Robust LiDAR-based Localization in Architectural Floor Plans

Federico Boniardi* Tim Caselitz* Rainer Kümmerle† Wolfram Burgard*

Abstract— **Modern automation demands mobile robots to be robustly localized in complex scenarios. Current localization systems typically use maps that require to be built and interpreted by experienced operators, increasing deployment costs as well as reducing the adaptability of robots to rearrangements in the environment. In contrast, architectural floor plans can be easily understood by non-expert users and typically represent only the non-rearrangeable parts of buildings. In this paper we propose a system for robot localization in architectural CAD drawings. Our method employs a simultaneous localization and mapping approach to online augment the floor plan with a map represented as a pose-graph with LiDAR measurements. Whenever the environment is accurately mapped in the vicinity of the robot, we use the graph to perform relative localization. We thoroughly evaluate our system in challenging real-world scenarios. Experiments demonstrate that our method is able to robustly track the robot pose even when the floor plan shows major discrepancies from the real-world. We show that our system achieves sub-centimeter accuracy and is suitable for real-time application.**

I. Introduction

Autonomous navigation is a crucial technology for modern flexible automation, with applications ranging from logistics to reconfigurable factories. Most of these scenarios require vehicles to be accurately localized during their operation. Traditional solutions for localization in industrial settings often employ installations in the factory hall, such as magnetic spots, markers, or guiding wires in the floor [1], [2]. However, modern approaches have proven their robustness and precision employing solely the safety LiDAR sensors on-board the vehicles and a map of the environment built with a comparable sensor modality [3], [4]. These maps are typically obtained during a preliminary process which requires supervision by an expert operator. They are built by solving the simultaneous localization and mapping (SLAM) problem and often encoded as occupancy grids which are not always intuitive for users due to noise and local distortions [5], as well as their intrinsic probabilistic representation. In fact, occupancy grid maps significantly differ from floor plans of buildings, which are commonly used human-readable architectural drawings created with modern CAD software. In order to understand occupancy maps, additional training for shop floor workers might be needed, resulting in an increased deployment time. Moreover, modifications of the environment are costly as they require new maps to be built, thus limiting the flexibility of rearranging building interiors or factory floors. Conversely, architectural drawings are often available and can therefore be leveraged for mobile robot navigation, reducing the set-up costs and the burden on the workers. Furthermore, they can be easily adapted to represent only the immutable features of a building, such as walls and doorways, thus providing an abstract representation which is independent of the actual configuration of the factory floor.

*Autonomous Intelligent Systems Lab, University of Freiburg, Germany
†KUKA Roboter GmbH, R&D Mobile Robotics, Augsburg, Germany

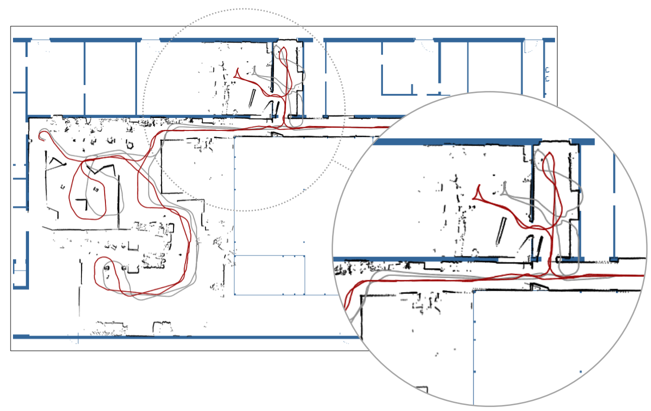

Fig. 1. The trajectory obtained with our localization system (red) in an architectural floor plan (blue) of a factory-like scenario. The map of LiDAR observations (black) shows also structures not represented in the floor plan. The map is aligned online to the CAD drawing to localize the robot. Our system works robustly even when the floor plan is fully occluded and in situations where Monte Carlo Localization (gray) fails.

In this work, we propose a robust LiDAR-based system for localization in architectural floor plans. Our approach combines mapping and localization techniques to exploit the information encoded in the CAD drawing as well as the observations from the real world, which are obtained during navigation and are not represented in the floor plan. In complex indoor environments only parts of the architectural CAD drawing match the current observations of the robot. Often, furniture and large equipment are located close to walls, covering them partially or even occluding them completely. This can significantly affect localization approaches that directly compare the sensor observations against a floor plan. To overcome the issue, our approach not only relies on pure localization techniques, but also performs mapping using a pose-graph-based SLAM formulation [6]. We propose a scan-to-map-matching method based on Generalized ICP (GICP) [7] to obtain constraints that align the estimated pose-graph to the CAD drawing as well as to overcome potential metrical inconsistencies of the floor plan. In order to reduce the computational burden, we perform *maximum a posteriori* (MAP) estimation of the current pose using relative measurements to the pose-graph once the mapping process is locally stabilized. Fig. 1 shows our system localizing a robot in a factory-like environment, including situations where the floor plan is fully occluded by large structures.

II. Related Work

The field of mobile robot localization has been extensively studied for several decades. Traditionally, the localization problem has been modeled as a Bayesian filtering problem by matching sensor readings with a globally consistent map of the environment. The most widespread methods assume underlying Markov models and use Kalman filters, histogram filters, or particle filters [8]. The latter approach is commonly known as Monte Carlo Localization (MCL). More recent works borrowed the idea of relative measurements exploited in pose-graph-based SLAM [6] and multi-robot localization [9], [10] and proposed the concept of *relative localization* to relax the need of globally consistent maps. Sprunk *et al.* [11] introduced the idea of localization against user-taught trajectories, encoded as a chain of consecutive odometry and 2D LiDAR measurements (anchor points). During the replay of the taught trajectory, the robot determines its pose with respect to the anchor points by matching the current scan with the reading stored during the teaching phase. The current anchor point is selected as the closest one to the robot. Mazuran *et al.* [12] extended this approach using an improved topological transition model for selecting anchor-points, allowing the robot to localize itself in a complex network of 2D LiDAR scans associated to a SLAM pose-graph.

