A decentralized framework for simultaneous calibration, localization and mapping with multiple LiDARs

Jiarong Lin, Xiyuan Liu and Fu Zhang

***Abstract*— LiDAR is playing a more and more essential role in autonomous driving vehicles for objection detection, self localization and mapping. A single LiDAR frequently suffers from hardware failure (e.g., temporary loss of connection) due to the harsh vehicle environment (e.g., temperature, vibration, etc.), or performance degradation due to the lack of sufficient geometry features, especially for solid-state LiDARs with small field of view (FoV). To improve the system robustness and performance in self-localization and mapping, we develop a decentralized framework for simultaneous calibration, localization and mapping with multiple LiDARs. Our proposed framework is based on an extended Kalman filter (EKF), but is specially formulated for decentralized implementation. Such an implementation could potentially distribute the intensive computation among smaller computing devices or resources dedicated for each LiDAR and remove the single point of failure problem. Then this decentralized formulation is implemented on an unmanned ground vehicle (UGV) carrying 5 low-cost LiDARs and moving at $1.3m/s$ in urban environments. Experiment results show that the proposed method can successfully and simultaneously estimate the vehicle state (i.e., pose and velocity) and all LiDAR extrinsic parameters. The localization accuracy is up to 0.2% on the two datasets we collected. To share our findings and to make contributions to the community, meanwhile enable the readers to verify our work, we will release all our source codes[1] and hardware design blueprint[2] on our Github.**

I. Introduction

With the ability of localizing positions and constructing local maps, simultaneous locomotion and mapping (SLAM) using sensors like camera, IMU, LiDAR, etc., are serving as the pillars for missions in autonomous driving [1], field survey [2] and 3D reconstruction [3]. Though visual SLAM has been widely applied in exploration and navigation tasks [4, 5], LiDAR SLAM [6]–[8] is still of significant essence. Compared with visual sensor, LiDAR is capable of providing high frequency 6 DoF state estimation with low-drift and simultaneously yielding a high resolution environment map. Furthermore, LiDAR is more robust to environments with illumination variations, poor light conditions or few optical textures [9].

Driven by these widespread robotic applications [10, 11], LiDARs have undergone unprecedented developments. In particular, solid state LiDARs have received the most interests [12]–[14]. Compared with conventional multi-line spining LiDARs, solid state LiDARs are more cost effective while retaining similar level of performance (e.g., map accuracy, point density). Nevertheless, a major drawback is their small FoV that they are prone to degeneration when facing geometrical feature-less scenes (e.g., wall, sky or grassland). To overcome this, multiple LiDARs are usually embedded at different locations of the vehicle and communicate via the vehicle bus (e.g., controller area network (CAN)), naturally forming a distributed sensor system. An illustrative example is shown in Fig. 1, where 5 LiDARs are installed on a robotic ground vehicle moving in 6 DoF.

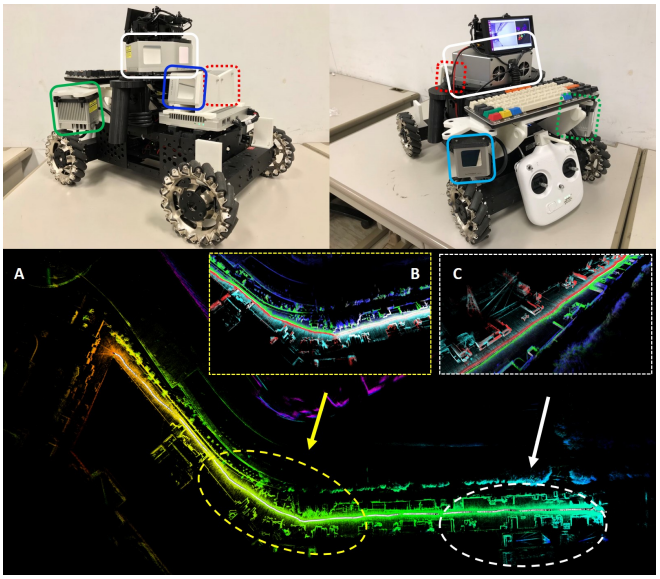

Fig. 1: Top: Our decentralized multi-LiDAR vehicle platform. The color of the bounding box is in accordance with the point cloud produced from the same LiDAR (as shown in B, C below). Bottom: A): The map built from $Scene$-1. The point cloud color is calculated by reflectivity. B), C): The detailed point cloud at the start and corner of the map. The color indicates the origin of the LiDAR. Our accompanying video is available at `https://youtu.be/n8r4ch8h8oo`

The use of multiple distributed LiDARs brings many new challenges in its localization and mapping: (1) extrinsic calibration. Since LiDARs are installed at different (and usually far apart) locations of the vehicle body, their relative pose is not perfectly rigid and may drift over time. This requires an online extrinsic calibration. This will be even more challenging when two adjacent LiDARs have very small overlap; (2) high network bandwidth and computation requirements. A LiDAR is usually generating raw point data at a very fast rate. Sending all LiDAR data to a central computer could not only easily jam the vehicle network and the central computer, causing single point of failure, but

J. Lin, X. Liu and F. Zhang are with the Department of Mechanical Engineering, Hong Kong University, Hong Kong SAR., China. {jiarong.lin, xliuba, fuzhang}@hku.hk

[1]https://github.com/hku-mars/decentralized_loam

[2]https://github.com/hku-mars/lidar_car_platfrom

also dramatically increases the computation power (meaning high power consumption, large noise, etc.). A potentially more robust way is to process each LiDAR's point data in its dedicated computer (e.g., electronic computing unit) and communicate the processed data (e.g., vehicle state, which is usually very small data) via the vehicle network.

In this paper, we present a decentralized multi-LiDAR calibration, localization and mapping system. The system is based on a decentralized formulation of EKF algorithm, which simultaneously runs on all LiDAR computers (or its allocated computing resources). All EKF copies perform the same procedures: maintaining an augmented state vector consisting the pose (and velocity) of the geometric center and the extrinsic parameters of all LiDARs, predicting from the most recent state update received from the rest EKF copies in the network, updating the state vector with new coming frames from its local LiDAR, and publishing the updated state vector to the network for other EKF copies to use.

In summary, our contributions are:

- We have proposed a calibration, localization and mapping system utilizing constant velocity model and EKF, which is capable of online estimating and updating LiDAR extrinsic w.r.t. geometric center.
- We present a decentralized multi-sensor calibration and fusion framework, which could be implemented in a distributed way and are potentially more robust to failures of central computers or individual sensors.
- We have verified the convergence and accuracy of the proposed framework on actual systems and have achieved high precision localization and mapping results when compared with previous single LiDAR SLAM solution.

