Landmark Graph-Based Indoor Localization[1]

Luca Polese

Alma Mater Studiorum - Università di Bologna

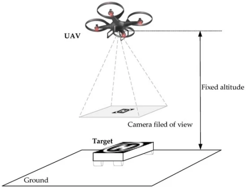
Tuesday 18th June, 2024

Global Navigation Satellite System

- **GNSS** has been successfully applied in many fields:

(a) Car navigation

(b) Geofencing



(c) Target tracking

Figure 1: Some examples of application

- It's **difficult to use** for inside location: **signal** is *blocked* by buildings, trees, obstacles, ...

Indoor localization

- It's even more **challenging**:
  - Indoor spaces are more **complicated** in terms of *layout*, *topology*, *space constraint*;
  - Indoor applications need more **accuracy**.
- Multiple systems have been proposed in recent years:

(a) WiFi (b) Zigbee (c) Bluetooth (d) Ultra-wideband

Figure 2: Some examples of indoor localization systems

- Each technique has **drawbacks** in terms of accuracy, cost coverage, complexity and applicability.

Landmark

Landmark for **Indoor Localization**

A landmark is a **spatial constraint**.
It's a location point where *at least one sensor* shows a **distinctive**, **stable**, and **identifiable pattern**.

Advantages

- **Naturally distributed** location points in indoor environments;
- **Easy to bound** the localization **error** with no extra cost.

Disadvantages

- **Economically** and **computationally expensive** systems;
- **Performance** depends on **completeness of landmarks**.

LG-Loc

Landmark-Guided Localization

It's a novel **graph-based indoor localization** method for smartphones

Compared with existing landmark-based localization methods:

- Computationally **efficient**;
- Handles **incomplete landmarks**.

Landmark graph

It's a **directed graph** where nodes are landmarks and edges are accessible paths with heading information

LG-Loc ∼ Phases & Challenges

This method consists of two **phases:**

1. *Offline:*
   - **Collect data** from several **smartphone sensors**;
   - **Construct** the **initial landmark graph** and **update it** with more landmarks.
2. *Online:*
   - Use newly collected data for **location initialization**, **estimation**, and **calibration**.

Challenges:

1. Infer the **initial location** without **manual input**.
2. **Recognize** landmarks **effectively**.
3. Handle **missing landmarks**.

LG-Loc System Architecture

- Users launch the **localization application** upon entering a **building** and obtains his **location**;
- Application requests the building's **landmark graph**;
- Data used for **landmark detection** and **location estimation**;
- Initial location obtained via:
    - *WiFi fingerprinting* (if database is **available**);
    - *HMM-based algorithm* (if database is **unavailable**).

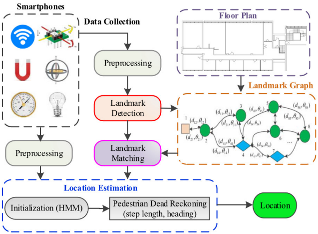


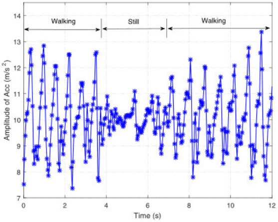
Figure 3: System architecture of LG-Loc

Phase A $\sim$ Definition and Recognition of Landmarks

- $\forall$ landmark $\exists$ 3 features:
  - **Distinctiveness**: Unique patterns distinguishable from surroundings;
  - **Stability**: It doesn't change dynamically over time;
  - **Identifiability**: Detectable by one or more sensors.
- Mathematical definition: $v = \langle (x, y), (R_1, \ldots, R_M) \rangle$
  - (x, y): coordinate of the landmark;
  - $(R_1, \ldots, R_M)$: detection rule in different types of sensor readings;
  - M is the number of rules that this landmark possesses.

