# An Android Application for Pedestrian Localization

## using WiFi fingerprinting and inertial navigation

developed in Android Lollipop 5.0.1

Computer Engineering, Università di Pisa

Mobile And Pervasive Systems

A.Y. 2017

Nicola Messina

# Motivation

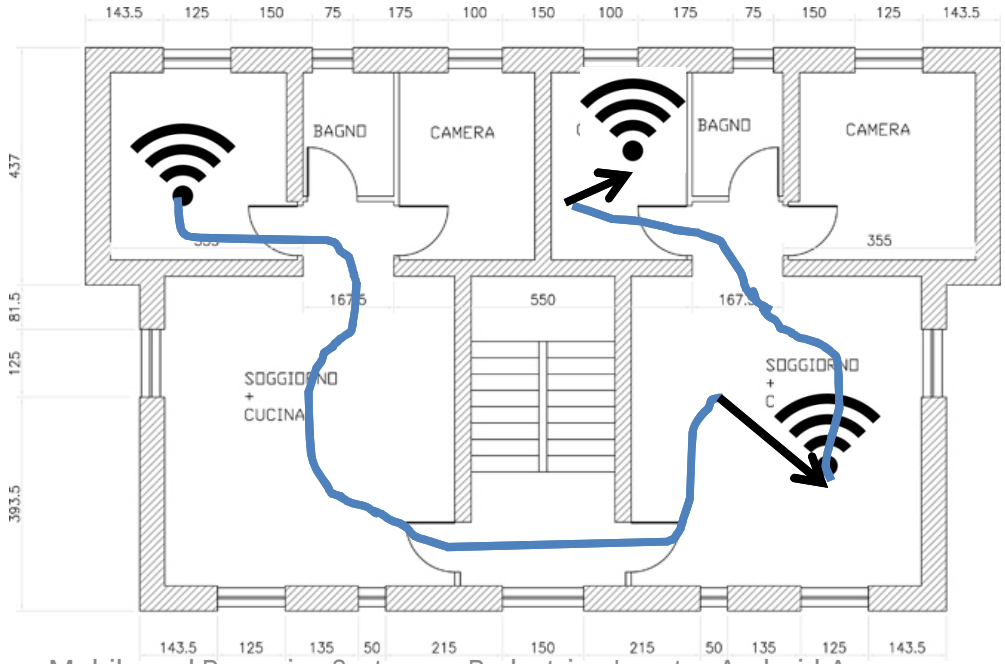
- Requirement of obtaining location-aware service is rapidly increasing, due to
  - Rapid development of mobile communication and pervasive computing technology
  - Need for location awareness in most real life applications
    - Navigation
    - Object or people tracking

**Pedestrian tracking**

- **GPS** is the most common positioning service
  - Power hungry and **not always available**
- What if we want to locate peoples **indoor?**
  - Firefighters inside a building
  - Old people movements inside their apartments

# Application Concept

- Use smathphone capabilities to reconstruct pedestrian movement without relying on GPS
  - **Wifi** for absolute, coarse grain location synchronization
  - **Inertial Sensors** (compass, gyro, accelerometer) for relative, fine grain location estimation

