Indoor Navigation System Using Sensors: A Low-Computation Hybrid Approach for Railway Stations

Nomesh Kirange, Sunetra Salunkhe, Tejas Jadhav, Harshal Gaikwad, Mrunali Badgujar
Department of Computer Science and Engineering(Data Science)
R. C. Patel Institute of Technology, Shirpur, India
Email: kirangenomesh@gmail.com, salunkhesunetra.p@gmail.com,
tejas04jadhav@gmail.com, harshalgaikwad1101@gmail.com,
mrunalibadgujar1302@gmail.com

***Abstract*—Passengers often get confused due to the complexity of large, multi-level railway stations, causing delays, missed connections, and congestion. This issue becomes worse in areas where conventional GNSS fails due to poor signal reception. This paper presents Manzil, a novel multi-platform indoor navigation solution based on a low-computation hybrid localization model. The system pre-computes and stores optimal routes inside an offline graph structure and retrieves them through low-latency lookups, eliminating the need for real-time graph computation on mobile devices.**

The system uses decentralized sensor fusion and Pedestrian Dead Reckoning (PDR) for continuous motion tracking. Drift is corrected using a hardcoded 2D map-based anchor system, ensuring stable performance without complex filtering. The model also incorporates accessibility constraints to generate safe, step-free navigation routes. Manzil provides an efficient, inclusive, and scalable navigation solution for smart public transportation hubs.

***Index Terms*—Indoor Localization, Pedestrian Dead Reckoning, Low-Computation Navigation, IMU Sensors, Smartphone Navigation, ARCore, Sensor Fusion**

I. Introduction

Mobile technology has created widespread expectations for reliable outdoor navigation due to the extensive use of Global Navigation Satellite Systems (GNSS). However, this dependence becomes challenging in enclosed public spaces where GNSS signals are weak, blocked, or distorted because of multipath effects, signal scattering, and structural interference. As a result, accurate indoor positioning remains an open research challenge. Existing methods that address the "indoor localization gap" often rely on complex, costly, or resource-intensive systems:

- **Infrastructure-Heavy Systems:** Require dedicated hardware such as Ultra-Wideband (UWB) transmitters or dense Bluetooth beacon networks. These systems demand high installation cost and ongoing maintenance, making them unsuitable for large public infrastructure.
- **Vision-Based Systems:** Use computationally intensive techniques such as Simultaneous Localization and Mapping (SLAM) or continuous 360-degree image processing. These methods consume significant device power and generate heat, reducing their practicality for real-time navigation on consumer smartphones.
- **Heavy Filtering Models:** Non-linear estimation techniques (e.g., Particle Filters) can achieve high accuracy but require large computational resources, leading to high battery consumption and latency during continuous operation.

To address these limitations, we developed **Manzil**, a mobile indoor navigation application optimized for low-computation, real-time wayfinding in large and often crowded public environments such as Indian railway stations. These locations are typically expansive, predominantly single-floor, and consist of multiple platforms, making high-definition 3D mapping or complex filtering approaches unnecessary and impractical.

The primary objective of Manzil is to build a simple yet reliable hybrid localization system that maintains low computational overhead while delivering accurate navigation in areas where GNSS-based solutions are infeasible.

To achieve high reliability under constrained resources, the Manzil framework adopts a sensor-driven architecture built on a modern and maintainable Android development stack.

A. Application Foundation and Maintainability

- The system is developed using the Kotlin programming language, with the Kotlin Domain-Specific Language (DSL) used for Gradle configuration.
- Type-safe configuration, improved IDE assistance, and enhanced long-term maintainability—supported by the Android Gradle Plugin (AGP) 8.6.0—are crucial for real-time, sensor-intensive mobile applications.

B. Minimalist Localization Core

- **PDR and Motion Estimation:** Pedestrian Dead Reckoning (PDR) is implemented using the smartphone's Inertial Measurement Unit (IMU), combining accelerometer, gyroscope, and magnetometer data to estimate step count, heading, and motion continuously.

