Towards Practical and Deployable Infrastructure-Free Indoor Localization

Maimoonah Al-Mashhadani, Yaqoob Ansari, Eduardo Feo-Flushing, and Khaled A. Harras
Carnegie Mellon University
{maimoonm, yansari, efeoflus, kharras}@andrew.cmu.edu

***Abstract*—Indoor localization solutions are constrained by infrastructure requirements, privacy concerns, or accumulating drift. We present GraphWalker, a system that aligns commodity smartphone IMU trajectories to semantic floor plan graphs through constrained beam search. GraphWalker operates on coarse topological building representations where nodes encode semantic locations and edges represent traversable connections, requiring no WiFi fingerprinting, Bluetooth beacons, or dense sensor networks. The system transforms IMU-derived trajectories through coordinate alignment, then employs iterative beam search to maintain and score multiple path hypotheses based on topological constraints and heading consistency. Evaluation demonstrates 45% reduction in endpoint localization error (from 11.5 to 6.3 meters) and 35% reduction in graph distance compared to direct IMU projection, with median along-trajectory error of 1.4 meters. These results establish semantic graph-constrained localization as a scalable solution bridging infrastructure-free inertial tracking and high-precision infrastructure-dependent systems.**

***Index Terms*—indoor localization, inertial measurement unit, semantic graph alignment, beam search localization, infrastructure-free positioning, trajectory-to-graph matching**

I. Introduction

Indoor localization remains constrained by competing requirements. Infrastructure-dependent approaches achieve high accuracy but face deployment barriers: radio fingerprinting [1]–[5] requires extensive site surveys that degrade with environmental changes, Bluetooth beacons [6]–[8] and UWB systems [9] demand dense deployment and continuous calibration, and vision-based methods [10], [11] raise privacy concerns while requiring image databases. Pure inertial navigation [12] avoids infrastructure but accumulates unbounded drift, limiting utility beyond short trajectories.

Deep learning has transformed IMU-based trajectory estimation by mitigating sensor noise and integration drift. RoNIN [13], a learned inertial odometry system, employs recurrent networks trained on extensive, large-scale pedestrian datasets to predict displacement from raw IMU sequences, achieving relative position estimates without external references. IONet [14] extends this through temporal convolution for longer-range dependencies, while TLIO [15] fuses learned odometry with visual-inertial filtering. However, these methods produce trajectories in arbitrary local coordinate frames with unknown scale and orientation, creating a gap between relative trajectory quality and absolute placement accuracy on building floor plans.

Map-constrained localization can help bridge this gap. Graph-based formulations convert the problem to discrete inference: Hidden Markov Models on spatial graphs [16] apply Viterbi decoding but assume known transition probabilities, constraint-based optimization [17] matches trajectory segments to corridor geometry, particle filtering [18] tracks distributions on spatial maps, and InLoc [11] combines visual odometry with map matching. GraphSLAM [19] builds pose graphs but requires loop closure detection. Constrained motion planning [20] uses graph search with geometric constraints. These approaches require dense maps, known probabilistic models, or iterative optimization with extensive calibration.

We present GraphWalker, a system that aligns IMU-derived trajectories to semantic floor plan graphs through constrained beam search. Unlike dense map-matching approaches requiring detailed geometry, GraphWalker operates on sparse topological graphs where nodes represent semantic locations (rooms, corridor intersections) and edges encode traversable connections. A trajectory alignment component transforms IMU predictions into floor plan coordinates via scaling, rotation, and anchor node snapping. Beam search maintains multiple candidate path hypotheses scored by local heading consistency and trajectory-to-graph alignment, progressively pruning implausible routes under structural constraints. This formulation trades geometric precision for robust topological correctness, enabling deployment with minimal calibration and coarse building models.

