IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

Waliner: Lightweight and Resilient Plugin Mapping Method With Wall Features for Visually Challenging Indoor Environments

DongKi Noh[1], *Member, IEEE*, Byunguk Lee[2], Hanngyoo Kim[2], SeungHwan Lee[2*], *Member, IEEE*, HyunSung Kim[1], JuWon Kim[2], Jeongsik Choi[1], and SeungMin Baek[1]

***Abstract*— Vision-based indoor navigation systems have been proposed previously for service robots. However, in real-world scenarios, many of these approaches remain vulnerable to visually challenging environments such as white walls. In-home service robots, which are mass-produced, require affordable sensors and processors. Therefore, this paper presents a lightweight and resilient plugin mapping method called *Waliner*, using an RGB-D sensor and an embedded processor equipped with a neural processing unit (NPU). *Waliner* can be easily implemented in existing algorithms and enhances the accuracy and robustness of 2D/3D mapping in visually challenging environments with minimal computational overhead by leveraging a) structural building components, such as walls; b) the Manhattan world assumption; and c) an extended Kalman filter-based pose estimation and map management technique to maintain reliable mapping performance under varying lighting and featureless conditions. As verified in various real-world in-home scenes, the proposed method yields over a 5 % improvement in mapping consistency as measured by the map similarity index (MSI) while using minimal resources.**

***Index Terms*— Manhattan world assumptions, building components, line measurements, and mapping.**

I. Introduction

Vision-based simultaneous localization and mapping (SLAM) is a fundamental problem in robotics, where a robot estimates its position and generates a map of its environment using a camera sensor. Traditional visual SLAM algorithms often struggle in indoor environments with ambiguous visual features, such as blank walls, as shown in Fig. 1(a). These ambiguous features hinder feature extraction and matching, which are critical steps in SLAM (see Fig. 1(b)). However, even in indoor environments with ambiguous visual features, structural regularities such as the Manhattan world assumption (MWA) [1]–[8] can serve as strong constraints to enhance the accuracy and robustness of the existing visual SLAM algorithms [9]–[13] (see Fig. 1(b)). Prior studies [11]–[13] address ambiguous environments such as long corridors by utilizing line features. However, conventional line-feature extraction methods often incur additional computational costs and generate unreliable line segments, especially from moving objects. In some cases [14], other sensor information such as LiDAR data has also been incorporated. To overcome these limitations and enhance the robustness in visually challenging in-home environments, we propose a novel line-based odometry and refinement technique that leverages semantic information using an RGB-D sensor without additional sensors (e.g., LiDAR sensors). Our method applies semantic segmentation to identify meaningful objects and surfaces in a scene and extracts lines from these identified features to enable robust pose estimation and mapping.

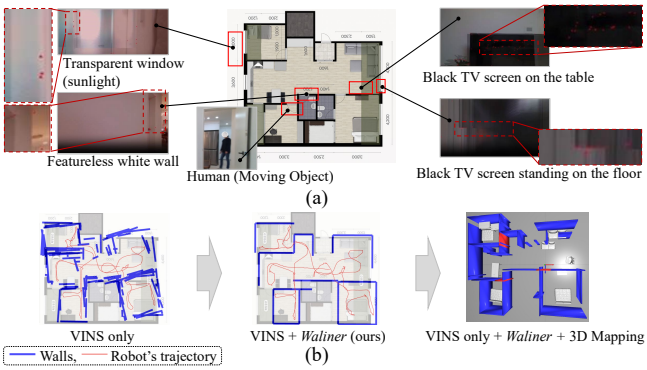


Fig. 1. Visually challenging environments and qualitative performance comparison: (a) Room plan serving as the ground truth with images captured by a robot. (b) Mapping results obtained using VINS [9] and our proposed method, along with a 3D map for environment visualization. Red dots represent visual features extracted from images captured by a robot.

This paper presents several key contributions to embedded SLAM systems operated on resource-constrained platforms, which are particularly applicable to commercial service robots functioning in complex indoor environments.

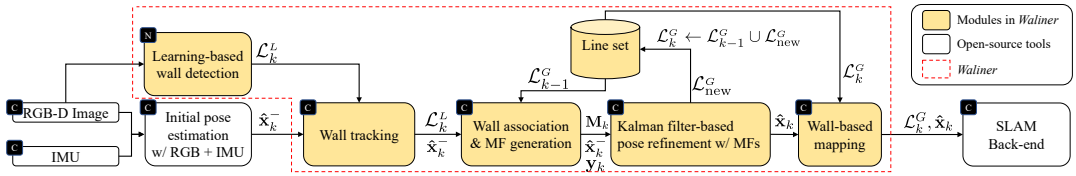
- We introduce a novel line extraction algorithm with semantic segmentation for real-time SLAM on resource-constrained platforms such as the Cortex-A9 using a vision sensor. This algorithm leverages specific building components, such as walls, to ensure efficient feature extraction and mapping.
- With semantic line features extracted from walls, we propose a lightweight odometry refinement algorithm to enhance the robustness. This algorithm is applied to robust pose estimation, map management, and a loop closure technique that significantly improves mapping performance in visually challenging environments with limited features. Moreover, it can be easily implemented into existing SLAM frameworks such as RTAB-MAP.

The remainder of this paper is organized as follows: Section II introduces related SLAM algorithms from the perspective of MWA. Section III provides an overview of the proposed SLAM framework. Section IV describes the method of line extraction to generate Manhattan frame (MF). Sections V and VI describe a priori and posteriori of the pose estimation processes, respectively. Sections VII and VIII describe the experimental setups and results. Finally, Section IX

*Corresponding author: SeungHwan Lee.

[1]DongKi Noh, HyunSung Kim, Jeongsik Choi, and SeungMin Baek are with the Advanced Robotics Lab., CTO Division, LG Electronics Inc., Seoul, 07796, Republic of Korea (e-mail: {dongki.noh, hs9767.kim, jeongs.choi, seungmin2.baek}@lge.com).

[2]Byunguk Lee, Hanngyoo Kim, SeungHwan Lee, and JuWon Kim are with the School of Electronic Engineering at KIT (Kumoh National Institute of Technology), Gumi, 39177, Republic of Korea (e-mail: {qud159, rlarb100, leesh}@kumoh.ac.kr; lambertkim@naver.com).



C: Embedded CPU equipped with 2GB memory (Cortex-A53, similar to Cortex-A9) N: Embedded neural processing unit (NPU, NXP i.MX 8M Plus, 2.3 TOPS) enabling parallel processing independent of the CPU

Fig. 2. Overall SLAM pipeline with the proposed method, *Waliner*: The colored box represents our proposed method, which aims to refine the initial pose $\hat{\mathbf{x}}_k$ using the Manhattan frame (MF) $\mathbf{M}_k$ generated from lines $\mathcal{L}$ extracted from walls using a deep learning-based method and a neural processing unit (NPU). Additionally, we leverage VINS [9] and RTAB-MAP [15] for the initial pose $\hat{\mathbf{x}}_k$ estimation at the k-th step and 3D mapping, respectively. The refined pose enhances the accuracy of the SLAM process, particularly in environments with challenging visual features.

presents our findings and conclusions.

II. Related Works

The methods for leveraging strong structural constraints are generally categorized into two approaches: geometry-based and learning-based. Researchers have explored various techniques within these categories to enhance the SLAM performance.