Despite the extensive research on robot localization, few works have addressed the problem of localization using CAD floor plans. Luo *et al.* [13] proposed an integrated system for localization on floor plan maps fusing camera and ultrasonic measurements. The floor plan image is processed to extract useful features, namely room plates and corners in doorways and passages, which are used as landmarks for localization. The robot estimates its current pose using a Bayesian filter that encodes the confidence to observe the landmarks. Ito *et al.* [14] use Monte Carlo Localization to localize a device using only a CAD drawing, an RGB-D camera, an inertial measurement unit (IMU) and the WiFi signal strength. The proposal distribution is obtained using visual odometry and the likelihood of each particle is computed by extracting sections of the point cloud obtained from the depth camera. The WiFi signal is used to initialize the particle weights during the global localization phase in order to speed up convergence to a uni-modal distribution. Winterhalter *et al.* [15] proposed a similar method to localize a Google Tango tablet using the RGB-D and IMU data of the device. Inertial-visual odometry is utilized to define the proposal distribution and a simple 3D model shaped on the floor plan is used to define the expected measurements of a beam sensor model. Hile *et al.* [16] localize a mobile phone on a CAD drawing employing GPS and the camera of the phone. The method is purely geometric and relies on detecting landmarks based on "cornerity" features of the environment and matching them with the CAD drawing, thus inferring the relative pose of the camera with respect to those landmarks.

Our method for localization in floor plan drawings differs significantly from the approaches described above. Instead of addressing the problem using sensor models that are robust to the missing information in the drawings, we adopt SLAM techniques and concepts from relative localization to fit a map onto the floor plan in order to overcome the lack of features in CAD drawings. We also localize with respect to previously acquired LiDAR measurements whenever the map provides a sufficiently complete representation of the environment. To the best of our knowledge, the work that is algorithmically most similar to our approach was proposed by Vysotska *et al.* [17]. The authors used SLAM with prior information from OpenStreetMap to align the SLAM map of an outdoor urban environment to the map provided by OpenStreetMap, thus improving the robustness of the SLAM process and the quality of the generated map. Our work uses different prior maps and extends this approach presenting a long-term pure localization perspective.

This paper provides three contributions to the problem of robot localization in poorly informative maps. First, we propose a pose-graph-based framework for localization in floor plans using information from measurements irrespectively of being represented on the CAD drawing or not. Second, we introduce a method for scan-to-map-matching to obtain pose information relative to the floor plan. Third, we propose and evaluate a unified system for mapping and pure localization that is robust, accurate, and suitable for long-term operation.

TABLE I

The notation used in the paper.

Symbol	Description
$\mathbb{SE}(2)$	Manifold of 2D rigid transformations.
$\oplus, \ominus$	Compound operations in $\mathbb{SE}(2)$ (i.e., Cheeseman symbols).
$\mathbf{t}^{tr}$	Translation term of $\mathbf{t} \in \mathbb{SE}(2)$.
$\mathbf{t}^{R}, \mathbf{t}^{\phi}$	Rotation matrix and rotation angle of $\mathbf{t} \in \mathbb{SE}(2)$.
$\mathbf{t}x$	Rigid transformation of a vector $x \in \mathbb{R}^2$: $\mathbf{t}^R x + \mathbf{t}^{tr}$.
$[\![\mathbf{t}]\!]$	Element of $\mathbb{SE}(2)$ as a vector in $\mathbb{R}^3$.
Σ, Ω	Covariance matrix and related information matrix.
$\|x\|_2$	Standard Euclidean ℓ^2-norm: $(x^\top x)^{1/2}$.
$\|x\|_\Sigma$	Mahalanobis norm: $(x^\top \Omega x)^{1/2}$.
$\xi_{t:T}$	The sequence $(\xi_{t_1}, ..., \xi_{t_n})$, $t = t_1 < t_2 < \ldots < t_n = T$.
$(\xi_i)_i$	Shorten notation for $(\xi_i)_{i=1}^m$. Similarly for multi-indices.

III. Proposed Method

The goal of this work is to track the pose of a robot in an architectural floor plan using a 2D LiDAR sensor. Given a coarse estimate of the starting pose $\mathbf{x}_0$ in the reference frame of the floor plan and the current LiDAR reading $\mathcal{Z}_t$, we want to estimate the 2D pose $\mathbf{x}_t$ of the robot in that reference frame. The proposed method is not intended to solve the global localization problem since for many applications, such as industrial ones, robots start their operation at a known location, for instance a charging station. Henceforth, we refer to Tab. I for the notation and symbols adopted in this section.

The backbone of the method consists in a *maximum a posteriori* pose-graph-based SLAM system [18], which uses priors obtained from a CAD floor plan. Following the formulation of pose-graph-based SLAM with prior knowledge proposed in [19], given the trajectory of the robot represented as a sequence of poses $\mathbf{x}_{0:t} \in \mathbb{SE}(2)^n$ and relative measurements $(\Delta\mathbf{x}_{t_i,t_j})_{ij} \in \mathbb{SE}(2)^m$ between pairs of poses, we compute the trajectory that maximizes the posterior distribution of the relative measurements. Formally, assuming conditional independence of the measurements given a trajectory as well as prior independence of the related poses, we estimate the posterior trajectory as

$$\widehat{\mathbf{x}}_{0:t} \triangleq \operatorname*{arg\,max}_{\mathbf{x}_{0:t}} \prod_{t_i,t_j} p(\Delta\mathbf{x}_{t_i,t_j} \mid \mathbf{x}_{0:t}) \prod_{t_k} p(\mathbf{x}_{t_k}). \quad (1)$$

Assuming Gaussian noise in the relative measurements, namely $p(\Delta\mathbf{x}_{t_i,t_j} \mid \mathbf{x}_{0:t}) \sim \mathcal{N}([\![\mathbf{x}_{t_j} \ominus \mathbf{x}_{t_i}]\!], \Sigma_{ij})$, and either Gaussian or uniform noise of the priors, that is $p(\mathbf{x}_{t_k}) \sim \mathcal{N}([\![\overline{\mathbf{x}}_k]\!], \Sigma_k)$ or $p(\mathbf{x}_{t_k}) \sim \mathcal{U}$, the problem in Eq. 1 can be solved using non-linear least squares optimization as follows:

$$\begin{aligned}\widehat{\mathbf{x}}_{0:t} &= \operatorname*{arg\,min}_{\mathbf{x}_{0:t}} \sum_{ij} \|\mathbf{e}_{ij}\|^2_{\Sigma_{ij}} + \sum_k \|\mathbf{e}_k\|^2_{\Sigma_i} \\ &\approx \operatorname*{arg\,min}_{\mathbf{x}_{0:t}} \sum_{ij} \rho_m\left(\|\mathbf{e}_{ij}\|_{\Sigma_{ij}}\right) + \sum_k \rho_p\left(\|\mathbf{e}_k\|_{\Sigma_k}\right),\end{aligned} \quad (2)$$

where $\mathbf{e}_{ij} \triangleq [\![\ominus\Delta\mathbf{x}_{t_i,t_j} \ominus \mathbf{x}_{t_i} \oplus \mathbf{x}_{t_j}]\!]$, $\mathbf{e}_k \triangleq [\![\overline{\mathbf{x}}_k \ominus \mathbf{x}_{t_k}]\!]$ and the right summation is to be intended only for normally distributed priors and $\rho_m, \rho_p : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ are robust kernels used to reduce the influence of outliers.