II. Related Work

To date, multi-LiDAR sensors have been implemented in obstacle detection and tracking [15, 16], computing occupancy map [17] and natural phenomenon observation [18, 19]. All these setups rely on a central processing unit for computation and data exchange. Few research attention has been focused on the decentralization property of the multi-LiDAR system, however, which makes the above mentioned applications vulnerable to sensor message delay or loss. Furthermore, the unsupervised extrinsic calibration and sensor fusion of multi-LiDAR system remain to be discovered.

Combining several sensors has led to the issue of multi-sensor data fusion. The simplest way is to use loosely coupled sensor fusion [20], though computationally efficient, the decoupling of multi-sensor constraints may cause information loss. Tightly coupled sensor fusion model have also been discussed in [21] to improve the map accuracy. The joint optimization of entire sensor measurements and state vector is too time consuming, however, especially for high frequency sensor like LiDAR. Originated from statistics, EKF based sensor fusion [8] has become dominant in LiDAR SLAM due to their simple, linear and efficient recursive algorithm. In our approach, we implement EKF to maintain an augmented state vector to achieve a balance between productivity and precision.

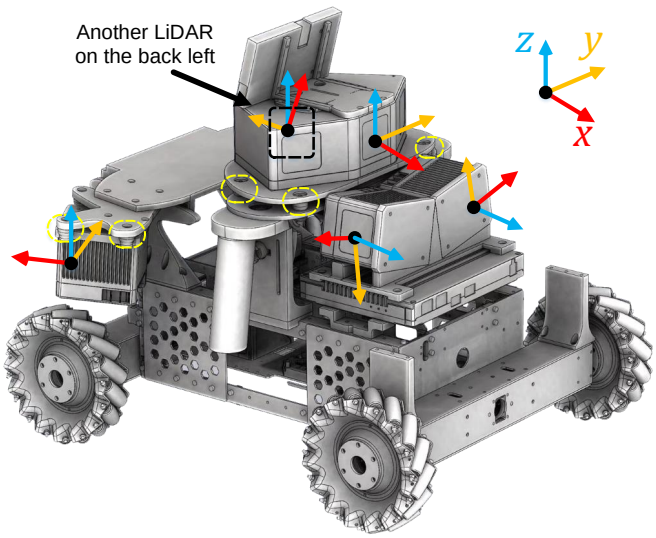

Fig. 2: Our platform for data sampling, with 5 LiDARs installed on the car platform. To prevent the mechanical vibration caused by the rotating wheels on the rough ground, we add some damper ball between the connection of LiDAR and the platform (marked inside the yellow dashed circle).

In addition to filtering, extrinsic calibration (recover rigid-body transformation between sensors) has been widely implemented to improve the SLAM precision. The majority of current LiDAR extrinsic calibration involve the following assumptions: known retroreflective targets or artificial environments [22]. This requirement is hard to meet if users want to customize the mounting position that unsupervised calibration is preferred. Motion based approaches have been described in [23], however, their results are easily affected by the cumulated drift from the motion. Appearance based approaches have been addressed in [24] that the optimal extrinsic is solved by maximizing overall point cloud quality. In contrast, our approach starts from a given initial value and iteratively utilizes EKF to refine extrinsic online. To the best our knowledge, our work is the first work that fuses data from multiple LiDARs in a decentralized framework, which can not only address the problem of localization and mapping, but can also online calibrate the extrinsic of 6-DoF. The results shown in Section. VII demonstrates that our approach is of high-precision and effectiveness.

III. Overview

The configuration of our system is shown in Fig. 2, we have five different LiDARs installed on the car platform, with their front face looking "front", "left", "right", "back-left" and "back-right". LiDAR-1 is Livox-MID100[3], with $98.4°$ of horizontal and $38.4°$ of vertical FoV. Other LiDARs are all Livox-MID40, with $38.4°$ circular FoV. Due to the limited FoV, there is no overlapping areas between any two LiDARs (see Fig. 3).

[3]https://www.livoxtech.com/mid-40-and-mid-100/specs

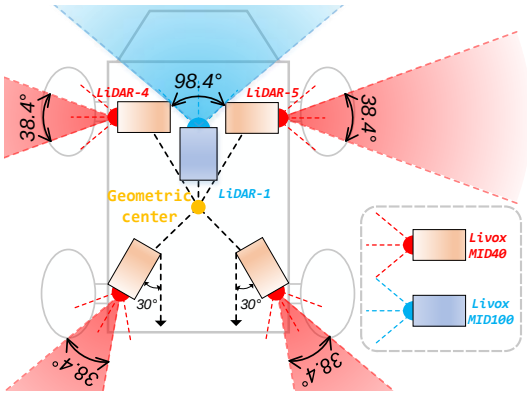

Fig. 3: The configurations of our LiDAR installation.

	Livox MID100	Livox MID40	3D printing & others	Total
Price	$ 1499	$ 599 × 4	≤ $100	≤ $3995

TABLE I: List of the materials (not including the robot platform) in our multi-LiDAR system, with the total price about 4k USD, including 5 LiDARs and the fees of 3D printing.

To prevent the vibrations caused by rotating the *mecanum wheels*[4] on rough ground, which could cause high-frequency motion blur effect on the LiDAR point cloud. We add the rubber damping-ball at the connection of LiDAR and car platform, which could effectively compensate the vibration. In addition, most of our mechanical parts are 3D printed with the PLA material, which can be easily distorted by the applied force. By this, we do not treat our LiDAR group as a rigid system.

Our platform is of low-cost, with all of our mechanical parts being 3D printed, whose total price is about 4k USD (details are shown in Table. I). With the algorithm proposed in our following sections, we can achieve a precision around 0.2%. For more details of our platform, we strongly recommend the readers visiting the project on our GitHub.[5]

IV. Vehicle Model

A. Notation

In our work, we use the 4×4 matrix $\mathbf{T}$ to denote the pose and rigid transformation in 6 degrees of freedom (DoF):

$$\mathbf{T}=\begin{bmatrix}\mathbf{R} & \mathbf{t}\\ \mathbf{0} & 1\end{bmatrix}\in SE(3)$$

Since the 3×3 rotation matrix $\mathbf{R}$ is of 3 DoF, we use a rotation vector $\mathbf{r}\in\mathbb{R}^3$ as a minimal parameterization, which represents the rotation in the form of angle$-$axis.

$$\mathbf{R}=e^{\widehat{\mathbf{r}}}\in SO(3) \tag{1}$$

where $\widehat{\cdot}$ transforms a vector into a skew-symmetric matrix. The conversions between $\mathbf{R}$ and $\mathbf{r}$ are denoted as $\mathbf{R}\{\cdot\}$ and $\mathbf{r}\{\cdot\}$ for convenience.

$$\mathbf{R}=\exp(\mathbf{r})=\mathbf{R}(\mathbf{r}) \tag{2}$$
$$\mathbf{r}=\log(\mathbf{R})=\mathbf{r}(\mathbf{R}) \tag{3}$$

[4]https://en.wikipedia.org/wiki/Mecanum_wheel
[5]https://github.com/hku-mars/lidar_car_platfrom

Using the minimal parameterization of $\mathbf{R}$, the full pose $\mathbf{T}$ can also be parameterized minimally by $\mathbf{x}=[\mathbf{r},\mathbf{t}]^T\in\mathbb{R}^6$, and the conversions between these two can be denoted as:

$$\mathbf{T}=\mathbf{T}(\mathbf{x}) \tag{4}$$
$$\mathbf{x}=\mathbf{x}(\mathbf{T}) \tag{5}$$

Sometimes, we also use the notation $\mathbf{T}=(\mathbf{r},\mathbf{t})$ to represent the minimal parameterization of $\mathbf{T}$.