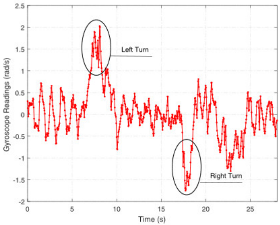
Types of Landmarks

- **Accelerometer Landmark**: Detects changes in **motion state**;



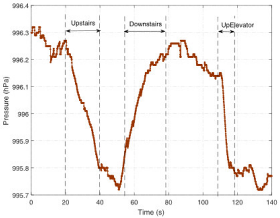
Types of Landmarks

- **Accelerometer Landmark**: Detects changes in **motion state**;
- **Gyroscope Landmark**: Detects changes in **walking direction** using magnetometer & gyroscope;



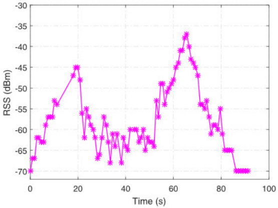
Types of Landmarks

- **Accelerometer Landmark**: Detects changes in **motion state**;
- **Gyroscope Landmark**: Detects changes in **walking direction** using magnetometer & gyroscope;
- **Barometer Landmark**: Measures **air pressure**, which changes with **altitude/height**;



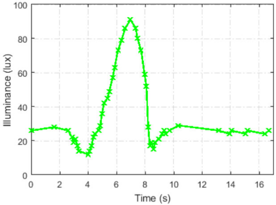
Types of Landmarks

- **Accelerometer Landmark**: Detects changes in **motion state**;
- **Gyroscope Landmark**: Detects changes in **walking direction** using magnetometer & gyroscope;
- **Barometer Landmark**: Measures **air pressure**, which changes with **altitude/height**;
- **WiFi Landmark**: Detects location point that overhears the **strongest RSS from an AP**;



Types of Landmarks

- **Accelerometer Landmark**: Detects changes in **motion state**;
- **Gyroscope Landmark**: Detects changes in **walking direction** using magnetometer & gyroscope;
- **Barometer Landmark**: Measures **air pressure**, which changes with **altitude/height**;
- **WiFi Landmark**: Detects location point that overhears the **strongest RSS from an AP**;
- **Light Landmark**: Detects changes in **light intensity** using light sensor;



Phase B ∼ Construction of Initial Landmark Graph

- Location of landmarks correspond to the **location of an element** inside the **floor map**;
- The **landmark graph** is constructed starting from **map information**:
  - *Incomplete*: lacks room-level details (e.g., furniture, desks);
  - Cannot locate WiFi and light landmarks from floor plan.
- **Ingredients**:
  - Nodes: Landmarks;
  - Edges: Accessible paths between landmarks;
  - Graph Representation: $G = (V, E)$:
    - $V = \{v_1, \ldots, v_N\}$ (set of landmarks);
    - $E = \{e_1, \ldots, e_M\}$ (set of edges).

Phase C ∼ Updating of Landmark Graph

- **Learning New Landmarks**:
  - Crowdsourcing: Collect N user trajectories;
  - Each trajectory contains n_i potential landmarks $\{v_1^i, v_2^i, \ldots, v_{n_i}^i\}$.
- **Potential Landmarks**:
  - Location points **satisfying** (at least) one **detection rule**;
  - **Not yet included** in the current **landmark graph**.
- **Updating Algorithm**:
  - *Process*: Updates a landmark graph by **clustering potential landmarks** based on *distance* and *rules*, removing clusters with insufficient elements. Then ads new nodes and edges to the graph based on the cluster centers and detection rules.

Challenges in Using Landmarks for Localization

- **Relevant challenges to tackle**:
  - **Data Association Issue**: multiple landmarks nearby $\rightarrow$ difficult to determine the detected landmark;
  - **Missing Landmarks**: Addressing cases where one or more landmarks are absent.
- To solve the previous points it has been defined a **belief**:
  - Belief (*bel*): Indicates **trust level** that a location point **matches a landmark**;
  - Belief Calculation Formula:

$$bel(v_k) = \delta(R_k, R_t) \cdot r(\theta_k, \theta_t) \cdot g(d_k, d_t)$$

  - Components:
    - $\delta(R_k, R_t)$: Dirac delta function for *detection rule matching*;
    - $r(\theta_k, \theta_t)$: Rectangle function for *heading comparison*;
    - $g(d_k, d_t)$: Exponential function for *distance comparison*.