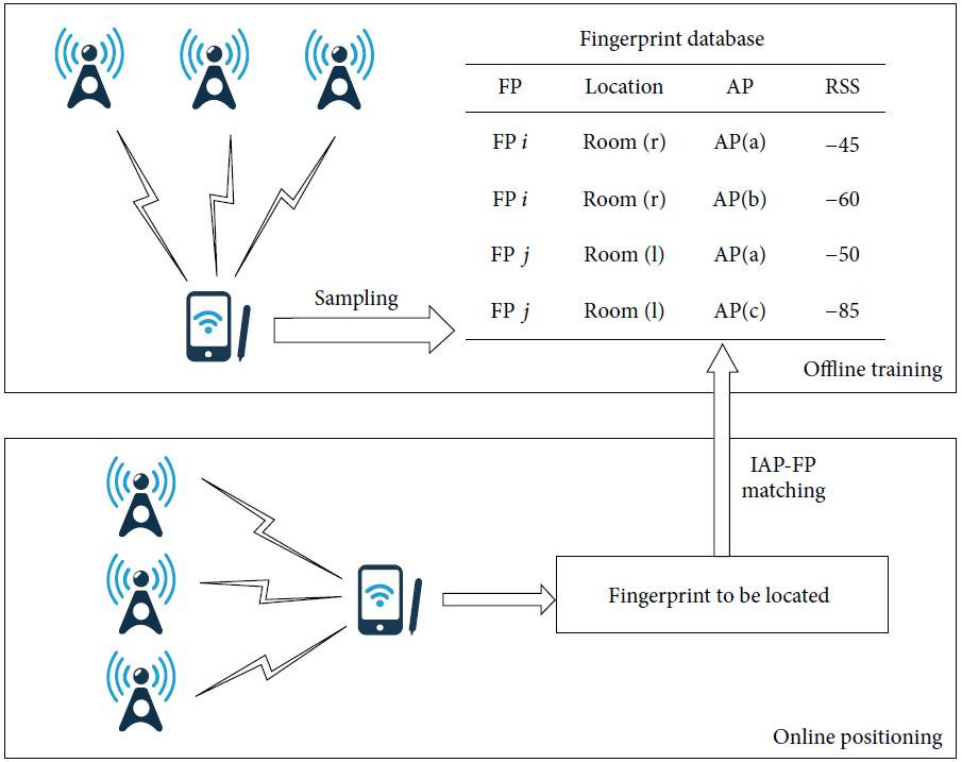


# Wifi Localization

- Range free method based on **wifi fingerprints**
  - **Fingerprint:**
    - set of access point s(APs) BSSID
    - received signal strength from each BSSID
    - geographical coordinates
- Wifi AP can be sensed since periodically they emit beacon frames carrying the BSSID and used by the Wifi receiver to estimate the RSSI.
  - Delay between scans is high (between 3-5 seconds) since receiver must listen for a certain period of time on all channels sequentially

# Wifi Phases

- **Offline phase**: the fingerprint is acquired, associated to a certain position and stored in a database
- **Online phase**: the fingerprint in an unknown location is built and compared with the ones in the database



# Wifi Matching

- **Online** phase:
  - The best K fingerprints found in the database are used to estimate current phone location
  - Similarity criterion needed for defining the best K fingerprints
    - Needed the concept of **distance** among fingerprints. Lots of distances introduced in fingerprinting literature

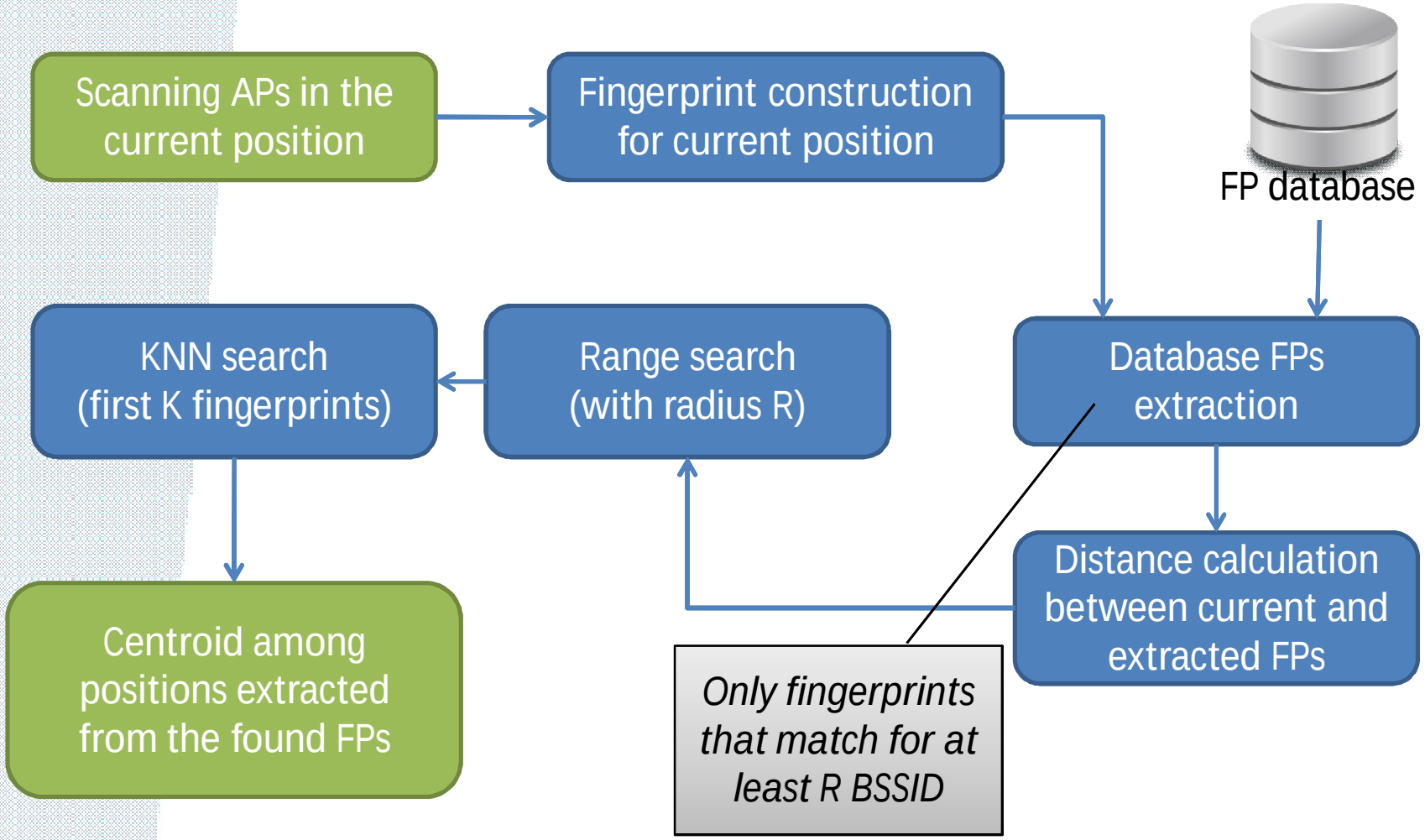
$$d = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(RSS_{mi} - RSS_{pi})^2}$$

