguyen, F. Mascarich, T. Dang, and K. Alexis, "Complementary multi–modal sensor fusion for resilient robot pose estimation in subterranean environments," in *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2020, pp. 1024–1029.

[11] T. Shan, B. Englot, C. Ratti, and D. Rus, "LVI-SAM: Tightly-coupled Lidar-Visual-Inertial Odometry via Smoothing and Mapping," *2021 IEEE International Conference on Robotics and Automation (ICRA)*, vol. 00, pp. 5692–5698, 2021.

[12] D. Wisth, M. Camurri, S. Das, and M. Fallon, "Unified multi-modal landmark tracking for tightly coupled lidar-visual-inertial odometry," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1004–1011, 2021.

[13] X. Zuo, Y. Yang, P. Geneva, J. Lv, Y. Liu, G. Huang, and M. Pollefeys, "LIC-Fusion 2.0: LiDAR-Inertial-Camera Odometry with Sliding-Window Plane-Feature Tracking," *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 00, pp. 5112–5119, 2021.

[14] J. Lin and F. Zhang, "R3LIVE: A Robust, Real-time, RGB-colored, LiDAR-Inertial-Visual tightly-coupled state Estimation and mapping package," *arXiv*, 2021.

[15] J. Zhang and S. Singh, "Laser–visual–inertial odometry and mapping with high robustness and low drift," *Journal of Field Robotics*, vol. 35, no. 8, pp. 1242–1264, 2018. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21809

[16] S. Zhao, H. Zhang, P. Wang, L. Nogueira, and S. Scherer, "Super Odometry: IMU-centric LiDAR-Visual-Inertial Estimator for Challenging Environments," *arXiv*, 2021.

[17] Y. Wang, W. Song, Y. Zhang, F. Huang, Z. Tu, and Y. Lou, "MetroLoc: Metro Vehicle Mapping and Localization with LiDAR-Camera-Inertial Integration," *arXiv*, 2021.

[18] T.-M. Nguyen, S. Yuan, M. Cao, T. H. Nguyen, and L. Xie, "VIRAL SLAM: Tightly Coupled Camera-IMU-UWB-Lidar SLAM," *arXiv*, 2021.

[19] Z. Wang, J. Zhang, S. Chen, C. Yuan, J. Zhang, and J. Zhang, "Robust high accuracy visual-inertial-laser slam system," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 6636–6641.

[20] Defense Advanced Research Projects Agency (DARPA), "DARPA Subterranean Challenge." [Online]. Available: https://subtchallenge.com/

[21] F. Dellaert, "Factor graphs and gtsam: A hands-on introduction," Georgia Institute of Technology, Tech. Rep., 2012.

[22] S. Khattak, F. Mascarich, T. Dang, C. Papachristos, and K. Alexis, "Robust thermal-inertial localization for aerial robots: A case for direct methods," in *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2019, pp. 1061–1068.

[23] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, "isam2: Incremental smoothing and mapping using the bayes tree," *The International Journal of Robotics Research*, vol. 31, no. 2, pp. 216–235, 2012. [Online]. Available: https://doi.org/10.1177/0278364911430419

[24] Locus Robotics, "fuse: A graph-based sensor fusion framework." [Online]. Available: https://github.com/locusrobotics/fuse

[25] H. Carrillo, I. Reid, and J. A. Castellanos, "On the Comparison of Uncertainty Criteria for Active SLAM," *2012 IEEE International Conference on Robotics and Automation*, pp. 2080–2087, 2012.

[26] S. Lynen, M. Achtelik, S. Weiss, M. Chli, and R. Siegwart, "A robust and modular multi-sensor fusion approach applied to mav navigation," in *Proc. of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, 2013.

[27] P. D. 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," 2022.

[28] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng, "Ros: an open-source robot operating system," in *Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA) Workshop on Open Source Robotics*, Kobe, Japan, may 2009.

[29] ANYbotics, "ANYmal Legged Robots," available at https://www.anybotics.com/.

[30] Sevensense Robotics, "Alphasense Position," available at https://www.sevensense.ai/product/alphasense-position.