In the remainder of this section we outline how this framework can be used and adapted for efficient and robust pose tracking even when floor plans are metrically inaccurate. Sec. III-A describes our localization method, while Sec. III-B presents how we estimate the trajectory prior using GICP-based scan-to-map-matching.

A. Localization Using Floor Plan Pose-Graphs

Although incrementally solving the SLAM with priors problem for the whole robot trajectory $\mathbf{x}_{0:t}$ can provide an MAP estimate of the current robot pose $\mathbf{x}_t$, it might be computationally intractable for long-term applications, unless the pose-graph is maintained sparse during navigation. Furthermore, the number of constraints in Eq. 2 related to relative measurements will eventually outweigh those generated by the priors, thus decoupling the pose-graph from the CAD drawing. In the following sections we propose two efficient methods to address these problems.

1) Maximum a Posteriori Localization: Assuming that a stable pose-graph anchored to the CAD drawing is available, we can adapt the MAP estimation in Eq. 1 to track the current robot pose without altering the underlying pose-graph and thus keeping the computational complexity bounded. Given the relative measurements $(\Delta\mathbf{x}_{t,t_j})_j$ at time t with respect to some previous poses in the trajectory $(\mathbf{x}_{t_j})_j$, we can estimate the current robot pose as

$$\widehat{\mathbf{x}}_t \triangleq \operatorname*{arg\,max}_{\mathbf{x}_t} \prod_{t_j} p(\Delta\mathbf{x}_{t,t_j} \mid \mathbf{x}_t) p(\mathbf{x}_t). \quad (3)$$

Under the assumptions of Eq. 2, the estimation above can be computed by solving the following optimization:

$$\widehat{\mathbf{x}}_t \approx \operatorname*{arg\,min}_{\mathbf{x}_t} \sum_j \rho_m\left(\|\mathbf{e}_j\|_{\Sigma_j}\right) + v_t \rho_p\left(\|\mathbf{e}_t\|_{\Sigma_t}\right), \quad (4)$$

where $\mathbf{e}_j$, $\mathbf{e}_t$, ρ_m and ρ_p are defined as above. The term $v_t \in \{0, 1\}$ encodes whether a Gaussian or a uniform prior is used, or, more concretely, if the scan-to-map-matching is valid or not. As for the optimization problem in Eq. 2, we compute $\widehat{\mathbf{x}}_t$ using a non-linear least squares optimization.

In order to decide whether a pure localization or a full pose-graph optimization should be executed, we use the following simple heuristic based on the associations computed by the front-end: pure localization is performed at time t only if enough associations can be obtained in the vicinity of the robot, that is, if the number of relative measurements $(\Delta\mathbf{x}_{t,t_j})_j$ exceeds a threshold value N_{loc}. This approach efficiently maintains the number of vertices and edges of the pose-graph bounded without any complex and computationally expensive pose-graph sparsification procedure. The pseudo-code of the whole system is reported in Algorithm 1.

Algorithm 1 MAP Localization in Architectural Floor Plans.

procedure LOCALIZATION($\mathcal{Z}_t$)
    $\langle \Delta\mathbf{x}_{t-1,t}, \Sigma_{t-1,t} \rangle \leftarrow \text{relative\_motion}(\mathcal{Z}_{t-1}, \mathcal{Z}_t)$
    if $\|\Delta\mathbf{x}^{tr}_{t-1,t}\|_2 < d \wedge |\Delta\mathbf{x}^{\phi}_{t-1,t}| < \alpha$ **then**
        return
    $\mathbf{x}_t \leftarrow \widehat{\mathbf{x}}_{t-1} \oplus \Delta\mathbf{x}_{t-1,t}$
    $\mathcal{G}_t \leftarrow \emptyset$
    if $t - 1 = \tilde{t}$ **then** ▷ *$\tilde{t}$: time of the last full optimization*
        $\mathcal{G}_t \leftarrow \mathcal{G}_t \cup \{\langle \Delta\mathbf{x}_{t-1,t}, \Sigma_{t-1,t} \rangle\}$
    $\langle \overline{\mathbf{x}}_t, \Sigma_t, v_t \rangle \leftarrow \text{prior}(\mathbf{x}_t, \mathcal{Z}_t, \textit{floorplan})$
    $\mathcal{G}_t \leftarrow \mathcal{G}_t \cup \{\langle \overline{\mathbf{x}}_t, \Sigma_t, v_t \rangle\}$
    $n \leftarrow 0$
    for $\mathbf{x}_\tau$ in $\widehat{\mathbf{x}}_{0:\tilde{t}}$ **do**
        if $\|\mathbf{x}^{tr}_\tau - \mathbf{x}^{tr}_t\|_2 \leq R \wedge |\tau - t| > S$ **then**
            $\langle \Delta\mathbf{x}_{t,\tau}, \Sigma_{t,\tau} \rangle \leftarrow \text{relative\_measure}(\mathcal{Z}_t, \mathcal{Z}_\tau)$
            $\mathcal{G}_t \leftarrow \mathcal{G}_t \cup \{\langle \Delta\mathbf{x}_{t,\tau}, \Sigma_{t,\tau} \rangle\}$
            $n \leftarrow n + 1$
    if $n \geq N_{loc}$ **then**
        $\widehat{\mathbf{x}}_t \leftarrow \text{estimate\_pose}(\mathcal{G}_t)$
    else
        $\mathbf{x}_{0:t} \leftarrow (\widehat{\mathbf{x}}_{0:\tilde{t}}, \mathbf{x}_t)$
        $\mathcal{G}_{0:t} \leftarrow (\mathcal{G}_{0:\tilde{t}}, \mathcal{G}_t)$
        $\widehat{\mathbf{x}}_{0:t} \leftarrow \text{estimate\_trajectory}(\mathcal{G}_{0:t})$

