Wall-Based BIM Constraints for Drift Reduction in Indoor RGB-D SLAM

Asier Bikandi-Noya
SnT, University of Luxembourg
Luxembourg City, Luxembourg
asier.bikandi@uni.lu

Miguel Fernandez-Cortizas
SnT, University of Luxembourg
Luxembourg City, Luxembourg
miguel.fernandez@uni.lu

Muhammad Shaheer
SnT, University of Luxembourg
Luxembourg City, Luxembourg
muhammad.shaheer@uni.lu

Ali Tourani
SnT, University of Luxembourg
Luxembourg City, Luxembourg
ali.tourani@uni.lu

Holger Voos
SnT, University of Luxembourg
Luxembourg City, Luxembourg
holger.voos@uni.lu

Jose Luis Sanchez-Lopez
SnT, University of Luxembourg
Luxembourg City, Luxembourg
joseluis.sanchezlopez@uni.lu

***Abstract*—Visual SLAM is prone to trajectory drift over extended operation in large indoor environments, degrading map quality and limiting geometric accuracy. We propose *ivS-Graphs*, a visual SLAM system that exploits wall-level correspondences between the mapped environment and a Building Information Model (BIM) to bound this drift. Detected and mapped walls are associated with their BIM counterparts through a two-stage strategy combining plane-parameter and spatial proximity measures, and the resulting correspondences are incorporated as geometric constraints in the back-end factor graph. The method operates in real time and is validated on eight real indoor sequences across two office buildings, achieving a 32.29% reduction in trajectory error and a 7.89% improvement in map accuracy over the visual SLAM backbone.**

***Index Terms*—Visual SLAM, Building Information Model, Architectural Priors, Factor Graph Optimization, Indoor Localization**

I. Introduction

Visual SLAM provides a framework for jointly reconstructing an environment and localizing a sensor within it [1]. However, trajectory drift accumulates over extended operation in large indoor environments, producing maps that are geometrically inconsistent with the true environment [2]. When the expected layout of the environment is known in advance, architectural priors can be incorporated into the SLAM pipeline to enforce geometric consistency and mitigate this drift.

Building Information Models (BIM) [3] encode the *as-planned* structural layout of a building — walls, floors, and other architectural elements — and are available from the design phase onward, making them a suitable source of architectural priors for indoor deployments.

Several works have leveraged such architectural priors for robot localization. Prior map-based approaches commonly estimate the robot pose within a pre-existing map without continuously refining an evolving one [4]. More tightly coupled formulations integrate BIM directly into LiDAR SLAM to regularize the trajectory [5], [6], but rely on room-level semantics that fail in symmetric environments or small areas. On the vision side, [7] integrates visual SLAM with detailed digital twins to suppress drift, requiring a prebuilt model with rich visual content unavailable in typical deployments.

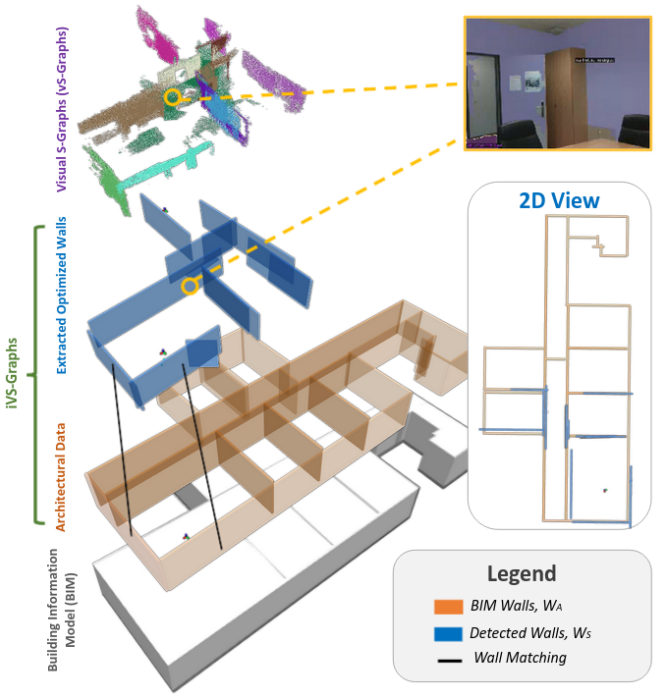

Fig. 1: Overview of *ivS-Graphs* on an office sequence. The BIM reference (bottom) provides architectural priors that are matched to extracted wall planes (middle) from the reconstructed map (top). The 2D projection illustrates the alignment between the SLAM map and the BIM layout.