We share some preliminary results that demonstrate the promise of GraphWalker. Our sensitivity analysis shows that performance critically depends on the radius of valid hallway transitions and the window size used for scoring. With tuned parameters, GraphWalker reduces endpoint localization error by 45% in metric distance (from 11.5 to 6.3 m) and 35% in graph distance compared to direct IMU projection baselines, while maintaining a median along-trajectory error of 1.4 m. These findings demonstrate that semantic graph constraints effectively mitigate directional drift, providing a scalable solution bridging infrastructure-free inertial tracking and infrastructure-dependent high-precision localization.

Overall, our contributions are: (i) A beam search formulation for aligning drift-prone IMU trajectories to semantic graphs under structural constraints, (ii) A trajectory alignment pipeline that establishes coordinate correspondence without manual feature engineering, and (iii) Preliminary Evaluation on 16 real-world walking trajectories averaging 50 m.

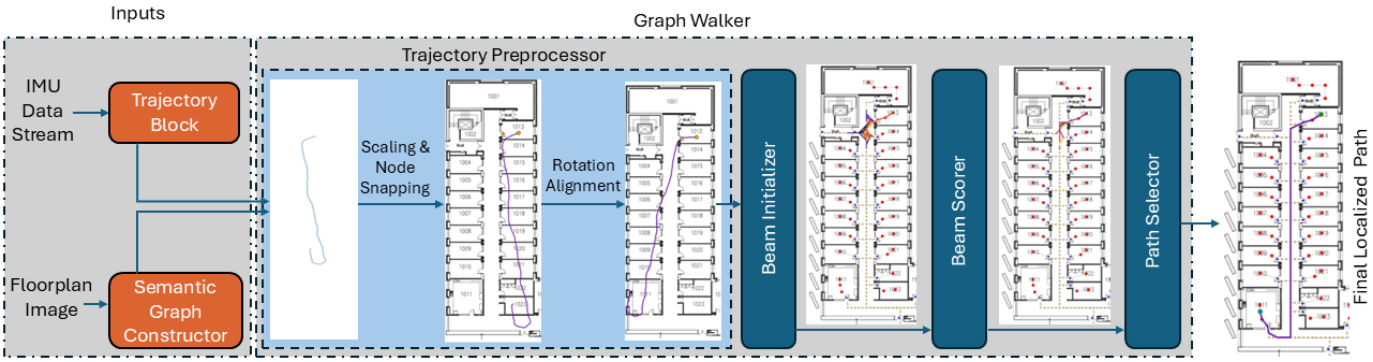

Fig. 1. *GraphWalker* system overview. IMU data and floor plan are processed through trajectory encoding and graph construction (orange blocks), then iteratively refined via beam search (blue blocks). Candidate paths (red) are progressively pruned until the final localized trajectory (purple) is selected.

II. The GraphWalker System

GraphWalker localizes users within buildings by aligning IMU-derived trajectories with semantic graph representations via iterative beam search (Fig. 1). The system transforms raw sensor data through coordinate alignment, generates candidate path hypotheses, scores them against trajectory observations, and selects the highest scoring localization. This formulation trades geometric precision for robust topological correctness, enabling deployment with minimal calibration and preserving user privacy through on-device processing.

A. Problem Setting and Inputs

GraphWalker operates on two inputs. The first is an IMU stream $\mathcal{I} = \{i_1, \ldots, i_T\}$ from a smartphone containing accelerometer and gyroscope measurements. These are processed by learned odometry methods such as RoNIN [13] into the trajectory $\mathcal{T} = \{(x_t, y_t)\}_{t=1}^{T}$ in local coordinates. Although such trajectories capture path geometry with high relative accuracy, they exhibit scale drift and operate in arbitrary coordinate frames with unknown global orientation. The second input is a floor plan represented as a semantic graph $G = (V, E)$ where nodes $v \in V$ represent locations (room centroids, corridors, doorways) and edges $e \in E$ encode traversable connections with distance and bearing. The graph may be constructed manually or derived automatically from floor plan imagery using existing tools (e.g., [21]).

