LoLa-SLAM: Low-Latency LiDAR SLAM Using Continuous Scan Slicing

Mojtaba Karimi, *Member, IEEE*, Martin Oelsch, *Member, IEEE*, Oliver Stengel, Edwin Babaians, and Eckehard Steinbach, *Fellow, IEEE*

***Abstract*—Real-time 6D pose estimation is a key component for autonomous indoor navigation of Unmanned Aerial Vehicles (UAVs). This letter presents a low-latency LiDAR SLAM framework based on LiDAR scan slicing and concurrent matching, called LoLa-SLAM. Our framework uses sliced point cloud data from a rotating LiDAR in a concurrent multi-threaded matching pipeline for 6D pose estimation with high update rate and low latency. The LiDAR is actuated using a 2D Lissajous spinning pattern to overcome the sensor's limited FoV. We propose a two-dimensional roughness model to extract the feature points for fine matching and registration of the point cloud. In addition, the pose estimator engages a temporal motion predictor that assists in finding the feature correspondences in the map for the fast convergence of the non-linear optimizer. Subsequently, an Extended Kalman Filter (EKF) is adopted for final pose fusion. The framework is evaluated in multiple experiments by comparing the accuracy, latency, and the update rate of the pose estimation for the trajectories flown in an indoor environment. We quantify the superior quality of the generated volumetric map in comparison to the state-of-the-art frameworks. We further examine the localization precision using ground truth pose information recorded by a total station unit.**

***Index Terms*—SLAM, aerial systems, perception and autonomy, low-latency localization.**

I. Introduction

THE demand for unmanned aerial vehicles (UAVs) for autonomous exploration and inspection is growing. The compact design-factor, relatively low cost, and maneuverability of UAVs make them well suited for various tasks [1]. Autonomous UAVs require a reliable navigation system to operate in challenging environments, such as GPS-denied or cluttered indoor areas [2], [3]. Tackling the general navigation mission, one needs to address a set of problems ranging from 6D pose estimation to trajectory planning [4], [5]. In practice, the navigation task's performance depends mainly on the accuracy, update rate, and latency of the perception unit (see Fig. 1). For instance, the trajectory tracking controller of the UAV requires a high-rate pose estimation (in the range of 0.1 to 1.0 kHz) with comparatively low latency (maximum 50 ms) to achieve a fully autonomous flight [6]–[8]. In this context, latency is the time difference from the sensor observation until the localization algorithm provides the estimated pose.

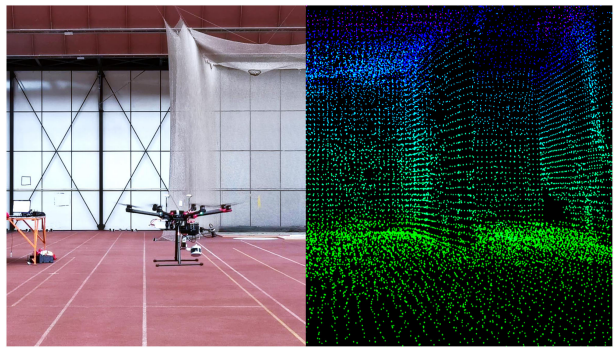


Fig. 1. Real-time localization of a UAV in an indoor environment using the proposed SLAM framework. It provides low-latency 6D pose estimation with a high temporal update rate for autonomous navigation while generating a full-scale 3D map of the environment.

Visual-inertial localization and mapping is well studied for drone localization as it can meet the aforementioned requirements [9], [10]. However, in large indoor areas, the accuracy of visual-inertial localization approaches tends to degrade dramatically [11], [12]. This is mainly due to the considerable distance between the camera and the scene, which causes feature tracking failures in the camera frames. As an alternative, laser scanning technology is employed to capture precise range measurements enabling LiDAR-based localization and mapping systems. To address the 6D pose estimation task, three-dimensional simultaneous localization and mapping (3D SLAM) is the preferred way. 3D SLAM is mainly addressed in the literature by using multiple laser scanners, which are installed in both horizontal and vertical frames [13], [14]. However, due to the weight and processing constraints in UAVs, in practice, the use of an actuated LiDAR is preferred rather than installing multiple or heavier sensors with an inherently larger FoV [15], [16]. In this regard, different spinning mechanisms have been developed to be mounted on drones [17], [18]. Although generally successful, existing solutions come with some shortcomings, such as long revisit time, blind spots, inconsistencies in scanning the environment, and skewing issues. These shortcomings cause problems

Manuscript received October 13, 2020; accepted February 8, 2021. Date of publication February 19, 2021; date of current version March 12, 2021. This letter was recommended for publication by Associate Editor N. Kottege and Editor P. Pounds upon evaluation of the reviewers' comments. This work was supported by the German Aerospace Center (DLR) with funds from the Federal Ministry of Economic Affairs and Energy (BMWi) on the basis of a resolution of the German Bundestag under the Reference '20X1707 C'. *(Corresponding author: Mojtaba Karimi.)*

The authors are with the Department of Electrical and Computer Engineering, Chair of Media Technology, Technical University of Munich (TUM), Munich 80333, Bavaria, Germany (e-mail: mojtaba.karimi@tum.de; martin.oelsch@tum.de; oliver.stengel@tum.de; edwin.babaians@tum.de; Eckehard.Steinbach@tum.de).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2021.3060721, provided by the authors.

Digital Object Identifier 10.1109/LRA.2021.3060721

in determining the correspondences within the point cloud data for 6D positioning [19], [20].

While LiDAR-based SLAM has been studied widely in the past decade for mobile robots [19], [21]–[24], there is still a major gap in utilizing these systems for real-time navigation of UAVs. This is mainly because it is not possible to rely on an aerial vehicle remaining sufficiently motionless. The drone position must be represented at a high temporal update rate with low-latency to achieve stable closed-loop control. Addressing the challenge mentioned above, continuous-time trajectory estimators were developed in the literature [25], [26]. However, due to the limited FoV and the substantial time needed for a complete scan period of the LiDAR, estimating continuous odometry and global registration introduces a significant latency and low-fidelity 6D pose estimation in such a system. To address this issue, LiDAR-inertial SLAM systems use an additional Inertial Measurement Unit (IMU) to produce high-rate pose updates [24]. They provide acceptable results for ground-based mobile robots. However, as UAVs experience high dynamic motions and inevitable high-frequency vibrations due to the propellers' rotation, IMU-based approaches leads to having a fluctuating pose estimation in such applications [27].