Authors are with the Automation and Robotics Research Group, Interdisciplinary Centre for Security, Reliability and Trust (SnT), University of Luxembourg. Holger Voos is also associated with the Faculty of Science, Technology and Medicine, University of Luxembourg.

This work was partially funded by the FNR Luxembourg under the project BRIDGES/2025-1/IS/19685965/BARCODE.

For Open Access, the author has applied a CC BY 4.0 public copyright license to any Author Accepted Manuscript version arising from this submission.

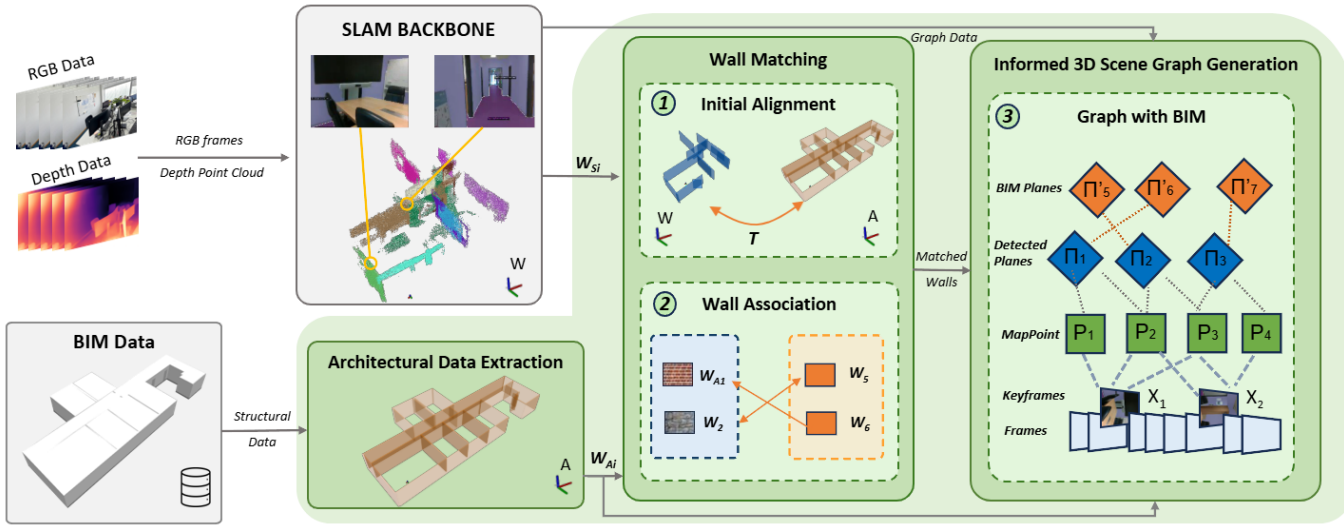

Fig. 2: *ivS-Graphs* system architecture. Architectural walls $\mathbf{W_{A_i}}$ are extracted from the BIM offline and provided as input to the system, while the SLAM backbone processes RGB-D data online, extracting keyframes, map points, and wall segments $\mathbf{W_{S_i}}$. The contributions of this work are highlighted in green: (**1**) initial alignment of the BIM and SLAM coordinate frames via a least-squares transformation $\mathbf{T}$, (**2**) continuous wall association establishing correspondences $\mathbf{W_A} \leftrightarrow \mathbf{W_S}$, which are introduced as (**3**) fixed BIM plane nodes into the back-end factor graph, constraining the optimization alongside keyframes, map points, and detected wall planes.

To address the lack of BIM-informed drift correction in visual SLAM, this work proposes *ivS-Graphs* (illustrated in Fig. 1), which extends an existing visual SLAM backbone [8] with back-end constraints derived from associations between detected and mapped walls and their BIM counterparts.

- Integration of architectural BIM priors as geometric constraints in a visual SLAM back-end to reduce trajectory drift.
- A two-stage wall association strategy that requires only the specification of two BIM wall identifiers expected to be observed at the start of operation, enabling immediate alignment without prior environment exploration.