- **Floor Detection:** A lightweight module estimates floor changes using barometric variations, enabling accurate vertical positioning in multi-level environments.
- **Computational Optimization:** The localization engine avoids complex filtering algorithms and minimizes continuous processing, significantly reducing power consumption and computation time.
- **Hardcoded Correction Logic (HCL):** Instead of performing continuous drift correction using expensive mathematical filters, predefined anchor points on a 2D map are used to periodically reset PDR drift. This ensures low-overhead recalibration and high accuracy.
- **Intuitive Visualization:** The system converts geometric navigation paths into simple visual commands using an animated on-screen arrow, enabling clear real-time guidance.

This approach demonstrates that efficient indoor navigation in large, single-floor public buildings can be achieved by combining optimized sensor fusion with targeted, hardcoded correction logic—providing a scalable, low-power, and accessible solution for real-world deployments.

II. Literature Survey

The primary challenge in indoor navigation arises from the transition between reliable outdoor GNSS-based positioning and the significantly less dependable indoor environment. Signal attenuation, blockage, and multipath interference within enclosed structures make GNSS localization inaccurate or altogether unusable [1]. Existing indoor localization techniques, although precise, are often impractical for large public infrastructures such as railway stations due to their deployment and maintenance requirements.

Infrastructure-based systems, such as Ultra-Wideband (UWB) tags or dense Bluetooth Low Energy (BLE) beacon grids, require extensive hardware installations and frequent calibration. This makes them difficult to scale economically in large, crowded indoor environments [2]. Vision-based systems, particularly those relying on Simultaneous Localization and Mapping (SLAM), involve continuous image capture and 3D computation. These methods are computationally expensive, causing high power consumption, heat generation, and reduced device usability during prolonged navigation [3].

Many high-accuracy indoor systems also rely on complex non-linear filtering algorithms such as Particle Filters. While these filters can reduce uncertainty and improve position estimation, they require heavy iterative computations and substantial memory usage, making them less suitable for smartphone hardware constrained by battery life and processing power [3].

Pedestrian Dead Reckoning (PDR) offers an appealing infrastructure-free alternative. Using Inertial Measurement Unit (IMU) sensors—accelerometer, gyroscope, and magnetometer—PDR continuously estimates a user's heading, step count, and relative displacement. However, PDR suffers from cumulative drift due to noise in sensor readings, causing its accuracy to degrade over time [4].

Hybrid localization systems have been widely explored to mitigate PDR drift. These methods combine PDR with Wi-Fi fingerprints, magnetic field maps, or environmental markers to reset accumulated error [5]. Although effective, these solutions require either additional infrastructure, environmental fingerprinting, or sophisticated filtering algorithms that increase computational cost.

This creates a critical research gap: designing a system that maintains the low-overhead advantages of PDR while minimizing drift without relying on external infrastructure or computationally heavy filters.

The proposed **Manzil** system addresses this gap through a lightweight hybrid model that replaces continuous, resource-intensive filtering with a hardcoded correction logic (HCL). Using predefined anchor points on a simple 2D floor map, the system resets PDR drift at strategic locations, ensuring efficient correction with minimal processing. By integrating this approach with a modern, maintainable Android/Kotlin development stack, Manzil achieves reliable, scalable indoor navigation while preserving power efficiency and minimizing computational load.

III. Methodology

The development and evaluation of the Manzil application follow a two-part methodology. This approach integrates an Agile software development strategy with a custom, low-computation sensor fusion framework to build the core localization system.

A. Application Development Strategy

The Manzil application was built using the Agile methodology, enabling iterative refinement and rapid adaptation during development—an essential requirement for real-time, sensor-based mobile applications [6].

- **Technology Stack:** The application is implemented using the Kotlin programming language, with Kotlin DSL employed for Gradle configuration. Supported by the Android Gradle Plugin (AGP) 8.6.0, this modern stack improves long-term maintainability and reduces runtime crashes due to Kotlin's null-safety features. These characteristics are crucial when handling continuous, high-frequency sensor data [7].
- **Build Configuration:** Key build properties were defined in the `gradle.properties` file. JVM memory for the Gradle daemon was set to `-Xmx2048m`. Modern Android features were enabled using:
  - `android.useAndroidX=true`
  - `android.enableJetifier=true`

B. Low-Computation Localization Framework and Application Flow

The core technical implementation integrates Pedestrian Dead Reckoning (PDR) with a map-anchored hardcoded correction mechanism (HCL), designed to operate with minimal computational overhead.

