Versatile 3D Multi-Sensor Fusion for Lightweight 2D Localization

Patrick Geneva,* Nathaniel Merrill,* Yulin Yang, Chuchu Chen, Woosik Lee, and Guoquan Huang

Abstract— **Aiming for a lightweight and robust localization solution for low-cost, low-power autonomous robot platforms, such as educational, or industrial ground vehicle, under challenging conditions (e.g., poor sensor calibration, low lighting and dynamic objects), we propose a 2-stage localization system which incorporates both offline prior map building and online multi-modal localization. In particular, we develop an occupancy grid mapping system with probabilistic odometry fusion, accurate scan-to-submap covariance modeling, and accelerated loop-closure detection, which is further aided by 2D line features that exploit the environmental structural constraints. We also develop an versatile EKF-based online 3D localization system which can optimally fuse multi-modal information provided, e.g., by the pre-built occupancy grid map, and the IMU, odometry and 2D LiDAR measurements with low computational requirements. In addition, online spatial-temporal calibration between these sensors are also estimated to account for poor initial guess and "plug-and-play" application, which both improves the accuracy and flexibility of the proposed multi-sensor fusion framework. Our mapping system is qualitatively compared to the state-of-the-art Google Cartographer, then, extensive Monte-Carlo simulations are performed to verify both accuracy, consistency and efficiency of the proposed map-based localization system with full spatial-temporal calibration. We also validate the complete system (prior map building and online localization) with building-scale real world datasets.**

I. INTRODUCTION

Enabling versatile, lightweight, robust and centimeter-accuracy localization for indoor ground robots holds potentially huge implications for the practical development of autonomous systems. Within the service, educational, and commercial sectors, ground vehicles are a fundamental transportation platform which enable higher level tasks (e.g. package delivery, inspection, or environmental mapping). Accurate localization is crucial to robotic autonomy, but is limited by both the sensor payload limit, cost, and computational requirements for processing sensor data and multi-sensor fusion for state estimation. Hence, an efficient localization system which fuses information of multiple modalities (e.g., inertial, odometry, range, camera, ultra-wide band) has been a research focus over the past decade [1].

A particular application that has attracted large amounts of attention due to high accuracy requirements in challenging large scale dynamic environments is warehouse robotic localization [2]. Light detection and ranging (LiDAR)-based localization systems have become a focus of indoor ground robots due to LiDAR's robustness to external factors, such as poor lighting conditions, complementary use in safety systems that prevent collisions with the surrounding structure, and simplicity of the collected measurements. However these LiDAR only systems suffer from degenerate cases, such as long corridors – requiring additional sensing information to be fused to overcome these cases and increase localization accuracy. Additionally, the lower signal-to-noise ratios of low-end sensors requires the careful modeling of sensor errors and the fusion of multiple noisy sources to reduce the overall state uncertainty to acceptable levels. Thus, in this work we focus on fusing multiple low-cost multi-modal sensors to provide accurate localization while running on computationally limited devices.

To that end, we propose a two part system: (1) an offline 2D mapping algorithm which builds an occupancy grid and sparse line feature map in a tightly coupled nonlinear optimization framework (see Sec. III); (2) a versatile efficient filter which leverages inertial information to handle non-2D irregularities such as bumps and fuses odometry and LiDAR information and leverages the pre-built map to limit the drift of localization, while refining all sensor calibration parameters for improved robustness (see Sec. IV). In particular, we design a complete localization system encompassing both mapping and online localization functionalities:

- Inspired by Cartographer [3], we propose a submap-based occupancy grid mapping system, but accurately model the scan-to-submap covariance and perform probabilistic fusion with odometry readings. We additionally exploit the environmental structure and extract lines which provide geometric constraints between poses. Finally, we accelerate the loop closure detection by leveraging the estimate state uncertainty for limiting the scan matching search window.
- We propose a 3D extended Kalman filter (EKF)-based online localization system which optimally fuses inertial, odometry, and 2D LiDAR sensors for accurate online state estimation. The generated prior map is leveraged to bound the drift of localization without the computational burden of simultaneously building it alongside the state estimation. We additionally handle inaccurate sensor spatial and temporal calibrations through online estimation of these parameters and allowing for "plug and play" robots with hand-measured calibration guesses.
- Extensive simulated evaluation of the localization system is performed with analysis of different sensor configurations and the convergence of calibration parameters. Additionally the impact of update frequency of prior map constraints is investigated.

* These authors contributed equally to this work.

This work was partially supported by the University of Delaware (UD) College of Engineering, the NSF (IIS-1924897), the ARL (W911NF-19-2-0226, JWS 10-051-003), and Geek+. Yang was partially supported by the University Doctoral Fellowship and Geneva by the Delaware Space Grant College and Fellowship Program (NASA Grant NNX15AI19H).

The authors are with the Robot Perception and Navigation Group (RPNG), University of Delaware, Newark, DE 19716, USA. {pgeneva, nmerrill, yuyang, ccchu, woosik, ghuang}@udel.edu

- The system as a whole is evaluated on a real-world dataset where we are able to localize within the prior map. We evaluate the time consumption of the proposed localization algorithm and quality of generated map.

II. RELATED WORKS

2D LiDAR-based localization has received significant attentions over the past years [4], and its solutions can be approximately categorized into two major families: particle-based [5] and graph-based [3], [6], [7]. The former including the well-known FastSLAM [8], GMapping [5], TinySLAM [9], [10], and VinySLAM [11], has been of particular interest due to the ability to run on low-power devices. However, their superior performance often comes at a higher computational cost due to a large number of particles needed.

Google Cartographer [3] introduced a more efficient scan-to-submap loop closure detection algorithm and optimized both scans in a local submap frame along with a global sparse pose graph, which included loop closure constraints; however, they do not explicitly model the covariance of their measurements, and instead use equal weighting. In this paper we present a few improvements and additions to this system to construct an offline map suitable for lightweight online localization. Specifically, we model of the uncertainty of the scan-to-submap matching problem and perform *weighted* least squares optimization in our mapping system.

As one of the closest work, HectorSLAM [12] combines 2D multi-level occupancy mapping alongside a 3D EKF which estimates the full 3D trajectory of the sensor through 2D LiDAR and IMU measurements. The 3D EKF propagates forward with inertial measurements and updates using covariance intersection of the optimized scan matching result from their 2D mapping module. As compared to this work, we look to perform online estimation of the extrinsic and time offset calibration between all sensors to facilitate the easy deployment to new robots. We additionally leverage a pre-computed prior map in our online localization which takes into account loop closures allowing for increased accuracy in large-scale environments without the extra cost of building it online.

III. 2D LINE AND OCCUPANCY GRID MAPPING

The proposed mapping system improves upon Cartographer [3] and its architecture is outlined in Fig. 1. At each timestep we optimize incoming scans to the current submap while background threads perform loop closure detection and optimization of the global pose graph which contains relative pose, loop closure, and line cost terms. The non-linear optimization problem is formulated and solved using the Ceres Solver [13], with the state vector $\mathbf{x}_{map}$ given by:

$$\mathbf{x}_{map} = \begin{bmatrix} \bar{\mathbf{x}}_1 & \cdots & \bar{\mathbf{x}}_k & \bar{\mathbf{x}}_{f_1} & \cdots & \bar{\mathbf{x}}_{f_\ell} \end{bmatrix} \tag{1}$$

where $\bar{\mathbf{x}}_i,\ i \in \{1 \dots k\}$ contains the 2D position ${}^G\bar{\mathbf{p}}_{L_i}$[1] and the yaw angle ${}^G\theta_{L_i}$ (${}^G\bar{\mathbf{R}}_{L_i}$ in matrix form) of the LiDAR in a global frame at time t_i and $\mathbf{x}_{f_j}, j \in \{1 \dots \ell\}$ is a 2D line feature, which will be explained in Sec. III-C.

[1]Note that throughout the paper, $\bar{(\cdot)}$ denotes either a 2D vector or 2×2 $SO(2)$ matrix, and in its absence refers to a full 3D position or 3×3 $SO(3)$ rotation matrix.