B. Constant velocity model

Viewing the robot as a rigid body, its pose can be represented by a reference frame (e.g., at the geometric center in Fig. 3). Furthermore, we use a constant velocity model as in [25] to model the 6 DoF motion of the robot. Denote $\mathbf{r}_k^c$ the robot attitude , $\mathbf{t}_k^c$ is the translation, $\boldsymbol{\omega}_k^c$ the angular velocity, and $\mathbf{v}_k^c$ the linear velocity, all at time k, then a constant velocity model yields a state equation as below:

$$\mathbf{r}_{k+1}^c=\mathbf{r}\left(\mathbf{R}\left(\mathbf{r}_k^c\right)\exp\left(\widehat{\boldsymbol{\omega}}_k^c\cdot\Delta t\right)\right) \tag{6}$$
$$\mathbf{t}_{k+1}^c=\mathbf{t}_k^c+\mathbf{v}_k^c\cdot\Delta t \tag{7}$$
$$\boldsymbol{\omega}_{k+1}^c=\boldsymbol{\omega}_k^c+\boldsymbol{\epsilon}_\omega \tag{8}$$
$$\mathbf{v}_{k+1}^c=\mathbf{v}_k^c+\boldsymbol{\epsilon}_\mathbf{v} \tag{9}$$

where Δt is the time difference from the last update at t_k and the current update at t_{k+1} (i.e., $\Delta t=t_{k+1}-t_k$), $\boldsymbol{\epsilon}_\omega$ and $\boldsymbol{\epsilon}_\mathbf{v}$ are the force and torque impulse applied to the robot. They are usually modeled as zero-mean Gaussian noise:

$$[\boldsymbol{\epsilon}_\omega,\boldsymbol{\epsilon}_\mathbf{v}]^T\sim\mathcal{N}(\mathbf{0},\boldsymbol{\Sigma}_\mathbf{w})$$

The above state model can be rewritten as a more compact form as below:

$$\mathbf{x}_{k+1}^c=\mathbf{f}\left(\mathbf{x}_k^c,\mathbf{w};\Delta t\right)\in\mathbb{R}^{12} \tag{10}$$

where $\mathbf{x}_k^c=[\mathbf{r}_k^c,\mathbf{t}_k^c,\boldsymbol{\omega}_k^c,\mathbf{v}_k^c]^T$ and $\mathbf{w}=[\boldsymbol{\epsilon}_\omega,\boldsymbol{\epsilon}_v]^T$.

C. Extrinsic model

Assuming there are N LiDARs and $\mathbf{T}^{ei}=\left(\mathbf{r}^{ei},\mathbf{t}^{ei}\right)$ denotes the extrinsic of i-th LiDAR frame w.r.t the reference frame, we have the pose $\mathbf{T}_k^i$ of i-th LiDAR at time k as:

$$\begin{aligned}\mathbf{T}_k^i&=\mathbf{T}_k^c\mathbf{T}_k^{ei}\\&=\begin{bmatrix}\mathbf{R}_k^c\mathbf{R}^{ei} & \mathbf{R}_k^c\mathbf{t}^{ei}+\mathbf{t}_k^c\\ \mathbf{0} & 1\end{bmatrix}\end{aligned} \tag{11}$$

where $\mathbf{T}_k^c=(\mathbf{r}_k^c,\mathbf{t}_k^c)$ is the pose at time t_k.

D. Full state model

Denote $\mathbf{x}^{ei}=\left[\mathbf{r}^{ei},\mathbf{t}^{ei}\right]^T\in\mathbb{R}^6$ as the state associated the i-th LiDAR extrinsic parameters, then the full state is

$$\mathbf{x}=\begin{bmatrix}\mathbf{x}^c & \mathbf{x}^{e1} & \mathbf{x}^{e2} & \cdots & \mathbf{x}^{eN}\end{bmatrix}^T\in\mathbb{R}^{12+6N} \tag{12}$$

and the state model is

$$\mathbf{x}_{k+1}^c=\mathbf{f}\left(\mathbf{x}_k^c,\mathbf{w};\Delta t\right) \tag{13}$$
$$\mathbf{x}_{k+1}^{e1}=\mathbf{x}_k^{e1} \tag{14}$$
$$\mathbf{x}_{k+1}^{e2}=\mathbf{x}_k^{e2} \tag{15}$$
$$\vdots \tag{16}$$
$$\mathbf{x}_{k+1}^{e2}=\mathbf{x}_k^{eN} \tag{17}$$

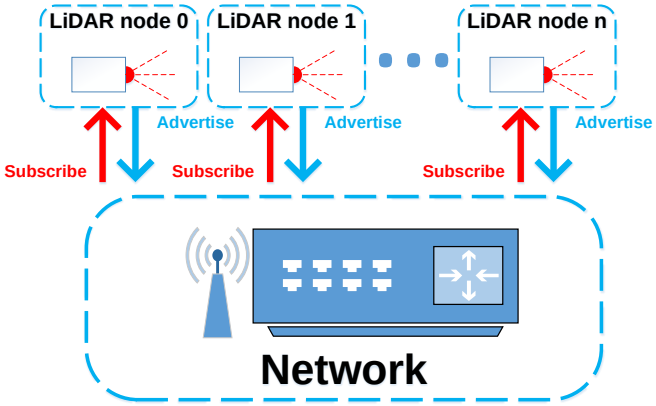

Fig. 4: The framework of our decentralized system, each LiDAR refreshes the newest state $\mathbf{x}$ by subscribing the message from the network. Once the point cloud registration with latest coming data is complete, it will advertise the selected feature points and the updated state to the network.