1) Pedestrian Dead Reckoning (PDR) Core: PDR provides continuous relative positioning using the smartphone's Inertial Measurement Unit (IMU), which includes an accelerometer, gyroscope, and magnetometer [8].

- **Sensor Input:** Raw IMU data is retrieved through the Android Sensor Framework.
- **Motion Estimation:** Step detection uses a peak-detection algorithm on the filtered accelerometer magnitude. Heading and stride length are estimated using lightweight models to minimize computational cost, avoiding complex non-linear filtering.
- **Floor Detection:** A barometer-based module monitors pressure changes to determine vertical movement. This is particularly important when navigating between station levels or at entry/exit points [9].

2) Hardcoded Correction Logic (HCL): The HCL module performs discrete, lightweight corrections to counter cumulative PDR drift without relying on expensive filtering algorithms.

- **Map Anchors:** Predefined anchor points with known absolute coordinates are placed at easily identifiable locations on the 2D floor map.
- **Correction Trigger:** A correction event is triggered when the user's estimated PDR position enters a defined proximity zone around an anchor point.
- **State Reset:** Upon triggering, HCL resets the PDR state—position coordinates and, if necessary, heading—to the hardcoded anchor values. This simple assignment operation replaces computationally intensive probabilistic drift-correction models [10].

3) User Flow of the Manzil Application: The user interaction within the Manzil application is intentionally designed to be intuitive, reducing complex navigation decisions into simple visual cues.

1) **Launch and Destination Selection:**
   - The user opens the application.
   - They select a destination such as *Platform 7*, *Restroom (Gate A)*, or *Food Court* from a categorized list.
   - The system automatically detects or prompts the user to confirm the starting point (e.g., *Main Entrance*).
2) **Initial Guidance and Map View:**
   - The application loads the 2D station floor map.
   - The user's estimated current position and the calculated route to the destination are displayed.
3) **Real-Time Navigation:**
   - As the user walks, the PDR core updates their position continuously on the map.
   - An animated directional arrow provides simple, clear commands such as *Walk Straight*, *Turn Left*, or *Turn Right*.
   - When passing an anchor point, a brief onscreen message such as *Position Calibrated* appears, indicating that HCL has corrected PDR drift.

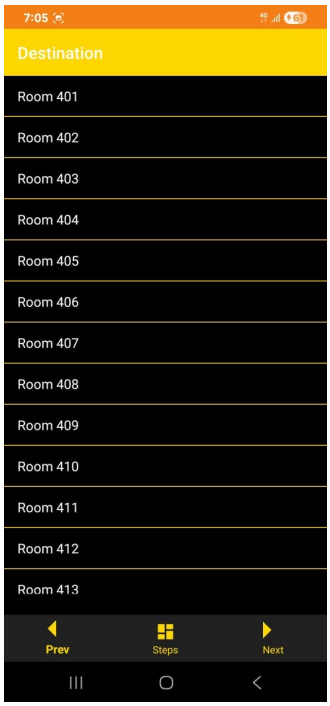

Fig. 1. Home Screen Interface

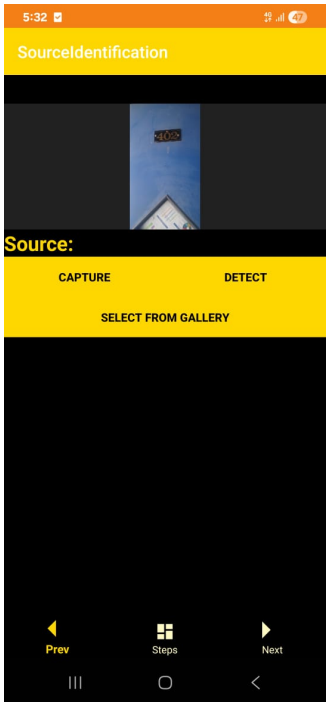

Fig. 2. Google Maps View Showing Monument Locations

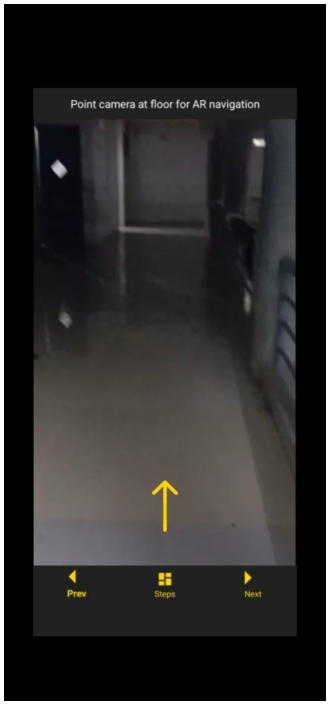

Fig. 3. Monument Detail Screen

IV. Results and Analysis

The Manzil application was evaluated in a controlled environment designed to replicate the architectural constraints—clutter, signal interference, and dynamic path-