II. Methodology

A. System Overview

Our system builds upon *vS-Graphs* [8] as a SLAM backbone, which provides a feature-based front-end derived from *ORB-SLAM3* [9] and a hierarchical graph-based back-end for structural mapping. The front-end processes RGB-D frames and extracts observed walls $\mathbf{W_S}$ from depth data via the *Scene Segmentor*, while the *ATLAS* map manager maintains the evolving map. The back-end employs a factor graph encoding keyframes $\mathbf{X}$, map points $\mathbf{P}$, and structural wall planes $\mathbf{W_S}$, connected via visual odometry, reprojection, and pose-to-wall constraints. The overall architecture is illustrated in Fig. 2.

The system takes three inputs: (i) RGB-D data from the camera, (ii) architectural walls extracted from the BIM, and (iii) two initial wall IDs specifying the expected first wall observations, used to initialize the alignment between the BIM and the SLAM map (detailed in Section II-B). Both detected and BIM-extracted walls are represented as:

$$w_i = \begin{cases} \mathrm{id}_i, & \text{unique wall identifier} \\ \pi_i = (\mathbf{n}_i, d_i), & \text{supporting plane: normal and distance} \\ \mathbf{c}_i, & \text{centroid position} \\ l_i, t_i, & \text{length and thickness} \end{cases} \tag{1}$$

where $\mathbf{n}_i \in \mathbb{R}^3$ and $d_i \in \mathbb{R}$ define the plane via the Hessian normal form $\mathbf{n}_i^\top \mathbf{p} + d_i = 0$. We denote $\mathbf{W_A}$ as BIM walls and $\mathbf{W_S}$ as detected walls. Three modules extend the backbone:

- **BIM-to-SLAM Wall Association (Section II-B).** Establishes and maintains correspondences between BIM walls and detected walls.
- **BIM-Constrained Back-End (Section II-C).** Incorporates matched wall pairs as constraints into the factor graph optimization.

B. BIM-to-SLAM Wall Association

The matching procedure operates in two stages: initial alignment and continuous matching.

Initial Alignment. Before any BIM correction is applied, the two coordinate frames must be aligned. The system waits until two approximately perpendicular walls are detected and matches them to the two user-specified BIM wall IDs. The transformation ${}^{S}\hat{T}_A \in \mathrm{SE}(3)$ aligning the BIM frame $\mathbf{A}$ to the SLAM frame $\mathbf{S}$ is then estimated by minimizing plane-parameter residuals via least squares. All BIM walls $\mathbf{W_A}$ are subsequently re-expressed in $\mathbf{S}$ and stored in the map manager for continuous association.

Continuous Matching. After initialization, wall associations are updated as new walls are detected. Each detected

wall w_{S_i} is matched against candidate BIM walls w_{A_j} using two complementary measures. The *Plane-Parameter Distance* (PPD) evaluates infinite surface alignment:

$$d_{\text{plane}}(\pi_{S_i}, \pi_{A_j}) = \sqrt{(\pi_{S_i} \ominus \pi_{A_j})^\top (\pi_{S_i} \ominus \pi_{A_j})} \quad (2)$$

where $\ominus$ computes the minimal distance between plane parameters accounting for sign ambiguity. The *Lateral Centroid Distance* (LCD) disambiguates walls on parallel planes by measuring tangential displacement between centroids:

$$d_{\text{lat}}(w_{S_i}, w_{A_j}) = \sqrt{\|\mathbf{c}_{S_i} - \mathbf{c}_{A_j}\|^2 - \left((\mathbf{c}_{S_i} - \mathbf{c}_{A_j})^\top \mathbf{n}_{S_i}\right)^2} \quad (3)$$

Both metrics are combined into a normalized weighted score:

$$s(w_{S_i}, w_{A_j}) = \alpha \frac{d_{\text{plane}}(\pi_{S_i}, \pi_{A_j})}{\tau_p} + (1-\alpha) \frac{d_{\text{lat}}(w_{S_i}, w_{A_j})}{\tau_c} \quad (4)$$

where τ_p and τ_c are normalization thresholds and $\alpha \in [0, 1]$ controls the trade-off between surface alignment and spatial proximity. Only candidates satisfying $d_{\text{plane}} < \tau_p$ and $d_{\text{lat}} < \tau_c$ are considered, and the best match $w_A^* = \arg\min_{w_{A_j}} s(w_{S_i}, w_{A_j})$ is selected.