A. Geometry-based Approach

First, Wu *et al.* [2] proposed a 3D LiDAR-based SLAM algorithm that utilizes MWA using planar constraints. Joo *et al.* [7] introduced a SLAM framework based on the Atlanta world assumption, which is more general than MWA. Yunus *et al.* [3] proposed a Manhattan SLAM approach for robust tracking in both MF and non-MF environments. Li *et al.* [4] decoupled rotation and translation estimation using MWA, first estimating the rotation of the robot and then deriving the translation based on this estimation. Kim *et al.* [5] exploited vertical dominant directions (2D) to efficiently extract a 3D MF by leveraging Manhattan structure's 90° periodicity. Kim *et al.* [6] introduced a method for line estimation using UAVs, grouping 2D projected LiDAR data using a hierarchical clustering technique [8] and RANSAC. Li *et al.* [1] proposed the Hong Kong world, which combines multiple Manhattan worlds for outdoor environments by integrating sloped structures. Recently, a novel SLAM approach that leverages LiDAR data and geometric constraints derived from MWA was proposed [14]. This work is closely related to our approach, as it serves as a plug-in module designed to enhance pose accuracy. Their method, called linear four-point LiDAR SLAM, introduced an efficient algorithm for pose estimation using only four LiDAR points, significantly reducing the computational overhead.

However, for in-home (2D) service robots, adding LiDAR sensors is neither cost-effective nor mechanically feasible. Moreover, point-based line and planar features used in previous methods may be ambiguous or redundant due to various planar objects (e.g., refrigerators). In contrast, our approach extracts lines oriented perpendicularly to gravity from only the regions identified by deep learning-based wall masks.

B. Learning-Based Approach

Mahmoud *et al.* [11] introduced a visual SLAM technique that integrated semantic segmentation and layout estimation. The authors employed RoomNet [16] for semantic segmentation and layout estimation, which helped correct drift errors in the SLAM system and aligned the tracking mechanism more closely with natural human movement patterns. However, because it utilizes an RNN-like structure, it requires a large amount of memory. Therefore, it is not suitable for embedded systems. Yue *et al.* [17] at ETH Zurich explored room reconstruction using a deep learning architecture based on two-level queries, each focusing on polygonal structures and their corner points, whereas it requires high quality 3D point cloud data for high quality mapping. Several previous studies [18], [19] have used semantic segmentation to predict the intersection between walls and the floor. Li *et al.* [20] investigated deep learning-based line feature detection across various camera models, including pinhole, fisheye, and spherical types. However, it is limited to detecting only straight lines. In contrast, our approach is designed with a lightweight backbone using ESANet [21] and is less dependent on high-quality depth data, making it more suitable for embedded systems.

III. Overview of *Waliner*: Wall Line-Based Odometry Refinement

We propose a lightweight odometry refinement method that incorporates semantic features for accurate mapping, as shown in Fig. 2. This method is designed for use in embedded systems and deployment of commercial service robots. In particular, this method focuses exclusively on measurements of structured environments such as walls and doors, unlike previous approaches that integrate measurements with building information models [22], [23]. Our method is called *Waliner*, a combination of the words ***wa***ll ***li***nes and ***align***ment. *Waliner* was developed and validated at the component level for integration into existing SLAM frameworks. *Waliner* uses several open-source tools, including VINS [9]-based visual odometry for initial pose estimation and RTAB-MAP [15] as the SLAM back-end to generate a 3D map, as shown in Fig. 2. Because RTAB-MAP supports various optimization methods, such as g2o and GTSAM, it is well-suited for evaluating the flexibility and compatibility of our plugin-based approach with different optimization backends (see our GitHub repository).

The mapping process with *Waliner* is briefly described as follows. A learning-based wall detector processes RGB-D sensor data, and VINS provides the initial odometry. Subsequently, walls are detected from the RGB-D data and projected onto the ground as 2D line sets $\mathbf{L}^L_{k,n}$ at the k-th time step, where L and n denote local coordinates and the n-th line in the set, respectively. The odometry and projected lines are synchronized in terms of the time step.

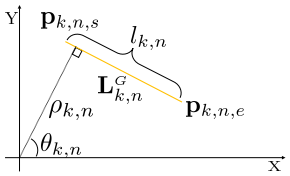


Fig. 3. Line representation in Cartesian coordinates: A line is characterized by its length $l_{k,n}$, distance $\rho_{k,n}$ from the origin, angle $\theta_{k,n}$, and endpoints $\mathbf{p}_{k,n,s}$ and $\mathbf{p}_{k,n,e}$. This information is crucial for defining wall structures in the proposed SLAM method.

These synchronized data (e.g., pose and lines) are used as the inputs for *Waliner*'s pose refinement module. By leveraging the line features generated from building structures, *Waliner* utilizes a strong assumption related to MWA [1]–[8], [20]. Using MWA, *Waliner* generates Manhattan frames (MFs) based on wall-based lines. The k-th refined robot pose $\hat{\mathbf{x}}_k = (\hat{x}_k, \hat{y}_k, \hat{\theta}_k) \in \mathrm{SE}(2)$ is estimated using Kalman filter-based pose refinement. Based on the wall lines and refined poses, a wall-based 2D map is generated. In addition, RTAB-MAP is used to generate a 3D map, such as the OctoMap.

IV. Manhattan Frame Generation in *Waliner*

We propose a Manhattan frame-based Kalman filter SLAM for pose estimation, building upon MWA approaches [1]. We leverage VINS [9] for initial pose estimation without MWA assumptions. *Waliner* generates line set $\mathcal{L}^G_k = \{\mathbf{L}^G_{k,1}, ..., \mathbf{L}^G_{k,n}\}$, where n is the number of lines at the current step k and superscript G denotes the global coordinate system.

A. Notation

The n-th line $\mathbf{L}^G_{k,n}$ in the k-th line set $\mathcal{L}^G_k$ in the global coordinate is defined as follows:

- $\mathbf{L}^G_{k,n} = [\rho^G_{k,n}, \theta^G_{k,n}, l^G_{k,n}, \mathbf{p}^G_{k,n,s}, \mathbf{p}^G_{k,n,e}, \mathrm{ID}^G_{k,n}, w^G_{k,n}]$,
- $\mathbf{p}^G_{k,n,s} = [x^G_{k,n,s}, y^G_{k,n,s}]$, $\mathbf{p}^G_{k,n,e} = [x^G_{k,n,e}, y^G_{k,n,e}]$,

where $w_{k,n}$ and $\mathrm{ID}^G_{k,n}$ denote the weight on the n-th line $\mathbf{L}^G_{k,n}$ and corresponding identifier, respectively; the other notations are illustrated in Fig. 3. The k-th MF $\mathbf{M}_k$ in the local coordinates is expressed by the angle $\theta_{\mathrm{MF},k}$ as follows:

$$\mathbf{M}_k = \left[\theta_{\mathrm{MF},k}\ \theta_{\mathrm{MF},k} + \tfrac{\pi}{2}\right], \tag{1}$$

where the subscript MF denotes the Manhattan frame.

B. Wall Detection-based Line Feature Extraction

Under the MWA, we generate a set of lines $\mathcal{L}^G_k$ representing walls projected onto the ground at each time step k. The procedure for obtaining reliable lines is described in detail below. Initially, learning-based semantic segmentation is applied to identify areas recognized as walls within the input image. For this purpose, we employ the pre-trained ESANet [21], which is designed for both high segmentation performance and low inference time. The network produces a mask $\mathbf{M}$, where the areas identified as walls are labeled as $\mathbf{1}$, and all other areas are labeled as $\mathbf{0}$. Using this mask, *Waliner* focuses only on walls and their position vectors, excluding non-wall areas. This process is represented by the following equation:

$$\mathbf{W} = f_b(\mathbf{I} \otimes \mathbf{M}, \mathbf{D}, c_x, c_y, f_x, f_y), \tag{2}$$

where $\mathbf{W}$ is a masked depth map in the form of a 2D array, matching the size of the input image, which contains the wall information as a position vector extracted from the front-facing data of the robot's perspective; f_b is a backward projection function; $\mathbf{I}$ and $\mathbf{D}$ are the input RGB and depth images, respectively; $\otimes$ denotes the Hadamard product; and c_x, c_y, f_x, and f_y are the intrinsic parameters of the RGB camera.