AP orientated Euclidean Distance [3]

BSSIDs of fingerprint A that don't compare in fingerprint B are considered having strength of -120 db

# Wifi Matching, position estimation

- Wifi matching flow:



# Inertial Navigation

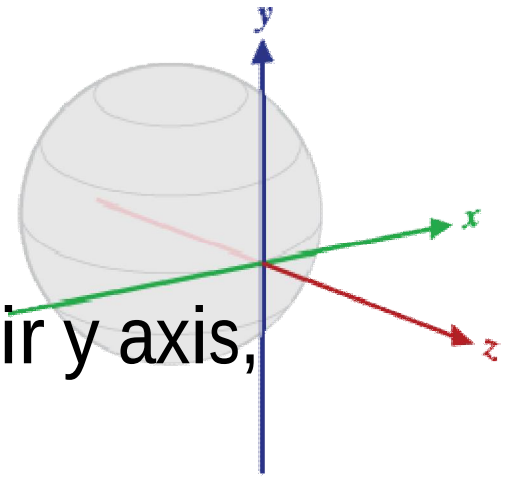
- Once initial position is known, if we know **magnitude**(Δs) of movement and its **direction**(θi), we car reconstruct the pedestrian path.

$$\begin{cases} x_{i+1} = x_i + \Delta s \cos\theta_i \\ y_{i+1} = y_i + \Delta s \sin\theta_i \end{cases}$$

  - Initial position given by previously seen wifi location algorithm
  - Magnitude given by step length (step detected through accelerometer)
  - Direction estimated using Android rotation sensor

# Direction Estimation (1)

- Android rotation sensor uses **sensor fusion** (using compass, accelerometer, gyroscope) to obtain the phone rotation to respect earth reference frame
  - It outputs a rotation matrix $R_{wp}$ that express rotation of phone to respect world
- We need to estimate $R_{wu}$
  - Rotation of **user** to respect world
- If phone and users are aligned on their y axis, there is no problem.
  - What if the phone is facing an arbitrary direction to respect the user? (e.g. user has the phone in the pocket)



# Direction Estimation (2)

- We need to estimate $R_{up}$, the rotation matrix that transforms user space into phone space
  - We estimate it in a **calibration process,**
    - The user keeps the phone facing his own direction ($R_{wu}$ is known), then he puts it in the pocket ($R_{wp}$ is known)

$$R_{up} = \bar{R}_{wu}^{T} \bar{R}_{wp}$$

  - Given $R_{up}$ then, the rotation $R_{wu}$ can be estimated:

$$R_{wu} = R_{wp} R_{up}^{T}$$

  - So now direction unit vector(x,y) can be estimated:

$$[x, y, 0]^{T} = R_{wu}[0, 1, 0]^{T}$$

Y points toward north

# Direction Filtering

- Calibration matrix $R_{up}$ filtering
  - Both $R_{wu}$ and $R_{wp}$ are sensed 15 times. Then, their mean is computed