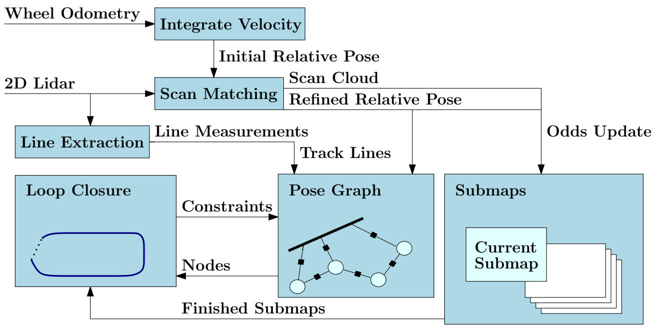

Fig. 1: We process integrated odometry measurements along with a new scan to obtain a single relative pose edge for the pose graph and insertion of the scan into the current submap. Extracted line features from the current scan are tracked to lines in the state vector and added to the global pose graph. Background threads detect loop closures between current scans and finished submaps using the correlative scan matcher, and merge lines after loop closure.

A. 2D Odometry Measurements

The 2D wheel odometry measurement provides local yaw angular velocity ${}^{O_\tau}\omega$ and x-direction linear velocity ${}^{O_\tau}v$. The readings at timestep τ can be described as:

$${}^{O_\tau}\omega_m = {}^{O_\tau}\omega + n_{\omega\tau}, \quad {}^{O_\tau}v_m = {}^{O_\tau}v + n_{v\tau} \tag{2}$$

where $n_{\omega\tau} \sim \mathcal{N}(0, \sigma_\omega^2)$, $n_{v\tau} \sim \mathcal{N}(0, \sigma_v^2)$. Hence, the relative pose measurement $\bar{\mathbf{z}}_{k-1,\tau+1}$ (with corresponding covariance $\bar{\mathbf{Q}}_{k-1,\tau+1}$) from odometry between t_{k-1} and $t_{\tau+1}$ (with $t_{k-1} \leq t_\tau \leq t_{\tau+1} \leq t_k$) can be integrated as:

$$\bar{\mathbf{z}}_{k-1,\tau+1} = \begin{bmatrix} {}^{O_{k-1}}\theta_{O_\tau} + {}^{O_\tau}\omega\delta t_\tau \\ {}^{O_{k-1}}\bar{\mathbf{p}}_{O_\tau} + \begin{bmatrix} \cos({}^{O_{k-1}}\theta_{O_\tau}) \\ \sin({}^{O_{k-1}}\theta_{O_\tau}) \end{bmatrix} {}^{O_\tau}v\delta t_\tau \end{bmatrix} \tag{3}$$

$$\bar{\mathbf{Q}}_{k-1,\tau+1} = \mathbf{H}_\tau \bar{\mathbf{Q}}_{k-1,\tau} \mathbf{H}_\tau^\top + \mathbf{G}_\tau \bar{\mathbf{Q}}_o \mathbf{G}_\tau^\top \tag{4}$$

where $\bar{\mathbf{Q}}_o = \mathbf{diag}\{\sigma_w^2, \sigma_v^2\}$. The Jacobians are omitted here for brevity. Iterating over Eq. (3) and (4) with all the odometry readings between t_{k-1} and t_k, we get the 2D relative pose measurements $\bar{\mathbf{z}}_{k-1,k}$ with covariance $\bar{\mathbf{Q}}_{k-1,k}$ and can be written w.r.t the state as:

$$\bar{\mathbf{z}}_{k-1,k} = \mathbf{h}_o\left(\bar{\mathbf{x}}_{k-1}, \bar{\mathbf{x}}_k\right) + \bar{\mathbf{n}}_{ok} \tag{5}$$

$$\mathbf{h}_o(\bar{\mathbf{x}}_{k-1}, \bar{\mathbf{x}}_k) = \begin{bmatrix} {}^G\theta_{O_k} - {}^G\theta_{O_{k-1}} \\ {}^O_L\bar{\mathbf{R}}{}^G_{L_{k-1}}\bar{\mathbf{R}}^\top \left({}^G\bar{\mathbf{p}}_{L_k} - {}^G\bar{\mathbf{p}}_{L_{k-1}} - {}^G_{L_k}\bar{\mathbf{R}}{}^O_L\bar{\mathbf{R}}^{\top O}\bar{\mathbf{p}}_L\right) + {}^O\bar{\mathbf{p}}_L \end{bmatrix}$$

where $\bar{\mathbf{n}}_{ok} \sim \mathcal{N}(\mathbf{0}, \bar{\mathbf{Q}}_{k-1,k})$, where ${}^O_L\bar{\mathbf{R}}$ and ${}^O\bar{\mathbf{p}}_L$ denotes the 2D rigid transformation between the LiDAR and odometry. The 2D relative pose odometry cost term can be defined as:

$$c_o = \|\bar{\mathbf{z}}_{k-1,k} - \mathbf{h}_o(\bar{\mathbf{x}}_{k-1}, \bar{\mathbf{x}}_k)\|^2_{\bar{\mathbf{Q}}^{-1}_{k-1,k}} \tag{6}$$

B. Occupancy Grid Map

We store occupancy grids in a local submap frame to allow for map corrections in the event of a loop closure. Each cell in the occupancy grid represents an $r \times r$ square of the world, where r is the chosen grid resolution. We can calculate the discrete cell index given a 2D vector ${}^S\bar{\mathbf{p}}$ in the submap frame:

$$\begin{bmatrix} x_d \\ y_d \end{bmatrix} = \frac{1}{r}{}^S\bar{\mathbf{p}} + \begin{bmatrix} x_o \\ y_o \end{bmatrix} - \frac{1}{2}\mathbf{1}_{2\times 1} \tag{7}$$

where $[x_o\ y_o]^\top$ is the cell index of the origin of the submap, which is initialized to the center of the grid cells, and moved as necessary when the grid matrix increases in size due to inserting scans that are out of its boundaries. A probability in the occupancy grid M of submap S at location ${}^{S}\bar{\mathbf{p}}$, using the indexing function in Eq. (7), is denoted as $M({}^{S}\bar{\mathbf{p}})$.

The submap occupancy grid, stores the probability that there is an object in the $r \times r$ square of the world and is initialized to a probability of 0.5. To update the map with a LiDAR scan, we use the registered pose of the LiDAR $\{{}^{S}_{L_i}\bar{\mathbf{R}}, {}^{S}\bar{\mathbf{p}}_{L_i}\}$ and trace along the ray between the current LiDAR position and the scan point ${}^{S}\bar{\mathbf{p}}_j$. For the end point ${}^{S}\bar{\mathbf{p}}_j$, we update the occupancy with a user-defined probability that a LiDAR range reading is a hit p_{hit}, and for all of the points in the rasterized line that lies along the ray we similarly update with a miss probability p_{miss}. For both hit and miss points, the probability update follows the form:

$$p_{new} = \mathbf{odds}^{-1}(\mathbf{odds}(p_{old})\mathbf{odds}(p_{update})) \tag{8}$$

where $\mathbf{odds}(p) = p/(1-p)$ and p_{update} takes either the hit or miss probability depending on the case. We set a threshold on the maximum number of scan insertions to a submap in order to keep them small and able to change the global map in a useful way.

1) Scan Matching: In order to accurately determine the relative pose between LiDAR scans, we register new scans to the current submap that contains some recently inserted scans as described above. Similarly to Google Cartographer [3], we use a nonlinear optimization to perform scan registration; however, unlike the Cartographer system, we consider the uncertainty of the scan points, initial guess, and the sampled submap. We perform the registration in a relative frame in order to avoid the reuse of information due to covariance propagation of the relative pose with the previous pose in the submap frame. The scan matching cost can be defined as:

$$c_{scan} = \left\| \begin{bmatrix} {}^{L_j}\bar{\mathbf{p}}_{L_k} - {}^{L_j}\bar{\mathbf{p}}_{L_k}^{(0)} \\ \mathrm{Log}({}^{L_j}_{L_k}\bar{\mathbf{R}}^{\top}\,{}^{L_j}_{L_k}\bar{\mathbf{R}}^{(0)}) \end{bmatrix} \right\|^2_{\mathbf{\Omega}_{init}} + \sum_{i=1}^{n} \frac{\left(1 - M({}^{S}\bar{\mathbf{x}}_{L_j}\,{}^{L_j}\bar{\mathbf{x}}_{L_k}, {}^{L_k}\bar{\mathbf{p}}_i)\right)^2}{\sigma_{si}^2}$$

where $\mathrm{Log}(\cdot)$ is the rotation matrix logarithm for special orthogonal group and ${}^{L_j}\bar{\mathbf{x}}_{L_k}$ denotes the relative pose between LiDAR frames $\{L_j\}$ and $\{L_k\}$. Note that ${}^{L_j}\bar{\mathbf{p}}_{L_k}^{(0)}$ and ${}^{L_j}_{L_k}\bar{\mathbf{R}}^{(0)}$ denote the initial guess of the relative pose, which can either come from integrating odometry measurements, described in Sec. III-A, or by performing a correlative scan match with an exhaustive search [14] or depth-first search [3]. Since we require the information of the initial guess, $\mathbf{\Omega}_{init}$, we use Olson's method [14] to compute the covariance for both types of correlative scan matchers. The cost function c_{scan} optimizes the relative pose ${}^{L_j}\bar{\mathbf{x}}_{L_k}$ between frames $\{L_j\}$ and $\{L_k\}$, which may not necessarily be consecutive depending on the application (i.e., scan matching for loop closure).