2) Balancing the Constraints: As already mentioned, the constraints related to the measurements must be balanced with the prior constraints if CAD drawings are metrically inaccurate (see Fig. 5). We can assume that inaccuracies act as bias terms $\boldsymbol{\beta}(\mathbf{x}_{t_i}, \mathbf{x}_{t_j}) \in \mathbb{R}^3$ in the relative measurements. As a consequence, the relative measurement are now distributed as $p(\Delta\mathbf{x}_{t_i,t_j} \mid \mathbf{x}_{0:t}) \sim \mathcal{N}([\![\mathbf{x}_{t_j} \ominus \mathbf{x}_{t_i}]\!] + \boldsymbol{\beta}(\mathbf{x}_{t_i}, \mathbf{x}_{t_j}), \Sigma_{ij})$. Setting $\boldsymbol{\beta}_{ij} \triangleq \boldsymbol{\beta}(\mathbf{x}_{t_i}, \mathbf{x}_{t_j})$ and omitting the robust kernels for simplicity, the MAP estimation becomes

$$\begin{aligned}\widehat{\mathbf{x}}_{0:t} &= \operatorname*{arg\,min}_{\mathbf{x}_{0:t}} \sum_{ij} \|\mathbf{e}_{ij} + \boldsymbol{\beta}_{ij}\|^2_{\Sigma_{ij}} + \sum_k \|\mathbf{e}_k\|^2_{\Sigma_k} \\ &= \operatorname*{arg\,min}_{\mathbf{x}_{0:t}} \sum_i \left[\sum_j \|\mathbf{e}_{ij} + \boldsymbol{\beta}_{ij}\|^2_{\Sigma_{ij}} + \|\mathbf{e}_i\|^2_{\Sigma_i}\right].\end{aligned} \quad (5)$$

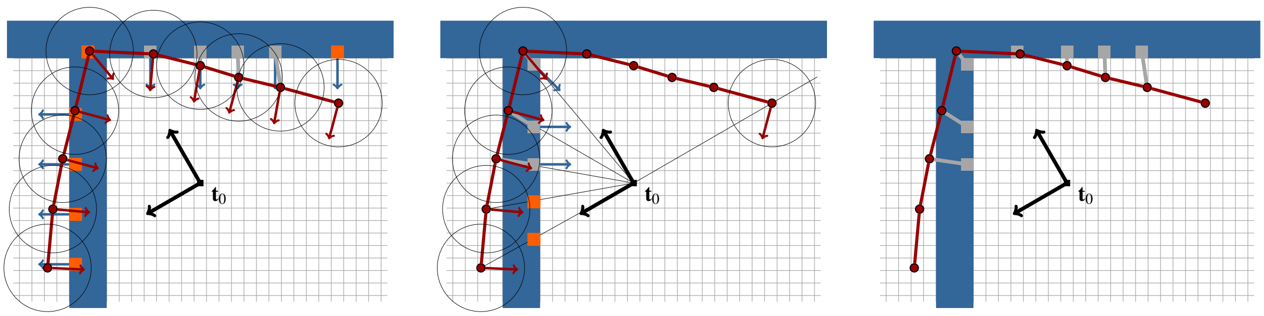

Fig. 2. The construction of the association set (Sec. III-B.1-**2a/2b**). The grid represents the pixel discretization of the CAD drawing (in blue). In gray the accepted associations, in orange the rejected ones. First the candidate $m_s^{\mathbf{a}}$ is selected using the proximity map (left). When the association is rejected, the new candidate $m_s^{\mathbf{b}}$ is obtained via ray-tracing (center). The final associations contains both $m_s^{\mathbf{a}}$ and $m_s^{\mathbf{b}}$ (right).

The summations can be merged as the error terms $\mathbf{e}_k$ are indexed on the vertices of the pose-graph.

As the number of relative measurements increases, the prior term becomes negligible with respect to the inner summation. It is therefore necessary to introduce a weighting term $W_{ij} \in \mathbb{R}^{3\times 3}$ on the information matrix Ω_{ij} that reduces the effect of the bias terms $\boldsymbol{\beta}_{ij}$ in the relative measurements. A natural choice is to multiply the each information matrix by $W_{ij} \triangleq n_i^{-1} I_{3\times 3}$, where n_i is the number of measurements $(\Delta \mathbf{x}_{t_i,t_j})_j$ related to the pose $\mathbf{x}_{t_i}$.

B. Trajectory Priors Using Generalized ICP

As described above, trajectory prior $p(\mathbf{x}_{0:t})$ is modeled as a joint distribution of independent Gaussian terms $p(\mathbf{x}_{t_k})$. The mean $[\![\overline{\mathbf{x}}_{t_k}]\!]$ and covariance Σ_k of each term are computed using scan-to-map-matching. For this, we adapted the GICP framework proposed by Segal *et al.* [7].

We assume a floor plan to be encoded as a binary image $FP : \mathcal{I} \subset \mathbb{N}^2 \longrightarrow \{0, 1\}$ with known resolution $\sigma > 0$, where $\mathcal{I} \triangleq \{1, \dots, H\} \times \{1, \dots, W\}$ is the set of pixels. Having a know resolution is not restrictive as common CAD drawings have metrical annotations and can therefore automatically be exported as images with desired scale. Moreover, we assume the floor plan to be endowed with a reference frame, say $\mathscr{F}_w$, which can be assumed as the world reference frame. Accordingly, we can define $\lfloor \cdot \rfloor_{\sigma,\mathscr{F}_w}$ to be the discretization operator that converts a point in world coordinates into the corresponding pixel on the floor plan image. Similarly, $\lceil \cdot \rceil_{\sigma,\mathscr{F}_w}$ will denote the pseudo-inverse of the discretization operator, which converts a pixel to the corresponding grid point in world coordinates. Given a floor plan image, we define $\Pi_{\mathcal{I}} : \mathcal{I} \longrightarrow \mathcal{I}$ to be the function that associates to every pixel p the closest occupied pixel $\Pi_{\mathcal{I}}(p)$ in the image. We can extend $\Pi_{\mathcal{I}}$ to a map Π defined as $\Pi(x) \triangleq \lceil \Pi_{\mathcal{I}}(\lfloor x \rfloor_{\sigma,\mathscr{F}_w}) \rceil_{\sigma,\mathscr{F}_w}$, for any point $x \in \mathbb{R}^2$ in the floor plan, expressed in world coordinates. Finally, we define the image normal field ν as the function that maps every point on the floor plan to the normalized image gradient of FP at $\lceil p \rceil_{\sigma,\mathscr{F}_w}$ whenever that normalized gradient exists, $\perp$ otherwise. We assume that image normals point towards the free space of the floor plan.