To meet these challenges, we propose a low-latency localization and mapping framework using scan slicing (LoLa-SLAM). In our approach, the point cloud data from a rotating laser scanner with 360° horizontal FoV is continuously sliced and used for estimating the real-time 6D pose. Besides, we introduce a novel sensor payload design in which a small, lightweight multi-line 360° laser scanner is actuated with a Lissajous pattern. This structure is employed to reduce the revisit time and provides a dense point cloud with consistency in scanning the environment [28], [29]. To this end, while UAV localization using actuated LiDARs has been presented previously [25], [30]–[32], to the best of our knowledge, this is the first work on scan slicing of an actuated rotating LiDAR, which is utilized for real-time low-latency indoor localization. This work has three main contributions. First, it presents an actuated LiDAR sensor platform's design and implementation based on a centralized servo mechanism and a 2D Lissajous pattern to improve the FoV and reduce the revisit time. Second, this letter proposes a novel real-time LiDAR-based low-latency SLAM framework based on scan slicing and a concurrent multi-threaded matching pipeline. Third, multiple experiments were designed and conducted to evaluate the proposed framework's performance toward the reliable localization of a UAV in an indoor environment.

The rest of this letter is organized as follows: Section II describes the problem and notation used in this work. Section III introduces the sensor platform and the Lissajous LiDAR actuation pattern, and Section IV details the continuous sliced scanning model. The low-latency localization framework is introduced in Section V. Experimental evaluations are presented in Section VI. Finally, Section VII concludes this letter and examines potential future works.

II. Problem Statement and Sensor Platform

Rotating laser scanners use the time-of-flight (ToF) concept that operates by emitting laser light and capturing its reflection

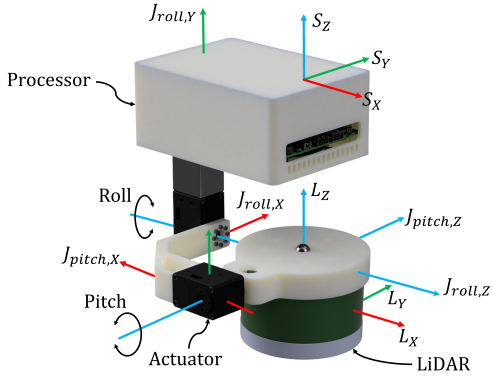


Fig. 2. CAD model of the sensor payload design for UAVs represented with the coordinate frames. A 2-DoF centralized servo mechanism is utilized for LiDAR actuation. The system includes a rotating 16 beam-line LiDAR, 2-DoF centralized servo mechanism for LiDAR actuation, motor drivers, and a processing unit.

to measure distances to nearby objects. In the case of multi-line scanning, multiple laser beams are used with a rotating structure at a fixed velocity to provide a more extended FoV compared to single-line planar LiDARs. Each of the ToF measurements is associated with the horizontal and vertical angle of the corresponding beam. This information is combined to compute the 3D coordinates of the sensed point, and accordingly, a LiDAR generates a point cloud corresponding to the set of observed points. The LoLa-SLAM system studied in this work is validated on the actively actuated LiDAR system shown in Fig. 2. The Velodyne VLP-16 laser scanner has a vertical FoV of 30° and a horizontal FoV of 360° with approximately 0.4° angular resolution and 16 laser beams. The laser scanner is connected to a centralized hinge joint with 2-DoF. Each of the joints can be controlled individually with a maximum velocity of 360°/s in a range of ±45° with 0.087° resolution.

We use right-handed coordinate systems with an uppercase subscript. All laser points are measured in the LiDAR coordinate frame $\{L\}$. As depicted in Fig. 2, the origin of this frame is in the geometric center of the laser scanner. The collected points that belong to one complete rotation of a multi-beam laser in 360° are represented as a single full scan. Classically, each full scan is used in the SLAM algorithms to determine the ego-motion of the platform. In this work, instead of using a full scan of a rotating LiDAR, we use continuously sliced scans as a set of points $\mathcal{P}$ with right subscript $q, q \in Z^+$ to indicate the slice number. Each slice is a subset of an upcoming full scan. Each point in the sliced point cloud $\mathcal{P}$ is defined as $i, i \in \mathcal{P}_q$ in the LiDAR frame $\{L\}$ and is marked by $\mathcal{P}^L_{(q,i)}$. Similarly, the sensor coordinate frame $\{S\}$ is a 3D coordinate system coinciding with $\{L\}$ and the joint frame $\{J_{pitch}\}$ and $\{J_{roll}\}$. Map coordinate frame $\{M\}$ is a coordinate system coinciding $\{S\}$ with the initial position $\{0\}$. A point in the sliced point cloud $\mathcal{P}_q$, in the map coordinate is denoted as $\mathcal{P}^M_{(q,i)}$. With the specified notation defined above, the low-latency localization and mapping is defined as: given a sequence of continuous sliced scans in the sensor frame $\mathcal{P}^M_q$, estimate the 6D pose of the platform in the map coordinate frame $\{M\}$ as ${}^M\boldsymbol{T}_{S,q}(t)$, and create a representative map of the environment.

III. LiDAR Actuation Using Lissajous Pattern

The limited vertical FoV of state-of-the-art laser scanners is one of the main challenges for 6D positioning. In robotic applications, this issue is normally addressed by an articulated sensor, where it is driven by one or more dedicated actuators in order to increase the FoV. However, such observation requires a non-negligible amount of time to capture. This becomes problematic for drones while they are floating in the air. Periodic actuation of the LiDAR sensor in roll and/or pitch direction repeats the observation of the surface patches and allows for a larger FoV. To enhance this procedure, researchers have developed non-raster scan actuation motions, called Lissajous pattern [28], [29], [33]. Considering the fact that a state-of-the-art rotating LiDAR provides a constant sampling rate, the main benefit of using a Lissajous scan pattern-based actuation of the LiDAR is the reduced revisit time (by 54%) within the defined FoV [29]. In addition, compared to the other actuation models, avoiding the orderly nature of the raster scan by using a simultaneously periodic sinusoidal trajectory can be defined as another advantage of the Lissajous pattern.

Motivated by these works, we propose a 2D Lissajous pattern for a rotating 360° LiDAR. This pattern allows us to rotate the LiDAR to observe the vertical and horizontal surfaces equitably. We designed and developed an actuated LiDAR payload with 2-DoF, as shown in Fig. 2, to be able to precisely execute this pattern. Based on the systematic study of the scan skewing problem [34], we developed a mechanism that actuates the laser scanner in a combined roll and pitch motion around a centralized axes to minimize the measurement distortion. The platform is manufactured using printed ABS material. The proposed 2D Lissajous scan pattern at time t for the roll (α) and pitch (β) axes are defined by

$$\begin{aligned} \alpha(t) &= A_{roll}(t)\sin(2\pi f_c t), \\ \beta(t) &= A_{pitch}(t)\cos(2\pi f_c t), \end{aligned} \tag{1}$$

where the frequency f_c is the period of one full sweep, and the amplitudes A_{roll} and A_{pitch} are defined as the maximum angles of the motion. We set the period of one full sweep to 8.0 seconds and the maximum amplitude to 45°. These values are obtained heuristically based on the dynamics of the drone and the FoV of the sensor. We use a slow start mechanism, which increases the amplitude of the Lissajous actuation pattern from zero to its maximum value within the first three sweeps. We use the smooth start mainly to prevent damage to the rotating laser scanner as the sudden and fast motion will cause hardware damage in long term use. Although it is not necessary, this smooth increase of the amplitude at the start, while the robot is stationary (e.g., before takeoff), helps in the initialization of the SLAM algorithm. This is because, at the start, there is a large number of the newly observed points, which must be added to the map. The generated Lissajous pattern in comparison with a single axis raster scan is illustrated in Fig. 3.