To determine a real wall among the areas detected as walls, we employ two assumptions: first, walls are generally parallel to the direction of gravity, and second, if a wall exists within the input image, it typically occupies the deepest position in each column of $\mathbf{W}$. Based on these assumptions, we infer that among the multiple point clusters within a single column of $\mathbf{W}$, the cluster aligned with gravity and exhibiting the maximum depth is most likely to represent the wall. Thus, to identify point clusters distributed similarly to the direction of gravity, we initially group points within a column of $\mathbf{W}$ into clusters that satisfy the following simple criteria:

$$0 \leq (\mathbf{p}_{1,c_i} - \mathbf{p}_{2,c_i}) \cdot \mathbf{g} \leq l^v_{\mathrm{TH}}, \tag{3}$$

$$0 \leq \frac{(\mathbf{p}_{1,c_i} - \mathbf{p}_{2,c_i}) \cdot \mathbf{g}}{|\mathbf{p}_{1,c_i} - \mathbf{p}_{2,c_i}|} \leq \theta^v_{\mathrm{TH}}, \tag{4}$$

where $\mathbf{p}_{1,c_i}$ and $\mathbf{p}_{2,c_i}$ are the 3D position vectors of two different points in the same column c_i of $\mathbf{W}$; $\mathbf{g}$ is the unit vector of gravity calculated from the inertial measurement unit (IMU); l^v_{TH} and θ^v_{TH} are the thresholds for distance and angle along the vertical direction, respectively. We select the cluster with the greatest depth among all clusters. Then, we compute the average position vector of its points and project it onto the ground plane as $\mathbf{p}_{p,c_i}$ by removing the height component. This process is repeated for each column in $\mathbf{W}$, selecting the deepest cluster in each column and projecting it onto the ground, resulting in at most one projected point $\mathbf{p}_{p,c_i}$ per column, as shown in Fig. 4(a). After identifying wall locations $\mathbf{p}_{p,c_i}$ for each column, the next step is to cluster the projected points into planar surfaces corresponding to the same wall. Suppose there are four projected points $\mathbf{p}_{p,c_1}$, $\mathbf{p}_{p,c_2}$, $\mathbf{p}_{p,c_3}$, and $\mathbf{p}_{p,c_4}$, as shown in Fig. 4(b). The projected

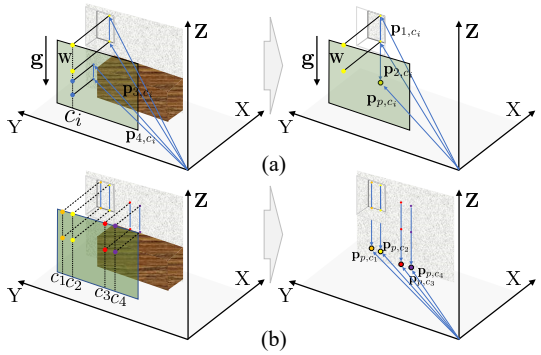


Fig. 4. Illustration of the wall detection and grouping process: **(a)** The method identifies walls by projecting wall information onto the ground using depth data and the gravity vector from the inertial measurement unit (IMU). The algorithm selects the deepest point in each column as the most likely wall candidate, demonstrating its accuracy in detecting vertical surfaces. **(b)** Projected points on the ground that represent different walls are clustered. Points with similar depths are grouped to form continuous planar surfaces, allowing for the precise reconstruction of wall locations in the environment.

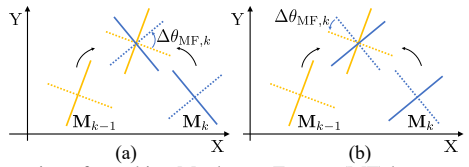


Fig. 5. Illustration of matching Manhattan Frames (MF) between consecutive time steps. The rotation angle $\Delta\theta_{\mathrm{MF},k}$ is determined from two possible alignments of MFs. The alignment with the smaller absolute value of $\Delta\theta_{\mathrm{MF},k}$ is selected to minimize pose estimation errors.

points are grouped into clusters according to the following criteria:

$$|\mathbf{p}_{p,c_1} - \mathbf{p}_{p,c_2}| \leq l_{\mathrm{TH}}^{\mathrm{GP}},\ |\mathbf{p}_{p,c_2} - \mathbf{p}_{p,c_3}| \leq l_{\mathrm{TH}}^{\mathrm{GP}}, \tag{5}$$

$$0 \leq \frac{(\mathbf{p}_{p,c_1} - \mathbf{p}_{p,c_2}) \cdot (\mathbf{p}_{p,c_2} - \mathbf{p}_{p,c_3})}{|\mathbf{p}_{p,c_1} - \mathbf{p}_{p,c_2}||\mathbf{p}_{p,c_2} - \mathbf{p}_{p,c_3}|} \leq \theta_{\mathrm{TH}}^{\mathrm{GP}}, \tag{6}$$

where $l_{\mathrm{TH}}^{\mathrm{GP}}$ and $\theta_{\mathrm{TH}}^{\mathrm{GP}}$ are the thresholds for distance and angle on the ground plane, respectively. By satisfying the criteria (5) and (6), we can group the projected points that are close to each other and aligned in a straight line, which is a characteristic feature of walls. Finally, the parameters representing the shape of each line are derived through the least squares method using the projected points within each cluster. The start and end points $\mathbf{p}_{k,n,s}$ and $\mathbf{p}_{k,n,e}$ of a line are defined by dropping perpendicular lines from the rightmost and leftmost projected points, respectively, within the associated cluster.

C. Manhattan Frame Generation and Matching

The MF generated by MWA is used to obtain the orientation changes between consecutive MFs via MF matching. To generate a MF, we first select a line with high confidence. In this study, we assume a strong correlation between line confidence and line length. Longer lines consist of a greater number of closely spaced samples, which enhances their reliability. A selected line $\mathbf{L}_{\mathrm{MF},k}$ at the k-th step of MF is as follows:

$$\mathbf{L}_{\mathrm{MF},k} = \underset{\mathbf{L}_{k,i} \in \mathcal{L}_k}{\operatorname{argmin}} \sigma^2(\mathbf{L}_{k,i}) \text{ s.t. } \sigma(\mathbf{L}_{k,i}) \propto \frac{1}{l_{k,i}}, \tag{7}$$

where $\sigma(\cdot)$ denotes the confidence measurement function.

Accordingly, the k-th MF is given by $\mathbf{M}_k$ generated by the angle $\theta_{\mathrm{MF},k}$ of the selected line $\mathbf{L}_{\mathrm{MF},k}$. This is matched with the previous MF $\mathbf{M}_{k-1}$. The result of this matching is the rotation angle $\Delta\theta_{\mathrm{MF},k}$. Because the MF is represented by a two-dimensional vector with two lines, $\Delta\theta_{\mathrm{MF},k}$ can assume two possible values, as shown in Fig. 5. Given that $\Delta\theta_{\mathrm{MF},k}$ between consecutive frames does not vary significantly, *i.e.*, the angle change is typically small, the value with the smaller absolute magnitude is selected as $\Delta\theta_{\mathrm{MF},k}$.