1) The Algorithm: Given an initial guess $\mathbf{t}_0 \in \mathbb{SE}(2)$ and a 2D LiDAR scan as a set of endpoints $\mathcal{Z} \triangleq \{z_s\}_{s=1}^N \subset \mathbb{R}^2$ expressed in the sensor reference frame $\mathscr{F}_l$, repeat for N_{max} iterations the following optimization procedure:

1 – Compute the scan normal field $\nu_{\mathcal{Z}}$ that associates to every endpoint z_s the normal vector $\nu_{\mathcal{Z}}(z_s)$ to the scan curve pointing towards the origin of $\mathscr{F}_l$.

2 – Compute the association set $\mathcal{A} \triangleq \{(z_s, m_s)\}_s \subset \mathcal{Z} \times \mathbb{R}^2$ as follows: given $\theta_{\text{iter}} > 0$ dependent on the current iteration and a candidate association (z_s, m_s), add the association if the following conditions are satisfied

$$\begin{cases} \|\mathbf{t}_0 z_s - m_s\|_2 < \theta_{\text{iter}}, \\ \nu(m_s) \neq \perp \;\text{ and } \nu(m_s)^\top \mathbf{t}_0^R \nu_{\mathcal{Z}}(z_s) \geq 0, \end{cases} \tag{6}$$

that is, the Euclidean distance between the beam endpoint and the corresponding map point does not exceed a threshold value and the related normals are not pointing in opposite directions. The map correspondence candidate m_s for a beam endpoint $\mathbf{t}_0 z_s$ is selected according to the following heuristics (see Fig. 2):

- **2a –** first set $m_s \triangleq m_s^{\mathbf{a}} \triangleq \Pi(\mathbf{t}_0 z_s)$, that is, the closest occupied point on the floor plan,
- **2b –** whenever **2a** fails to satisfy Eq. 6, set $m_i \triangleq m_i^{\mathbf{b}} \triangleq \lceil p(\mathbf{t}_0 z_s) \rceil_{\sigma,\mathscr{F}_w}$, where $p(\mathbf{t}_0 z_s)$ is the pixel obtained by ray-tracing the floor plan image from the LiDAR pose along the beam direction of $\mathbf{t}_0 z_s$.

3 – Update the initial guess with the transformation that optimally aligns the associated points in $\mathcal{A}$. That is,

$$\widehat{\mathbf{t}} \triangleq \operatorname*{arg\,min}_{\mathbf{t} \in \mathbb{SE}(2)} \sum_{(z_s, m_s) \in \mathcal{A}} \rho\left(\|\mathbf{t} z_s - m_s\|_{\Sigma_{s,\mathbf{t}}}\right), \tag{7}$$

where $\rho : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is again a robust kernel and

$$\Sigma_{s,\mathbf{t}} \triangleq R_{m_s} \begin{bmatrix} \eta & 0 \\ 0 & 1 \end{bmatrix} R_{m_s}^\top + R_{\mathbf{t}z_s} \begin{bmatrix} \delta & 0 \\ 0 & 1 \end{bmatrix} R_{\mathbf{t}z_s}^\top. \tag{8}$$

As in the work of Segal *et al.* [7], R_m and $R_{\mathbf{t}z_s}$ are the rotation matrices that align the standard Euclidean basis to the normals $\nu(m_s)$ and $\mathbf{t}^R \nu_{\mathcal{Z}}(z_s)$, which is the normal to the LiDAR endpoint transformed using $\mathbf{t}$.

Finally, $\overline{\mathbf{x}}_{t_k}$ is set to be $\widehat{\mathbf{t}}$ at the last iteration, expressed in the reference frame of the CAD drawing.

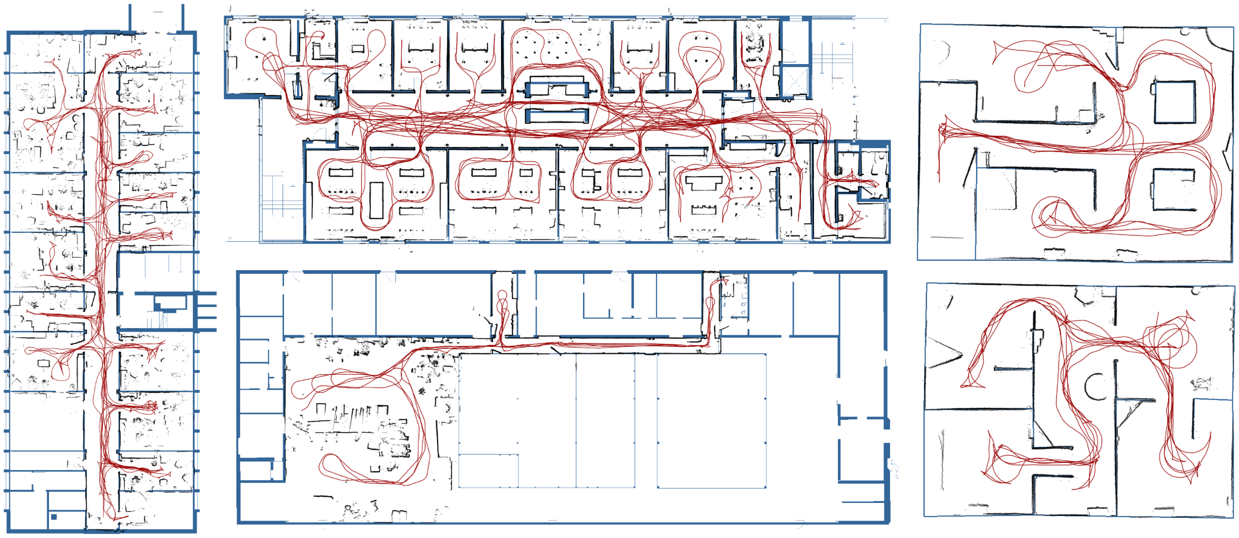

Fig. 3. The datasets used in the experiments. In red the trajectories estimated with our method, in black the map built by the robot to localize in the CAD drawings (blue). Left: *Lab079*. Center-top: *Irc080*. Center-bottom: *Hall078*. Right-top/bottom: *ground truth dataset* (cluttered scenarios). The figures have different scales.