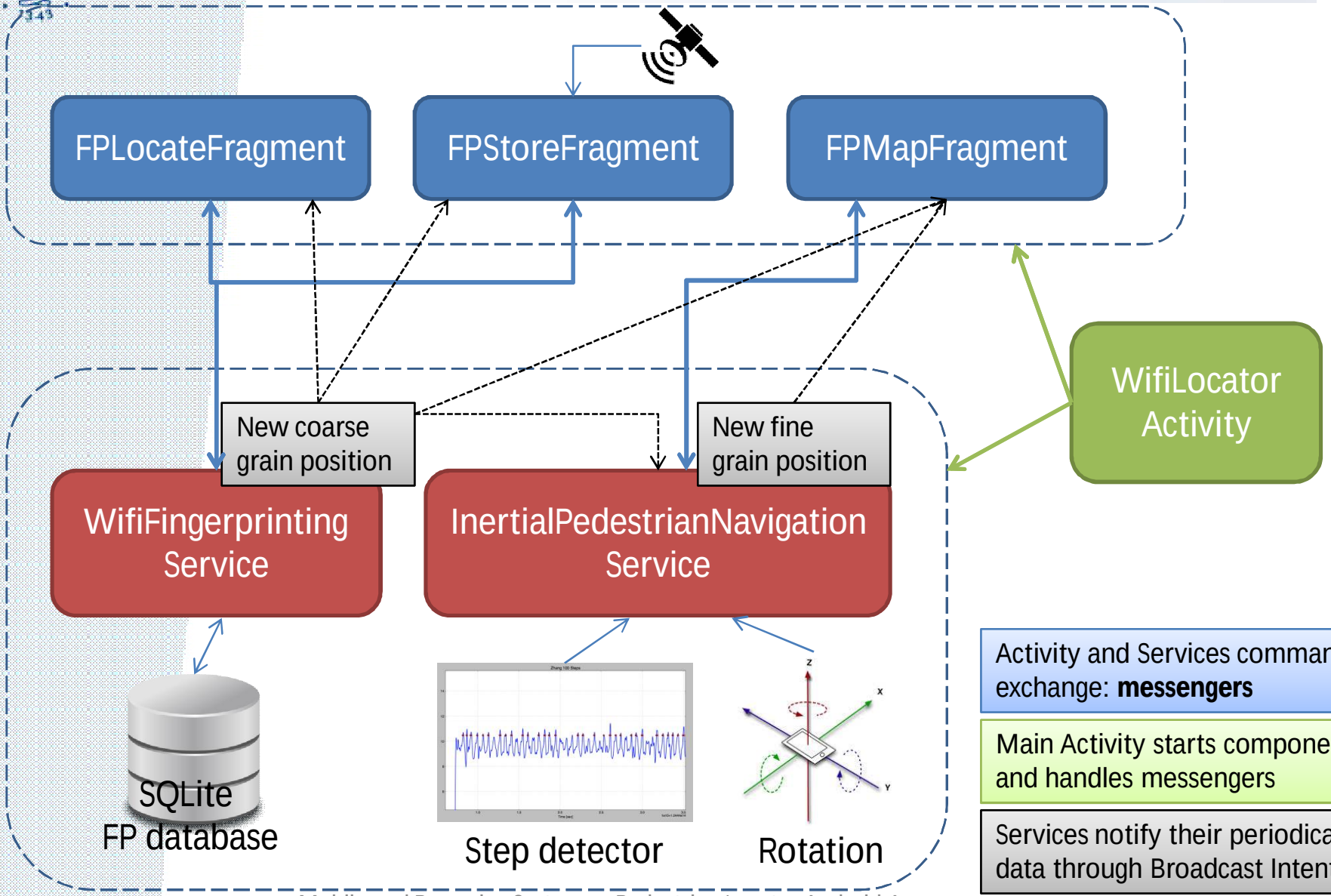
- Direction vector (x,y) filtering:
  - exponential moving average filtering applied on both dimensions

$$S_t = \begin{cases} Y_1, & t = 1 \\ \alpha \cdot Y_t + (1 - \alpha) \cdot S_{t-1}, & t > 1 \end{cases}$$