B. Trajectory Preprocessor

Learned odometry produces trajectories in egocentric coordinate frames bearing no relationship to floor plan coordinates, preventing direct projection due to scale mismatch, arbitrary rotation, and unknown absolute position. The trajectory alignment component establishes coordinate correspondence through three operations (Fig. 1). Scale calibration converts pixel coordinates to metric units: $\mathcal{T}' = r \cdot \mathcal{T}$ where $r = 0.03$ m/px is obtained from the Tesseract graph construction process. Node snapping identifies the nearest graph node to the known starting location: $v_{\text{anchor}} = \arg\min_{v \in V} \|v - \mathcal{T}'_0\|$ where $\mathcal{T}'_0$ is the trajectory origin. Rotation alignment computes the trajectory heading from two points $\mathcal{T}'_0, \mathcal{T}'_k$ as $\theta_{\text{traj}} = \arctan 2(\mathcal{T}'_k - \mathcal{T}'_0)$ and obtains the anchor node's dominant edge bearing θ_{anchor} by selecting the edge most aligned with the initial trajectory direction based on heading consistency scoring. The rotation angle $\Delta\theta = \theta_{\text{anchor}} - \theta_{\text{traj}}$ yields the aligned trajectory: $\mathcal{T}'' = R(\Delta\theta) \cdot (\mathcal{T}' - v_{\text{anchor}}) + v_{\text{anchor}}$ where $R(\Delta\theta)$ is the 2D rotation matrix.

Algorithm 1 Beam Initialization and Expansion

Require: Aligned trajectory $\mathcal{T}''$, graph $G = (V, E)$, anchor v_{anchor}
Require: Segment t, active beams $\mathcal{B}$, pending beams $\mathcal{P}$, width K
Ensure: Executable moves $\mathcal{Q}$, updated pending $\mathcal{P}'$
Initialize $\mathcal{Q} \leftarrow \emptyset$, $\mathcal{P}' \leftarrow \emptyset$
for all beam $b \in \mathcal{P}$ **do**
  if enough distance is available for b to advance **then**
    Add b to $\mathcal{Q}$
  else
    Add b to $\mathcal{P}'$
  end if
end for
for all beam $b \in \mathcal{B}$ **do**
  $v_{\text{last}} \leftarrow$ last node in b
  for all $v' \in$ neighbors(v_{last}) **do**
    $b' \leftarrow b \oplus v'$
    if edge (v_{last}, v') reachable in segment t **then**
      Add b' to $\mathcal{Q}$
    else
      Add b' to $\mathcal{P}'$
    end if
  end for
end for
return $\mathcal{Q}$, $\mathcal{P}'$

C. Search Beam Initializer

Trajectory drift and local ambiguities prevent single-hypothesis tracking, requiring the system to maintain multiple plausible paths while remaining computationally tractable. Beam search maintains K candidate hypotheses initialized at v_{anchor} and expanded per trajectory segment. Each beam $b = \langle v_1, \ldots, v_m \rangle$ represents a partial node sequence. Algorithm 1 extends beams to neighbors: for a beam terminating at v, an extension $b' = b \oplus v'$ is considered valid if (i) the edge (v, v') exists in the semantic graph and is therefore traversable, or (ii) v and v' correspond to hallway nodes within a small spatial radius for which traversal is allowed. In both cases, the candidate edge length must not exceed the distance covered by the current trajectory segment. Valid extensions whose length fits within the current segment enter the executable

Algorithm 2 Beam Scoring

Require: Executable moves $\mathcal{Q}$, beam width K
Ensure: Updated beam set $\mathcal{B}$
1: Initialize new beam set
2: **for all** move in $\mathcal{Q}$ **do**
3: Retrieve old score s_{old}
4: Compute alignment α and windowed trajectory similarity β
5: Combine: $s_{\text{new}} = 0.60 s_{\text{old}} + 0.25\alpha + 0.15\beta$
6: Add move with s_{new} to new beam set
7: **end for**
8: Keep top K beams by score
9: **return** updated beam set

queue $\mathcal{Q}$ for scoring, while valid extensions that exceed the current segment length are placed in the pending set $\mathcal{P}$ and re-evaluated once additional trajectory segments become available. This dual-queue mechanism handles mismatches between trajectory discretization and graph edge boundaries.

