MASARYK UNIVERSITY
FACULTY OF INFORMATICS

Indoor Navigation for Android

MASTER'S THESIS

Michal Holčík

Brno, spring 2012

Declaration

Hereby I declare, that this paper is my original authorial work, which I have worked out by my own. All sources, references and literature used or excerpted during elaboration of this work are properly cited and listed in complete reference to the due source.

Advisor: Mgr. Jonáš Ševčík

Thanks

I would like to thank my advisor Mgr. Jonáš Ševčík for kind supervision and leading towards the thesis objective, Peter Trško for mathematical advice and discussions on the subject, Mgr. Petr Kroutil and Mgr. Martin Vytrhlík from the Geographical Information Systems department ÚVT MU who provided plans for university buildings and useful advice on the data conversion and last but not least Ing. Leoš Vojíř from Siemens Enterprise Comms. company for lending me the Samsung Galaxy Tab device on which all of the developed code was tested.

Abstract

The aim of this thesis was to study available navigation methods for localization within indoor environments and to implement chosen methods in an application for Android smartphones equipped with inertial sensors and network interfaces for connection to wireless networks. Due to planned use as a localization component for augmented reality applications, emphasis was placed on as high localization precision as possible.

We implemented a localization application based on step-based dead reckoning combined with a WiFi RSS fingerprinting method in a particle filter type system.

Test results demonstrate that the precision of our system does not fullfil augmented reality requirements, nevertheless the system still can be used as a basis for room-to-room indoor navigation.

Keywords

pedestrian navigation, indoor localization, step detection, sequential Monte Carlo localization, particle filter, wifi fingerprinting

Contents

Chapter 1

Introduction

This thesis project builds on the work of Mgr. Jonáš Ševčík in investigating the ways of user localization in indoor environments and implementing a prototype of an indoor localization system. His thesis [18] identified the need to determine the location of a library visitor or employee, both users of an augmented reality system, that provides information and locations of books in the library.

1.1 Augmented reality

Augmented reality (AR) is real-time integration of an image of the real world with an artificially created information. In present the most common way of augmenting the reality is by composing a digitally aquired image of the real environment with a computer generated visualization of the added information and presenting them on a computer screen, smartphone display or so called augmented reality glasses. In order to compose well the real and generated images, AR applications depend on the knowledge of the camera location. In AR surgery or car-fixing application, where the camera position is near the studied object, relative position may be determined by digitally processing the input image, while in outdoor applications the objects of interest (providing information about landmarks, monuments, or finding the nearest pub) are usually big and far enough so that the precision of the GPS localization is sufficient. In indoor applications however, localization precision requirements rise due to smaller distances and scale of the target objects and at the same time the GPS signal is obstructed by walls and other building structural parts. As a result there is currently no simple or reliable method of determining user location.

1.2 Target precision

Mgr. Ševčík in his thesis [18] uses a section in library shelf as the unit for addressing book location. The shelves are approximately 1 meter apart, and the shelf sections are also almost meter wide as it can be seen on the library photo, see Figure 1.1. A user of the librarian augmented reality system standing in this area would receive contextual information about the books contained in the shelf. We will use this area as our localization precision criterion. The localization precision, we will aim for, should therefore be within 1 meter from the reference location.

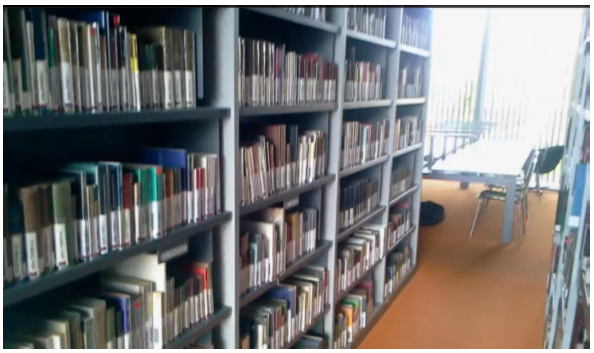

Figure 1.1:

1.3 Navigation and localization

Wikipedia, defines navigation [**?**] as "the process of monitoring and controlling the movement of a craft or vehicle from one place to another." The knowledge of the navigator's own position is critical for the all the navigational techniques, however the process of finding a location relative to other reference locations or patterns is called localization. Sometimes the term navigation is used as totum pro parte when referring to the task of localization, though localization is only a subtask of the navigation process. We, in this thesis, are solely concentrating on the task of localization (or positioning or position/location tracking).

1.4 Thesis chapters

The thesis begins with a general overview of localization methods (Chapter 2, "Localization methods"), followed by a description of characteristics of harware sensors found in smartphone devices, Chapter 3, "Sensors". In the next Chapter 4, "Localization model" a navigational model is presented, whose implementation is covered in Chapter 5, "Implementation of the Pedestrian Navigation System". Evaluation of the localization precision of the implemented system comes in the final Chapter 6, "Precision evaluation".