To probabilistically weight the scan cost. We compute the uncertainty of each occupancy score residual z_i, σ_{si}, by taking into account both the uncertainties of the scan and the submap.

$$z_i = 1 - M({}^{S}\bar{\mathbf{x}}_{L_j}\,{}^{L_j}\bar{\mathbf{x}}_{L_k}, {}^{L_k}\bar{\mathbf{p}}_i - \mathbf{n}_i) - n_{map} \tag{9}$$

$$\tilde{z}_i \simeq -\frac{\partial M}{\partial^{L_j}\tilde{\bar{\mathbf{x}}}_{L_k}}\,{}^{L_j}\tilde{\bar{\mathbf{x}}}_{L_k} - \frac{\partial M}{\partial \mathbf{n}_i}\mathbf{n}_i - n_{map} \tag{10}$$

where $\tilde{\bar{\mathbf{x}}}$ represents the error states of relative pose, $\mathbf{n}_i \sim \mathcal{N}(0, \mathbf{Q}_i)$ represents the LiDAR point measurement noise, and n_{map} denotes the scalar occupancy noise. Note that the map uncertainty σ_{map} is computed by sampling the current submap occupancy grid across the 4×4 grid used for the bicubic interpolation. Hence $\sigma_{si}^2 = \frac{\partial M}{\partial \mathbf{n}_i}\mathbf{Q}_i\left(\frac{\partial M}{\partial \mathbf{n}_i}\right)^{\top} + \sigma_{map}^2$. The information matrix of the relative pose of ${}^{L_j}\bar{\mathbf{x}}_{L_k}$ can be computed with the final Jacobian values as:

$$\mathbf{\Omega}_{j,k} = \mathbf{J}_i^{\top}\mathbf{J}_i + \mathbf{\Omega}_{init}, \quad \mathbf{J}_i = \frac{-1}{\sigma_{si}}\,\frac{\partial M}{\partial^{L_j}\tilde{\bar{\mathbf{x}}}_{L_k}} \tag{11}$$

Since the indexing function M for the map is not differentiable, we use bicubic interpolation with Hermite splines to achieve a smooth version. Denoting the scan point transformed into the submap frame by ${}^{S}\bar{\mathbf{p}}_i$, we apply a finite difference formula to calculate the necessary gradient $\frac{\partial M}{\partial^{S}\bar{\mathbf{p}}_i}$.

C. Line Map

In order to further exploit the geometrical constraints of structured indoor environments, line features can also be utilized to enhance the mapping. Inspired by our previous work [15], we propose to use 2D closest point (CP) to describe the line features. If we use $\mathbf{n}$ and d to denote the line normal direction and distance of line to origin, with the closest point representation , the transformation of a CP line from global to LiDAR frame can be written as:

$$\mathbf{x}_f = d \cdot \mathbf{n}, \quad \begin{bmatrix} {}^{L}\bar{\mathbf{n}} \\ {}^{L}d \end{bmatrix} = \begin{bmatrix} {}^{L}_{G}\bar{\mathbf{R}}^{\top} & \mathbf{0}_{2\times 1} \\ -{}^{G}\bar{\mathbf{p}}_L^{\top} & 1 \end{bmatrix} \begin{bmatrix} {}^{G}\bar{\mathbf{n}} \\ {}^{G}d \end{bmatrix} \tag{12}$$

Given a scan at time t_k, we extract lines from the point cloud using the method of Pfister et. al. [16]. We can express the CP line measurement $\bar{\mathbf{z}}_{f,k}$ as a function of the state and the cost function of LiDAR as follows:

$$\bar{\mathbf{z}}_{f,k} = \mathbf{h}_f(\bar{\mathbf{x}}_k, \bar{\mathbf{x}}_f) + \bar{\mathbf{n}}_{f,k} \tag{13}$$

$$c_f = \|\bar{\mathbf{z}}_{f,k} - \mathbf{h}_f(\bar{\mathbf{x}}_k, \bar{\mathbf{x}}_f)\|^2_{\mathbf{Q}_{f,k}^{-1}} \tag{14}$$

where $\bar{\mathbf{n}}_{f,k} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{f,k})$.

1) Line Tracking: As compared to using descriptor or χ^2-based line matching, we opt for a simpler and more efficient method which relies on thresholding the distances of tracked lines to new lines projected into the global frame. We consider two lines being a match if (1) the absolute difference of the two line distances is below a threshold, (2) the dot product of the two line normal vectors is above a threshold, and (3) if the minimum Euclidean distance between the two lines' end points is below a threshold. Otherwise, a new line is added to the state vector. We have found that this method is suitable to track lines over long periods of time.

D. Loop Closure

We perform two types of loop closure in our mapping system. The first kind is based on using a correlative scan matcher to match new scans to a finished submap (i.e., a submap that will receive no more scan insertions). Using the marginal covariance for both states that are being considered for a loop closure, $\bar{\mathbf{x}}_i$ and $\bar{\mathbf{x}}_j$, we calculate the search window for the correlative scan matcher as the 3σ bound of the current estimated relative pose and its covariance:

$${}^{i}\bar{\mathbf{x}}_j = \begin{bmatrix} {}^{G}\theta_j - {}^{G}\theta_i \\ {}^{G}_{i}\bar{\mathbf{R}}^{\top}({}^{G}\bar{\mathbf{p}}_j - {}^{G}\bar{\mathbf{p}}_i) \end{bmatrix}, \quad \mathbf{P}_{ij} = \begin{bmatrix} \mathbf{H}_i & \mathbf{H}_j \end{bmatrix} \begin{bmatrix} \mathbf{P}_{ii} & \mathbf{P}_{ij} \\ \mathbf{P}_{ij}^{\top} & \mathbf{P}_{jj} \end{bmatrix} \begin{bmatrix} \mathbf{H}_i^{\top} \\ \mathbf{H}_j^{\top} \end{bmatrix}$$

$$\mathbf{H}_j = \begin{bmatrix} 1 & \mathbf{0}_{1\times 2} \\ \mathbf{0}_2 & {}^{G}_{i}\bar{\mathbf{R}}^{\top} \end{bmatrix}, \qquad \mathbf{H}_i = \begin{bmatrix} -1 & \mathbf{0}_{1\times 2} \\ \mathbf{J}\,{}^{G}_{i}\bar{\mathbf{R}}^{\top}({}^{G}\bar{\mathbf{p}}_j - {}^{G}\bar{\mathbf{p}}_i) & -{}^{G}_{i}\bar{\mathbf{R}}^{\top} \end{bmatrix}$$

where $\mathbf{J} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$. $\mathbf{P}_{ii}$, $\mathbf{P}_{ij}$ and $\mathbf{P}_{ii}$ are covariances and correlations of state $\bar{\mathbf{x}}_i$ and $\bar{\mathbf{x}}_j$. By limiting the search window for scan matching, we are able to close more loops faster as compared to using a fixed search window and avoid invalid loop closures. When the search window is large, we use the depth-first search scan matcher proposed by [3], and when it is small we use an exhaustive search – which will be faster for smaller bounds due to the overhead of pre-computing the upper bounds for the depth-first search. Note that for large loops, the public implementation of Cartographer will search the entire submap – the entire length and width of the submap and in every possible orientation. This can be quite slow, even with the efficient depth-first search, and can lead to incorrect loop constraints in symmetric areas which our proposed method avoids. To select which pose nodes we wish to perform scan registration to, we look in a fixed radius of the current state estimate rather than using it's covariance since the correlative scan matcher is able to detect matches outside of the current state's 3σ ellipse due to the large size of the submaps.