D. Beam Scorer

Not all topologically valid paths are physically plausible given observed motion, requiring differentiation through consistency evaluation between graph structure and trajectory geometry. Beam scoring quantifies how well a candidate beam explains recent IMU-derived motion (Algorithm 2). For an extension from node v to neighbor v', the scorer computes two components. First, an *alignment* term α measures the cosine similarity between the displacement vector associated with traversing edge (v, v') and the edge's unit direction, capturing local heading agreement. Second, a *trajectory-similarity* term β compares the recent trajectory segment to the corresponding graph subpath using an adaptive lookback, accounting for both displacement direction and path efficiency. The new beam score is then computed as $\text{s}_{\text{new}} = 0.60 \cdot \text{s}_{\text{old}} + 0.25 \cdot \alpha + 0.15 \cdot \beta$. After scoring all candidate extensions, only the top K beams (K=50) are retained.

E. Beam Selector

At trajectory completion, the system maintains K candidate paths that have survived iterative pruning and must select the globally optimal hypothesis. The beam selector identifies the final localized path after processing all trajectory segments. At termination, the beam set $\mathcal{B}_{\text{final}}$ contains K candidate paths. The selector returns $b^* = \arg\max_{b \in \mathcal{B}_{\text{final}}} s(b)$ where $s(b)$ is the cumulative score. This beam represents the most likely path through the semantic graph given the observed IMU trajectory under physical connectivity constraints. The output path $P^* = \langle v_1, \ldots, v_n \rangle$ corresponding to b^* provides the final localization result mapped to floor plan coordinates.

III. Preliminary Evaluation

A. Evaluation Protocol and Setup

We evaluate GraphWalker on 16 walking trajectories collected in a university building. The trajectories span offices, corridors, lecture halls, and multi-room sections, with mean length 50 m. We record IMU data using Samsung Galaxy S20 5G smartphones at 200 Hz. Ground truth locations are manually annotated at key points and turns, aligned to IMU timestamps. The semantic graph is derived from building floor plans. Nodes represent room centers, corridor intersections, and waypoints; edges encode traversable connections. All trajectories begin and end at known anchor points corresponding to graph nodes. The known starting position is a realistic assumption for practical deployment scenarios such as entering a building at a marked entrance, and enables initial trajectory anchoring and orientation alignment.

For comparison, we implement a baseline in which the IMU-derived velocity trajectory is scaled and rotated to best align with the floor plan, after which the final estimated position is snapped to the nearest graph node. As a sensitivity analysis, we additionally evaluate the effect of two internal parameters on localization accuracy: (i) the hallway-transition radius that allows transitions between adjacent hallway nodes without explicit edges, and (ii) the trajectory similarity window size used by the beam scorer.

TABLE I
Sensitivity analysis. Base: hallway radius = 2.2 m (70 px), similarity window = 12.6 m (400 px). Only one parameter varies at a time.

Parameter	Size [m/px]	Mean Dist Error [m]	Median Dist Error [m]	Max Dist Error [m]	Endpoint Error [m]	Endpoint Error [nodes]
Radius	0.0 / 0	13.48	14.67	25.49	46.12	79.67
	1.1 / 35	3.66	2.23	9.42	17.76	30.87
	2.2 / 70	**1.71**	**1.42**	**4.18**	**6.31**	**9.69**
	3.2 / 100	2.63	1.85	6.72	9.56	15.31
Window	0.0 / 0	2.25	1.75	5.67	10.08	16.56
	6.3 / 200	2.33	1.84	5.74	9.00	14.44
	12.6 / 400	**1.71**	**1.42**	**4.18**	**6.31**	**9.69**
	18.9 / 600	1.73	1.40	4.30	6.30	9.75

B. Metrics

We measure localization accuracy using five metrics: the averaged mean, median, and maximum distance errors along the trajectory; endpoint error in meters; and endpoint error in graph-path distance (number of nodes). The along-trajectory distance errors quantify how well the system maintains accurate localization throughout the path. The endpoint error in meters reflects how close the final predicted position is to the true destination, while the endpoint error in nodes captures correctness with respect to the building's structural layout: two endpoints may be only a few meters apart in Euclidean space yet belong to different rooms, corridors, or hallway branches. For the trajectory projection baseline, only the final endpoint is available, so intermediate distance errors are undefined.