V. State and Measurement Models in *Waliner*

The state and measurement models for robot pose estimation are defined using the k-th robot pose $\mathbf{x}_k = [x_k\ y_k\ \theta_k]^{\mathrm{T}}$, odometry $\mathbf{o}_k$ estimated by VINS, and the line set $\mathcal{L}_{k-1}$ obtained by the learning-based detector as follows:

$$\mathbf{x}_{k|k-1} = f(\mathbf{x}_{k-1}, \mathbf{o}_k) + \mathbf{w}_k,\ \mathbf{w}_k \sim N(0, \mathbf{R}_k), \tag{8}$$

$$\mathbf{z}_k = \begin{bmatrix} \mathbf{z}_{k,1} \\ \vdots \\ \mathbf{z}_{k,N} \end{bmatrix} = h(\mathbf{x}_k, \mathcal{L}_{k-1}) + \mathbf{v}_k,\ \mathbf{v}_k \sim N(0, \mathbf{Q}_k), \tag{9}$$

where $f(\cdot)$ and $h(\cdot)$ denote the motion and sensor models; $\mathbf{z}_{k,i}$ denotes the i-th measurement $[\rho_{k,i}\ \theta_{k,i}]^{\mathrm{T}}$ predicted by the function $h(\cdot)$; $\mathbf{w}_k$ and $\mathbf{v}_k$ denote the noise terms, characterized the covariance matrices $\mathbf{R}_k \in \mathbb{R}^{3\times3}$ and $\mathbf{Q}_k \in \mathbb{R}^{2N\times2N}$ in the motion and measurement models, respectively, where N is the total number of measurements. In particular, the covariance matrix $\mathbf{Q}_k$ represents the uncertainty in wall-based line extraction results. This is defined as follows:

$$\mathbf{Q}_k = \begin{bmatrix} \mathbf{Q}_{k,1} & & \\ & \ddots & \\ & & \mathbf{Q}_{k,N} \end{bmatrix},\ \mathbf{Q}_{k,i} = \beta_i \begin{bmatrix} \sigma_{\rho,k,i} & 0 \\ 0 & \sigma_{\theta,k,i} \end{bmatrix}, \tag{10}$$

where β_i denotes the scale factor used to correct the variance of the measurement error. $\sigma_{\rho,k,i}$ and $\sigma_{\theta,k,i}$ denote the covariances of the line measurement related to the distance and angle, respectively, estimated in the line extraction process.

A priori estimation is performed using odometry $\mathbf{o}_k$ estimated by VINS and is expressed as $\mathbf{o}_k = [\Delta x_k\ \Delta y_k\ \Delta\theta_k]^{\mathrm{T}}$, where Δx_k, Δy_k, and $\Delta\theta_k$ represent the changes in the robot's position along the x-axis and y-axis, and heading, respectively, since the previous measurement. A priori estimation of the robot's state is then calculated using (8):

$$\hat{\mathbf{x}}_k^- = f(\hat{\mathbf{x}}_{k-1}, \mathbf{o}_k),\ \hat{\mathbf{P}}_k^- = \mathbf{F}^{\mathrm{T}}\hat{\mathbf{P}}_{k-1}\mathbf{F} + \mathbf{R}_k, \tag{11}$$

where $f(\cdot)$ and $\mathbf{F}$ denote the state transition function and matrix, respectively; and $\hat{\mathbf{P}}_k$ represents the covariance matrix estimated by *Waliner*. A priori estimation of the robot's state is then used to predict its state at the next time step.

VI. Posteriori Estimation in *Waliner*

A. Pose Correction with Manhattan Frames

The rotation angle $\theta_{\mathrm{MF},k}$ between consecutive frames is obtained via MF matching, which is used to modify the orientation term of (11) as follows:

$$\hat{\mathbf{x}}_k^- = f(\hat{\mathbf{x}}_{k-1}, \mathbf{o}_k) + \begin{bmatrix} 0 \\ 0 \\ \Delta\theta_{\mathrm{MF},k} \end{bmatrix} \text{ s.t. } |\Delta\theta_{\mathrm{MF},k}| < \theta_{\mathrm{TH}}^{\mathrm{MF}}, \tag{12}$$

where $\theta_{\mathrm{TH}}^{\mathrm{MF}}$ denotes the threshold of the angle change, which is used to check whether MF matching is successful. Empirically, in our case, $\theta_{\mathrm{TH}}^{\mathrm{MF}}$ ranges from 10 to 15 degree.

B. Measurement Acquisition

To update the predicted robot pose, $\hat{\mathbf{x}}_k$, and its covariance, $\hat{\mathbf{P}}_k$, using the measurement $\mathbf{z}_k$, it should be represented in the global coordinate as follows:

$$\begin{bmatrix} x_{s,k,n}^G & x_{e,k,n}^G \\ y_{s,k,n}^G & y_{e,k,n}^G \\ 1 & 1 \end{bmatrix} = \mathbf{T}_k^G \begin{bmatrix} x_{s,k,n}^L & x_{e,k,n}^L \\ y_{s,k,n}^L & y_{e,k,n}^L \\ 1 & 1 \end{bmatrix}, \tag{13}$$

where the superscripts G and L denote the global and local coordinates, respectively.

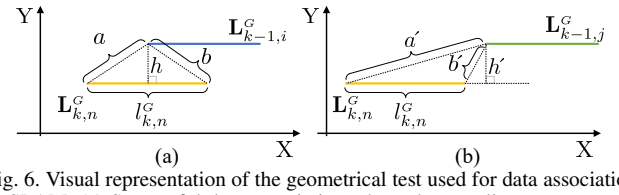


Fig. 6. Visual representation of the geometrical test used for data association in SLAM. (a) Successful data association where the new line measurement matches the existing line. (b) Failed data association where the geometrical condition described in (17) is not satisfied, leading to a matching failure.

$$\mathbf{T}^G_k = \begin{bmatrix} \cos\hat{\theta}^-_k & -\sin\hat{\theta}^-_k & \hat{x}^-_k \\ \sin\hat{\theta}^-_k & \cos\hat{\theta}^-_k & \hat{y}^-_k \\ 0 & 0 & 1 \end{bmatrix}. \tag{14}$$

Accordingly, based on the changed start and end points of the line measurement, the distance between the line and the origin and the angle of the line are changed as follows:

$$\theta^G_{k,n} = \tan^{-1}\left(\frac{y^G_{e,k,n} - y^G_{s,k,n}}{x^G_{e,k,n} - x^G_{s,k,n}}\right), \tag{15}$$

$$\rho^G_{k,n} = \frac{y^G_{e,k,n} - x^G_{e,k,n}\tan\theta^G_{k,n}}{\tan^2\theta^G_{k,n} + 1}. \tag{16}$$

Finally, using (13), (15), and (16), a line vector is defined as $\mathbf{L}^G_{k,n}=\{\rho^G_{k,n}, \theta^G_{k,n}, l^G_{k,n}, \mathbf{p}^G_{k,n,s}, \mathbf{p}^G_{k,n,e}, \mathrm{ID}^G_{k,n}, w^G_{k,n}\}$,where $\mathbf{p}^G_{k,n,s}=(x^G_{s,k,n}, y^G_{s,k,n})$; $\mathbf{p}^G_{k,n,e}=(x^G_{e,k,n}, y^G_{e,k,n})$. IDs are identified at the wall detection stage.