The second type of loop closure is performed with the line features. Once a scan-to-submap loop closure is closed with the above method, we exhaustively search all lines in the state vector to see if any of them have moved enough to be considered the same line by using the same criteria as line tracking, but with different thresholds. If we find that two or more lines should be merged, we remove all but one from the state vector, and reroute the edges that were connected to the removed lines to single merged line. This can help to fix any lines that incorrectly lost tracking and improve the overall solution. Additionally, even though this is an exhaustive search, our method to compare the lines is highly efficient, so this procedure typically runs in a few microseconds – even on large maps.

IV. ONLINE LOCALIZATION

In this section, we present our versatile EKF-based multi-sensor fusion for online localization, which can incorporate any multi-modal information from a pre-built map, odometry, or LiDAR, if available, while automatically compensating for spatial/temporal calibration variations. In particular, the state vector of the proposed EKF consists of the current inertial navigation state, two historical LiDAR poses, and the set of extrinsic parameters.

$$\mathbf{x}_k = \begin{bmatrix} \mathbf{x}_I & \mathbf{x}_L & \mathbf{x}_W & {}^L t_I & {}^L t_O \end{bmatrix} \tag{15}$$

$$\mathbf{x}_I = \begin{bmatrix} {}^{I_k}_G\mathbf{R} & {}^G\mathbf{p}_{I_k} & {}^G\mathbf{v}_{I_k} & \mathbf{b}_{\omega_k} & \mathbf{b}_{a_k} \end{bmatrix} \tag{16}$$

$$\mathbf{x}_L = \begin{bmatrix} {}^{I_k}_G\mathbf{R} & {}^G\mathbf{p}_{I_k} & {}^{I_{k-1}}_G\mathbf{R} & {}^G\mathbf{p}_{I_{k-1}} \end{bmatrix} \tag{17}$$

$$\mathbf{x}_W = \begin{bmatrix} {}^L_I\mathbf{R} & {}^L\mathbf{p}_I & {}^O_I\mathbf{R} & {}^O\mathbf{p}_I \end{bmatrix} \tag{18}$$

where $\mathbf{b}_\omega$ and $\mathbf{b}_a$ are the gyroscope and accelerometer biases, and ${}^G\mathbf{v}_{I_k}$ is the velocity in the global frame. We use the right orientation error state $\mathbf{R} = \hat{\mathbf{R}}\ \mathrm{Exp}(-\boldsymbol{\delta\theta})$, where $\mathrm{Exp}(\cdot)$ is the $SO(3)$ matrix exponential [17], while for all other vector quantities the error state is addition $\mathbf{v} = \hat{\mathbf{v}} + \tilde{\mathbf{v}}$.

We propagate the inertial state $\mathbf{x}_I$ forward using incoming IMU measurements of linear accelerations ${}^I\mathbf{a}_m$ and angular velocities ${}^I\boldsymbol{\omega}_m$ based on the following generic nonlinear IMU kinematics [18] propagating the state from timestep $k-1$ to k and covariance $\mathbf{P}_{k-1|k-1}$ forward in time:

$$\hat{\mathbf{x}}_{k|k-1} = f(\hat{\mathbf{x}}_{k-1|k-1}, {}^I\mathbf{a}_m, {}^I\boldsymbol{\omega}_m, \mathbf{0}) \tag{19}$$

$$\mathbf{P}_{k|k-1} = \boldsymbol{\Phi}_{k-1}\mathbf{P}_{k-1|k-1}\boldsymbol{\Phi}_{k-1}^\top + \mathbf{Q}_{k-1} \tag{20}$$

where $\hat{\cdot}$ denotes the estimated value and the subscript $k|k-1$ denotes the predicted estimate at time t_k given the measurements up to time t_{k-1}. $\boldsymbol{\Phi}_{k-1}$ and $\mathbf{Q}_{k-1}$ are the system Jacobian and discrete noise covariance matrices of the linearized system [19]. We purposely choose the IMU for propagation, as compared to using the wheel odometry, since the the IMU is likely to have less spurious readings, while wheel odometry is dependent on the environment and wheel slip could cause incorrect state propagation and inconsistent estimation.

A general non-linear measurement function relates to the states (e.g., relative and global pose measurements from ICP, odometry integration measurements, etc) can be written as:

$$\mathbf{z}_{m,k} = \mathbf{h}(\mathbf{x}_k) + \mathbf{n}_{m,k} \tag{21}$$

where we the measurement noise $\mathbf{n}_{m,k} \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_{m,k})$. We can linearize Eq. (21) and use it for standard EKF update. We note that before any update we check if the measurement passes a 95 percentile χ^2-distribution gating test to prevent bad measurements from corrupting our state estimates. In what follows, we explain the pertinent measurements used in our EKF update.

A. *Pointcloud Projection*

One of the challenges of using a 2D LiDAR sensor in a 3D world is that the alignment of 2D clouds are only valid if they are measured in the same plane. If a 2D LiDAR records a scan of a room, and then pitches upwards by 45 degrees, the alignment result will be non-trivial even though the robot has not moved. For ground vehicles this can be the case when we go up and down a slope or hit a bump on the ground. Thus, in order to properly find the alignment between two scans, both need to be in the same 3D plane. Thus we project all features from the current LiDAR frame into the global xy plane assuming the walls are aligned with gravity as follows:

$${}^{L'_k}\bar{\mathbf{p}}_f = {}^L_I\bar{\mathbf{R}}{}^{I_k}_G\bar{\mathbf{R}}\boldsymbol{\Lambda}\left({}^{I_k}_G\mathbf{R}^\top {}^L_I\mathbf{R}^\top {}^{L_k}\mathbf{p}_f\right) \tag{22}$$

where $\boldsymbol{\Lambda} = [\mathbf{e}_1^\top\ \mathbf{e}_2^\top]^\top$, $\mathbf{e}_i$ is the i-th standard basis, and ${}^{L'_k}\bar{\mathbf{p}}_f$ is the 2D position of the feature in the xy plane, as seen from yaw only local frame $\{L'_k\}$.

B. *Relative LiDAR ICP*

We leverage the point-to-plane variant of iterative closest point (ICP) [20] with covariance estimation of the resulting transformation following [21], [22]. We use the *libpointmatcher* library point-to-plane minimizer [23] and the MatLab derivation code of [22] to implement the ICP algorithm. We use a max of four neighbor points to compute the pointcloud normal vectors and force the minimization to only optimize a 2D transformation.

We first project the last $\{L_{k-1}\}$ and current $\{L_k\}$ scans into the global xy plane. We denote these two 2D pointclouds as ${}^{L'_{k-1}}\boldsymbol{\mathcal{P}}$ and ${}^{L'_k}\boldsymbol{\mathcal{P}}$ respectively. We then run the ICP algorithm to get the following result:

$$\begin{bmatrix} {}^{L'_{k-1}}_{L'_k}\bar{\mathbf{R}}_m & , {}^{L'_{k-1}}\bar{\mathbf{p}}_{L'_k,m} & , \mathbf{Q}_{icp} \end{bmatrix} = icp\left({}^{L'_{k-1}}\boldsymbol{\mathcal{P}}, {}^{L'_k}\boldsymbol{\mathcal{P}}\right) \quad (23)$$

where $\mathbf{Q}_{icp}$ is the calculated measurement covariance which is based on the uncertainty of the points in each pointcloud. As noted in [24], this covariance result can be extremely overconfident and if directly incorporated can make the estimator become highly inconsistent. Thus we manually inflate this covariance by a fixed amount over all datasets to ensure that we are properly capturing the noise of this ICP alignment.