The selection of map candidates for the association set $\mathcal{A}$ is designed to trade-off between efficiency (**2a**) and robustness (**2b**). While the associations obtained via the proximity map can be computed in constant time, the method might result in many rejections, for instance, if the transformed endpoint $\mathbf{t}_0 z_s$ lies on the innermost pixels of thick walls, where the normalized gradient does not exist. Conversely, associations obtained via ray-tracing always have valid normals, at the price of more expensive computations. The combination of the two reduces the depletion of associations while keeping the system efficient.

2) Computing the Information Matrix: Following Grisetti *et al.* [6], we set the covariance matrix Σ_k to be the inverse of the information matrix of the system obtained by first-order linearization of the error function in Eq. 7 at its last iteration. To do so, we observe that, set $\epsilon_s(\mathbf{t}) \triangleq \mathbf{t} z_s - m_s$, the following equations hold:

$$\|\epsilon_s(\mathbf{t})\|^2_{\Sigma_{s,\mathbf{t}}} = \|L_{s,\mathbf{t}}^\top \epsilon_s(\mathbf{t})\|_2^2 = e_s(\mathbf{t})^\top e_s(\mathbf{t}), \qquad (9)$$

where $L_{s,\mathbf{t}} L_{s,\mathbf{t}}^\top$ is the Cholesky decomposition of the information $\Omega_{s,\mathbf{t}}$ related to the covariance matrix in Eq. 8 and $e_s(\mathbf{t}) \triangleq L_{s,\mathbf{t}}^\top \epsilon_s(\mathbf{t})$. As a consequence, the cost function in Eq. 7 can be linearized as described in [6] since the information matrices weighting the error terms $e_s(\mathbf{t})$ are now independent from the optimization variables, in fact, identity matrices. Setting $E_s(\mathbf{t}) \triangleq \|e_s(\mathbf{t})\|_2^{-1} e_s(\mathbf{t}) \sqrt{\rho(\|e_s(\mathbf{t})\|_2)}$, the information matrix simplifies to

$$\Omega_k \triangleq \sum_s \left[\frac{\partial E_s}{\partial \boldsymbol{\delta}}(\widehat{\mathbf{t}} \oplus \boldsymbol{\delta})\Big|_{\boldsymbol{\delta}=\mathbf{0}} \right]^\top \left[\frac{\partial E_s}{\partial \boldsymbol{\delta}}(\widehat{\mathbf{t}} \oplus \boldsymbol{\delta})\Big|_{\boldsymbol{\delta}=\mathbf{0}} \right], \qquad (10)$$

where $\frac{\partial \cdot}{\partial \boldsymbol{\delta}}$ is the Jacobian operator with respect of $\boldsymbol{\delta} \in \mathbb{SE}(2)$.

C. Role of the Initial Guess

The MAP estimation at Eq. 1 requires the priors $p(\mathbf{x}_{t_k})$ to be independent of $(\Delta\mathbf{x}_{t_i,t_j})_{ij}$ for any t_k. Since the initial guess at t_k is chosen as $\widehat{\mathbf{x}}_{t_{k-1}} \oplus \Delta\mathbf{x}_{t_{k-1},t_k}$, $\overline{\mathbf{x}}_k$ depends on $\widehat{\mathbf{x}}_{0:t_{k-1}}$, that is, on the previous relative measurements. However, assuming a good initial guess, GICP converges to the same estimate for small perturbations of $(\Delta\mathbf{x}_{t_i,t_j})_{ij}$, hence $\overline{\mathbf{x}}_k$ is (locally) independent of $(\Delta\mathbf{x}_{t_i,t_j})_{ij}$. Thus, $p(\mathbf{x}_{0:t})$ defined as joint distribution of independent Gaussian terms obtained using scan-to-map-matching is a valid trajectory prior. In Eq. 3, $p(\mathbf{x}_t)$ never depends on $(\Delta\mathbf{x}_{t,t_j})_j$ for any initial guess, therefore, it is again a valid prior.

IV. Experimental Evaluation

Since no CAD floor plan is available for almost any public 2D LiDAR-based SLAM dataset, we evaluated our localization approach using seven datasets recorded in different buildings at the campus of the University of Freiburg. We collected the data by teleoperating a Pioneer® P3-DX equipped with SICK® S300 Professional laser rangefinder with $270°$ of field of view, 540 beams and $30.0\,$m range. For this, we collected two groups of datasets (see Fig. 3):

- *Ground truth datasets*: Four 10 minutes long datasets recorded in two environments created in our lab using movable panels, each with and without clutter. We used a motion capture system with ten Raptor-E cameras to precisely track the ground truth pose of the robot. We used the same motion tracker to measure the environment and create a floor plan drawing accordingly.
- *Performance datasets*: Three datasets, named *Hall078*, *Lab079* and *Irc080*, recorded in real buildings and respectively 15, 35 and 60 minutes long.

We used only the *ground truth datasets* to assess the accuracy of the method since no ground truth is available for the *performance datasets*. We exploited the latter to benchmark the runtime performance of our system in longer runs. The quantitative evaluation of our results are reported in Sec. IV-C and Sec. IV-D.