IV. Continuous Scan Slicing

The laser scanner used in this work rotates with a maximum of 1200 rounds per minute (rpm) and provides 300 000 points per second. A 360° scan is available every $\omega_{scan} = 50$ milliseconds and contains 15 thousand measurements. The laser scanner is continuously measuring and creates a constant output stream associated with time and angle. These measurements are accumulated over time at the driver level. When the laser scanner completes a full 360° scan, it releases all the recorded points as a single scan. State-of-the-art SLAM frameworks are developed based on these 360° scans for ego-motion estimation [22], [23], [27]. In practice, the point cloud arrival rate defines the frequency of the pose estimation. However, the ego-motion estimation can be performed after any other measurement, which means 360° is an arbitrary margin, and individual measurements can be bundled freely.

Although it is reasonable to compare complete 360° scans in the odometry unit; we show later in the proposed localization framework that our SLAM model does not depend on the odometry unit. The proposed SLAM framework uses continuous slices of a full 360° scan for low-latency localization using direct scan matching on the global map. The main intention of the proposed continuous slicing point cloud (CSPC) is to gather smaller collections of measurements that can be used in a successive manner instead of waiting for a full 360° scan. In this context, describing the term continuous in other words, we collect the measurements from the upcoming scan into a small point cloud slice for the last θ angle (i.e., only the data of the measurements within the last $\theta = 45°$ of the rotation), and publish them one after another. As shown in Fig. 4, slicing of the points is developed in the driver level of the laser scanner, and the CSPC is published at a rate of $\frac{1}{\Delta q}$ Hz. Considering each slice

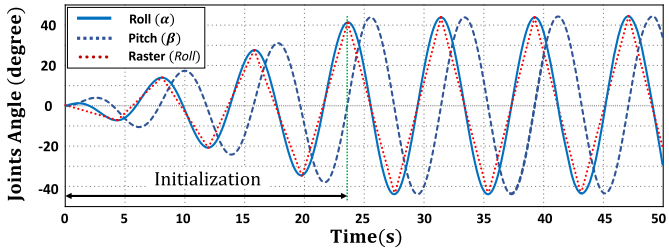


Fig. 3. The proposed 2D Lissajous pattern for the roll and pitch joints of the sensor platform. The frequency of the pattern is defined to perform one sweep in 8.0 seconds. In the start, a smooth increase in the amplitude is used to minimize potential hardware damage and also to generate the initial map.

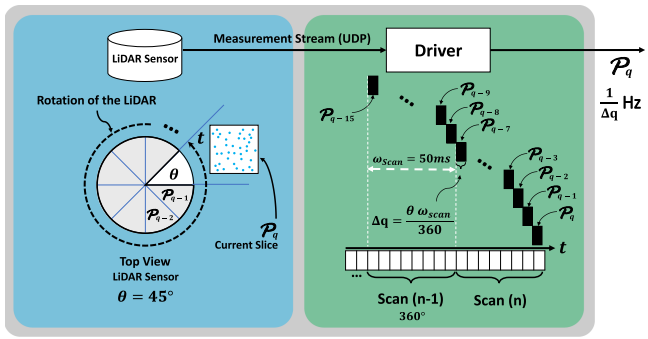


Fig. 4. Continuous Slicing Point Cloud (CSPC) model. This model gathers smaller collections of measurements that can be used in a concurrent pipeline instead of the normally used full 360° scan. Each slice $\mathcal{P}_q$ is defined for the last $\theta°$ of the rotation of the LiDAR.

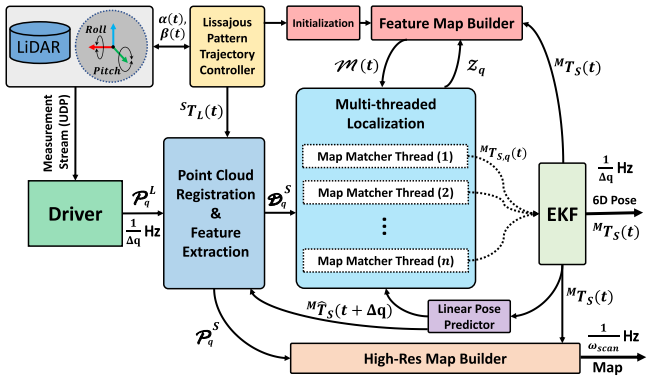


Fig. 5. Overview of the LoLa-SLAM framework. The results of the multi-threaded localization pipeline are fused using the EKF. The feature map is a voxelized map of the collected feature points with respect to the corresponding pose at time t. The voxelized high-resolution map is generated directly from the input CSPC.

is generated from the last $\theta = 45^\circ$ of the rotation of the LiDAR, each slice is available every $\Delta q = 6.25$ ms and noted by $\mathcal{P}_q$ with q indicating the last observed slice. With less angular coverage, the point cloud becomes more ambiguous because opposing features are missing to prevent misguided shifts during optimization in the pose estimation. However, the new measurements can be integrated with old ones to achieve the necessary number of measurements to ensure correct matching. For instance, we can add the new slice ($\theta = 45^\circ$) to the previously registered slices within the last 315°, in order to obtain a repeatedly full 360°scan, however, at a much higher update rate. While the reuse of past measurements does not affect the achieved update rate, the system's real-time performance needs to be carefully investigated due to the increase in computational usage.

V. Multi-Threaded Low-Latency Localization

A. System Overview

The overall framework of the proposed system is depicted in Fig. 5. Let's consider $\mathcal{P}_q^L$ to be the points in the LiDAR coordinate frame $\{L\}$ published in real-time with the update rate of $\frac{1}{\Delta q}$ Hz and the LiDAR to follow the motion with the generated Lissajous pattern. The point cloud $\mathcal{P}_q^L$ is first registered in the sensor coordinate frame using the transformation provided by the absolute status of the joints. During the start phase, the generated map points at time t are denoted with $\mathcal{M}(t)$, and as we assume the platform's pose is in the stationary state; we only collect the extracted feature points. The feature extraction node takes the point cloud $\mathcal{P}_q^S$ in the sensor coordinate frame and extracts the feature points. Let us assume for the first iteration that the minimum initial map is already available from the initialization step.