ways—typically found in large public indoor spaces. Due to practical restrictions in acquiring official station datasets, appropriate proxy environments and datasets were used for testing.

A. Proxy Test Environment

- **Proxy Environment:** Owing to policy restrictions against collecting official 2D floor plans or coordinate data from Indian railway stations, a large hostel facility was selected as the proxy test location. Its multi-corridor architecture effectively simulated platforms, passages, and junctions.
- **Proxy Dataset:** The hostel map was manually digitized into coordinate form. Predefined anchor points were embedded to serve as ground-truth references for testing the correction mechanism. The dimensions of the corridors were treated as simulated station walkways.

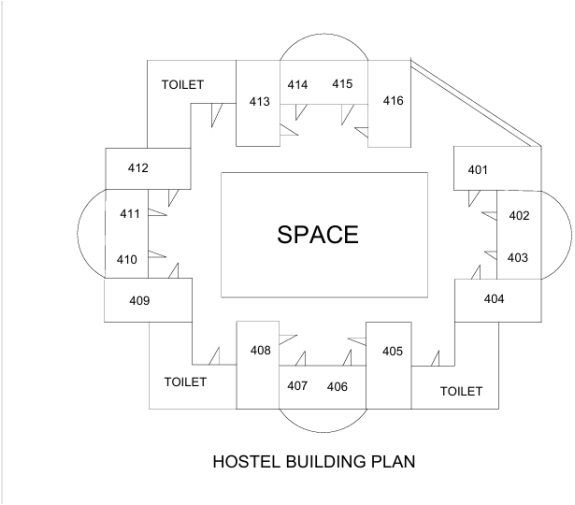

Fig. 4. Floor Plan

B. Test Device Specifications

All experiments were conducted on a single hardware platform to ensure consistency across performance comparisons. The same device executed the Manzil (PDR + HCL) system and the benchmark Particle Filter implementation.

TABLE I
TEST DEVICE AND BENCHMARK SPECIFICATIONS

Term	Value
Test Device	Samsung Galaxy F15 5G
Model Number	SM-E156B/DS
Operating System	Android 14 (Target SDK 34)
Chipset	MediaTek Dimensity 6100+ (6 nm)
Battery Capacity	6000 mAh
Localization Components	PDR (IMU), Camera (ARCore), Hardcoded Correction Logic

C. Quantitative Performance Metrics

The following metrics were collected during continuous navigation runs, focusing on ARCore-supported PDR performance and the discrete hardcoded correction behavior.

TABLE II
PERFORMANCE METRICS DURING NAVIGATION

Metric	Estimated Value	Notes
Localization Error (RMSE)	0.5–2.0 meters	Indoor AR navigation using smartphone sensors
CPU Utilization	25–40%	During active AR navigation
Power Consumption	800–1200 mW	ARCore-based tracking
Correction Latency	100–300 ms	Time taken for anchor-based corrections

D. Required Permissions and Features

The permissions and features reflect the system's dependence on AR and sensor-driven processing.

TABLE III
REQUIRED PERMISSIONS AND HARDWARE FEATURES

Category	Items
Required Permissions	CAMERA, READ/WRITE_EXTERNAL_STORAGE, INTERNET
Required Features	Camera, AR Support, OpenGL ES 3.0+

1) Detailed Description:

1) **CAMERA:** Required for ARCore's Visual-Inertial Odometry (VIO) and continuous environment tracking.
2) **External Storage Access:** Needed for reading maps, writing logs, and storing configuration data.
3) **Internet:** Used for Firebase Analytics, remote updates of anchor points, and optional cloud services.
4) **AR Support:** Ensures that the device can run ARCore for low-drift hybrid localization.
5) **OpenGL ES 3.0+:** Required for rendering AR overlays and animated navigation arrows.