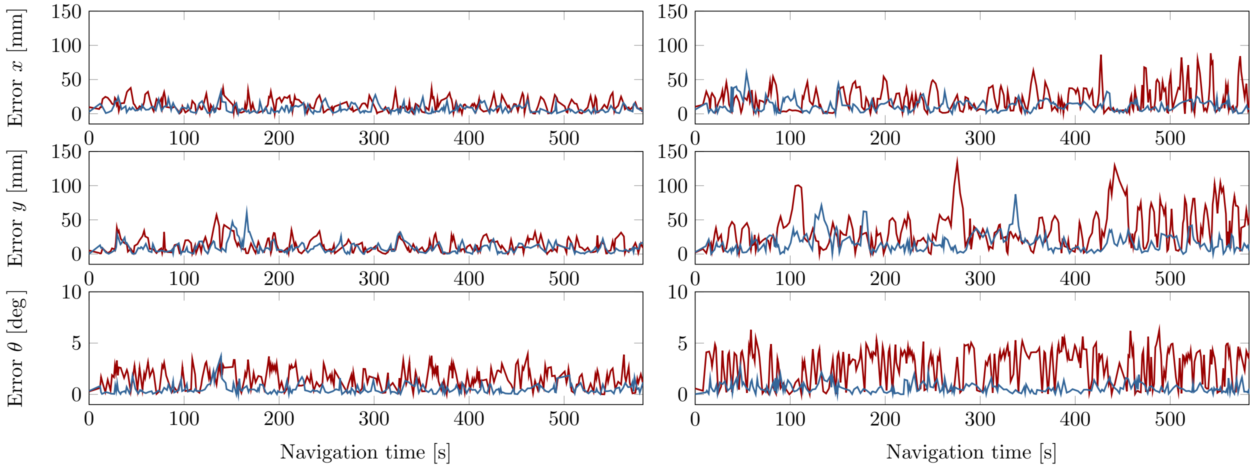

Fig. 4. Errors over time for the two *ground truth datasets* (left and right). In blue the uncluttered settings, in red the cluttered ones.

A. Implementation Details and Parameter Selection

For the optimizations in Eq. 2, 4, and 7 we used the implementation of the Levenberg-Marquardt algorithm provided by the g^2o framework [20]. We utilized ICP with point-to-line metric from the *C(anonical) Scan Matcher* by Censi [21] to estimate the relative measurements $(\Delta\mathbf{x}_{t_i,t_j})_{ij}$, both for loop closures and incremental scan matching, and our implementation of GICP-based scan-to-map-matching.

The parameters used for the evaluation have been empirically selected and are the same for all the experiments. Referring to the notation used in Sec. III-A and Algorithm 1, we set $d = 1\,\mathrm{m}$, $\alpha = 0.5\,\mathrm{rad}$, $R = 1.5\,\mathrm{m}$, $S = 10$, $N_{loc} = 10$, and Huber kernels both for ρ_m and ρ_p. To compute the prior for the trajectory, we exported the floor plans with resolution $\sigma = 10\,\mathrm{mm/px}$ and used 3×3 Scharr kernels to calculate the image normal field ν. We chose $N_{max} = 10$, $\theta_{\text{iter}} = \frac{0.25}{\sqrt{\text{iter}}}\mathrm{m}$, $\eta = 0.05$, $\delta = 0.05$, and Huber kernels for the optimization in Eq. 7.

B. Robustness

No significant failure has been encountered for any dataset. From a qualitative standpoint, the robot was always localized during navigation with only few situations where the localization was poor, mainly due to wrong associations in the scan-to-map-matcher. Failures happened only temporarily when a portion of the environment was newly observed. No errors appeared after the map was corrected by optimizing the map with new loop closures. According to the experiments, neither small metrical inconsistencies of the CAD drawing (see Fig. 5), nor significant or even full occlusion due to clutter or large installations have a substantial effect on the method. We compared our method with saturated likelihood field MCL. We fine-tuned the parameters of our implementation of MCL (250 mm saturation and 5000 particles) to be able to track the robot even in a highly occluded scenario. A comparison of our system with the best run of MCL is shown in Fig. 1. The results of our method are substantially more consistent with the floor plan.

MCL performs poorly since CAD drawings can significantly differ from the observations due to not represented structures. Consequently, the particle weighting does not necessarily peak around the true robot pose. This results in a wrong pose estimate, particularly when the proposal distribution has a high covariance. However, choosing a distribution with low covariance reduces the capability of MCL to recover from failures. In contrast, although our method can also be temporarily affected by a bad initial guess for computing the trajectory prior, the approach benefits from the online built pose-graph to retrieve a good estimate of the current pose, making the system robust to failures. In essence, the role of the optimization in Eq. 1 is to provide a good initial guess for the prior estimation.

C. Localization Accuracy

The results of the experiments on the *ground truth datasets* are shown in Fig. 4. In the best scenario the robot localized with an average error of (7.5 ± 5.8) mm and (9.5 ± 8.5) mm along the x and y axes respectively and a yaw average error of $(0.52 \pm 0.47)^\circ$. As expected, the presence of unmapped clutter reduces the performance of the system. In the worst case we obtained an average error of (22.8 ± 18.7) mm in x, (35.3 ± 26.4) mm in y and $(2.90 \pm 1.73)^\circ$ in yaw. We compared the above results with MCL on the same maps. The best run outperformed MCL, which localized at its best with (21.8 ± 24.0) mm error in x, (33.9 ± 26.1) mm in y

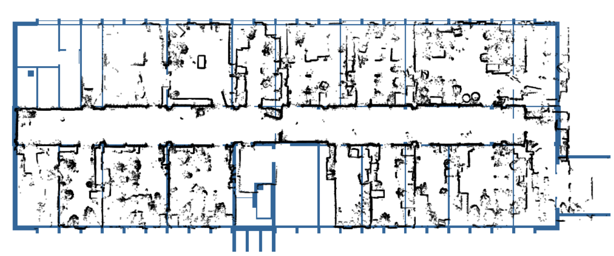

Fig. 5. Metric inaccuracy of the floor plan for *Lab079*. The SLAM map (black) shows that the CAD is 0.6 m shorter than the real world.