The multi-threaded localization pipeline takes the previously registered feature points $\mathcal{D}_{q-1}^M$ in combination with the extracted feature points from the current slice $\tilde{\mathcal{D}}_q$, for registration in the map. While, in principle, the current slice could be combined with more than one previous slice, for complexity and latency reasons, we limit it in our implementation to one previous scan slice. The combined feature points $\mathcal{D}_q^S$ are used to compute the precise motion of the LiDAR from matching the last sweeps within the map. With the slicing process described previously, a single thread is unable to handle the computation in real-time. The computation time increases because the diminished point clouds do not result in equally reduced optimization time, and the reuse of measurements adds additional calculations. Thus, we run the pose estimation in multiple threads with a modern multi-core processor and then fuse the individual concurrent estimates. Each thread extracts the pose transform concerning the map and uses the initial condition of the predicted 6D pose from the linear predictor unit. The Extended Kalman Filter (EKF) unit takes all the concurrent transformations published by the localization unit and fuses them for estimating the final 6D pose. Finally, the map builder unit takes and adds the new feature points into the map with respect to the associated final pose from the EKF unit. Similar to [22], considering that our approach uses a voxelized feature map for pose estimation, and as this unit runs in a dedicated process, a growing map does not affect the real-time localization procedure.

B. Point Cloud Registration and Feature Extraction

The measured point cloud is registered in the sensor coordinate frame using the transformation ${}^S\boldsymbol{T}_L(t)$. This transformation is obtained by considering the kinematics of the LiDAR actuation. As shown in Fig. 2, the laser scanner is attached to the sensor frame with two actively controlled joints to follow the pattern generated by Eq. (1). The homogeneous transformation is performed for all points i in $\mathcal{P}_{(q,i)}^L$. Let us define the homogeneous 3D affine transformation matrix $\boldsymbol{T}(t) = (\boldsymbol{R}(t), \boldsymbol{t}(t)) \in SE(3)$ as a rigid body transform composed of a rotation matrix $\boldsymbol{R}(t) \in SO(3)$ and a translation vector $\boldsymbol{t}(t) \in \mathbb{R}^3$, as

$$\boldsymbol{T} = \begin{pmatrix} \boldsymbol{R} & \boldsymbol{t} \\ 0 & 1 \end{pmatrix}, \tag{2}$$

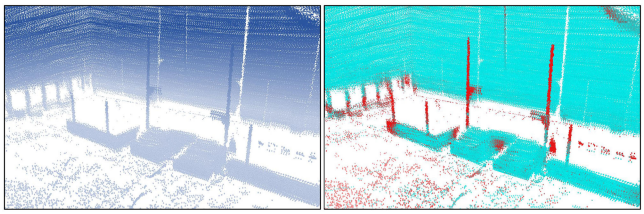
and accordingly, the points in the sensor coordinate frame $\mathcal{P}_{(q,i)}^S$ can be defined as

$$\mathcal{P}_{(q,i)}^S = {}^L\boldsymbol{T}_{S,q}(t)^{-1}\mathcal{P}_{(q,i)}^L, \tag{3}$$

while the transformation ${}^L\boldsymbol{T}_{S,q}(t)$ can be extracted from

$${}^L\boldsymbol{T}_{S,q}(t) = {}^L\boldsymbol{T}_{J_{pitch},q}(t)^{J_{pitch}}\boldsymbol{T}_{J_{roll},q}(t)^{J_{roll}}\boldsymbol{T}_{S,q}(t). \tag{4}$$

The geometric properties of the point cloud $\mathcal{P}_{(q,i)}^S$ in the sensor coordinate frame (i.e., roughness and normal vector) remain unchanged after performing the rigid transformations. Before the feature extraction, we use linear interpolation between the predicted pose and the previously estimated pose to remove the distortion of the point cloud similar to the approach from [22]. We extract the roughness value as a feature representation in this work. In this context, roughness value defines a term to evaluate the smoothness of the local surface. The feature extraction process is similar to the method used in [21]. However, instead of extracting the one-dimensional roughness of the points along the LiDAR's beam-lines, we extract a two-dimensional roughness value from the organized point cloud. We extract the two-dimensional roughness value by taking all neighboring points within a predefined spherical area and calculate the roughness



(a) (b)

Fig. 6. Illustration of the feature points in the LiDAR point cloud data. For better visualization, we show the accumulated CSPC data. ***a***) Shows the raw point cloud from the LiDAR measurements color-coded along the z axis. ***b***) Shows the calculated roughness of the points color-coded with cyan for low and red for high roughness values.

value in both vertical and horizontal directions. First, we find the set of adjacent points using the nearest neighbor search along the KD-tree. To improve the system's performance, as the size of each slice is smaller compared to the full scan mode, we filter correspondences to maintain small patches using ties broken by distance. This allows us to reduce the estimation time of the feature points for each slice. We define the two-dimensional roughness value $\mathcal{Z}$ for the point i by calculating the differences of the neighboring point set $\mathcal{N}^S_{(q,j),n}$ and normalize the term according to the distance as

$$\mathcal{Z}_i = 1 - \left\| \frac{1}{n\mathcal{P}^S_{(q,i)}} \sum_{j=1}^{n} \mathcal{N}^S_{(q,j)} \right\|, \tag{5}$$

where n is the number of the considered nearest neighbors of point i. The feature point set $\tilde{\mathcal{D}}_{(q,i)}$ is stored with the corresponding calculated roughness value. Recalling that each slice is a small portion of one full rotation of the LiDAR, the extracted feature points, which are sorted based on their roughness value, are well distributed within the observed point cloud for the entire FoV. In addition, we use the local entropy to assess the saliency of the feature points [35], and we remove those that have a higher local entropy value. This procedure helps us to avoid selecting points that have a high roughness score while not being reliable feature points. This can be inferred from the fact that their nearby points are not well aligned. For a simple example, consider the points distributed along with the objects on a crowded table. These points might have a high roughness score, but they are not as reliable as the corner points on the wall's edge for the localization. Fig. 6 illustrates the result of the proposed method for feature extraction.