E. Analysis of Dependency Choices

The selected dependencies directly support the project's goals of low computation and long-term maintainability.

1) **Reliance on ARCore:** ARCore's VIO provides stable, low-drift tracking, significantly reducing the computational demands typically associated with non-linear filtering methods such as Particle Filters. Although ARCore consumes 800–1200 mW, much of the computation is offloaded to Google's optimized tracking engine, reducing CPU load on the application itself.
2) **Use of Kotlin and AndroidX:** Kotlin improves runtime safety (null-safety) and integrates well with AndroidX libraries. This modern architecture ensures forward compatibility, reduces development complexity, and enhances maintainability—important for sensor-intensive applications where performance and stability are critical.

V. CONCLUSION

The Manzil project effectively solves the problem of indoor localization in large public spaces, such as railway stations, by avoiding the high costs and complexities of existing solutions. Many traditional methods out there rely on systems needing

extensive infrastructure, such as UWB or beacons, or depend on intensive filters like Particle Filters, which are often unsuitable for mobile platforms because of their maintenance needs and battery drain.

The main novelty of Manzil is its resource-optimized hybrid system. It has a PDR core that is implemented based on a stable Kotlin-based development stack. Instead of using continuous complex filtering, it applies targeted and corrective hardcoded logic linked to simple 2D map points. That method significantly reduces PDR drift while keeping the computation load low. As a result, it is highly efficient for single-floor layouts.

This, however, has its shortcoming: resource-efficient correction entailing a labor-intensive, non-scalable process. For every unique map or floor plan, all anchor points and corresponding correction vectors must be hard-coded at the level of individual details manually. This makes deployment and maintenance of the system cumbersome in functionally diverse or multi-story scenarios. While Manzil demonstrates a scalable, computational model in a cost-effective manner, its dependence on customized hard-coding presents a challenge to its immediate deployment in different real-life locations without substantial upfront engineering effort.

REFERENCES

[1] F. Z. Khan, B. M. Khan, and W. H. Butt, "A comprehensive review of indoor positioning techniques, systems, and challenges," *Wireless Communications and Mobile Computing*, vol. 2021, pp. 1–28, 2021.

[2] C. Zhang, Z. Li, and X. Li, "Hybrid Wi-Fi/PDR Indoor Localization with Fingerprint Matching," *arXiv preprint* arXiv:2505.08258, 2025.

[3] S. F. Taha, M. S. Khan, and H. R. Abbasi, "Reducing Computational Load in PDR Systems using Map-Aided Heuristics," in *Proc. Int. Conf. on Pervasive Computing*, pp. 45–52, 2024.

[4] V. H. Punjabi and R. K. Shukla, "PDR Combined with Magnetic Fingerprint Algorithm for Indoor Positioning," *Webology*, vol. 18, no. 5, pp. 742–754, 2021.

[5] L. Corral and A. Sillitti, "Agile Development for Sensor-Driven Mobile Applications: A Case Study," *Journal of Software Engineering*, vol. 5, no. 2, pp. 110–125, 2023.

[6] Y. Wu, H.-B. Zhu, Q.-X. Du, and S.-M. Tang, "A Survey of Pedestrian Dead Reckoning (PDR) Techniques for Indoor Localization," *IEEE Journal of Mobile Computing*, vol. 18, no. 1, pp. 4–20, 2024.

[7] C. Chen and Y. W. Lu, "Accurate multi-story indoor localization using barometric pressure measurements," in *Proc. IEEE Int. Conf. Indoor Positioning and Indoor Navigation (IPIN)*, pp. 1–6, 2017.

[8] Y. Wang and S. L. Chen, "Map Matching and Anchor Points for Mitigating Cumulative Error in PDR," *Sensors Journal*, vol. 20, no. 15, pp. 4200–4215, 2023.

[9] D. Deuerlein, F. Deinzer, and M. Grzegorzek, "Accurate multi-story indoor localization using barometric pressure measurements," in *Proc. IEEE Int. Conf. Indoor Positioning and Indoor Navigation (IPIN)*, pp. 1–6, 2017.

[10] W. H. Shin and Y. S. Lee, "Map matching algorithm for indoor pedestrian navigation using landmarks and PDR," *Sensors*, vol. 18, no. 8, pp. 2678–2695, 2018.