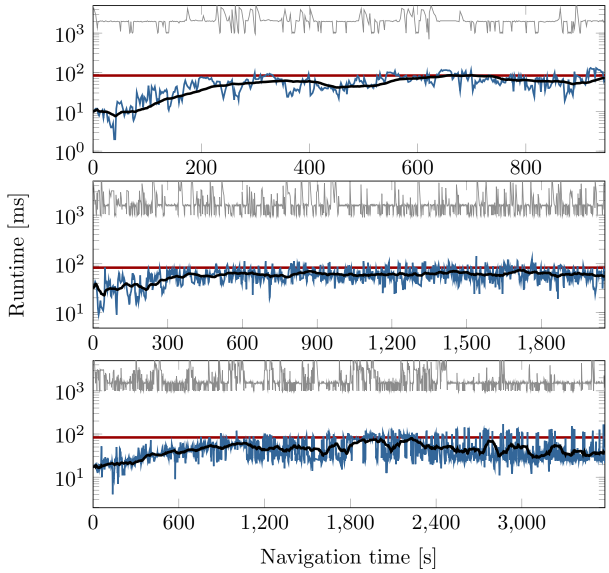

Fig. 6. Runtime performance for *Hall078* (top), *Lab079* (center) and *Irc080* (bottom). The runtime of single updates (blue) is less than the available time between two consecutive updates (gray). The moving average of the runtime (black) is below the scan time (red), which is 83 ms/scan.

and $(2.44 \pm 18.70)^\circ$ in yaw, showing that the accuracy of our method is comparable with that of MCL.

D. Runtime Performance

The experiments were run on an 8-core 4.0 GHz Intel® Core™ i7 CPU. The moving average of the system runtime is bounded even for long navigation runs, which shows that our approach is suitable for long-term applications. The elapsed time at each update step for *Hall078*, *Lab079* and *Irc080* is shown in Fig. 6. The average runtimes over each entire experiment were (54 ± 27) ms, (58 ± 19) ms, and (44 ± 26) ms respectively, showing that the system is on average comparable to standard likelihood field MCL ((52 ± 8) ms in our implementation with same parameters of Sec. IV-C) and usable in near real-time.

V. Conclusions

In this paper we presented a LiDAR-based system for localization in architectural floor plans. While the robot is moving through its environment, we use a SLAM method to online generate a map, represented as a pose-graph with LiDAR readings. The priors for the trajectory are computed with the proposed GICP-based scan-to-map-matcher to fit the generated pose-graph onto the floor plan. When the map sufficiently covers the area in the vicinity of the robot, we estimate its relative pose with respect to the matching nodes of the pose-graph without a global optimization process. This combination makes the system robust to missing information in CAD drawings and also computationally efficient. We evaluated our approach in several real-world scenarios and showed that the method works robustly in complex environments and is as accurate and efficient as common state-of-the-art localization systems.

References

[1] H. Mäkelä and T. von Numers, "Development of a navigation and control system for an autonomous outdoor vehicle in a steel plant," *Control Engineering Practice*, vol. 9, no. 5, pp. 573–583, 2001.

[2] S.-Y. Lee and H.-W. Yang, "Navigation of automated guided vehicles using magnet spot guidance method," *Robotics and Computer-Integrated Manufacturing*, vol. 28, no. 3, pp. 425–436, 2012.

[3] J. Roewekaemper, C. Sprunk, G. D. Tipaldi, C. Stachniss, P. Pfaff, and W. Burgard, "On the position accuracy of mobile robot localization based on particle filters combined with scan matching," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Villamoura, Protugal, 2012, pp. 3158–3164.

[4] C. Sprunk, B. Lau, P. Pfaff, and W. Burgard, "An accurate and efficient navigation system for omnidirectional robots in industrial environments," *Autonomous Robots*, pp. 1–21, 2016.

[5] L. Bowen-Biggs, S. Dazo, Y. Zhang, A. Hubers, M. Rueben, R. Sowell, W. D. Smart, and C. M. Grimm, *A Method for establishing correspondences between hand-drawn and sensor-generated maps*. Springer International Publishing, 2016, pp. 1003–1013.

[6] G. Grisetti, R. Kümmerle, C. Stachniss, and W. Burgard, "A tutorial on graph-based SLAM," *IEEE Intelligent Transportation Systems Magazine (ITS)*, vol. 2, no. 4, pp. 31–43, 2010.

[7] A. Segal, D. Haehnel, and S. Thrun, "Generalized-ICP," in *Proc. of Robotics: Science and Systems (RSS)*, vol. 2, 2009.

[8] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. MIT Press, 2005.

[9] A. Martinelli, F. Pont, and R. Siegwart, "Multi-robot localization using relative observations," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2005, pp. 2797–2802.

[10] A. Howard, "Multi-robot mapping using manifold representations," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, vol. 4. IEEE, 2004, pp. 4198–4203.

[11] C. Sprunk, G. D. Tipaldi, A. Cherubini, and W. Burgard, "Lidar-based teach-and-repeat of mobile robot trajectories," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Tokyo, Japan, Nov. 2013, pp. 3144–3149.

[12] M. Mazuran, F. Boniardi, W. Burgard, and G. D. Tipaldi, "Relative topometric localization in globally inconsistent maps," in *Proc. of the International Symposium of Robotics Research (ISRR)*, 2015.

[13] R. C. Luo, Y.-C. Lin, and C.-C. Kao, "Autonomous mobile robot navigation and localization based on floor plan map information and sensory fusion approach," in *Proc. of the IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems*. IEEE, 2010, pp. 121–126.

[14] S. Ito, F. Endres, M. Kuderer, G. D. Tipaldi, C. Stachniss, and W. Burgard, "W-rgb-d: floor-plan-based indoor global localization using a depth camera and wifi," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 417–422.

[15] W. Winterhalter, F. Fleckenstein, B. Steder, L. Spinello, and W. Burgard, "Accurate indoor localization for RGB-D smartphones and tablets given 2D floor plans," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015, pp. 3138–3143.

[16] H. Hile and G. Borriello, "Positioning and orientation in indoor environments using camera phones," *IEEE Computer Graphics and Applications (CG&A)*, vol. 28, no. 4, 2008.

[17] O. Vysotska and C. Stachniss, "Exploiting building information from publicly available maps in graph-based SLAM," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 4511–4516.

[18] N. Sünderhauf and P. Protzel, "Towards a robust back-end for pose graph SLAM," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2012, pp. 1254–1261.

[19] R. Kümmerle, B. Steder, C. Dornhege, A. Kleiner, G. Grisetti, and W. Burgard, "Large scale graph-based SLAM using aerial images as prior information," *Autonomous Robots*, vol. 30, no. 1, pp. 25–39, 2011.

[20] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g^2o: A general framework for graph optimization," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2011, pp. 3607–3613.

[21] A. Censi, "An ICP variant using a point-to-line metric," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2008.