Notice that state vector $\mathbf{x}$ in (12) retains all information for determining the robot state in the future, therefore being a valid state. For example, the i-th LiDAR pose can be determined from $\mathbf{x}$ as follows:

$$\mathbf{T}_{k+1}^{i} = \mathbf{T}\left(\mathbf{x}_{k}^{c}\right)\mathbf{T}\left(\mathbf{x}_{k}^{ei}\right) \tag{18}$$

E. Measurement model

Our decentralized EKF runs a LiDAR odometry and mapping (LOAM) algorithm [12] for each LiDAR on its dedicated computing devices, usually an onboard computer with modulate computing performance or a virtual computation resource allocated from a high performance server. Taking the i-th LiDAR as an example, the LOAM solves the i-th LiDAR pose at time t_{k+1} (i.e., $\mathbf{T}_{k+1}^{i}$) by minimizing the distance of edge features $\mathbf{r}_{p2e}$ and plane features $\mathbf{r}_{p2p}$ between the current scan and a local map

$$\min_{\mathbf{T}_{k+1}^{i}\in SE(3)} \left(\sum \mathbf{r}_{p2e} + \sum \mathbf{r}_{p2p}\right) \tag{19}$$

To accelerate the optimization process, the predicted i-th LiDAR pose $\bar{\mathbf{T}}_{k+1}^{i}$ from *Section. V-A* is usually used as the initial estimate, and the error pose from which is solved. i.e.,

$$\mathbf{T}_{k+1}^{i} = \bar{\mathbf{T}}_{k+1}^{i}\mathbf{T}(\delta\mathbf{x}_{k+1}^{i}) \tag{20}$$

Substituting this into (19) leads to

$$\delta\hat{\mathbf{x}}_{k+1}^{i} = \underset{\delta\mathbf{x}_{k+1}^{i}\in\mathbb{R}^{6}}{\arg\min} \left(\sum \mathbf{r}_{p2e} + \sum \mathbf{r}_{p2p}\right) \tag{21}$$

Assume the Hessian matrix of (21) at convergence is $\hat{\mathbf{\Sigma}}_{\delta}^{-1}$, then $\hat{\mathbf{\Sigma}}_{\delta}$ is the covariance matrix associated to the measurement $\delta\hat{\mathbf{x}}_{k+1}^{i}$. That is to say, the measurement model is

$$\delta\hat{\mathbf{x}}_{k+1}^{i} = \delta\mathbf{x}_{k+1}^{i} + \mathbf{v} \tag{22}$$

where $\mathbf{v}\sim\mathcal{N}(\mathbf{0},\hat{\mathbf{\Sigma}}_{\delta})$ and $\delta\mathbf{x}_{k+1}^{i}$ is solved from (20)

$$\delta\mathbf{x}_{k+1}^{i} = \mathbf{x}\left(\left(\bar{\mathbf{T}}_{k+1}^{i}\right)^{-1}\mathbf{T}\left(\mathbf{x}_{k}^{c}\right)\mathbf{T}\left(\mathbf{x}_{k}^{ei}\right)\right) \tag{23}$$

notice that $\delta\bar{\mathbf{x}}_{k+1}^{i} = \mathbf{0}$.

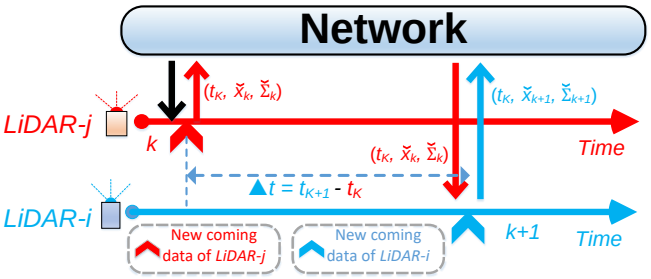

Fig. 5: One step update of our decentralized EKF algorithm: once receiving a point cloud scan at time t_{k+1}, the i-th LiDAR retrieves the newest state update $(\check{\mathbf{x}}_k, \check{\mathbf{\Sigma}}_k)$, which was received from the network at its local time t_k. Then it uses its scan to update the state, and advertise the updated state $(\check{\mathbf{x}}_k, \check{\mathbf{\Sigma}}_{k+1})$ to the network.

V. Decentralized Extended Kalman Filter

In this section, we introduce our decentralized EKF algorithm. Unlike existing EKF algorithm which often runs a single instance on a central computer, our system runs multiple EKF instances in parallel, one per LiDAR. Individual instance usually runs on the respective LiDAR dedicated computing resources and is responsible for processing that LiDAR data. As shown in Fig. 4, each EKF reads the full state vector $\mathbf{x} = \begin{bmatrix}\mathbf{x}^{c} & \mathbf{x}^{e1} & \mathbf{x}^{e2} & \cdots & \mathbf{x}^{eN}\end{bmatrix}^{T}$ from the network, updates it by registering the respective LiDAR data, and publishes the updated state to the network for other EKF instances to use. In the following, we explain in detail how the full state is updated for each LiDAR (e.g., i-th LiDAR).

A. State prediction

As shown in Fig. 5, assume the i-th LiDAR obtains a scan at time t_{k+1}. Moreover, assume the most recent state update $\check{\mathbf{x}}_k$ (and associated covariance $\check{\mathbf{\Sigma}}_k$) was published by LiDAR j (j could be equal to i). The $\check{\mathbf{x}}_k$ and $\check{\mathbf{\Sigma}}_k$ were received by the LiDAR i at its local time t_k ($t_k < t_{k+1}$).

Refer to the Section IV-D, we have:

$$\check{\mathbf{x}}_k = \begin{bmatrix}\check{\mathbf{x}}_k^{c} & \check{\mathbf{x}}_k^{e1} & \check{\mathbf{x}}_k^{e2} & \cdots & \check{\mathbf{x}}_k^{eN}\end{bmatrix}^{T} \tag{24}$$

Then, follow the standard EKF prediction, we have the predicted full state vector:

$$\bar{\mathbf{x}}_{k+1} = \begin{bmatrix}\bar{\mathbf{x}}_{k+1}^{c} & \bar{\mathbf{x}}_{k+1}^{e1} & \bar{\mathbf{x}}_{k+1}^{e2} & \cdots & \bar{\mathbf{x}}_{k+1}^{eN}\end{bmatrix}^{T} \tag{25}$$

computed as below:

$$\bar{\mathbf{x}}_{k+1}^{c} = \mathbf{f}\left(\check{\mathbf{x}}_k^{c}, \mathbf{0}; \Delta t\right) \tag{26}$$

$$\bar{\mathbf{x}}_{k+1}^{e1} = \check{\mathbf{x}}_k^{e1} \tag{27}$$

$$\vdots$$

$$\bar{\mathbf{x}}_{k+1}^{eN} = \check{\mathbf{x}}_k^{eN} \tag{28}$$

where $\Delta t = t_{k+1} - t_k$. The covariance matrix associated to the state prediction $\bar{\mathbf{x}}_{k+1}$ is

$$\bar{\mathbf{\Sigma}}_{k+1} = \mathbf{F}\check{\mathbf{\Sigma}}_k\mathbf{F}^{T} + \mathbf{G}\mathbf{\Sigma}_{\mathbf{w}}\mathbf{G}^{T} \tag{29}$$

where $\mathbf{F}\in\mathbb{R}^{(12+6N)\times(12+6N)}$ and $\mathbf{G}\in\mathbb{R}^{(12+6N)\times 6}$

$$\mathbf{F} = \begin{bmatrix}\dfrac{\partial\mathbf{f}\left(\mathbf{x}_k^{c}, \mathbf{0}; \Delta t\right)}{\partial\mathbf{x}_k^{c}} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_{6N}\end{bmatrix}, \quad \mathbf{G} = \frac{\partial\mathbf{f}\left(\check{\mathbf{x}}_k^{c}, \mathbf{w}; \Delta t\right)}{\partial\mathbf{w}} \tag{30}$$