C. Low-Latency Localization and Mapping

Pose Estimation. The localization framework proposed in this letter uses the predicted pose as an initial condition. It matches the feature points of the sliced scan into the map and extracts the corresponding transformation. This procedure is concurrently performed on each of the combined sliced point sets $\mathcal{D}^S_q$. The output of this process are synchronous individual pose estimates. The combined feature point set $\mathcal{D}^S_q$ can be determined by

$$\mathcal{D}^S_q = {}^M\boldsymbol{T}_{S,q-1}{}^{-1}\mathcal{D}^M_{q-1} + \tilde{\mathcal{D}}_q. \tag{6}$$

The estimated poses are later used for updating the state estimator in the EKF. Considering the point sets $\mathcal{D}^S_q$ and $\mathcal{M}^M_q$ in correspondence $\mathcal{D}^S_q \leftrightarrow \mathcal{M}^M_q$, and that they are related via a rigid body transform, in each thread we seek to estimate $\boldsymbol{R}$ and $\boldsymbol{t}$ such that

$$\mathcal{M}^M_q = \boldsymbol{R}\mathcal{D}^S_q + \boldsymbol{t} = {}^M\boldsymbol{T}_{S,q}(t)\mathcal{D}^S_q, \tag{7}$$

where the problem is well-studied with various closed-form solutions in the literature. To solve for the pose within the map, we need to establish a geometric relationship between $\mathcal{D}^M_q$ and $\mathcal{M}^M_q$. Using the predicted transformation ${}^M\hat{\boldsymbol{T}}_{S,q}(t+\Delta q)$, which we extracted from the previous observations of the motion, we first transform the feature points in $\mathcal{D}^S_q$ into the map coordinate frame using

$$\hat{\mathcal{D}}^M_q = {}^M\hat{\boldsymbol{T}}_{S,q}(t+\Delta q)\mathcal{D}^S_q. \tag{8}$$

Both of the point sets now are in the same coordinate frame, but they are not aligned perfectly as we used only the predicted transformation. However, we can find the nearest neighbor points for each feature point i in $\hat{\mathcal{D}}^M_q$ within the map point set. It is worth to mention that some of the points do not have any correspondence on the map. This means the closest point in the map is far from the selected feature point in $\hat{\mathcal{D}}^M_q$; therefore, we filter them using a simple threshold value based on Euclidean distance. Now we can derive a geometric relationship between all of the corresponding selected points in $\hat{\mathcal{D}}^M_q$, and the nearest points in the map as a non-linear function

$$f_{q,i}(\hat{\mathcal{D}}^M_{q,i}, {}^M\boldsymbol{T}_{S,q}) = \boldsymbol{d}_i, \quad i \in \hat{\mathcal{D}}^M_q \in \mathcal{M}^M_q, \tag{9}$$

where stacking Eq. (9) for each feature point in $\mathcal{D}^S_q$, we can obtain a non-linear function f_q and rewrite it as

$$f_q(\mathcal{D}^S_q, {}^M\hat{\boldsymbol{T}}_{S,q}{}^{-1}{}^M\boldsymbol{T}_{S,q}) = \boldsymbol{d}, \quad q \in Z^+, \tag{10}$$

where $\boldsymbol{d}$ contains the corresponding Euclidean distances for each point, and each row of f is associated with one feature point. We use the Levenberg-Marquardt trust-region algorithm as a non-linear least squares solver for this problem [36]. In our optimization problem, a groups of scalars must be converged together. This means the three components of a translation vector and the four components of the Quaternion that define the sensor's pose are used as a parameter set. We use the Huber loss function to reduce the influence of outliers on the non-linear least squares solver's solution. To solve Eq. (10), we compute the Jacobian matrix $\boldsymbol{J}$ for the non-linear function f with respect to ${}^M\boldsymbol{T}_{S,q}$ where

$$\boldsymbol{J} = \frac{\partial f}{\partial {}^M\boldsymbol{T}_{S,q}}, \tag{11}$$

and we further solve Eq. (10) with non-linear iteration toward minimizing the $\boldsymbol{d}$ to zero.

Linear Pose Predictor: We use a linear prediction based on the observed history of the estimated pose as an initial transformation for finding the correspondences within the map. We define the predicted transformation ${}^M\hat{\boldsymbol{T}}_{S,q}(t+\Delta q) = (\hat{\boldsymbol{R}}[\Theta^{-1}], \hat{\boldsymbol{t}})$

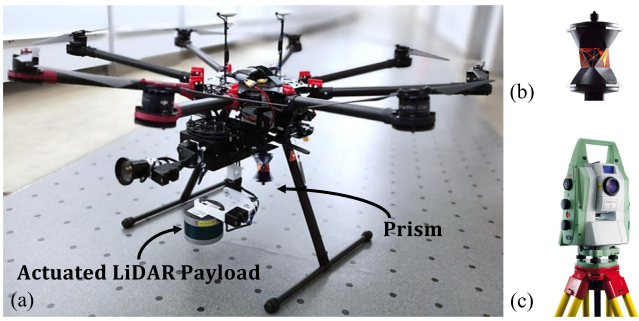


Fig. 7. (a) A UAV platform with an on-board computing unit and an active LiDAR payload. (b) The 360° reflector prism for tracking of the UAV (c) The Leica TS60 Tachymeter total station.

using the linear prediction model

$$ {}^{M}\hat{\boldsymbol{t}}_{S,q}(t+\Delta q) = \sum_{i=0}^{b} \boldsymbol{A}_i {}^{M}\boldsymbol{t}_{S,q}(t-i\Delta q), \tag{12} $$

$$ {}^{M}\hat{\boldsymbol{R}}_{S,q}(t+\Delta q) = {}^{M}\boldsymbol{R}_{S,q}\left[\sum_{i=0}^{b} \boldsymbol{A}_i \Theta^{-1}(t-i\Delta q)\right], \tag{13} $$

where b indicates the number of previous observations, $\boldsymbol{A}_i = \lambda_{i,b}\boldsymbol{I}_3$ is the predictor coefficients, and Θ is the element-wise Euler angle vector representation that is used to calculate the rotation matrix. Because the previously observed poses are equally-spaced values in time, a polynomial interpolation can be defined as a linear combination of the given observation. The predictor coefficient $\lambda_{i,b}$ is a scalar multiplication for the identity matrix $\boldsymbol{I}_3$. In our linear model these elements defined by the ith component of the bth row of Pascal's triangular matrix of the binomial coefficients [37].

VI. Experimental Evaluation

To assess the performance of the proposed framework, we tested our approach on a recorded data set from a large sports hall. To evaluate the pose estimation accuracy, an octocopter equipped with the sensor payload is tracked with a Leica TS60 Tachymeter total station. The total station in tracking mode has a stated measurement accuracy of 0.2 cm at a rate of 10 Hz. To enable tracking with the total station, a 360° reflector prism is mounted on the octocopter as shown in Fig. 7. The static transformation is applied to have the total station's measurements in the same coordinate system as the UAV. The octocopter is flown multiple times with different trajectories in a sports hall. Flight duration in total is about 36 minutes, and the velocity varies from 0.05 to 6.0 km/h. The data were recorded applying Lissajous, raster, and static modes. In the static mode, the LiDAR was held in a horizontal state. In the other two modes, the LiDAR was actuated using the defined patterns. Five data acquisition options are used for the LiDAR point cloud; the default mode (Full) is a 360° scan at 20 Hz, and the other modes are using scan slicing with a segment size of 30, 45, 90, and 180 degrees, respectively.