Functions Definitions

- Dirac Delta Function (δ):

$$\delta(R_k, R_t) = \begin{cases} 1, & \text{if } R_k == R_t \\ 0, & \text{otherwise} \end{cases}$$

- Rectangle Function (r):

$$r(\theta_k, \theta_t) = \begin{cases} 1, & \text{if } |\theta_k - \theta_t| < \theta \text{ threshold} \\ 0, & \text{otherwise} \end{cases}$$

- Exponential Distance Function (g):

$$g(d_k, d_t) = e^{-|d_k - d_t|}$$

Landmark Selection and Error Correction

- Select the **landmark** with the **highest** **(***bel***)**;
- Use (bel) threshold to **exclude fake landmarks**;
- **Ignore missing landmarks** and do not correct the location until the next landmark is detected.

Location Initialization

Standard methods

- *WiFi Fingerprinting Method*:
  - **Matches** new *WiFi fingerprints* with a *pre-collected database*
  - Requires **offline training**, which is time-consuming and labor-intensive
- *User Input Method*:
  - User inputs **initial location** when launching the app
  - Easy deployment but requires **active user participation**

Landmark Graph-based HMM Approach

- Initially the app has **no location information** of the user;
- User walks to **collect sensor readings** for location estimation;
- Sensor readings used to **fed** the **landmark recognition module**;
- This way it generates **observations** to detect landmarks.

Experimental Setup

- *Location*: Office building with eight floors;
- *Environment*: Elevators, staircases, corridors, common rooms, office rooms;
- *Testing path*: Two floors, approximately 362 meters long;
- *Device*: Google Nexus 6 smartphone;
- *Sensors*: WiFi, accelerometer, magnetometer, gyroscope, barometer, light sensor;
- *Participants*: Six volunteers;
- *Task*: Walk preset path, report markers for location accuracy.

Data Collection

How the experient was carried out

The participants walked along the preset path with the phone in hand and reported the preset markers they encountered to evaluate the location accuracy.

- Recorded data: **MAC addresses** of visible WiFi access points and **RSS**;
- All data timestamped for alignment and inference;
- Participants clicked markers on Android app to set the locations.

Step Counting and Step Length Estimation

- **Step counting step detection** and **step length** estimation have an impact on the *accuracy of localization*: participants were asked to walk *300 steps*;
- Accuracy of step counting:
  - Detection with/without constraints: $> 94\%$;
- Step length estimation:
  - Step length influenced by walking frequency (step periodicity);
  - Step periodicity stability: Most steps in [0.5, 0.7] interval.

Initial Location Determination

- Ten random starting points along the path;
- Tested the HMM-based method to initialize th localization system;
- **Results**: Average distance a user has to travel to determine initial location is ~ 9.95 m.

Numerical results

- *Performance*:
  - *Proposed method* : **88% accuracy** with **error** $<$ **1.5 m**;
  - *Map filtering method* : **63% accuracy**;
  - *WiFi fingerprinting method* : **33% accuracy**;
  - *PDR methods* : **Worst outcome** $\rightarrow$ lack of spatial constraints.
- *Mean error*:
  - *Proposed method* : **0.80 m**;
  - *Map filtering method* : $\sim$ **1.7 m**;
  - *WiFi fingerprinting method* : **3.5 m**;
  - *PDR methods* : $>$ **7 m**.
- *Efficiency* - comparison with *Map filtering method* :
  - $\sim$ 5 times faster;
  - Landmark graph-based correction, no wall/obstacle detection;
  - Suitable for resource-limited platforms.

Discussion on Landmark Graph-based Indoor Localization

Achievements

- Low-cost, robust, high accuracy (under 1 m);
- Reduces human effort for initial location input;
- Belief metric for accurate landmark matching.

Conclusion

- Novel, low-cost, high-accuracy indoor localization
- Sensor-based landmark detection with location correspondence
- Better accuracy and computational efficiency than existing methods

ToDos

- Sensor power consumption in smartphones;
- New thresholds for landmark detection: machine learning;
- Cooperation between multiple devices: exchange location and distance.

Bibliography

[1] F. Gu, S. Valaee, K. Khoshelham, J. Shang, and R. Zhang, "Landmark graph-based indoor localization," *IEEE Internet of Things Journal*, vol. 7, no. 9, pp. 8343–8355, 2020. DOI: 10.1109/JIOT.2020.2989501.