C. Data association (DA): Line Feature Association

Suppose that the n-th line $\mathbf{L}^G_{k,n}$ is extracted at the k-th time step. During the update step of the Kalman filter-based SLAM algorithm, the current measurements are associated with lines from the previous line set $\mathcal{L}^G_{k-1}$ using three methods as follows: First, the chi-squared (χ^2) test [24] is used for the basic DA as $\chi^2 = \Delta\mathbf{L}^T\mathbf{\Sigma}\Delta\mathbf{L}$, where $\Delta\mathbf{L} = \mathbf{L}^G_{k-1,i} - \mathbf{L}^G_{k,n}$ $(1 \le i \le M)$, with M denoting the number of lines in line set $\mathcal{L}^G_{k-1}$. The weight matrix $\mathbf{\Sigma}$ is a 9×9 diagonal matrix as $\mathbf{\Sigma} = \mathrm{diag}(\lambda_1, \lambda_2, \cdots, \lambda_9)$. The second element λ_2 related to an angle has the largest weight, and the weights λ_8 and λ_9 are set to 0. When the condition $\chi^2 < \chi^2_{\mathrm{TH}}$ is true, data association is successful. $\mathbf{\Sigma}$ and χ^2_{TH} should be determined according to the environment and sensor noise, thus, in general environments, they can only be used to filter out lines that are significantly misaligned with others. Second, suppose the foot of the perpendicular line can be drawn from the newly added line $\mathbf{L}^G_{k,n}$ to the two lines $\mathbf{L}^G_{k-1,i}$ and $\mathbf{L}^G_{k-1,j}$ with $\Delta\theta = |\theta^G_{k,n} - \theta^G_{k-1,i}| < \theta_{\mathrm{TH}}$, as shown in Fig. 6. DA is successful if the length h is smaller than h_{TH} and simultaneously satisfies the following:

$$l^G_{k,n} + \varepsilon > \sqrt{a^2 - h^2} + \sqrt{b^2 - h^2}, \tag{17}$$

where $0 < \varepsilon < 1$ and θ_{TH} denotes the threshold of the angle.

Lastly, IDs identified at the wall detection stage IV-B can be used for data association. In summary, with the three aforementioned methods, DA is conducted. The associated lines are merged into a single line.

D. Line Feature Measurement Update

If the corresponding lines are found using DA in the k-th frame, the individual lines $\mathbf{L}^G_{k,n}$ in $\mathcal{L}^G_k$ are updated. Therefore, although they initially appeared as separate lines (after failed DA and new registration), two or more registered lines may be merged into a single line as the registered individual lines are updated empirically.

E. New Line Feature Measurement

If the DA of a new line feature fails under all previous conditions, the line feature is denoted as a new line $\mathbf{L}^G_{k,\mathrm{new}}$, and it should be added to the line set $\mathcal{L}^G_{k-1}$. However, unlike the lines in $\mathcal{L}^G_{k-1}$, $\mathbf{L}^G_{k,\mathrm{new}}$ is not yet represented in the global Manhattan world, *i.e.*, it is only represented in the global space. Therefore, to represent the new line in the global Manhattan world, the angular information, $\theta^G_{k,\mathrm{new}}$, which is the second element of $\mathbf{L}^G_{k,\mathrm{new}}$, is modified as follows:

$$\theta^G_{k,\mathrm{new}} = \begin{cases} \theta_{\mathrm{MW},h}, & \text{if } |\theta^G_{k,\mathrm{new}} - \theta_{\mathrm{MW},h}| < \theta^{\mathrm{MW}}_{\mathrm{TH}} \\ \theta_{\mathrm{MW},v}, & \text{otherwise} \end{cases}, \tag{18}$$

where $\theta^{\mathrm{MW}}_{\mathrm{TH}}$ denotes a threshold that represents how close the angle $\theta^G_{k,\mathrm{new}}$ of $\mathbf{L}^G_{k,\mathrm{new}}$ is to the angle $\theta_{\mathrm{MW},h}$ of GMW. Using (22), all elements of $\mathbf{L}^G_{k,\mathrm{new}}$ should be updated such as the distance $\rho^G_{k,new}$ from the origin to the line. Finally, new line set $\mathcal{L}^G_k$ is generated as $\mathcal{L}^G_k \leftarrow \mathcal{L}^G_{k-1} \cup \mathbf{L}^G_{k,\mathrm{new}}$.

F. Loop Closure With Line Features

In cases where DA fails, this can be due to the observation of a new line. However, failure can also occur when the filter diverges, *i.e.*, the robot's positional information becomes inaccurate. As a result, even though the robot observes a previously registered line, its position can shift significantly, leading to the incorrect assumption that it is observing a new line. To prevent this, we propose a loop detection algorithm to identify when the robot is observing the same line. The detection condition is as follows: we consider the robot's heading, $\hat{\theta}^-_k$, and extend it by a range of $\pm\pi/2$ to obtain $\hat{\theta'}^-_k$. This assumes an error in heading estimation and adjusts by $\pm\pi/2$ using the MWA. With this $\hat{\theta'}^-_k$, the current measured line $\mathbf{L}^G_{k,n}$ is recomputed using the modified transformation, which is

$$T'_G = \begin{bmatrix} \cos\left(\hat{\theta'}^-_k\right) & -\sin\left(\hat{\theta'}^-_k\right) & \hat{x}^-_k \\ \sin\left(\hat{\theta'}^-_k\right) & \cos\left(\hat{\theta'}^-_k\right) & \hat{y}^-_k \\ 0 & 0 & 1 \end{bmatrix}. \tag{19}$$

If data association succeeds with the modified transformation, this indicates that the robot's heading has indeed diverged. Therefore, $\hat{\theta}^-_k$ is updated to $\hat{\theta'}^-_k$. Based on this corrected heading, the measurement acquisition and robot position update processes are then executed.

G. Posteriori Estimation in Waliner

If DA is successful for a new line measurement $\mathbf{L}^G_{k,\mathrm{new}}$, the k-th prior robot's pose $\hat{\mathbf{x}}^-_k$ and the $(k-1)$-th line set $\mathcal{L}^G_{k-1}$ are both updated with the associated line $\mathbf{L}^G_a$ in $\mathcal{L}^G_{k-1}$ and $\mathbf{L}^G_{k,\mathrm{new}}$. In addition, we defined an augmented measurement model as $\mathbf{z}^* = g(\mathbf{z}) + \mathbf{Q}^*_k$, where $\mathbf{z}^* \in \mathbb{R}^{3\times 1}$ and $g(\cdot)$ denotes the pose estimation function that utilizes MFs derived from lines extracted by (7). By decoupling the uncertainties for each component, where the x and y components are influenced by

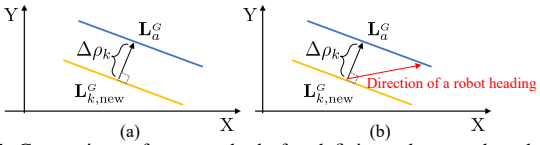


Fig. 7. Comparison of two methods for defining odometry based on line measurements. (a) Odometry estimation without considering the robot's heading, which can lead to potential inaccuracies. (b) Improved odometry estimation related to the robot's heading, resulting in more accurate pose updates.