An example of 6D trajectory tracking using LoLa-SLAM and the total station is illustrated in Fig. 8. The results show that

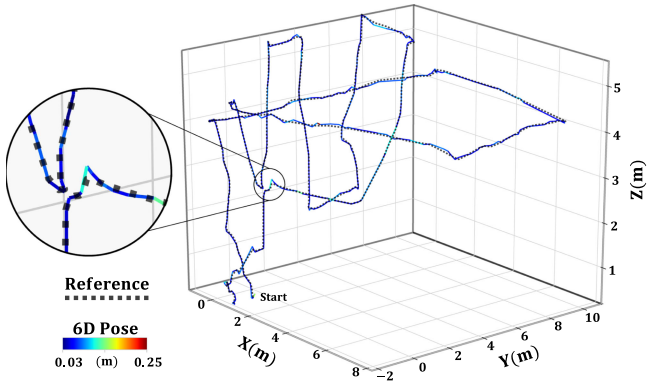


Fig. 8. Absolute trajectory error of the UAV in a sports hall in comparison to the GT pose trajectory from the total station.

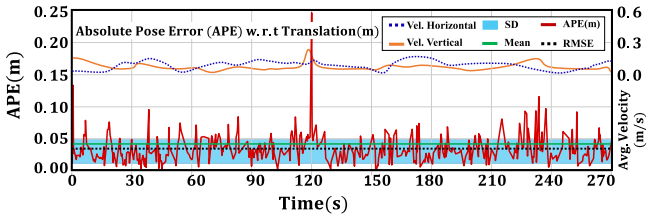


Fig. 9. Distribution of the 6D pose error with respect to the GT data of the total station. The root mean square error (RMSE) for this trajectory is equal to 3.6 cm.

TABLE I
Results of Multiple Experiments Using the Recorded Data Set in a Sports Hall. All the Experiments are Conducted Using the Onboard Core.i7 Intel Processor - 4 Cores and With 16 GB of RAM

	Experiment (θ° + Pattern)	Total Dist. (m)	CPU Usage Avg.	RMSE - R (°)	**RMSE - T (m)**	Max Error (m)	SD (m)	**Update Rate (Hz)**	**Latency (ms)**
LoLa-SLAM (Proposed)	**30°+Lissajous**	184	-	-	**Failed**	-	-	-	-
	30°+Raster	175	-	-	**Failed**	-	-	-	-
	30°+Static	193	92%	7.4	**0.079**	0.48	0.041	**240**	**16**
	45°+Lissajous	184	85%	4.3	**0.039**	0.27	0.028	**160**	**19**
	45°+Raster	175	83%	4.4	**0.057**	0.41	0.033	**160**	**21**
	45°+Static	193	77%	5.8	**0.070**	0.42	0.034	**160**	**18**
	90°+Lissajous	184	84%	4.4	**0.036**	0.27	0.033	**80**	**39**
	90°+Raster	175	83%	4.4	**0.055**	0.43	0.032	**80**	**43**
	90°+Static	193	75%	5.8	**0.067**	0.46	0.034	**80**	**38**
	180°+Lissajous	184	55%	4.1	**0.035**	0.25	0.027	**40**	**78**
	180°+Raster	175	51%	4.9	**0.052**	0.39	0.029	**40**	**85**
	180°+Static	193	52%	3.7	**0.066**	0.38	0.031	**40**	**77**
LOAM	**Full+Lissajous**	184	53%	4.6	**0.034**	0.26	0.024	**20**	**93**
	Full+Raster	175	56%	4.8	**0.050**	0.44	0.031	**20**	**97**
	Full+Static	193	49%	3.9	**0.065**	0.47	0.033	**20**	**89**

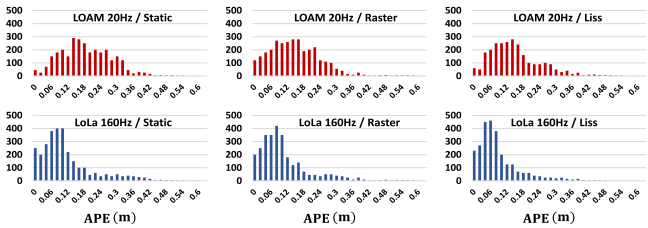


Fig. 10. Histogram of the absolute pose error (APE) distribution when performing active-controlled hovering (PD controller) with 20 Hz and 160 Hz pose feedback.

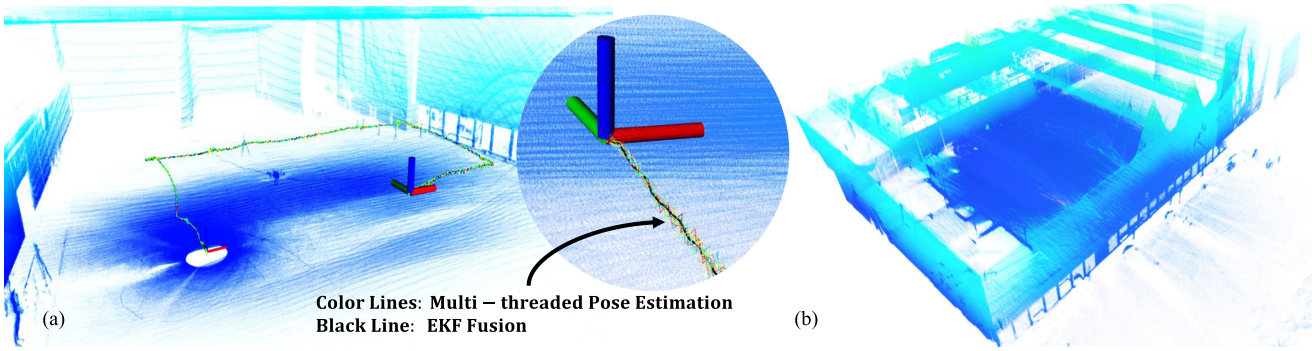


Fig. 11. Reconstructed 3D map of a sports hall using the LoLa-SLAM framework. $\boldsymbol{a}$) Concurrent pose estimations from the multi-threaded localization and the fused 6D pose of the UAV. $\boldsymbol{b}$) The dense point cloud map from a different perspective.

there is less deviation in the horizontal direction compared to the vertical direction. We can see that the larger errors mostly appear when the drone moves fast in the vertical direction. This issue could have two origins; First, for the linear predictor, as the map matching uses this prediction as an initial input in the localization step, a poor prediction could cause such a wrong matching. Second, imperfect observations caused by the LiDAR actuation: due to the insufficient points in the vertical face compared to the horizontal. To investigate more on this problem, we further compared the proposed method with the static LiDAR; this means even fewer points are observed in the vertical faces, and as expected, the error in the vertical direction increased considerably. In addition to this, as shown in Fig. 9, the RMSE of a selected trajectory is only 3.6 cm; however, due to the fast motion, the maximum pose error increased up to 26.8 cm at some points.