Then we can predict the pose of i-th LiDAR pose $\bar{\mathbf{T}}^i_{k+1}$ at time t_{k+1}:

$$\bar{\mathbf{T}}^i_{k+1} = \mathbf{T}\left(\bar{\mathbf{x}}^c_{k+1}\right)\mathbf{T}\left(\bar{\mathbf{x}}^{ei}_{k+1}\right) \tag{31}$$

which is used as the initial estimate of $\mathbf{T}^i_{k+1}$ used in the LOAM as explained in *Section. IV-E.*

B. Measurement update

A problem with the state equation (26∼28) is that it involves N extrinsic parameters. With measurements $\delta\widehat{\mathbf{x}}^i_{k+1}$ from the point registration in (21), the system is not observable (nor detectable), causing the EKF to diverge. This problem is usually resolved by fixing the reference frame at any one of the N LiDARs, removing the extrinsic estimation of that LiDAR. However, when the reference LiDAR fails, the rests have to agree on another reference LiDAR, which is usually a complicated process.

To avoid this, we choose the reference frame at the center of all LiDARs. i.e.,

$$\mathbf{t}^c_k = \frac{1}{N}\sum_{i=1}^{N}\mathbf{t}^i_k;\ \forall k = 0, 1, 2, \cdots \tag{32}$$

Substituting in (11) leads to

$$\sum_{i=1}^{N}\mathbf{t}^{ei} = \mathbf{0} \tag{33}$$

Besides the location, the attitude of the reference frame $\mathbf{R}^c_k$ are defined such that

$$\sum_{i=1}^{N}\mathbf{r}\left(\left(\mathbf{R}^c_k\right)^T\mathbf{R}^i_k\right) = \mathbf{0} \tag{34}$$

which leads to

$$\sum_{i=1}^{N}\mathbf{r}^{ei} = \mathbf{0} \tag{35}$$

As a result, in addition to the measurement $\delta\widehat{\mathbf{x}}^i_{k+1}$ in (22), two new measurements of $\mathbf{0}_{3\times1}$ should be added. The total measurement vector is

$$\mathbf{y}_m = \left[\delta\widehat{\mathbf{x}}^i_{k+1}, \mathbf{0}, \mathbf{0}\right]^T \tag{36}$$

and the respective output functions are

$$\mathbf{r}_{re} = \sum_{i=1}^{N}\mathbf{r}^{ei},\ \ \mathbf{t}_{re} = \sum_{i=1}^{N}\mathbf{t}^{ei} \tag{37}$$

Then we have the residual vector $\mathbf{z}$ and the associated covariance $\mathbf{\Sigma}_z$ are

$$\mathbf{z} = \begin{bmatrix}\delta\widehat{\mathbf{x}}^i_{k+1} & -\bar{\mathbf{r}}_{re} & -\bar{\mathbf{t}}_{re}\end{bmatrix} \in \mathbb{R}^{12} \tag{38}$$

$$\mathbf{\Sigma}_z = \begin{bmatrix}\hat{\mathbf{\Sigma}}_\delta & \mathbf{0}\\ \mathbf{0} & s\mathbf{I}\end{bmatrix} \in \mathbb{R}^{12\times12} \tag{39}$$

where s is a small value to prevent the EKF from being singular (set as 1 in our work), $\bar{\mathbf{r}}_{re}$ and $\bar{\mathbf{t}}_{re}$ are the sum of predicted extrinsic rotation and translation in (37), respectively.

As a result, the Kalman gain is

$$\mathbf{K} = \bar{\mathbf{\Sigma}}_{k+1}\mathbf{H}^T\left(\mathbf{H}\bar{\mathbf{\Sigma}}_{k+1}\mathbf{H}^T + \mathbf{\Sigma}_z\right)^{-1} \in \mathbb{R}^{(12+6N)\times12} \tag{40}$$

with $\mathbf{H} \in \mathbb{R}^{12\times(12+6N)}$ being

$$\mathbf{H} = \begin{bmatrix} \frac{\partial(\delta\mathbf{x}^i_{k+1})}{\partial\mathbf{x}^c_k} & \mathbf{0} & \cdots & \frac{\partial(\delta\mathbf{x}^i_{k+1})}{\partial\mathbf{x}^{ei}} & \cdots & \mathbf{0} \\ \mathbf{0} & \frac{\partial\mathbf{r}_{re}}{\partial\mathbf{x}^{e1}} & \cdots & \frac{\partial\mathbf{r}_{re}}{\partial\mathbf{x}^{ei}} & \cdots & \frac{\partial\mathbf{r}_{re}}{\partial\mathbf{x}^{eN}} \\ \mathbf{0} & \frac{\partial\mathbf{t}_{re}}{\partial\mathbf{x}^{e1}} & \cdots & \frac{\partial\mathbf{t}_{re}}{\partial\mathbf{x}^{ei}} & \cdots & \frac{\partial\mathbf{t}_{re}}{\partial\mathbf{x}^{eN}} \end{bmatrix} \tag{41}$$

Finally, we have the measurement update as follows:

$$\check{\mathbf{x}}_{k+1} = \bar{\mathbf{x}}_{k+1} + \mathbf{K}\mathbf{z} \tag{42}$$

$$\check{\mathbf{\Sigma}}_{k+1} = \left(\mathbf{I} - \mathbf{K}\mathbf{H}\right)\bar{\mathbf{\Sigma}}_{k+1} \tag{43}$$

The updated full state is then advertised to the network for other EKF instances to use.

C. Algorithm of decentralized calibration, localization, and mapping with multiple LiDARs

To sum up, we conclude the previous EKF formulation as the algorithm shown below:

Algorithm 1: Decentralized calibration, localization and mapping on the i-th LiDARs

Input : $\check{\mathbf{x}}_k$, $\check{\mathbf{\Sigma}}_k$ received from the network at time t_k; Current point cloud of i-th LiDAR received at time t_{k+1}.

Output : Advertise the updated state $\check{\mathbf{x}}_{k+1}$ and its associated covariance matrix $\check{\mathbf{\Sigma}}_{k+1}$ to the network.

Prediction:

Get $\bar{\mathbf{x}}_{k+1}$ from (25).
Get $\bar{\mathbf{\Sigma}}_{k+1}$ from (29).
Compute $\bar{\mathbf{T}}^i_{k+1}$ from (31).