C. Results

Table I summarizes the sensitivity analysis. The optimal parameters are hallway transition radius of 2.2 m (70 px) and trajectory similarity window of 12.6 m (400 px).

Table II presents localization performance for GraphWalker and the trajectory projection baseline. GraphWalker reduces endpoint error by 45% in metric distance and 35% in graph

TABLE II
PRELIMINARY LOCALIZATION RESULTS FOR GRAPHWALKER COMPARED TO THE TRAJECTORY ENDPOINT PROJECTION. DISTANCE ERRORS ALONG THE TRAJECTORY ARE NOT MEANINGFUL FOR THE BASELINE.

Metric	Mean Distance Error [m]	Median Distance Error [m]	Max Distance Error [m]	Endpoint Error [m]	Endpoint Error [nodes]
Baseline	–	–	–	11.5	14.9
GraphWalker	1.7	1.4	4.2	6.3	9.7

distance compared to the baseline. Along-trajectory errors remain below 4.2 m, with median error 1.4 m.

IV. DISCUSSION AND FUTURE DIRECTIONS

The results demonstrate that trajectory preprocessing and beam-search alignment on semantic graphs improve localization accuracy over endpoint projection. GraphWalker leverages the full trajectory with structural constraints to mitigate accumulated drift from dead reckoning. Sensitivity analysis in Table I highlights that performance depends critically on the choice of internal parameters. An overly small hallway-transition radius prevents plausible corridor-to-corridor traversal, while a too-large radius allows invalid transitions that violate the building's structural constraints encoded in the graph. Similarly, too-small similarity window sizes reduce the effectiveness of trajectory scoring, while overly large windows dilute the influence of recent trajectory segments.

Performance depends on semantic graph quality and completeness. In this work, graphs are manually constructed. Automatic graph extraction from floor plans could improve scalability but requires validation. IMU odometry is often sensitive to motion irregularities including pauses, sidestepping, and abrupt direction changes. These artifacts propagate through alignment and degrade localization accuracy.

Future work will address three directions. First, extensive data collection across diverse buildings and longer trajectories will test generalizability and improve robustness to complex environments. Second, we will investigate multi-modal integration with lightweight sensors while preserving privacy and low cost. Third, adaptive scoring mechanisms and richer semantic features may improve trajectory-to-graph matching. Extension from localization to real-time navigation remains an open challenge.

V. CONCLUSION

We presented GraphWalker, a system for indoor localization that aligns IMU trajectories with semantic floor plan graphs using beam search to mitigate drift and produce plausible mapped trajectories. Preliminary evaluation on real-world walking trajectories shows 45% reduction in endpoint error compared to baseline projection, demonstrating feasibility for low-cost, privacy-preserving deployment using commodity smartphones. Future work will improve robustness to complex environments, integrate richer semantic signals, and extend toward real-time navigation applications.

REFERENCES

[1] S. Mostafa, K. A. Harras, and M. Youssef, "Unicellular: An accurate and ubiquitous floor identification system using single cell tower information," in *ACM SIGSPACIAL*, 2023.

[2] O. Hashem, K. A. Harras, and M. Youssef, "Deepnar: Robust time-based sub-meter indoor localization using deep learning," in *IEEE SECON*, 2020, pp. 1–9.

[3] S. Mostafa, K. A. Harras, and M. Youssef, "A survey of indoor localization systems for multi-floor environments," *IEEE Access*, 2025.