the range noise $\sigma_{\rho,k,\text{MF}}$ and the orientation is governed by the angular noise $\sigma_{\theta,k,\text{MF}}$, we define the augmented measurement covariance matrix as follows:

$$\mathbf{Q}_k^* = \beta_{\text{MF}} \begin{bmatrix} \sigma_{\rho,k,\text{MF}} & 0 & 0 \\ 0 & \sigma_{\rho,k,\text{MF}} & 0 \\ 0 & 0 & \sigma_{\theta,k,\text{MF}} \end{bmatrix}. \tag{20}$$

To begin, $\Delta\theta_k$ is calculated by $\theta_a^G - \theta_{k,new}^G$. Subsequently, to obtain Δx_k and Δy_k, we calculate the difference between the two lines as follows:

$$\Delta\rho_k = \rho_a^G - \rho_{k,new}^G. \tag{21}$$

Consequently, from (21), Δx_k and Δy_k are calculated as $\Delta\rho_k \cos(\theta_a^G)$ and $\Delta\rho_k \sin(\theta_a^G)$, respectively. However, this approach does not account for the robot's heading, resulting in Δx_k and Δy_k being the same regardless of the robot's heading. This limitation restricts the accuracy of measurements and may cause the filter to diverge in the worst case. Therefore, we propose a method that updates the robot's position by calculating the change in the robot's heading based on the strongly predicted MF-based robot's angle, as shown on the right side of Fig. 7, as follows:

$$\begin{aligned} \Delta x_k' &= \left(\Delta\rho_k \cos\left(\hat{\theta}_k^- - \theta_a^G - \frac{\pi}{2}\right)\right)\cos\left(\hat{\theta}_k^-\right), \\ \Delta y_k' &= \left(\Delta\rho_k \cos\left(\hat{\theta}_k^- - \theta_a^G - \frac{\pi}{2}\right)\right)\sin\left(\hat{\theta}_k^-\right). \end{aligned} \tag{22}$$

Using (22), an innovation vector can be expressed as

$$\mathbf{y}_k = [\Delta x_k' \ \Delta y_k' \ \Delta\theta_k]^{\text{T}}. \tag{23}$$

Using (11) with the Kalman gain $\mathbf{K}_k$ and (23), the k-th robot pose is updated as follows:

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^- + \mathbf{K}_k\mathbf{y}_k, \ \hat{\mathbf{P}}_k = (\mathbf{I} - \mathbf{K}_k)\hat{\mathbf{P}}_k^-, \tag{24}$$

where a Kalman gain is defined as $\mathbf{K}_k = \hat{\mathbf{P}}_k^-(\hat{\mathbf{P}}_k^- + \mathbf{Q}_k^*)^{-1}$.

VII. Experiments

A. Hardware and Parameters Settings

The dataset was collected using Azure Kinect and IMU sensors mounted on a robot that traversed each room following scenarios as depicted in Fig. 8. The entire algorithm, including the proposed algorithm, is operated on an embedded processor (Cortex-A53) equipped with an NPU (NXP i.MX 8M Plus). For in-home environments, we empirically set the following parameters: χ^2_{TH} to range from 10 to 60; $\theta^{\text{MF}}_{\text{TH}}$=10 in (12); h_{TH}=0.4 in Sec.VI.C; β_{MF}=1; and the motion noise values in $\mathbf{R}_k$ to 10 cm and 0.1 rad, respectively.

B. Error Metrics

As a quantitative metric to quantify the discrepancy between the estimated map and the true environment layout

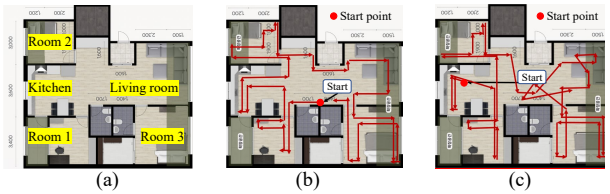


Fig. 8. Experimental environment and the paths followed by the robot in two scenarios. (a) Layout of the environment, including three rooms (Rooms 1, 2, and 3) and a living room. Each room, except for the living room, is approximately 4 m by 4 m. (b) Path for Scenario I, where walls are initially visible. (c) Path for Scenario II, where walls are less visible at the beginning. These scenarios were used to test the robustness of the proposed method under different initial visual conditions.

(expressed as a percentage), the map similarity index (MSI) is defined as

- $\text{MSI} = \frac{1}{M_R}\sum_{i=1}^{M_R}\left(1 - \frac{|R_{\text{est},i} - R_{\text{GT},i}|}{R_{\text{GT},i}}\right) \times 100,$

where M_R represents the total number of selected rooms. $R_{\text{est},i}$ denotes the estimated ratio of the room's horizontal to its vertical dimension for the i-th room as determined by our method, and $R_{\text{GT},i}$ is the corresponding ratio from the ground truth. A higher MSI indicates greater accuracy in the SLAM-generated map, bringing it closer to the true layout, thereby enhancing operational effectiveness in real-world scenarios.

In addition, we evaluate the absolute pose error, denoted as t_{APE}. The metric is defined as:

- $t_{\text{APE}} = \sqrt{\left(\sum_{m=1}^{M} ||\mathbf{t}_{m,\text{GT}} - \hat{\mathbf{t}}_m||^2\right)/M},$

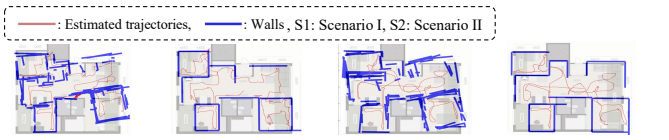
where $\mathbf{t}_{m,\text{GT}}$ and $\hat{\mathbf{t}}_m$ denote the ground truth position vector and estimated position vector, respectively; and M is the number of selected nodes.

VIII. Experimental Result and Discussion

The performance of the proposed method was evaluated and compared with that of the conventional methods in two different scenarios. To delineate the performance difference clearly, the conventional methods included for comparison are VINS odometry, VINS [9] with loop closure, and a fusion of VINS odometry and RTAB-MAP [15].

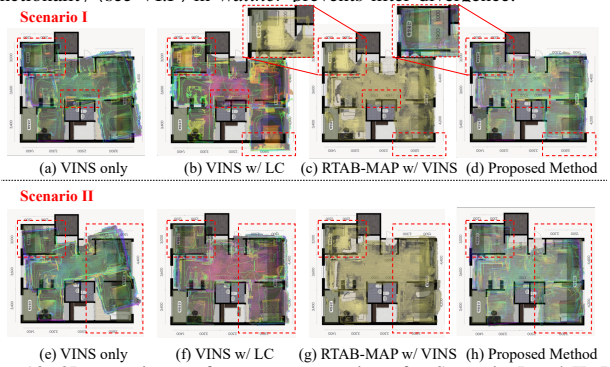
A. Qualitative Mapping Result on Our Dataset

The proposed method produces maps that more accurately represent the actual environmental layout compared with those generated by conventional methods. The maps produced by the proposed method on our dataset demonstrate more consistent and well-defined room boundaries, as shown in Figs. 9 and 10. For Scenario I in Rooms 2 and 3, as shown in Fig. 9(a), VINS encounters errors owing to accumulated inaccuracies and matching failures. However, the proposed method mitigates these errors by utilizing the wall information, as shown in Fig. 9(b). Unlike Scenario I, in Scenario II, the robot starts in the living room and moves towards the kitchen, with the far kitchen wall serving as the initial measurement. This initial condition may lead to more frequent loop closures later. After exploring the kitchen, the robot visits Rooms 1 and 2, intersecting with its previous path. As in Scenario I, the robot encounters accumulated errors and matching failures in Room 2, as shown in Fig. 9(c). Next, the robot then explores the living room and Room 3, where the errors intensify, leading to



(a) VINS only (S1) (b) VINS + *Waliner* w/ LC (S1) (c) VINS only (S2) (d) VINS + *Waliner* w/ LC (S2)

Fig. 9. Qualitative 2D mapping results for Scenarios I and II: (a) and (c) VINS-only method demonstrating significant inaccuracies. (b) and (d) Fusion of VINS and *Waliner* with LC, producing an accurate map that closely matches the actual room layout. This indicates that the loop closure functionality (see VI.F) in *Waliner* prevents filter divergence.