# Application Structure (1)

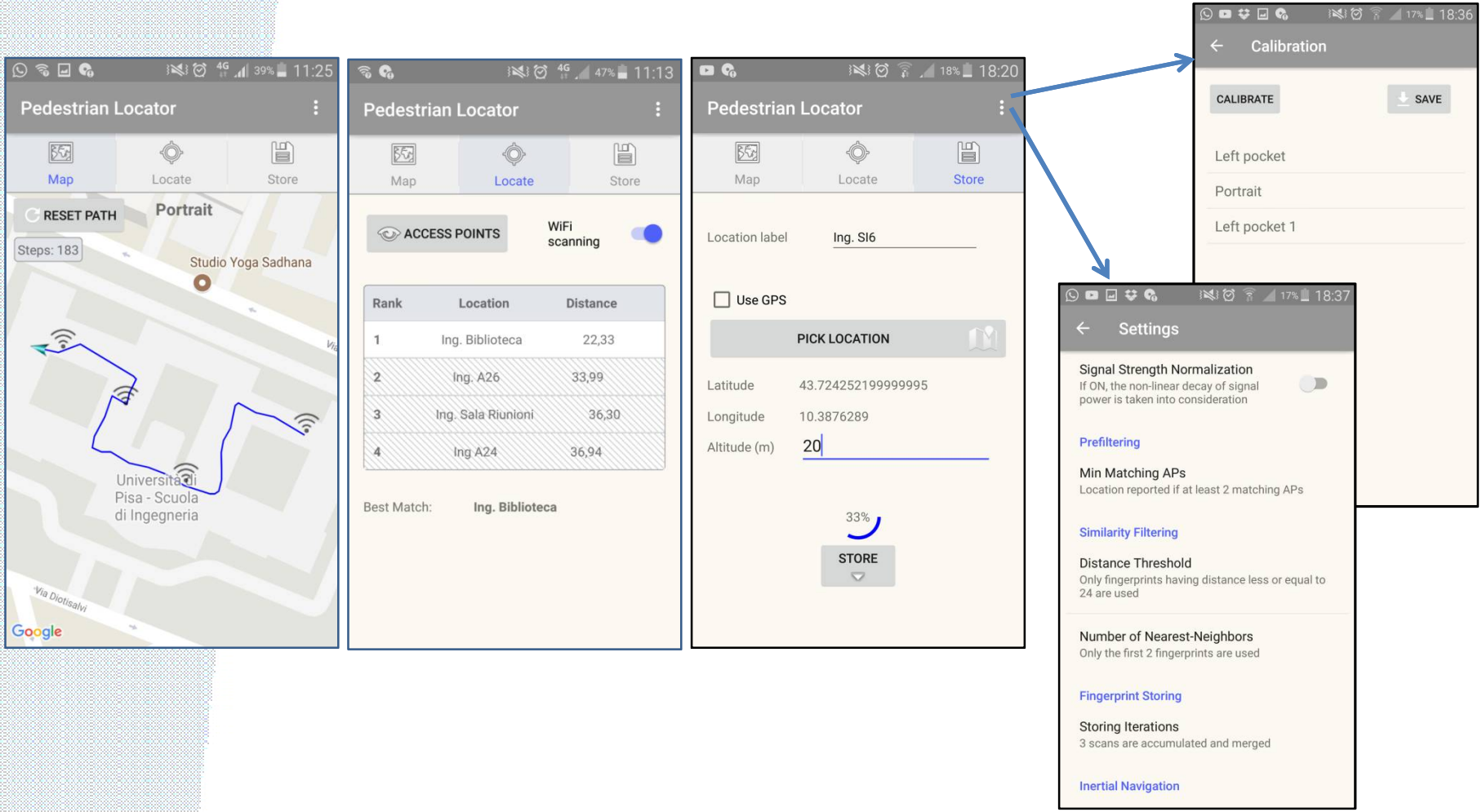
- **WifiLocatorActivity** (the main activity)
  - **FPMapFragment**, containing the map for visualizing movement
  - **FPLocateFragment**, UI for controlling fingerprint matching
  - **FPStoreFragment**, UI for storing new fingerprints
- **WifiFingerprintingService**
  - for wifi fingerprinting acquisition and matching)
- **InertialPedestrianNavigationService**
  - for steps counting/rotation acquisition