Furthermore, the latency and accuracy of the pose estimation are investigated. In Table I, we provide a comparison to the well known state-of-the-art SLAM framework LOAM [21]. By analyzing the results, we can see a trade-off between the chosen slice size, pose estimation accuracy, and the update rate. Choosing a smaller slice size results in a higher update rate; however, this means fewer features in each slice are used for localization. Consequently, the accuracy of the pose estimation starts deviating. In an extreme scenario, with using the slice size $\theta = 30^{\circ}$, the system cannot track the pose, and therefore the localization starts drifting. If the drift is larger than 0.5 m, we consider the localization to fail.

Considering a slice size $\theta = 45^{\circ}$ is a satisfactory candidate, we achieve a latency of less than 20 ms on average, which is acceptable for real-time drone navigation. In comparison, using the full scan mode, we observe a latency of 93 ms. Besides, we investigate the use of LoLa-SLAM in a closed-loop system while the drone was hovering using a PD controller. The histogram of the APE distribution is shown in Fig. 10. The figure shows that, while the accuracy of the pose estimation is lower using LoLa-SLAM compared to LOAM, the drone deviation is smaller due to the shorter latency and higher temporal update rate. While LoLa-SLAM provides a higher update rate and lower latency, it is computationally more expensive than the other approaches. For instance, on average, the proposed framework needs around 30% more processing power compared to LOAM. However, thanks

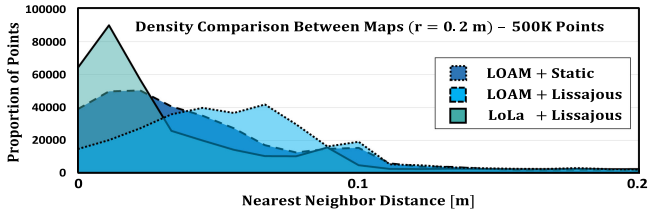


Fig. 12. Density comparison between the 3D maps. The distribution of the point spacing (nearest-neighbor distance) for randomly selected 500 K points within each map.

to the concurrent multi-threaded map matching, the system still runs in real-time.

The accuracy and overall validity of the proposed system are additionally assessed using the created dense 3D map. To evaluate the map quality, Fig. 11 shows the resulting point cloud from different perspectives, which allows for better interpretation of the scene. The tripod of the total station and pillars in the sports hall can be identified in the globally aligned and voxelized 3D map. It is worth stating that this level of 3D mapping is not obtainable without a precise localization. To evaluate the superior quality of the created map compared to LOAM, we further illustrate the density comparison between the point clouds similar to the method described in [25]. The point densities show the uniformity of the generated map. The distribution of the point spacing (nearest-neighbor distance) for randomly samples points is visualized in Fig. 12. The densities are represented as distributions of point spacing, as determined by each point's nearest neighbor. We use randomly selected 500 K points within each map and determined the density in a sphere with 0.2 m area around each point. This comparison shows that LoLa-SLAM has the highest proportion of points in the distance of less than 3.0 cm.

VII. Conclusion

In this work, we presented a customized system to localize a UAV in an indoor GPS-denied environment. We approached this challenge by employing an actuated LiDAR platform and utilizing the 2D Lissajous spinning pattern. We further developed a novel SLAM framework based on the continuous sliced point cloud to reduce the ego-motion estimation latency, named

LoLa-SLAM. We showed that the proposed SLAM system's accuracy and reliability remain adequate, and this is while the temporal update rate increased by at least 2X for the pose estimation. Besides, we demonstrated the proposed framework could simultaneously generate a high-quality 3D map. We demonstrate both the low-latency pose estimation and the overall accuracy of LoLa-SLAM by multiple experiments in an indoor environment. We showed that the update rate increases by choosing a smaller slice size; however, the proposed system can fail in the feature matching process if the slice size is too small. In addition, we examined the computational complexity which increases by 30% compared to the common methods.

In future work, to further improve the system, the odometry predictor and the EKF fusion can be integrated with redundant attitude and heading reference systems (AHRS) [24], [38]. Furthermore, adding an adaptive slice size controller based on the UAV's velocity can also be investigated to improve the system performance. Besides, environment aware attentive control of the LiDAR can be employed to improve the point cloud observation, map generation, and reliable pose estimation in complex indoor environments.

REFERENCES

[1] J. Miranda, S. Larnier, A. Herbulot, and M. Devy, "Uav-based inspection of airplane exterior screws with computer vision," in *Proc. 14th Int. Joint Conf. Comput. Vis., Imag. Comput. Graph. Theory Appl.*, Feb. 2019.

[2] F. Valenti, D. Giaquinto, L. Musto, A. Zinelli, M. Bertozzi, and A. Broggi, "Enabling computer vision-based autonomous navigation for unmanned aerial vehicles in cluttered gps-denied environments," in *Proc. Int. Conf. Intell. Transp. Syst.*, 2018, pp. 3886–3891.

[3] J. Siva and C. Poellabauer, "Robot and drone localization in gps-denied areas," in *Mission-Oriented Sensor Networks and Systems: Art and Science*. Berlin, Germany: Springer, 2019, pp. 597–631.

[4] Y. D. Yasuda, L. E. G. Martins, and F. A. Cappabianco, "Autonomous visual navigation for mobile robots: A systematic literature review," *ACM Comput. Surv.*, vol. 53, no. 1, pp. 1–34, 2020.

[5] X. Zhou, Z. Yi, Y. Liu, K. Huang, and H. Huang, "Survey on path and view planning for uavs," *Virtual Reality Intell. Hardware*, vol. 2, no. 1, pp. 56–69, 2020.

[6] S. Jung, S. Hwang, H. Shin, and D. H. Shim, "Perception, guidance, and navigation for indoor autonomous drone racing using deep learning," *IEEE Robot. Automat. Lett.*, vol. 3, no. 3, pp. 2539–2544, Jul. 2018.

[7] R. S. Dimitrova, M. Gehrig, D. Brescianini, and D. Scaramuzza, "Towards low-latency high-bandwidth control of quadrotors using event cameras," *IEEE Int. Conf. Robot. Autom. (ICRA)*, 2020, pp. 4294–4300.