(a) VINS only (b) VINS w/ LC (c) RTAB-MAP w/ VINS (d) Proposed Method

(e) VINS only (f) VINS w/ LC (g) RTAB-MAP w/ VINS (h) Proposed Method

Fig. 10. 3D mapping performance comparison for Scenario I and II. Red dashed-line boxes highlight the visual points of interest. Compared with the other methods, the proposed approach produces a more accurate and visually consistent map, particularly along structural boundaries.

significant matching failures. However, our approach corrects these errors using the proposed loop closure (see Fig. 9(d)).

B. Quantitative Performance Evaluation on Our Dataset

We quantify map quality in two scenarios with MSI and t_{APE} to analyze the experimental results by overlaying the mapping results on the architectural drawing as shown in Fig. 10. As shown in Tables I and II, the proposed method consistently achieves high MSI values and low pose error, indicating superior mapping accuracy. According to these result, the proposed method offers a significant improvement over conventional techniques, providing a more precise and reliable mapping solution for indoor environments, as shown in Fig. 10. In addition, Fig. 11 illustrates the results tested across diverse environments, further validating the robustness and effectiveness of the proposed method.

C. Performance Evaluation on Public Dataset

The proposed algorithm was evaluated on a public dataset (OpenLoris [26]), as represented in Table II, demonstrating its capability to enhance the mapping performance as a plugin method. In particular, our results indicate that the proposed method can improve the performance of even recent RGB-D–based odometry algorithms [25] tested on OpenLoris, as well as those of conventional RGB-based methods [9].

D. Computational Time in the Embedded System

Furthermore, in this study, our algorithm is operated on an embedded processor equipped with an NPU. Computational time was evaluated as shown in Table III. *Waliner* takes just 2 ms, compared with 60 ms for VINS [9] and 229.7 ms for VINS with RTAB-MAP [15]. This indicates the proposed method is lightweight. In particular, because the wall detection module runs in parallel on an NPU ($\approx$ 60 ms), it does not affect the overall computational time of the system.

TABLE I: PERFORMANCE COMPARISON OF DIFFERENT SLAM METHODS IN TWO SCENARIOS

Dataset	Method	Performance [MSI]
Scenario I (our dataset)	VINS [9] only	70
	VINS [9] w/ a loop closure	90
	VINS [9] + RTAB-MAP [15]	90
	VINS [9] + *Waliner* + RTAB-MAP [15]	**95.8**
Scenario II (our dataset)	VINS [9] only	50
	VINS [9] w/ a loop closure	90
	VINS [9] + RTAB-MAP [15]	87
	VINS [9] + *Waliner* + RTAB-MAP [15]	**93.5**

TABLE II: PERFORMANCE COMPARISON OF DIFFERENT SLAM METHODS ON A PUBLIC DATASET AND OUR DATASET

Dataset	Method	t_{APE} [m]
Extended scenario #2 (our dataset) in Fig. 11	VINS [9] only	1.66
	VINS [9] w/ a loop closure	1.86
	VINS [9] + RTAB-MAP [15]	0.89
	VINS [9] + *Waliner* + RTAB-MAP [15]	0.76
	S-VIO [25] + RTAB-MAP [15]	**0.20**
	S-VIO [25] + *Waliner* + RTAB-MAP [15]	**0.19**
Home sequence #1 in OpenLoris [26]	VINS [9] only	0.53
	VINS [9] w/ a loop closure	0.62
	VINS [9] + RTAB-MAP [15]	0.6
	VINS [9] + *Waliner* + RTAB-MAP [15]	0.52
	S-VIO [25] + RTAB-MAP [15]	**0.34**
	S-VIO [25] + *Waliner* + RTAB-MAP [15]	**0.32**
Home sequence #2 in OpenLoris [26]	VINS [9] only	0.36
	VINS [9] w/ a loop closure	0.32
	VINS [9] + RTAB-MAP [15]	0.33
	VINS [9] + *Waliner* + RTAB-MAP [15]	**0.28**
	S-VIO [25] + RTAB-MAP [15]	0.29
	S-VIO [25] + *Waliner* + RTAB-MAP [15]	**0.28**
Home sequence #3 in OpenLoris [26]	VINS [9] only	0.32
	VINS [9] w/ a loop closure	0.32
	VINS [9] + RTAB-MAP [15]	0.32
	VINS [9] + *Waliner* + RTAB-MAP [15]	**0.30**
	S-VIO [25] + RTAB-MAP [15]	0.33
	S-VIO [25] + *Waliner* + RTAB-MAP [15]	**0.32**
Home sequence #4 in OpenLoris [26]	VINS [9] only	0.30
	VINS [9] w/ a loop closure	0.30
	VINS [9] + RTAB-MAP [15]	0.29
	VINS [9] + *Waliner* + RTAB-MAP [15]	**0.28**
	S-VIO [25] + RTAB-MAP [15]	0.31
	S-VIO [25] + *Waliner* + RTAB-MAP [15]	**0.25**
Home sequence #5 in OpenLoris [26]	VINS [9] only	0.26
	VINS [9] w/ a loop closure	0.25
	VINS [9] + RTAB-MAP [15]	0.26
	VINS [9] + *Waliner* + RTAB-MAP [15]	**0.24**
	S-VIO [25] + RTAB-MAP [15]	0.25
	S-VIO [25] + *Waliner* + RTAB-MAP [15]	**0.24**

E. Limitation

Due to the limited sensing range of an RGB-D sensor, extremely long corridors and large open spaces can adversely affect performance. In addition, overall performance may be influenced by the accuracy of the initial wall extraction.

IX. CONCLUSIONS

This paper introduces *Waliner*, a lightweight mapping plugin designed for fast and robust mapping in visually challenging environments using an RGB-D sensor. To ensure reliable mapping, the proposed method incorporates several key strategies: leveraging structural building features, such as walls; implementing an efficient feature extraction algorithm; and applying robust pose estimation, refinement, and loop closure techniques. Experimental results on both our dataset and a public dataset (OpenLoris [26]) demonstrate

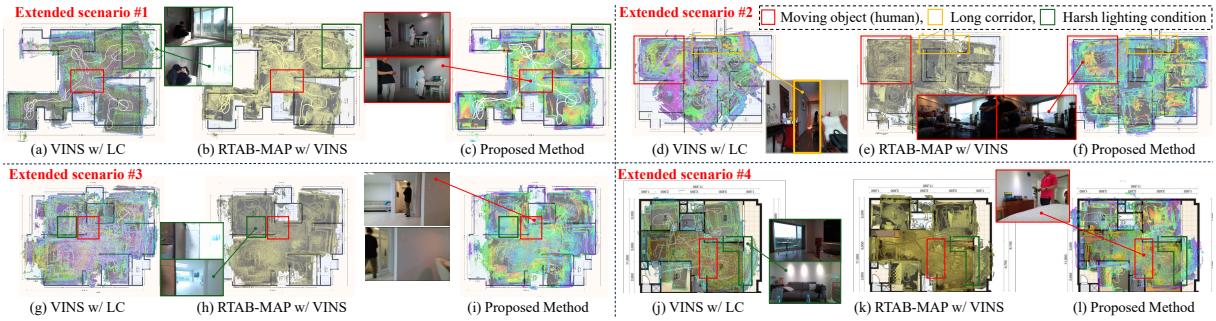


Fig. 11. Comparison of 3D mapping performance in diverse environments, including scenarios with moving humans, harsh lighting conditions, and a long corridor. The proposed method achieves improved mapping results compared to conventional techniques, significantly enhancing its applicability in typical home environments. Detailed images and videos are available on our GitHub repository (`https://github.com/Multiplanet-Robot`).