# Application Structure (2)



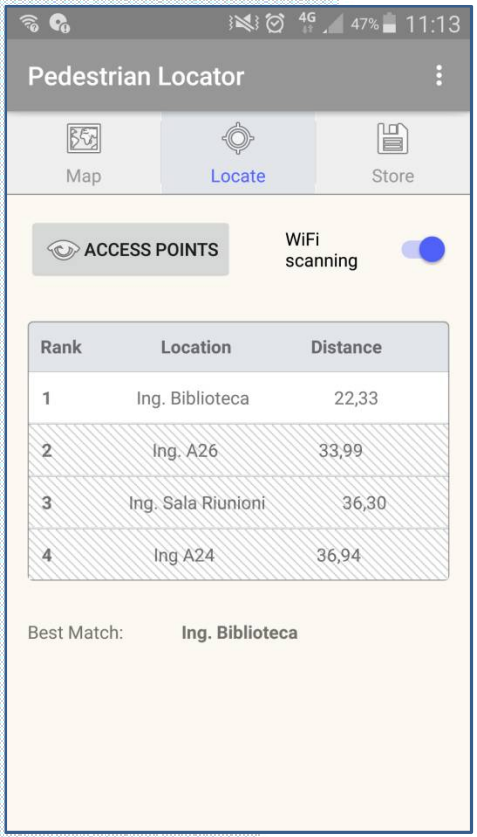
# Application GUI

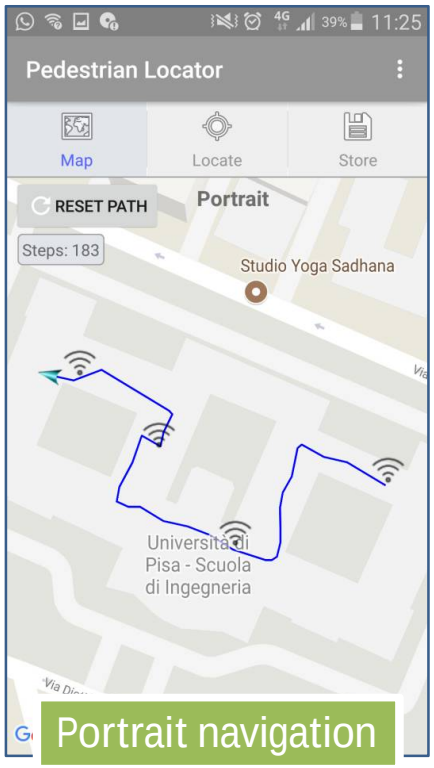
- The 3 fragments are organized in a tabbed activity
  - From menu other 2 activities (Calibration and Settings) can be accessed



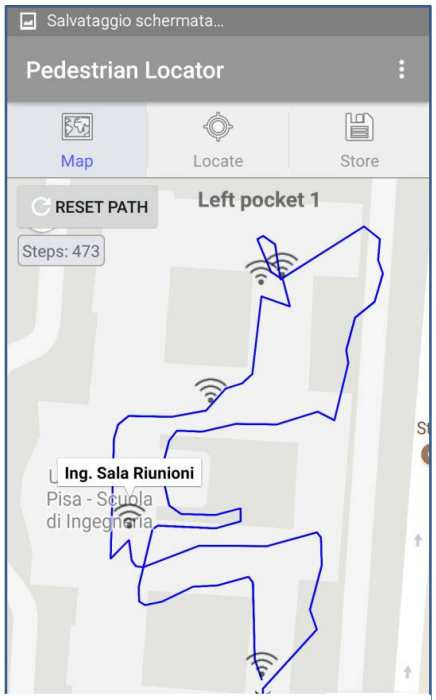
# Indoor Localization Test

- Very low physical distance between wifi sync points
  - Physical distance, however, is quite well reflected on fingerprint distance (for example, different floors are not mismatched)
- Lot of magnetic noise brings high errors in the inertial system




Portrait navigation




Left pocket navigation
**2 floors**

Min Matching APs: 2
Max number of NN: 2
Max FP distance: 26

Step length: 0.74m

# References


- [1] Pei Jiang, Yunzhou Zhang, Wenyan Fu, Huiyu Liu and Xiaolin Su, **Indoor Mobile Localization Based on Wi-Fi Fingerprint's Important Access Point,** *March 2015*
- [2] Xuxing Ding, Li Gao, and Zaijian Wang, **Modified Fingerprinting Algorithm for Indoor Location,** *Journal of Communications Vol. 12, No. 3, March 2017*
- [3] Aare Puussaar**, Indoor Positioning Using WLAN Fingerprinting with Post-Processing Scheme,** 2014
- [4] Joaquìn Torres-Sospedra, Raùl Montoliu, Sergio Trilles, Oscar Belmonte, Joaquìn Huerta **Comprehensive Analysis of Distance and Similarity Measures for Wi-Fi Fingerprinting Indoor Positioning Systems**
- [5] Alex T. Mariakakis **DirectMe: A Mobile Phone Algorithm for Direction Detection,** 2013