We now define the following measurement function:

$$ {}^{L'_{k-1}}_{L'_k}\bar{\mathbf{R}} = {}^{L}_{I}\bar{\mathbf{R}}{}^{I_{k-1}}_{G}\bar{\mathbf{R}}{}^{I_k}_{G}\bar{\mathbf{R}}^\top {}^{L}_{I}\bar{\mathbf{R}}^\top \quad (24)$$

$$ {}^{L'_{k-1}}\bar{\mathbf{p}}_{L'_k} = {}^{L}_{I}\bar{\mathbf{R}}{}^{I_{k-1}}_{G}\bar{\mathbf{R}}\boldsymbol{\Lambda}({}^{G}\mathbf{p}_{L_k} - {}^{G}\mathbf{p}_{L_{k-1}}) \quad (25)$$

where ${}^{G}\mathbf{p}_{L_i} = {}^{G}\mathbf{p}_{I_i} - {}^{I_i}_{G}\mathbf{R}^\top {}^{L}_{I}\mathbf{R}^\top {}^{L}\mathbf{p}_I$. We can now linearize the above measurement functions, such that we have the following linearized measurement error:

$$ {}^{L'_{k-1}}_{L'_k}\delta\theta_z = \mathbf{H}_{\theta L}\tilde{\mathbf{x}}_L + \mathbf{H}_{\theta W}\tilde{\mathbf{x}}_W + n_\theta \quad (26)$$

$$ {}^{L'_{k-1}}\tilde{\mathbf{p}}_{L'_k} = \mathbf{H}_{pL}\tilde{\mathbf{x}}_L + \mathbf{H}_{pW}\tilde{\mathbf{x}}_W + \mathbf{n}_p \quad (27)$$

where $[n_\theta \;\; \mathbf{n}_p^\top]^\top \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{icp})$. We have the following Jacobians in respect to our state:

$$\mathbf{H}_{\theta L} = \begin{bmatrix} -\mathbf{e}_3^\top {}^{L'_k}_{G}\hat{\mathbf{R}} & \mathbf{0}_{1\times 3} & \mathbf{e}_3^\top {}^{L'_k}_{G}\hat{\mathbf{R}} & \mathbf{0}_{1\times 3} \end{bmatrix}$$

$$\mathbf{H}_{\theta W} = \begin{bmatrix} \mathbf{e}_3^\top \left({}^{L'_k}_{L'_{k-1}}\hat{\mathbf{R}}{}^{L}_{I}\hat{\mathbf{R}} - {}^{L}_{I}\hat{\mathbf{R}}\right) & \mathbf{0}_{1\times 9} \end{bmatrix}$$

$$\mathbf{H}_{pL} = \begin{bmatrix} \mathbf{H}_1 & \mathbf{H}_2 & \mathbf{H}_3 & \mathbf{H}_4 \end{bmatrix}, \; \mathbf{H}_{pW} = \begin{bmatrix} \mathbf{H}_5 & \mathbf{H}_6 & \mathbf{0}_{2\times 6} \end{bmatrix}$$

$$\mathbf{H}_1 = \boldsymbol{\Lambda}\, {}^{L'_{k-1}}_{G}\hat{\mathbf{R}}\lfloor {}^{G}\hat{\mathbf{p}}_{L_k} - {}^{G}\hat{\mathbf{p}}_{L_{k-1}} + {}^{L'_{k-1}}_{G}\hat{\mathbf{R}}^\top {}^{L}\hat{\mathbf{p}}_I \rfloor$$

$$\mathbf{H}_2 = -\boldsymbol{\Lambda}\, {}^{L'_{k-1}}_{G}\hat{\mathbf{R}}, \qquad \mathbf{H}_3 = \boldsymbol{\Lambda}\, {}^{L'_{k-1}}_{G}\hat{\mathbf{R}}\lfloor {}^{L'_k}_{G}\hat{\mathbf{R}}^\top {}^{L}\hat{\mathbf{p}}_I \rfloor$$

$$\mathbf{H}_4 = \boldsymbol{\Lambda}\, {}^{L'_{k-1}}_{G}\hat{\mathbf{R}}, \qquad \mathbf{H}_6 = \boldsymbol{\Lambda}\, (\mathbf{I} - {}^{L_{k-1}}_{G}\hat{\mathbf{R}}{}^{L_k}_{G}\hat{\mathbf{R}}^\top)$$

$$\mathbf{H}_5 = \boldsymbol{\Lambda}\, \Big({}^{L}_{I}\hat{\mathbf{R}}\lfloor {}^{I_{k-1}}_{G}\hat{\mathbf{R}}({}^{G}\hat{\mathbf{p}}_{L_k} - {}^{G}\hat{\mathbf{p}}_{L_{k-1}})\rfloor$$
$$+ {}^{L_{k-1}}_{G}\hat{\mathbf{R}}{}^{I_k}_{G}\hat{\mathbf{R}}^\top \lfloor {}^{L}_{I}\hat{\mathbf{R}}^\top {}^{L}\hat{\mathbf{p}}_I \rfloor - {}^{L_{k-1}}_{G}\hat{\mathbf{R}}{}^{I_{k-1}}_{G}\hat{\mathbf{R}}^\top \lfloor {}^{L}_{I}\hat{\mathbf{R}}^\top {}^{L}\hat{\mathbf{p}}_I \rfloor\Big)$$

Using these linearized measurement residual and Jacobians we can perform our EKF update.

C. Global Prior Map LiDAR ICP

We first project the current $\{L_k\}$ scans into the global xy plane, getting the pointcloud ${}^{L'_k}\boldsymbol{\mathcal{P}}$. This cloud will then be aligned with the prior map pointcloud ${}^{G}\boldsymbol{\mathcal{M}}$ generated from our mapping system. We found that directly doing the ICP in the global frame of reference is very unstable and produces a covariance unsuitable for estimation. Thus, we transform a the points locally near to the current position in the prior map into the current frame of reference ${}^{L'_k}\boldsymbol{\mathcal{M}}$ and perform ICP as follows:

$$\begin{bmatrix} {}^{L'_k}_{L^+_k}\bar{\mathbf{R}}_m & , {}^{L'_k}\bar{\mathbf{p}}_{L^+_k,m} & , \mathbf{Q}_{icp} \end{bmatrix} = icp\left({}^{L'_k}\boldsymbol{\mathcal{P}}, {}^{L'_k}\boldsymbol{\mathcal{M}}\right) \quad (28)$$

where the frame $\{L^+_k\}$ is the corrected $\{L'_k\}$ frame that the current estimate should be at in the prior map. We can then directly compound this state to get the ICP measurement in the global frame:

$$ {}^{G}_{L^+_k}\bar{\mathbf{R}}_m = {}^{L'_k}_{G}\hat{\mathbf{R}}^\top {}^{L'_k}_{L^+_k}\bar{\mathbf{R}}_m, \; {}^{G}\bar{\mathbf{p}}_{L^+_k,m} = \boldsymbol{\Lambda}{}^{G}\mathbf{p}_{L_k} + {}^{L'_k}_{G}\hat{\mathbf{R}}^\top {}^{L'_k}\bar{\mathbf{p}}_{L^+_k,m}$$

Written as a function of the state we have:

$$ {}^{G}_{L'_k}\bar{\mathbf{R}} = {}^{I_k}_{G}\bar{\mathbf{R}}^\top {}^{L}_{I}\bar{\mathbf{R}}^\top, \; {}^{G}\bar{\mathbf{p}}_{L'_k} = \boldsymbol{\Lambda}({}^{G}\mathbf{p}_{L_k} - {}^{I_k}_{G}\mathbf{R}^\top {}^{L}_{I}\mathbf{R}^\top {}^{L}\mathbf{p}_I) \quad (29)$$

which can then be linearized to get the following measurement error state and Jacobians:

$$ {}^{G}_{L'_k}\delta\theta_z = \mathbf{H}_{\theta L}\tilde{\mathbf{x}}_L + \mathbf{H}_{\theta W}\tilde{\mathbf{x}}_W + n_\theta \quad (30)$$

$$ {}^{G}\tilde{\mathbf{p}}_{L'_k} = \mathbf{H}_{pL}\tilde{\mathbf{x}}_L + \mathbf{H}_{pW}\tilde{\mathbf{x}}_W + \mathbf{n}_p \quad (31)$$

$$\mathbf{H}_{\theta L} = \begin{bmatrix} -\mathbf{e}_3^\top {}^{L'_k}_{G}\hat{\mathbf{R}} & \mathbf{0}_{1\times 9} \end{bmatrix}, \; \mathbf{H}_{\theta W} = \begin{bmatrix} -\mathbf{e}_3^\top {}^{L}_{I}\hat{\mathbf{R}} & \mathbf{0}_{1\times 9} \end{bmatrix} \quad (32)$$

$$\mathbf{H}_{pL} = \begin{bmatrix} \boldsymbol{\Lambda}\lfloor {}^{L_k}_{G}\hat{\mathbf{R}}^\top {}^{L}\hat{\mathbf{p}}_I \rfloor & \boldsymbol{\Lambda}\mathbf{I}_{3\times 3} & \mathbf{0}_{2\times 6} \end{bmatrix} \quad (33)$$

$$\mathbf{H}_{pW} = \begin{bmatrix} \boldsymbol{\Lambda}{}^{I'_k}_{G}\hat{\mathbf{R}}^\top \lfloor {}^{L}_{I}\hat{\mathbf{R}}^\top {}^{L}\hat{\mathbf{p}}_I \rfloor & -\boldsymbol{\Lambda}{}^{L'_k}_{G}\hat{\mathbf{R}}^\top & \mathbf{0}_{2\times 6} \end{bmatrix} \quad (34)$$