TABLE III: COMPUTATIONAL TIME COMPARISON OF CORE MODULES IN COMPETITIVE SLAM METHODS ON SCENARIO I.

Method	Computational time [ms]
VINS only	60.0
VINS w/ LC	290.0 (60 (VINS) + 100 (LC) + 130 (PGO))
VINS + RTAB-MAP	229.7 (60 (VINS) + 169.7 (RTAB-MAP))
VINS + RTAB-MAP + *Waliner*	231.7 (229.7 + 2 (*Waliner*))
S-VIO only	120.0

that *Waliner* significantly enhances mapping consistency and mitigates absolute pose errors compared with conventional SLAM methods, while operating with minimal computational overhead on resource-constrained embedded systems. In our future work, we will focus on integrating additional semantic features to further improve mapping performance, particularly in challenging scenarios with limited geometric variation.

REFERENCES

[1] H. Li, J. Zhao, J.-C. Bazin, P. Kim, K. Joo, Z. Zhao, and Y.-H. Liu, "Hong Kong world: Leveraging structural regularity for line-based SLAM," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 11, pp. 13 035–13 053, 2023.

[2] H. Wu, W. Wu, X. Qi, C. Wu, L. An, and R. Zhong, "Planar constraint assisted LiDAR SLAM algorithm based on Manhattan world assumption," *Remote Sensing*, vol. 15, no. 1, p. 15, 2022.

[3] R. Yunus, Y. Li, and F. Tombari, "ManhattanSLAM: Robust planar tracking and mapping leveraging mixture of Manhattan frames," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2021, pp. 6687–6693.

[4] Y. Li, R. Yunus, N. Brasch, N. Navab, and F. Tombari, "RGB-D SLAM with structural regularities," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2021, pp. 11 581–11 587.

[5] P. Kim, H. Li, and K. Joo, "Quasi-globally optimal and real-time visual compass in Manhattan structured environments," *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 2613–2620, 2022.

[6] E. Jeong, S. Kang, D. Lee, and P. Kim, "Parsing indoor Manhattan scenes using four-point LiDAR on a micro UAV," in *Proc. IEEE Int. Conf. Control, Autom. Syst. (ICCAS)*, 2022, pp. 708–713.

[7] K. Joo, T.-H. Oh, F. Rameau, J.-C. Bazin, and I. S. Kweon, "Linear RGB-D SLAM for Atlanta world," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2020, pp. 1077–1083.

[8] S. C. Johnson, "Hierarchical clustering schemes," *Psychometrika*, vol. 32, no. 3, pp. 241–254, 1967.

[9] T. Qin, P. Li, and S. Shen, "VINS-Mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Trans. Robot.*, vol. 34, no. 4, pp. 1004–1020, 2018.

[10] D. Noh, H. Lim, G. Eoh, D. Choi, J. Choi, H. Lim, S. Baek, and H. Myung, "CLOi-Mapper: Consistent, lightweight, robust, and incremental mapper with embedded systems for commercial robot services," *IEEE Robot. Automat. Lett.*, vol. 9, no. 9, pp. 7541–7548, 2024.

[11] F. Zhou, L. Zhang, C. Deng, and X. Fan, "Improved point-line feature based visual SLAM method for complex environments," *Sensors*, vol. 21, no. 13, p. 4604, 2021.

[12] H. Lim, J. Jeon, and H. Myung, "UV-SLAM: Unconstrained line-based SLAM using vanishing points for structural mapping," *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 1518–1525, 2022.

[13] J. Jeon, H. Lim, D.-U. Seo, and H. Myung, "Struct-MDC: Mesh-refined unsupervised depth completion leveraging structural regularities from visual SLAM," *IEEE Robot. Automat. Lett.*, vol. 7, no. 3, pp. 6391–6398, 2022.

[14] E. Jeong, J. Lee, S. Kang, and P. Kim, "Linear four-point LiDAR SLAM for Manhattan world environments," *IEEE Robot. Automat. Lett.*, vol. 8, no. 11, pp. 7392–7399, 2023.

[15] M. Labbé and F. Michaud, "RTAB-Map as an open-source LiDAR and visual simultaneous localization and mapping library for large-scale and long-term online operation," *J. Field Robot.*, vol. 36, no. 2, pp. 416–446, 2019.

[16] C.-Y. Lee, V. Badrinarayanan, T. Malisiewicz, and A. Rabinovich, "Roomnet: End-to-end room layout estimation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 4865–4874.

[17] Y. Yue, T. Kontogianni, K. Schindler, and F. Engelmann, "Connecting the dots: Floorplan reconstruction using two-level queries," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 845–854.

[18] H. Yao, J. Miao, G. Zhang, and J. Chu, "3D layout estimation of general rooms based on ordinal semantic segmentation," *IET Computer Vision*, vol. 17, no. 8, pp. 855–868, 2023.

[19] S. Huang, J. Miao, H. Yao, Y. Zheng, J. Chu, and G. Zhang, "Room layout estimation based on planar semantic segmentation," in *Proc. IEEE World Conf. Applied Intelligence and Computing*, 2022, pp. 188–194.

[20] H. Li, H. Yu, J. Wang, W. Yang, L. Yu, and S. Scherer, "ULSD: Unified line segment detection across pinhole, fisheye, and spherical cameras," *J. Photogrammetry and Remote Sensing*, vol. 178, pp. 187–202, 2021.

[21] D. Seichter, M. Köhler, B. Lewandowski, T. Wengefeld, and H.-M. Gross, "Efficient RGB-D semantic segmentation for indoor scene analysis," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2021, pp. 13 525–13 531.

[22] F. Boniardi, A. Valada, R. Mohan, T. Caselitz, and W. Burgard, "Robot localization in floor plans using a room layout edge extraction network," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 5291–5297.

[23] R. Hendrikx, P. Pauwels, E. Torta, H. P. Bruyninckx, and M. van de Molengraft, "Connecting semantic building information models and robotics: An application to 2D LiDAR-based localization," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2021, pp. 11 654–11 660.

[24] J. Xu, R. Li, L. Zhao, W. Yu, Z. Liu, B. Zhang, and Y. Li, "CamMap: Extrinsic calibration of non-overlapping cameras based on SLAM map alignment," *IEEE Robot. Automat. Lett.*, vol. 7, no. 4, pp. 11 879–11 885, 2022.

[25] P. Gu and Z. Meng, "S-VIO: Exploiting structural constraints for RGB-D visual inertial odometry," *IEEE Robot. Automat. Lett.*, vol. 8, no. 6, pp. 3542–3549, 2023.

[26] Q. She, F. Feng, X. Hao, Q. Yang, C. Lan, V. Lomonaco, X. Shi, Z. Wang, Y. Guo, Y. Zhang, F. Qiao, and R. H. M. Chan, "OpenLORIS-Object: A robotic vision dataset and benchmark for lifelong deep learning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2020, pp. 4767–4773.