C. BIM-Constrained Back-End

After the initial alignment, BIM walls $\mathbf{W_A}$ are introduced as fixed nodes in the factor graph, representing the trusted reference, while detected walls $\mathbf{W_S}$ remain optimizable. Wall-to-wall factors are created from associations $\mathcal{M} = \{(w_{S_i}, w_{A_j})\}$ found during matching, with cost function:

$$c_{\text{wall-wall}}(w_{S_i}, w_{A_j}) = \left\| w_{S_i} \ominus w_{A_j} \right\|^2_{\Lambda_{w_{ij}}} \quad (5)$$

where the covariance $\Lambda_{w_{ij}} = \mathbf{I} \cdot \beta \cdot s(w_{S_i}, w_{A_j})$ grows proportionally with the matching score, so that high-confidence associations exert stronger corrections while ambiguous ones are automatically down-weighted. The total cost minimized by the optimizer is:

$$c_{\text{total}} = \sum_{x,p} c_{\text{reproj}}(x, p) + \sum_{x, w_S} c_{\text{pose-wall}}(x, w_S) + \sum_{(w_S, w_A) \in \mathcal{M}} c_{\text{wall-wall}}(w_S, w_A) \quad (6)$$

The optimizer adjusts $\{\mathbf{X}, \mathbf{P}, \mathbf{W_S}\}$ to minimize c_{total}, propagating BIM corrections through keyframes and map points according to their confidence. BIM walls remain fixed throughout, and corrections further propagate via the map merging and loop closing mechanisms inherited from the baseline [8]. Associations with large residuals are additionally down-weighted using a robust Huber kernel [10], reducing the influence of incorrect matches caused by occlusions or incomplete wall detections in structured indoor environments.

III. Evaluation

A. Validation Methodology

Datasets. Since no public dataset provides both RGB-D data and corresponding BIM, we collected real-world sequences combining RGB-D streams (Intel RealSense D435i) with LiDAR point clouds (Ouster OS0-64) for ground-truth generation, all integrated within the SMapper device [11]. The sequences span eight indoor trajectories across two office buildings with varying layouts, as summarized in Table I.

TABLE I: Summary of evaluation sequences. #W and #S denote the number of visible walls and distinct navigable spaces (rooms and corridors), respectively. Area, Length (L), and Duration (D) are in m^2, m, and s. office1 and office2 are separate buildings with multiple sequences each.

Env.	Sequence	#W	#S	Area (m^2)	L (m)	D (s)
Offices	office1-1	10	4	100	90	670
	office1-2	12	3	109	48	265
	office1-3	12	4	113	99	524
	office1-4	6	2	54	37	169
	office2-1	10	3	168	68	268
	office2-2	10	3	168	73	370
	office2-3	10	3	70	41	172
	office2-4	4	4	40	53	260
Total		74	27	822	509	2698

Baselines. Since no existing visual SLAM system integrates BIM priors, we evaluate against established RGB-D SLAM baselines: ORB-SLAM3 [9] and BAD-SLAM [12], selected for their reported performance and code availability. Comparison with vS-Graphs [8] directly isolates the contribution of BIM integration, as our system builds on it.

Implementation Details. All algorithms were executed on a workstation with an Intel Core i9-11950H (2.60 GHz), an NVIDIA T600 Mobile GPU, and 32 GB RAM.

B. Trajectory Estimation

We evaluate trajectory accuracy using the Absolute Trajectory Error (ATE), as drift accumulation directly degrades map quality. Reference trajectories are generated by processing Ouster OS0-64 LiDAR scans offline with S-Graphs [13], achieving an average ATE of 4.16 cm in similar indoor environments, substantially more accurate than any of the evaluated baselines.