Update :

Solve $\delta\widehat{\mathbf{x}}^i_{k+1}$ and $\hat{\mathbf{\Sigma}}_\delta$ from (21).
Get the Kalman gain $\mathbf{K}$ from (40).
Update $\check{\mathbf{x}}_{k+1}$ from (42).
Update $\check{\mathbf{\Sigma}}_{k+1}$ from (43).

Return : $\check{\mathbf{x}}_{k+1}$, $\check{\mathbf{\Sigma}}_{k+1}$

D. Initialization

1) Hand-eye calibration: To provide the well initialized extrinsic, we implement the hand-eye calibration algorithm introduced in [26]. However, due the damper ball and the 3D-printed modules, the rigid connection is not guaranteed. We do not assume the extrinsic result are well calibrated every time, however, we believe it is suitable to serve as the initial guess at the beginning of EKF and map alignment.

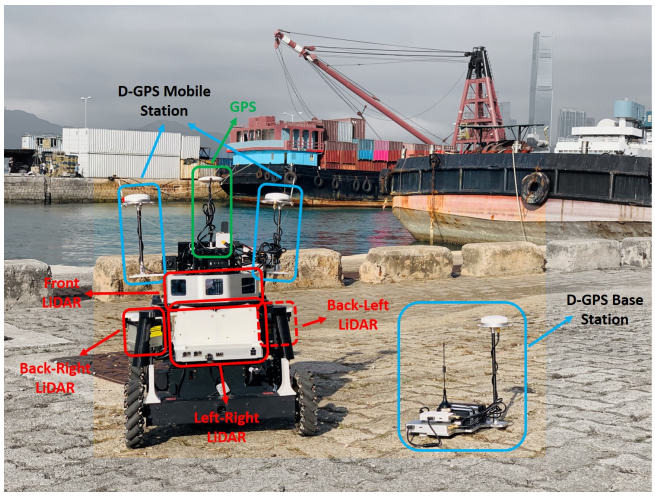

Fig. 6: Our remotely operated vehicle platform consisting 5 LiDARs, onboard mini-computer, D-GPS mobile station and monocular camera (for pilot preview only).

2) Map alignment: Map alignment can not only provide the initial estimation of extrinsic among LiDARs, but also can align the different coordinates of LiDARs, making the odomety of each LiDARs to the same reference frame.

Since there is no overlapping area among any two LiDARs, we can not directly find out the relative transformation between any two LiDARs. In our work, in the stage of initialization, each LiDAR node performs LOAM at their own frame coordinates, meanwhile, subscribes to the point cloud data published by others. Since the platform is moving, the mappings of LiDARs will have overlaps with others. Once the overlapping area is sufficient, the ICP algorithm is performed and we can align both the map and coordinate frames of each LiDAR.

VI. Experiments

Our custom-built robotic platform is shown in the Fig. 6, with a Differential Global Positioning System (D-GPS) mobile station mounted on the top, which is used to provide a high precision odometry reference to evaluate our algorithm. We implement our decentralized framework on a high-performance PC, which is embedded with Intel i7-9700K processor and 32GB RAM. Similar to a real distributed system, in our implementation each LiDAR EKF algorithm runs in an individual ROS node by publishing and subscribing messages from each other.

We ran our vehicle platform at a harbour area with relative constant speed, good GPS signal and some moving pedestrians. The satellite image of our test ground is shown in Fig. 11.B. Two trajectories, *Scene-1* and *Scene-2*, were recorded taking about $400s$ and $320s$ respectively. *Scene-1* is a one-way trajectory while in *Scene-2*, we chose to walk in relative straight lines and returned to an end point close to where we began, as shown in Fig. 8.

VII. Results

The 5 EKF copies maintain the same state vector and update it at different time (once receive the respective LiDAR data). In the following results, we collect each state estimate across all 5 EKFs and analyze its convergence over time as well as accuracy if ground truth is available (e.g., position).

A. Result of online extrinsic calibration

The extrinsic estimation of all 5 LiDARs with respect to the geometric center are plotted in Fig. 7. The left and right columns depict the extrinsic of rotation in Euler angle and the translation in meter, respectively. We test our algorithm with two sets of initial values: the first one is obtained from the result of map alignment (solid line) and the second one is directly set to 0 (dashed line). As shown in Fig. 7, both the extrinsic of rotation and translation converge to a stable value quickly, which demonstrates the ability of our algorithm to calibrate the extrinsic, even with inaccurate initial values. The converged extrinsic values also agrees with visual inspection of the location of each LiDAR.

B. Result of state estimation

Besides the extrinsic parameters, we further present the state estimation of the vehicle geometric center. In *Scene-1*, the pose and velocity estimation of the geometry are shown in Fig. 9, we can see that the estimation of velocity can reflect the change of poses very well. Taking the position of x-axis and its corresponding linear velocity for example, in the time interval $[18.9s, 300s]$, the value of *Pos_x* increases from $16m$ to about $400m$, with constant velocity around $1.36m/s$, which matches with the estimated velocity as shown in Fig. 9. The results of *Scene-2* are similar and not presented here due to space limit.

C. Evaluation of localization accuracy

While the proposed method converges qualitatively, in this sub-section, we perform quantitative evaluation on the localization accuracy of our algorithm by comparing with a differential Global Positioning System (D-GPS)[6], which can provide the localization reference with the precision of $1cm + 1ppm$. We evaluate our algorithm with different numbers of LiDARs on both of the two scenes. The comparison of different trajectories are shown in Fig. 8, where the trajectory of single front LiDAR follows the Ground-Truth well at first but fails in long run due to the lack of sufficient features within the small FoV.

Table II shows the maximum *absolute / relative error* among different configurations, showing that multiple LiDARs have great impact on improving the accuracy of localization. In addition, we plot the absolute error over distance in Fig. 10 for detailed reference.

In both scenes, we have achieved the precision of about 0.2%, which demonstrates that our algorithm is of high-accuracy.

D. Result of mapping

In the *Scene-1*, the maps we built are shown in Fig. 11, with the point cloud data sampled from different LiDARs being rendered with different colors. From both the bird's