D. LiDAR Time Offset Estimation

Note that for both relative and global ICP, we do not need to take into account the time offset. We clone the "true" time that the LiDAR scan was collected at from the current state pose from $\mathbf{x}_I$ into $\mathbf{x}_L$. Following the logic of Li and Mourikis [25], we say that the true will be near the current propagated estimate $\{{}^{I_k}_{G}\hat{\mathbf{R}}, {}^{G}\hat{\mathbf{p}}_{I_k}\}$ plus a small error:

$$ {}^{I_k}_{G}\mathbf{R} = \mathrm{Exp}(-\boldsymbol{\omega}{}^{L}\tilde{t}_I){}^{I_k}_{G}\hat{\mathbf{R}}, \quad {}^{G}\mathbf{p}_{I_k} = {}^{G}\hat{\mathbf{p}}_{I_k} + {}^{G}\hat{\mathbf{v}}_{I_k}{}^{L}\tilde{t}_I \quad (35)$$

E. Odometry Measurements

As compared to the mapping system we formulate a full 3D odometry measurement. In this case we create "psuedo" measurements where the angular rate about the roll and pitch and the velocity along the y and z-axes should be zero. We assign measurement uncertainties to these directions based on the environment ground conditions. In the case that we go up a hill and this "psuedo" measurement no longer holds, the χ^2 gating test will reject this invalid measurement. The general 3D odometry readings at time τ are:

$$ {}^{O_\tau}\boldsymbol{\omega}_m = {}^{O_\tau}\omega\mathbf{e}_3 + \mathbf{n}_{w\tau}, \quad {}^{O_\tau}\mathbf{v}_m = {}^{O_\tau}v\mathbf{e}_1 + \mathbf{n}_{v\tau} \quad (36)$$

where $\mathbf{n}_v$ and $\mathbf{n}_w$ denotes white Gaussian noises. The integrated relative pose and measurement covariance between t_{k-1} and $t_{\tau+1}$ is as follows:

$$ {}^{O_{k-1}}_{O_{\tau+1}}\mathbf{R} = {}^{O_{k-1}}_{O_\tau}\mathbf{R}\;\mathrm{Exp}\left(({}^{O_\tau}\boldsymbol{\omega}_m - \mathbf{n}_{w\tau})\delta t_\tau\right) \quad (37)$$

$$ {}^{O_{k-1}}\mathbf{p}_{O_{\tau+1}} = {}^{O_{k-1}}\mathbf{p}_{O_\tau} + {}^{O_{k-1}}_{O_\tau}\mathbf{R}({}^{O_\tau}\mathbf{v}_m - \mathbf{n}_{v\tau})\delta t_\tau \quad (38)$$

$$\mathbf{Q}_{k-1,\tau+1} = \mathbf{H}_\tau\mathbf{Q}_{k-1,\tau}\mathbf{H}_\tau^\top + \mathbf{G}_\tau\mathbf{Q}_{m\tau}\mathbf{G}_\tau^\top \quad (39)$$

where $[\mathbf{n}_{w\tau}^\top \;\; \mathbf{n}_{v\tau}^\top]^\top \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{m\tau})$ is the noise covariance matrix of a single odometry reading. Over all odometry readings between t_{k-1} and t_k, we integrate Eq. (37)-(39) to get the 3D relative pose measurement $\mathbf{z}_{k-1,k}$ with covariance $\mathbf{Q}_{k-1,k}$. The measurement function is:

$$\mathbf{z}_{k-1,k} = \begin{bmatrix} \mathrm{Log}({}^{O}_{I}\mathbf{R}{}^{I_{k-1}}_{G}\mathbf{R}{}^{I_k}_{G}\mathbf{R}^\top {}^{O}_{I}\mathbf{R}^\top) \\ {}^{O}_{I}\mathbf{R}{}^{I_{k-1}}_{G}\mathbf{R}({}^{G}\mathbf{p}_{O_k} - {}^{G}\mathbf{p}_{O_{k-1}}) \end{bmatrix} + \mathbf{n}_o \quad (40)$$

where $\mathbf{n}_o \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{k-1,k})$ and the positions are ${}^{G}\mathbf{p}_{O_i} = {}^{G}\mathbf{p}_{I_i} - {}^{I_i}_{G}\mathbf{R}^\top {}^{O}_{I}\mathbf{R}^\top {}^{O}\mathbf{p}_I$. We omit the Jacobians in respect to the state and calibration here for brevity. They are in the same form as the 2D LiDAR relative, Sec. IV-B, but without the projection matrices.

To handle calibration of the time offset we follow a logic close to that introduced in Sec. IV-D. In this case, the poses in our state are cloned at the "true" LiDAR clock time. Thus

TABLE I: Key simulation parameters for each sensor along with key estimator parameters.

Parameter	Value	Parameter	Value
IMU Freq. (hz)	200	Wheel Freq. (hz)	100
LiDAR Freq. (hz)	10	LiDAR Clones	2
Gyro. White Noise	1.6968e-04	Gyro. Rand. Walk	1.9393e-05
Accel. White Noise	2.0000e-03	Accel. Rand. Walk	3.0000e-03
Odom. Ang. Noise (rad/s)	8.0000e-03	Odom. Vel. Noise (m/s)	2.0000e-02
LiDAR Ray Noise (m)	3.0000e-02	LiDAR FOV (deg)	270
LiDAR Ang. Resolution (deg)	0.5	Prior Map Cell Size (m)	0.05

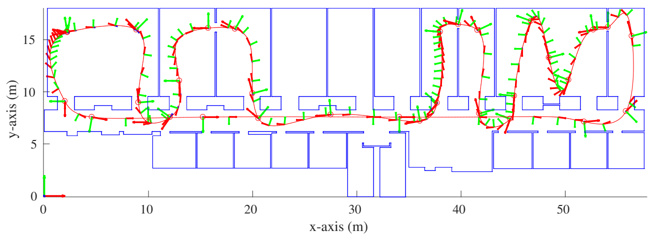

Fig. 2: Trajectory plot of the simulated structured environment containing 236 planes, 348 meter in length trajectory, and average speed of 1.8 m/s.

when we calculate our preintegrated odometry measurement, we propagate to the current estimate of the true LiDAR time. Using the expansions in Eq. (35) we can find the derivative of the propagated states in respect to the true LiDAR clones that are in our state.

V. ONLINE LOCALIZATION SIMULATION

Building off of the LIPS [26] and Open VINS [27] simulators we simulate a sensor suite moving through an indoor environment defined by a 2D floorplan and trajectory, see Fig. 2. We enforced that the orientation of the system is always along the velocity direction and is a pure 2D trajectory by setting the z-axis values to be zero, thus modeling our ground robot's nonholonomic motion constraint. LiDAR range measurements are generated by sending rays from the current true LiDAR pose and intercepted with the the 3D planes which have been extruded vertically from the 2D floorplan. We select the closest intersection as LiDAR range measurement and reject any that are further then the max range of the sensor. The prior map is generated using the 2D floorplan whose lines are sampled at the desired prior map cell size (i.e. along a line that defines a wall, we create a prior map point at fixed intervals along the line). All measurements are corrupted by some white Gaussian noise, while the IMU has additional time-varying bias term added. Table I, has the key sensor frequencies, sensor properties, and noise parameters used during simulation. For all experiments

TABLE II: Root mean squared error (RMSE) and normalized estimator error squared (NEES) for different system configurations averaged over 15 runs.

Configuration	RMSE Ori. (deg)	RMSE Pos. (m)	NEES Ori.	NEES Pos.
IMU + REL	14.034	4.344	4.122	4.845
IMU + REL + ODOM	3.714	1.221	2.574	1.693
IMU + PRIOR	0.201	0.047	1.979	0.595
IMU + PRIOR + ODOM	0.191	0.043	2.406	1.564
IMU + PRIOR + ODOM + REL	0.182	0.040	2.344	1.464

TABLE III: Root mean squared error (RMSE) for different frequency of prior map update averaged over 15 runs.