Table II summarizes the ATE results. *ivS-Graphs* reduces mean ATE by 32.29% relative to *vS-Graphs*, with improvements in 7 out of 8 sequences. The improvement correlates with trajectory length: longer sequences (e.g. *office1-1*, 90 m) benefit more from continuous wall constraints that bound drift accumulation, while the shortest sequence (*office1-4*, 37 m) shows a minor degradation as drift is already minimal and BIM constraints provide limited benefit. BAD-SLAM fails on most sequences due to sensitivity to textureless surfaces. The qualitative comparison in Fig. 3 further confirms this trend, showing a top-down overlay on *office1-1*: the trajectory estimated by *ivS-Graphs* is topologically consistent with the floor layout, whereas *vS-Graphs* exhibits deviations with the trajectory crossing through BIM walls.

C. Mapping Performance

We evaluate map quality by computing the Root Mean Square Error (RMSE) between the reconstructed point cloud

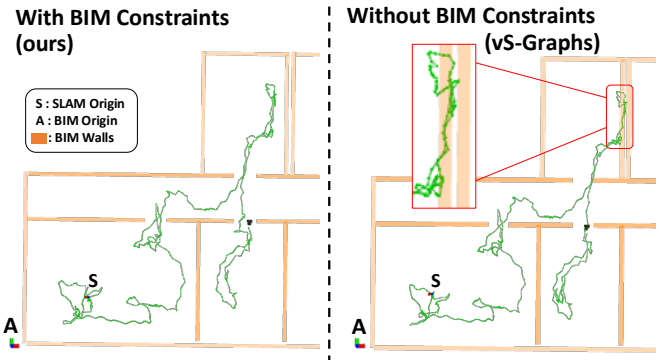

Fig. 3: Qualitative comparison of estimated trajectories on the *office1-1* sequence. Left: proposed method with BIM constraints; right: baseline without BIM constraints, just initial alignment integrated.

TABLE II: ATE in meters (m) for the evaluated RGB-D SLAM methods. Best and second-best results are **boldfaced** and underlined. Values averaged over five runs. Dashes indicate tracking failure. Δ reports improvement of *ivS-Graphs* over *vS-Graphs*.

Sequence	ATE (m)				
	BAD SLAM [12]	*ORB-SLAM3* [9]	*vS-Graphs* [8]	*ivS-Graphs* (ours)	Δ (%)
office1-1	-	0.462	0.343	**0.226**	+33.99%
office1-2	1.653	0.365	0.353	**0.183**	+48.15%
office1-3	-	0.272	0.253	**0.175**	+30.83%
office1-4	-	0.101	**0.090**	0.120	-33.33%
office2-1	1.879	**0.160**	0.223	0.214	+4.04%
office2-2	-	0.461	0.348	**0.224**	+35.76%
office2-3	-	0.393	0.280	**0.176**	+36.85%
office2-4	7.926	0.553	0.386	**0.123**	+68.10%
Mean	-	0.351	0.283	**0.178**	+32.29%

and the reference BIM point cloud. This metric directly reflects the geometric consistency of the map with respect to the *as-planned* layout.

Table III summarizes the results. *ivS-Graphs* achieves the lowest RMSE in 6 out of 8 sequences, with improvements over *vS-Graphs* ranging from +6.35% to +17.88%. Gains follow a clear size-dependent pattern: in larger environments such as *office1-1* (+17.88%), BIM constraints produce consistent map improvements as drift accumulates over extended operation. In smaller environments the contribution is marginal, as the baseline trajectory is already accurate. This also explains why relative gains are smaller than those observed for ATE: BIM constraints reduce drift-induced map distortion but cannot compensate for per-frame depth noise of the RGB-D sensor, which remains an independent source of map error.

TABLE III: Point cloud RMSE (m) for the evaluated RGB-D SLAM methods. Best and second-best results are **boldfaced** and underlined. Values averaged over five runs. Dashes indicate tracking failure. Δ reports improvement of *ivS-Graphs* over *vS-Graphs*.