[6]https://www.dji.com/d-rtk

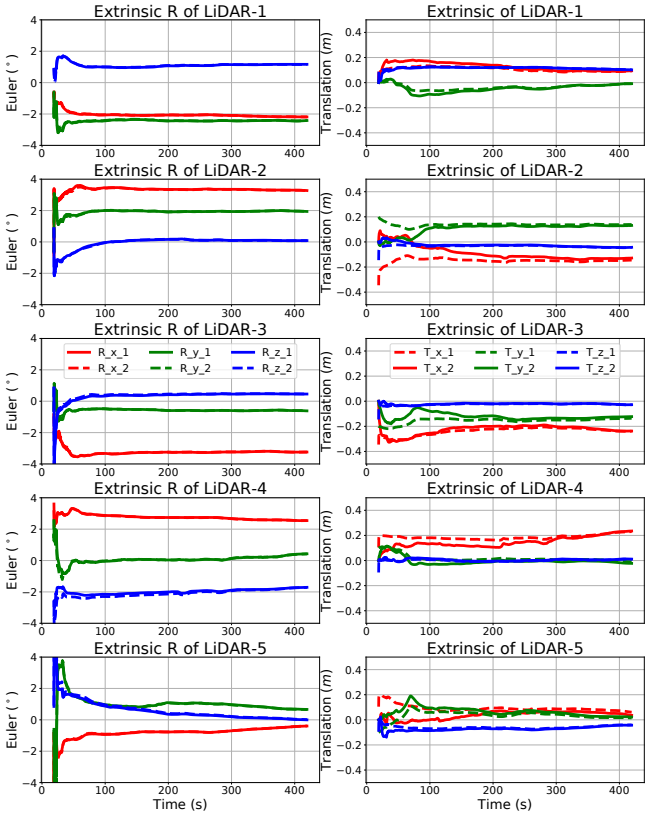

Fig. 7: The updates of extrinsic parameters (left: rotation, right: translation) among 5 LiDARs with respect to the geometric center. The solid line starts with the initial guess calculated from map-alignment. The dashed line starts with zero initial values.

eye-view (Fig. 11.A) and detailed view (Fig. 11.(C-E)), we can see that the point cloud data from different LiDARs is aligned well together and the consistency is kept both locally and globally. In summary, we have examined and verified the convergence and precision of the proposed algorithm on actual system with real world data.

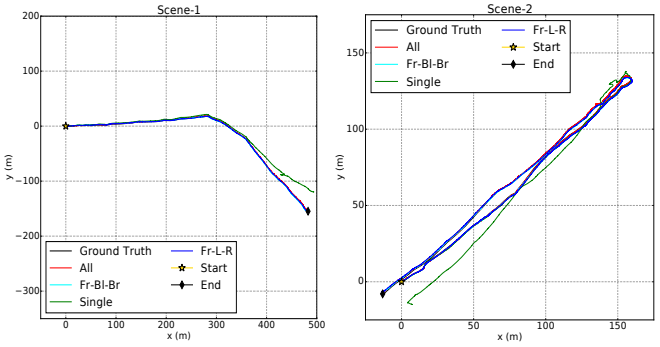

Fig. 8: The comparison of trajectories generated (from the view of bird's eye, since the trajectories are very close with others, we strongly recommend the readers to zoom in vector graph for further details) from D-GPS and our algorithm with different LiDAR configurations. Where, *GT* is the trajectory from D-GPS, *All* is with all LiDAR enabled, *Fr-Bl-Br* is the configuration with the front, back-left, back-right LiDARs enabled, *Fr-L-R* is the configuration with the front, left, right LiDARs enabled, *Single* is the configuration with only the front LiDAR enabled.

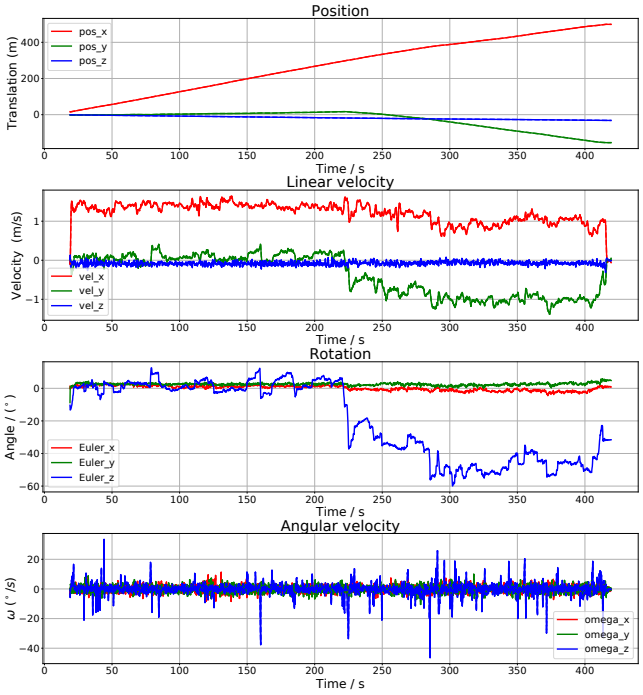

Fig. 9: The estimation of position, linear velocity, rotation and angular velocity of geometric center in *Scene-1*, with the configuration of *All* LiDARs being enabled. The data plot starts after the map alignments, which is at $t = 18.9s$.

	All	*Fr-Bl-Br*	*Fr-L-R*	Single
	Max (m / %)	Max (m / %)	Max (m / %)	Max (m / %)
Scene-1	**1.17 / 0.21%**	1.99 / 0.36%	1.29 / 0.24%	35.16 / 6.38%
Scene-2	**0.88 / 0.20%**	1.19 / 0.27%	1.33 / 0.31%	14.90 / 3.41%

TABLE II: The maximum absolute error (m) and relative error (%) among different LiDAR configurations of *Scene-1* and *Scene-2*, whose total length are *551.45m* and *436.47m*, respectively.

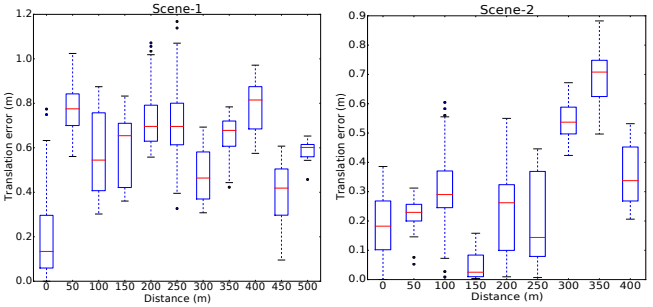

Fig. 10: The absolute translation error over time, with the configuration of all LiDAR being enabled (our best accuracy).

VIII. Conclusion and Discussion

This paper presents a decentralized EKF algorithm for simultaneous calibration, localization, and mapping with multiple LiDARs. Experiments in urban area are conducted. Results show that the proposed algorithm converges stably and has achieved 0.2% accuracy at low speed motion.