[4] O. Hashem, M. Youssef, and K. A. Harras, "Winar: Rtt-based sub-meter indoor localization using commercial devices," in *IEEE PerCom*, 2020, pp. 1–10.

[5] S. He and S.-H. G. Chan, "Wi-fi fingerprint-based indoor positioning: Recent advances and comparisons," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 466–490, 2015.

[6] R. Faragher and R. Harle, "Location fingerprinting with bluetooth low energy beacons," *IEEE journal on Selected Areas in Communications*, vol. 33, no. 11, pp. 2418–2428, 2015.

[7] A. Essameldin, M. Nurulhoque, and K. A. Harras, "More than the sum of its things: Resource sharing across iots at the edge," in *ACM/IEEE SEC*, 2020.

[8] K. Alkiek, M. Youssef, and K. A. Harras, "Earbender: Enabling rich imu-based natural hand-to-ear interaction in commodity earables," in *ACM PerCom*, 2023.

[9] A. Alarifi, A. Al-Salman, M. Alsaleh, A. Alnafessah, S. Al-Hadhrami, M. A. Al-Ammar, and H. S. Al-Khalifa, "Ultra wideband indoor positioning technologies: Analysis and recent advances," *Sensors*, vol. 16, no. 5, p. 707, 2016.

[10] D. Lymberopoulos, J. Liu, X. Yang, R. R. Choudhury, V. Handziski, and S. Sen, "A realistic evaluation and comparison of indoor location technologies: Experiences and lessons learned," in *Proceedings of the 14th international conference on information processing in sensor networks*, 2015, pp. 178–189.

[11] H. Taira, M. Okutomi, T. Sattler, M. Cimpoi, M. Pollefeys, J. Sivic, T. Pajdla, and A. Torii, "Inloc: Indoor visual localization with dense matching and view synthesis," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7199–7209.

[12] R. Harle, "A survey of indoor inertial positioning systems for pedestrians," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 3, pp. 1281–1293, 2013.

[13] S. Herath, H. Yan, and Y. Furukawa, "Ronin: Robust neural inertial navigation in the wild: Benchmark, evaluations, & new methods," in *2020 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2020, pp. 3146–3152.

[14] C. Chen, X. Lu, A. Markham, and N. Trigoni, "Ionet: Learning to cure the curse of drift in inertial odometry," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, no. 1, 2018.

[15] W. Liu, D. Caruso, E. Ilg, J. Dong, A. I. Mourikis, K. Daniilidis, V. Kumar, and J. Engel, "Tlio: Tight learned inertial odometry," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5653–5660, 2020.

[16] P. Newson and J. Krumm, "Hidden markov map matching through noise and sparseness," in *Proceedings of the 17th ACM SIGSPATIAL international conference on advances in geographic information systems*, 2009, pp. 336–343.

[17] G. Laignel, N. Pozin, X. Geffrier, L. Delevaux, F. Brun, and B. Dolla, "Floor plan generation through a mixed constraint programming-genetic optimization approach," *Automation in Construction*, vol. 123, p. 103491, 2021.

[18] P. Davidson, J. Collin, and J. Takala, "Application of particle filters for indoor positioning using floor plans," in *2010 Ubiquitous Positioning Indoor Navigation and Location Based Service*. IEEE, 2010, pp. 1–4.

[19] G. Grisetti, R. Kümmerle, C. Stachniss, and W. Burgard, "A tutorial on graph-based slam," *IEEE Intelligent Transportation Systems Magazine*, vol. 2, no. 4, pp. 31–43, 2011.

[20] S. M. La Valle, "Motion planning," *IEEE Robotics & Automation Magazine*, vol. 18, no. 2, pp. 108–118, 2011.

[21] Y. Ansari, A. Karkour, E. F. Flushing, and K. A. Harras, "Tesseract: Unfolding navigable graph representations from low-semantic floor plans," in *ACM SIGSPACIAL*, 2025.