[8] R. Pérez-Alcocer and J. Moreno-Valenzuela, "A novel lyapunov-based trajectory tracking controller for a quadrotor: Experimental analysis by using two motion tasks," *Mechatronics*, vol. 61, pp. 58–68, 2019.

[9] R. Mur-Artal and J. D. Tardós, "Visual-inertial monocular SLAM with map reuse," *IEEE Robot. Automat. Lett.*, vol. 2, no. 2, pp. 796–803, Apr. 2017.

[10] D. Van Opdenbosch, M. Oelsch, A. Garcea, T. Aykut, and E. Steinbach, "Selection and compression of local binary features for remote visual SLAM," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 7270–7277.

[11] S. Lynen *et al.*, "Large-scale, real-time visual-inertial localization revisited," *Int. J. Robot. Res.*, vol. 39, no. 9, pp. 1061–1084, 2020.

[12] M. W. Achtelik, S. Lynen, S. Weiss, L. Kneip, M. Chli, and R. Siegwart, "Visual-inertial SLAM for a small helicopter in large outdoor environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2012, pp. 2651–2652.

[13] D. Zhang, Z. Gong, Y. Chen, J. Zelek, and J. Li, "SLAM-based multisensor backpack lidar systems in gnss-denied environments," in *Proc. Geosci. Remote Sens. Symp.*, 2019, pp. 8984–8987.

[14] M. Velas, M. Spanel, T. Sleziak, J. Habrovec, and A. Herout, "Indoor and outdoor backpack mapping with calibrated pair of velodyne lidars," *Sensors*, vol. 19, no. 18, 2019, Art. no. 3944.

[15] J. Lin and F. Zhang, "Loam_livox: A fast, robust, high-precision lidar odometry and mapping package for lidars of small fov," *IEEE Int. Conf. Robot. Autom. (ICRA)*, 2020, pp. 3126–3131.

[16] J. Lin and F. Zhang, "A fast, complete, point cloud based loop closure for lidar odometry and mapping," 2019, *arXiv:1909.11811*.

[17] H. Qin *et al.*, "A stereo and rotating laser framework for UAV navigation in GPS denied environment," in *Proc. IECON 42nd Annu. Conf. IEEE Ind. Electron. Soc.*, 2016, pp. 6061–6066.

[18] W. Zhen and S. Scherer, "A unified 3D mapping framework using a 3D or 2D lidar," *Int. Symp. Experimental Robot.*, 2018, pp. 702–711.

[19] H. Qin *et al.*, "Autonomous exploration and mapping system using heterogeneous uavs and ugvs in gps-denied environments," *IEEE Trans. Veh. Technol.*, vol. 68, no. 2, pp. 1339–1350, Feb. 2019.

[20] R. Voges, C. S. Wieghardt, and B. Wagner, "Timestamp offset determination between an actuated laser scanner and its corresponding motor," *ISPRS Ann. Photogrammetry, Remote Sens. Spatial Inf. Sci.*, vol. 4, pp. 99–106, 2017.

[21] J. Zhang and S. Singh, "Loam: Lidar odometry and mapping in real-time," *Robot.: Sci. Syst.*, vol. 2, no. 9, 2014.

[22] J. Zhang and S. Singh, "Low-drift and real-time lidar odometry and mapping," *Auton. Robots*, vol. 41, no. 2, pp. 401–416, 2017.

[23] X. Ji, L. Zuo, C. Zhang, and Y. Liu, "Lloam: Lidar odometry and mapping with loop-closure detection based correction," in *Proc. IEEE Int. Conf. Mechatronics Automat.*, 2019, pp. 2475–2480.

[24] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, "Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping," 2020, *arXiv:2007.00258*.

[25] L. Kaul, R. Zlot, and M. Bosse, "Continuous-time three-dimensional mapping for micro aerial vehicles with a passively actuated rotating laser scanner," *J. Field Robot.*, vol. 33, no. 1, pp. 103–132, 2016.

[26] S. Anderson and T. D. Barfoot, "Towards relative continuous-time SLAM," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2013, pp. 1033–1040.

[27] M. Bosse, R. Zlot, and P. Flick, "Zebedee: Design of a spring-mounted 3D range sensor with application to mobile mapping," *IEEE Trans. Robot.*, vol. 28, no. 5, pp. 1104–1119, Oct. 2012.

[28] J. W. Anderson and G. M. Clayton, "Lissajous-like scan pattern for a gimballed lidar," in *Proc. IEEE/ASME Int. Conf. Adv. Intell. Mechatronics*, 2014, pp. 1171–1176.

[29] M. Benson, J. Nikolaidis, and G. M. Clayton, "Lissajous-like scan pattern for a nodding multi-beam lidar," in *Proc. ASME Dyn. Syst. Control Conf.*, 2018, pp. 1171–1176.

[30] D. Droeschel, M. Nieuwenhuisen, M. Beul, D. Holz, J. Stückler, and S. Behnke, "Multilayered mapping and navigation for autonomous micro aerial vehicles," *J. Field Robot.*, vol. 33, no. 4, pp. 451–475, 2016.

[31] W. Zhen, S. Zeng, and S. Soberer, "Robust localization and localizability estimation with a rotating laser scanner," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2017, pp. 6240–6245.

[32] H. Nakagomi *et al.* , "3D scan matching for mobile robot localization over rough terrain," *Elect. Eng. Jpn.*, vol. 209, no. 3/4, pp. 14–25, 2019.

[33] M. T. Benson, H. Sathishchandra, G. M. Clayton, and S. B. Andersson, "Compressive sensing-based reconstruction of lissajous-like nodding lidar data," in *Proc. Dyn. Syst. Control Conf.*, 2019, Art. no. V003T21A010.

[34] A. Al-Nuaimi, W. Lopes, P. Zeller, A. Garcea, C. Lopes, and E. Steinbach, "Analyzing lidar scan skewing and its impact on scan matching," in *Proc. IEEE Int. Conf. Indoor Positioning Indoor Navigation*, 2016, pp. 1–8.

[35] G. Li, Y. Geng, and W. Zhang, "Autonomous planetary rover navigation via active SLAM," *Aircraft Eng. Aerospace Tech.*, vol. 91, no. 1, pp. 60–68, 2018.

[36] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge, U.K.: Cambridge Univ. Press, 2003.

[37] C. K. Williams, "Prediction with gaussian processes: From linear regression to linear prediction and beyond," in *Learning in Graphical Models*. Berlin, Germany: Springer, 1998, pp. 599–621.

[38] M. Karimi, E. Babaians, M. Oelsch, T. Aykut, and E. Steinbach, "Skewed-redundant hall-effect magnetic sensor fusion for perturbation-free indoor heading estimation," in *Proc. 4th IEEE Int. Conf. Robot. Comput.*, 2020, pp. 367–374.