Prior Update Freq. (Hz)	RMSE Ori. (deg)	RMSE Pos. (m)
10	0.182	0.040
1	0.303	0.063
0.20	0.454	0.100
0.10	0.569	0.153

we report the error in the 2D plane while consistency metrics are for the full 3D estimated states.

A. Sensor Impact Comparison

Table II, shows the average root mean squared error (RMSE) of 15 simulated Monte-Carlo runs with all online calibration enabled and for a 2D trajectory through the simulated environment with different sensor configurations. As noted in Sec. IV-B, we inflated the covariance of the ICP algorithm by a fixed amount over all datasets and experiments. Even so, in the IMU + relative LiDAR we have slightly inconsistent estimation due to this inflation, while when additional sensors are used the filter becomes slightly under-confident in its estimates. We found that being under-confident provided far superior estimation accuracy as compared to having a over-confident filter. It is also interesting that the prior map greatly reduces the estimation error and that using all measurements provides the best accuracy.

B. Frequency of Prior Map Updates

Another key question is how often do we need to get prior map updates to obtain the impressive accuracy provided by the prior map? Table III shows the estimation error of the proposed system with inertial, odometry, relative LiDAR updates at 10 Hz and prior map updates at a specified fixed frequency. It can be seen that even when there are ten seconds between consecutive prior map updates the estimate is still able to retain high accuracy as compared to not including the prior map constraint.

C. Inspection of Online Calibration

We next looked to verify the ability of the proposed system to perform online calibration. As shown in Fig. 3, it is interesting that not all calibration parameters are able to calibrate. Specifically the LiDAR-IMU roll, pitch, and z-axis position are unable to quickly converge, which suggest that we might not be able to robustly calibrate these parameters online for a 2D LiDAR. The odometry calibration is unable to calibrate along the vertical direction which in this case is normal to the xy plane and is similar to what is seen for IMU-camera calibration [28].

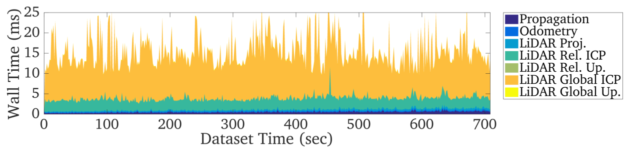

Fig. 6: Timing of different system components reported in milliseconds for the building floor dataset. Recorded on an Intel(R) Xeon(R) CPU E3-1505M v6 @ 3.00GHz processor in single threaded execution.

VI. REAL-WORLD EXPERIMENTAL RESULTS

A. Mapping Demonstration

To validate our mapping system, we compare our system to Google Cartographer [3] in a warehouse scenario. The result of this experiment can be observed in Fig. 4. Due to using line features and weighting by covariance, we are able to achieve a higher map quality in this case. Cartographer's

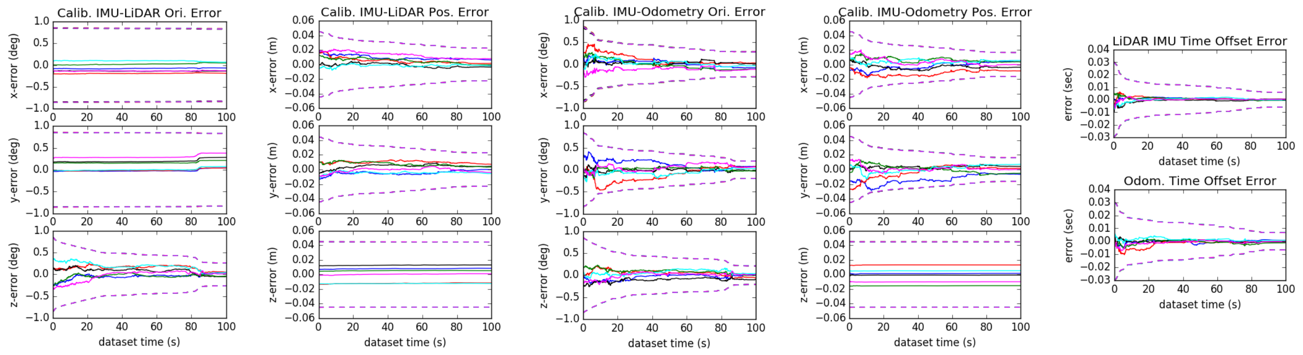

Fig. 3: Calibration errors of each parameter (solid) and 3σ bound (dotted) for six different runs under planar motion. Each colors denote runs with different realization of the measurement noise and the initial values.

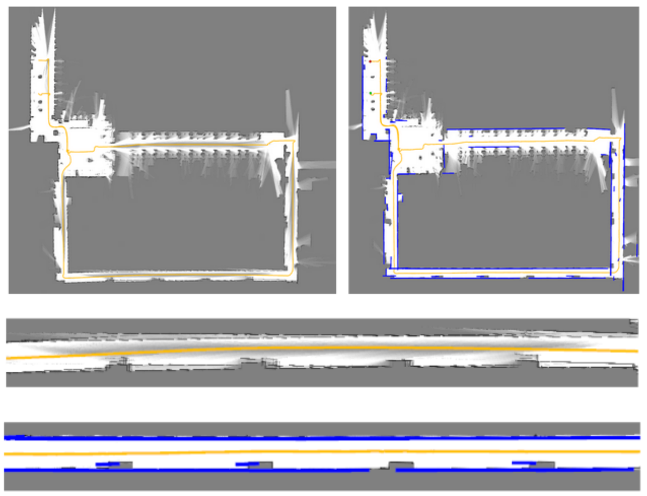

Fig. 4: A map generated by Google Cartographer (top left) and our mapping system (top right) in a warehouse environment. The LiDAR frame trajectory is shown in yellow with the start and end points in green and red, respectively. Line features are overlaid in blue for our map. Cartographer was forced to sacrifice the map quality along the hallway (middle) in order to close the loop, while our system successfully closed the loop and kept the hallway in-tact (bottom).

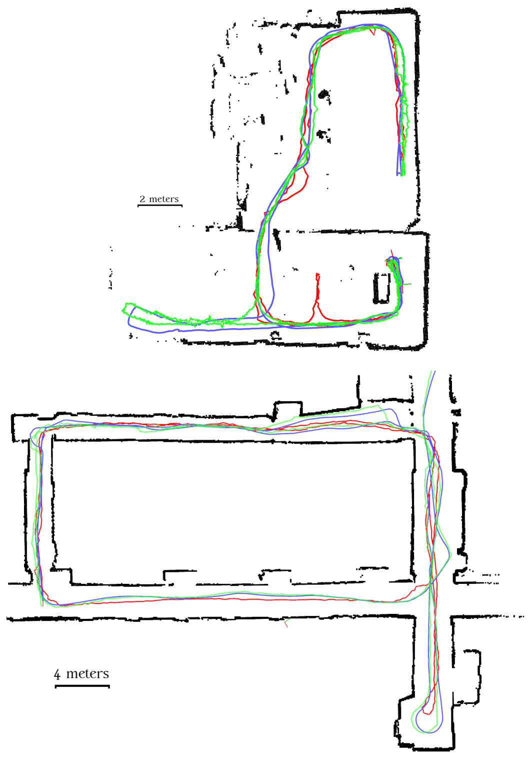

Fig. 5: A two room (top) and building floor (bottom) datasets collected with a Turtlebot3. Each map and the mapping trajectory (red) was used by the localization system (blue) on a second dataset through the environment. The batch optimized trajectory (green) of the second dataset can also be seen.

map of the lower hallway becomes corrupted after the loop closes, while our system correctly keeps the walls together. By estimating the uncertainty of the scan measurements and also including the line features, the nodes in the lower hallway are very well constrained in the vertical direction, and will tend to only move in the direction parallel to the walls during optimization, while Cartographer's equal weighting system causes the hallway to fall apart. Note that we used the values from Cartographer's PR2 demonstration, with the addition of wheel odometry.

B. Turtlebot3 Online Localization Demonstration

To further validate the system as a whole, we demonstrate its ability to easily run on a commercially available Turtlebot3 [2] system equipped with its integrate wheel encoders, inertial measurement unit, and an inexpensive RPLIDAR-A1[3]. The IMU runs at 118 Hz, odometry at 25 Hz, and LiDAR at 7 Hz, and all calibration initial values are hand measured. These parts cost no more then \$550 USD online, with the LiDAR making up only \$100 USD of the total. The Turtlebot3 is an excellent platform for robotic education, research, hobby, and product prototyping.