Sequence	Point Cloud RMSE (m)				
	BAD SLAM [12]	*ORB-SLAM3* [9]	*vS-Graphs* [8]	*ivS-Graphs* (ours)	Δ (%)
office1-1	-	0.416	0.403	**0.331**	+17.88%
office1-2	0.442	0.537	0.444	**0.396**	+10.72%
office1-3	-	0.360	**0.340**	0.354	-4.15%
office1-4	-	0.349	0.327	**0.306**	+6.35%
office2-1	0.394	0.407	0.432	**0.382**	+11.55%
office2-2	-	0.487	**0.470**	0.473	-0.80%
office2-3	-	0.412	0.481	**0.411**	+14.60%
office2-4	-	**0.386**	0.546	0.504	+7.64%
Mean	-	0.419	0.430	**0.395**	+7.89%

IV. Conclusion

ivS-Graphs reduces trajectory drift in visual SLAM by incorporating wall-to-BIM correspondences as geometric constraints in the factor graph back-end, achieving 32.29% ATE reduction and 7.89% map RMSE improvement over the visual SLAM backbone across eight real indoor sequences. The improvement correlates with trajectory length, confirming that BIM constraints are most effective in longer sequences where drift accumulates over extended operation.

Future work will extend the set of BIM elements used as constraints — incorporating floors, columns, and pipes — to improve alignment in smaller environments where wall coverage alone is insufficient. Additionally, leveraging these richer semantic cues will enable more robust initialization strategies, removing the requirement to specify two initial wall IDs. Validation on a broader range of indoor environments and under varying degrees of BIM completeness is also planned.

References

[1] C. Cadena *et al.*, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Trans. Robot.*, vol. 32, no. 6, pp. 1309–1332, 2016.

[2] T. Sun, Y. Hao, S. Huang, S. Savarese, K. Schindler, M. Pollefeys, and I. Armeni, "Nothing stands still: A spatiotemporal benchmark on 3D point cloud registration under large geometric and temporal change," *ISPRS J. Photogramm. Remote Sens.*, vol. 220, pp. 799–823, 2025.

[3] S. Azhar, M. Hein, and B. Sketo, "Building information modeling (BIM): Benefits, risks and challenges," in *Proc. 44th ASC Annual Conf.*, 2008, pp. 2–5.

[4] H. Blum, J. Stiefel, C. Cadena, R. Siegwart, and A. Gawel, "Precise robot localization in architectural 3D plans," *arXiv preprint arXiv:2006.05137*, 2020.

[5] M. Shaheer, J. A. Millan-Romera, H. Bavle, M. Giberna, J. L. Sanchez-Lopez, J. Civera, and H. Voos, "Tightly coupled SLAM with imprecise architectural plans," *IEEE Robot. Autom. Lett.*, vol. 10, no. 8, pp. 8019–8026, 2025.

[6] M. A. Vega Torres, A. Braun, and A. Borrmann, "BIM-SLAM: Integrating BIM models in multi-session SLAM for lifelong mapping using 3D LiDAR," *arXiv preprint arXiv:2408.15870*, 2024.

[7] R. Merat, G. Cioffi, L. Bauersfeld, and D. Scaramuzza, "Drift-free visual SLAM using digital twins," *IEEE Robot. Autom. Lett.*, vol. 10, no. 2, pp. 1633–1640, 2025.

[8] A. Tourani, S. Ejaz, H. Bavle, D. Morilla-Cabello, J. L. Sanchez-Lopez, and H. Voos, "vS-Graphs: Integrating visual SLAM and situational graphs through multi-level scene understanding," *arXiv preprint arXiv:2503.01783*, 2025.

[9] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM3: An accurate open-source library for visual, visual–inertial, and multimap SLAM," *IEEE Trans. Robot.*, vol. 37, no. 6, pp. 1874–1890, 2021.

[10] J. A. Gallego, F. A. González, and O. Nasraoui, "Robust kernels for robust location estimation," *Neurocomputing*, vol. 429, pp. 174–186, 2021.

[11] P. M. B. Soares, A. Tourani, M. Fernández-Cortizas, A. Bikandi Noya, J. L. Sánchez-López, and H. Voos, "SMapper: A multi-modal data acquisition platform for SLAM benchmarking," *J. Intell. Robot. Syst.*, vol. 112, 2025.

[12] T. Schöps, T. Sattler, and M. Pollefeys, "BAD SLAM: Bundle adjusted direct RGB-D SLAM," in *Proc. IEEE/CVF CVPR*, 2019, pp. 134–144.

[13] H. Bavle, J. L. Sanchez-Lopez, M. Shaheer, J. Civera, and H. Voos, "S-Graphs 2.0 – A hierarchical-semantic optimization and loop closure for SLAM," *arXiv preprint arXiv:2502.18044*, 2025.