As mentioned in our previous Section VI, our current implementation is based on a single high-performance PC where all communication are done within a PC, problems such as message synchronization or communication loss do not occur and are not considered. Moreover, limited by the computing power of the PC, the current implementation runs offline. Future work will implement on each LiDAR

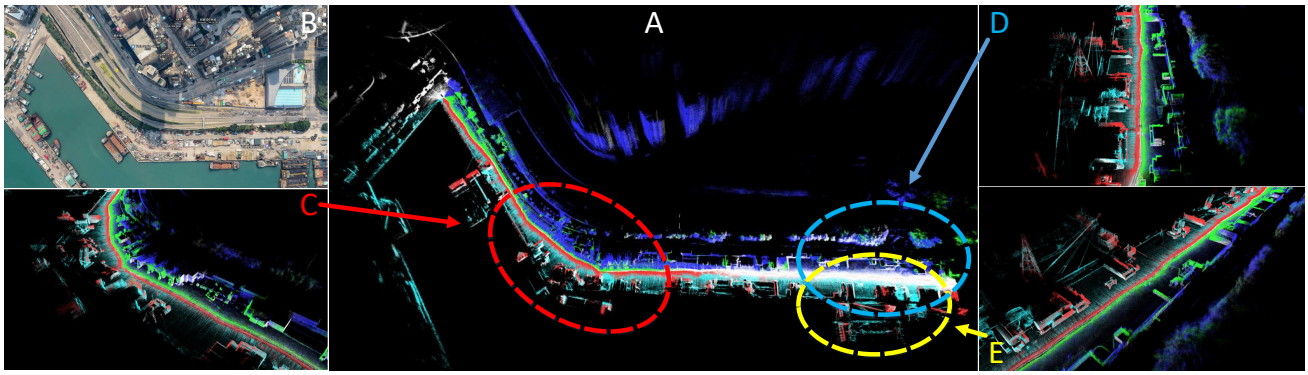

Fig. 11: A): The bird's eye-view of the map we reconstruct with the data collected in *Scene-1*. The point cloud data sampled from different LiDARs are rendered with different colors. The points of white, red, deep blue, cyan and green are the data sampled by the LiDAR installed on font, left, right, back-left and back-right, respectively; B): The satellite image of the experiment test ground; C)∼E): The detailed inspection of the area marked in dashed circle in A.

dedicated computer, solve the problem therein (e.g., time synchronization) and verify its robustness in presence of LiDAR failure.

IX. Acknowledgement

The authors would like to thank DJI Co., Ltd[7]. for donating devices and research fund.

References

[1] M. Dissanayake, P. Newman, S. Clark, H. Durrant-Whyte, and M. Csorba, "A solution to the simultaneous localization and map building (slam) problem," *IEEE Transactions on Robotics and Automation*, vol. 17, no. 3, p. 229–241, 2001.

[2] P. Farina, D. Colombo, A. Fumagalli, F. Marks, and S. Moretti, "Permanent scatterers for landslide investigations: outcomes from the esa-slam project," *Engineering Geology*, vol. 88, no. 3-4, p. 200–217, 2006.

[3] J. Engel, T. Schöps, and D. Cremers, "Lsd-slam: Large-scale direct monocular slam," *Computer Vision – ECCV 2014 Lecture Notes in Computer Science*, p. 834–849, 2014.

[4] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "Orb-slam: A versatile and accurate monocular slam system," *IEEE Transactions on Robotics*, vol. 31, no. 5, p. 1147–1163, 2015.

[5] C. Kerl, J. Sturm, and D. Cremers, "Dense visual slam for rgb-d cameras," *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013.

[6] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2d lidar slam," *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016.

[7] M. Pierzchala, P. Giguere, and R. Astrup, "Mapping forests using an unmanned ground vehicle with 3d lidar and graph-slam," *Computers and Electronics in Agriculture*, vol. 145, p. 217–225, 2018.

[8] J. Zhang and S. Singh, "Loam: Lidar odometry and mapping in real-time," *Robotics: Science and Systems X*, Dec 2014.

[9] T. Taketomi, H. Uchiyama, and S. Ikeda, "Visual slam algorithms: a survey from 2010 to 2016," *IPSJ Transactions on Computer Vision and Applications*, vol. 9, no. 1, Feb 2017.

[10] F. Gao, W. Wu, W. Gao, and S. Shen, "Flying on point clouds: Online trajectory generation and autonomous navigation for quadrotors in cluttered environments," *Journal of Field Robotics*, vol. 36, no. 4, pp. 710–733, 2019.

[11] Z. Xuexi, L. Guokun, F. Genping, X. Dongliang, and L. Shiliu, "Slam algorithm analysis of mobile robot based on lidar," *2019 Chinese Control Conference (CCC)*, 2019.

[12] J. Lin and F. Zhang, "Loam livox: A fast, robust, high-precision lidar odometry and mapping package for lidars of small fov," in *Proc. of The International Conference in Robotics and Automation (ICRA)*, 2020.

[13] M. Bosse, R. Zlot, and P. Flick, "Zebedee: Design of a spring-mounted 3-d range sensor with application to mobile mapping," *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1104–1119, Oct. 2012. [Online]. Available: https://doi.org/10.1109/tro.2012.2200990

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

[15] M. Sualeh and G.-W. Kim, "Dynamic multi-lidar based multiple object detection and tracking," *Sensors*, vol. 19, no. 6, p. 1474, 2019.

[16] S. Zeng, "A tracking system of multiple lidar sensors using scan point matching," *IEEE Transactions on Vehicular Technology*, vol. 62, no. 6, p. 2413–2420, 2013.

[17] J. Huang, M. Demir, T. Lian, and K. Fujimura, "An online multi-lidar dynamic occupancy mapping method," *2019 IEEE Intelligent Vehicles Symposium (IV)*, 2019.

[18] J. F. Newman, T. A. Bonin, P. M. Klein, S. Wharton, and R. K. Newsom, "Testing and validation of multi-lidar scanning strategies for wind energy applications," *Wind Energy*, vol. 19, no. 12, p. 2239–2254, 2016.

[19] F. Kopp, I. Smalikho, S. Rahm, A. Dolfi, J.-P. Cariou, M. Harris, R. I. Young, K. Weekes, and N. Gordon, "Characterization of aircraft wake vortices by multiple-lidar triangulation," *AIAA Journal*, vol. 41, no. 6, p. 1081–1088, 2003.

[20] S. Lynen, M. W. Achtelik, S. Weiss, M. Chli, and R. Siegwart, "A robust and modular multi-sensor fusion approach applied to mav navigation," *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013.

[21] S. Leutenegger, P. Furgale, V. Rabaud, M. Chli, K. Konolige, and R. Siegwart, "Keyframe-based visual-inertial slam using nonlinear optimization," *Robotics: Science and Systems IX*, 2013.

[22] N. Muhammad and S. Lacroix, "Calibration of a rotating multi-beam lidar," *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2010.

[23] Z. Taylor and J. Nieto, "Motion-based calibration of multimodal sensor extrinsics and timing offset estimation," *IEEE Transactions on Robotics*, vol. 32, no. 5, p. 1215–1229, 2016.

[24] J. Levinson and S. Thrun, "Unsupervised calibration for multi-beam lasers," *Experimental Robotics Springer Tracts in Advanced Robotics*, p. 179–193, 2014.

[25] A. J. Davison, "Real-time simultaneous localisation and mapping with a single camera," in *null*. IEEE, 2003, p. 1403.

[26] J. Jiao, Y. Yu, Q. Liao, H. Ye, and M. Liu, "Automatic calibration of multiple 3d lidars in outdoor environment," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019.

[7] https://www.dji.com