[2]https://www.turtlebot.com/
[3]http://www.slamtec.com/en/lidar/a1

To overcome the "kidnapped robot" problem on startup, we leverage a depth-first search on the loaded map occupancy grid with increase bounds. After finding the location in the prior map, we transform the yaw and xy location from the LiDAR to IMU sensor frame. The roll and pitch directions of the orientation, which are observable given the IMU sensor, are calculated using standard inertial frame initialization [27]. The orientation is the compound of the roll and pitch from this procedure and the yaw of the IMU in the prior map frame and the global xy of the IMU the position returned by the depth-first search.

To perform ICP on the prior map, the map occupancy grid from the mapping session is loaded from disk and converted into a pointcloud through thresholding the occupancy grid.

It is important to note that we only perform global ICP with points locally, as mentioned in Sec. IV-C, randomly sample a smaller 1.5k subset, try to match to the prior for every incoming LiDAR scan, and generate the prior map on a separate dataset through the environment.

In Fig. 5, the odometry trajectory can be seen overlaid along with the map trajectory used to generate the prior map. While there is no groundtruth, a visual inspection of the trajectory compared to the mapping system's output on the same localization dataset, shows good estimation and constraint within the prior map. Shown in Fig. 6, it is clear that the system is able to incorporate all three measurement sources at very high speed. For the two room dataset it took 12.3 ms to update on average and for the building floor dataset it took 15.4 ms. The ICP with the global map takes the most time being an average of 9.2 and 11.3 ms, respectively. It should also be noted that one can reduce the computation by only trying to get a prior map constraint at a lower frequency. We found that on the building floor dataset we could do prior map updates at 1 Hz and still get the same generated trajectory with an average update time of 4.2 ms.

VII. CONCLUSION AND FUTURE WORK

In this paper, we have developed a 2-stage multi-sensor localization system, which incorporates both offline prior map construction and online multi-sensor map-based localization, for low-cost low-power autonomous ground robots. For the mapping functionality, leveraging Cartographer [3], we have introduced the accurate scan-to-submap covariance modeling, probabilistic odometry reading fusion, and accelerated loop-closure detection. A 2D line map is built along with the occupancy grid map to exploit the environmental structural constraints. During online localization, a versatile 3D EKF-based system, which leverages the pre-built occupancy grid map, optimally fuse inertial, odometry and 2D LiDAR measurements, if available. Spatial-temporal calibration between these sensors is estimated online to handle poor initial guess and "plug and play" applications. Extensive Monte-Carlo simulations verified both the accuracy, consistency, and robustness of proposed localization system, while the complete system was validated on a two room and building floor realworld datasets. In the future we will exploit inertial measurements in the mapping and incorporate visual sensors in both mapping and localization.

REFERENCES

[1] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Transactions on robotics*, vol. 32, no. 6, pp. 1309–1332, 2016.

[2] P. Beinschob and C. Reinke, "Graph slam based mapping for agv localization in large-scale warehouses," in *2015 IEEE International Conference on Intelligent Computer Communication and Processing (ICCP)*. IEEE, 2015, pp. 245–248.

[3] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2d lidar slam," in *2016 IEEE International Conference on Robotics and Automation*. IEEE, 2016, pp. 1271–1278.

[4] J. M. Santos, D. Portugal, and R. P. Rocha, "An evaluation of 2d slam techniques available in robot operating system," in *2013 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*. IEEE, 2013, pp. 1–6.

[5] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with rao-blackwellized particle filters," *IEEE transactions on Robotics*, vol. 23, no. 1, pp. 34–46, 2007.

[6] K. Daun, S. Kohlbrecher, J. Sturm, and O. von Stryk, "Large scale 2d laser slam using truncated signed distance functions," in *2019 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*. IEEE, 2019, pp. 222–228.

[7] F. Boniardi, T. Caselitz, R. Kümmerle, and W. Burgard, "A pose graph-based localization system for long-term navigation in cad floor plans," *Robotics and Autonomous Systems*, vol. 112, pp. 84–97, 2019.

[8] M. Montemerlo, S. Thrun, D. Koller, B. Wegbreit *et al.*, "Fastslam: A factored solution to the simultaneous localization and mapping problem," *AAAI*, vol. 593598, 2002.

[9] B. Steux and O. El Hamzaoui, "tinyslam: A slam algorithm in less than 200 lines c-language program," in *2010 11th International Conference on Control Automation Robotics & Vision*. IEEE, 2010, pp. 1975–1979.

[10] O. El Hamzaoui and B. Steux, "Slam algorithm with parallel localization loops: Tinyslam 1.1," in *2011 IEEE International Conference on Automation and Logistics (ICAL)*. IEEE, 2011, pp. 137–142.

[11] A. Huletski, D. Kartashov, and K. Krinkin, "Vinyslam: an indoor slam method for low-cost platforms based on the transferable belief model," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 6770–6776.

[12] S. Kohlbrecher, O. Von Stryk, J. Meyer, and U. Klingauf, "A flexible and scalable slam system with full 3d motion estimation," in *2011 IEEE International Symposium on Safety, Security, and Rescue Robotics*. IEEE, 2011, pp. 155–160.

[13] S. Agarwal, K. Mierle, and Others, "Ceres solver," http://ceres-solver.org.

[14] E. Olson, "Real-time correlative scan matching," in *2009 IEEE International Conference on Robotics and Automation*, 06 2009, pp. 4387 – 4393.

[15] Y. Yang and G. Huang, "Observability analysis of aided ins with heterogeneous features of points, lines and planes," *IEEE Transactions on Robotics*, vol. 35, no. 6, pp. 399–1418, Dec. 2019.

[16] S. T. Pfister, S. I. Roumeliotis, and J. W. Burdick, "Weighted line fitting algorithms for mobile robot map building and efficient data representation," in *2003 IEEE International Conference on Robotics and Automation*, vol. 1, Sep. 2003, pp. 1304–1311 vol.1.

[17] T. D. Barfoot, *State estimation for robotics*. Cambridge University Press, 2017.

[18] A. B. Chatfield, *Fundamentals of High Accuracy Inertial Navigation*. AIAA, 1997.

[19] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint kalman filter for vision-aided inertial navigation," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*. IEEE, 2007, pp. 3565–3572.

[20] Y. Chen and G. Medioni, "Object modeling by registration of multiple range images," in *Proceedings. 1991 IEEE International Conference on Robotics and Automation*, 1991, pp. 2724–2725.

[21] A. Censi, "An accurate closed-form estimate of icp's covariance," in *Proceedings 2007 IEEE international conference on robotics and automation*. IEEE, 2007, pp. 3167–3172.

[22] S. M. Prakhya, L. Bingbing, Y. Rui, and W. Lin, "A closed-form estimate of 3d icp covariance," in *2015 14th IAPR International Conference on Machine Vision Applications (MVA)*. IEEE, 2015, pp. 526–529.

[23] F. Pomerleau, F. Colas, R. Siegwart, and S. Magnenat, "Comparing ICP Variants on Real-World Data Sets," *Autonomous Robots*, vol. 34, no. 3, pp. 133–148, Feb. 2013.

[24] M. Brossard, S. Bonnabel, and A. Barrau, "A new approach to 3d icp covariance estimation," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 744–751, April 2020.

[25] M. Li and A. I. Mourikis, "Online temporal calibration for camera–imu systems: Theory and algorithms," *The International Journal of Robotics Research*, vol. 33, no. 7, pp. 947–964, 2014.

[26] P. Geneva, K. Eckenhoff, Y. Yang, and G. Huang, "LIPS: Lidar-inertial 3d plane slam," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, Madrid, Spain, Oct. 1-5, 2018.

[27] P. Geneva, K. Eckenhoff, W. Lee, Y. Yang, and G. Huang, "Openvins: A research platform for visual-inertial estimation," in *Proc. of the IEEE International Conference on Robotics and Automation*, Paris, France, 2020. [Online]. Available: https://github.com/rpng/open_vins

[28] Y. Yang, P. Geneva, K. Eckenhoff, and G. Huang, "Degenerate motion analysis for aided INS with online spatial and temporal calibration," *IEEE Robotics and Automation Letters (RA-L)*, vol. 4, no. 2, pp. 2070–2077, 2019.