Chapter 2

Localization methods

This section contains an breif overview of available localization techniques. All sensor-based localization techniques are based on measuring physical quantities that in some way correlate with geographical location or the change of it.

2.1 Relative localization - Dead reckoning

Dead reckoning is calculates a position estimate by measuring the motion and the direction of motion from initial position.

2.2 Inertial navigation

Inertial navigation is a kind of dead reckoning navigation technique. Position is computing by double integrating input from accelerometers. In order to function well, it requires highly accurate inertial measurement unit (IMU).

In the start of the measurement, inertial navigation system, as any other dead reckoning system requires a position fix and initial orientation.

2.3 Electromagnetic wave-based techniques

Electromagnetic waves include visible and invisible light-based methods. Radio based techniques estimate position relative to known locations of radio emitters (satellites, wireless accesspoints, GSM basestations etc.).

2.3.1 Satellite-based techniques

The most prevalent solution today is the GPS. Followed by GALILEO and GLONASS as alternative solutions. Satellite signals are easily obstructed by walls and have poor coverage in indoor areas.

2.3.2 Radio methods

In this cathegory belong all known communication network technologies like WiFi, Bluetooth, ZigBee. Location estimate is computed by measuring the received signal strength

indication (RSSI) and comparing to a broadcast reference signal strength level.

Alternative technology based on radio waves is a RFID badge system. This however requires and prepared environment with regularly placed RFID tags.

2.4 Image processing methods

Optical methods use image information from camera, on which they perform analysis to extract location dependent patterns.

2.4.1 Marker detection

These method require active user involvement, finding and scanning optical markers (such as barcodes or QR codes). A prepared databases with mapping of the marker id to the geographical location is required.

2.4.2 Feature detection

This technique relies on computing the position from recognized features in an input image stream. This method is computationally demanding. However can result in sub-meter localization precision.

Chapter 3

Sensors

This chapter will cover basic information and characteristics of sensors usually available in current mobile communication devices (cellular phones or tablets) that can be used for indoor navigation.

To give an overview of sensors supported by the Android SensorManager API, the following list [http://developer.android.com/reference/android/hardware/Sensor.html]

Accelerometer and magnetometer are quite common sensors for estimating device's orientation in space and are a base sensors for UI controls and navigation. Gyroscope is present in the more expensive exemplars. Exotic sensors like pressure sensor or RFID reader are present in only few. Vast majority of the integrated inertial sensors use the MEMS – micro electro mechanical system.

3.1 Hardware sensors

To give examples of concrete sensors found in smartphones, we extracted a list of sensors found in the Samsung Galaxy Tab tablet. Hardware information was read from the device using the Android System Info application. [**?**]

AK8973 (by Asahi Kasei Corp.) compass/magnetic field sensor integrated

L3G4200 (by ST Microelectronics) gyroscopic sensor

BMA150 (by Bosch Sensortec) three-axis accelerometer.

The Galaxy Tab is also equipped with a temperature, proximity and light sensor. Intuitively there is little correlation between temperature and light intensity vs. geographical location, so we see no use in them for the purpose of indoor navigation.

3.2 Inertial sensors

- Linear accelerometer
- Gyroscope

3.2.1 Linear accelerometer

Linear accelerometer measures acceleration in relation to one axis of an inertial frame of reference. By using three orghogonal accelerometers, we receive information about the ac-

celeration in 3-dimensional space. All accelerometers use the same 2nd Newton's law: "The acceleration a of a body is parallel and directly proportional to the net force F and inversely proportional to the mass m."

3.2.1.1 Accelerometers and Gravity

All accelerometers on Earth's surface are subject to Earth's gravity.

3.2.2 Gyroscope

Mechanical gyroscope is an instrument consisting of a rapidly spinning flywheel attached to an axis that can turn freely. The angular momentum of the disk causes the axis to resist changes in the direction of its axis of rotation. Because of the gyroscope's tendency to maintain the same orientation in space regardless of the movement of the surrounding structure, it is used in the navigation and piloting systems of airplanes, ships, rockets, and other vehicles.

Electronic gyroscopes, also called angular rate sensors aren't themselves capable of maintaining the orientation, but rather measure the angular increments in the body's orientation, so that the actual orientation can be estimated by using a mathematical model.

There are several types of electronic gyroscopes

- Optical
- MEMS gyroscopes

Coriolis effect MEMS gyroscopes are used in todays personal communication devices. An object of the mass m moving at the velocity v in the frame of reference rotating at the angular velocity ω is subject to Coriolis force: $\mathbf{F_c} = -2m(\omega \times \mathbf{v})$

MEMS gyroscopes contain a vibrating element used to sense the Coriolis effect. When the gyroscope coordinate system turns, Coriolis force induces a secondary vibration on the axis perpendicular to the measurement axis.

Nowadays MEMS gyroscopes can't compete with precision to optical gyroscopes, nevertheless it is expected that technological progress will bring improvements in this area. Improvement of performance can be achieved by modifications in the design of circuit and sensing element.[1]

3.3 Non-inertial sensors

- Magnetometer
- Pressure sensor

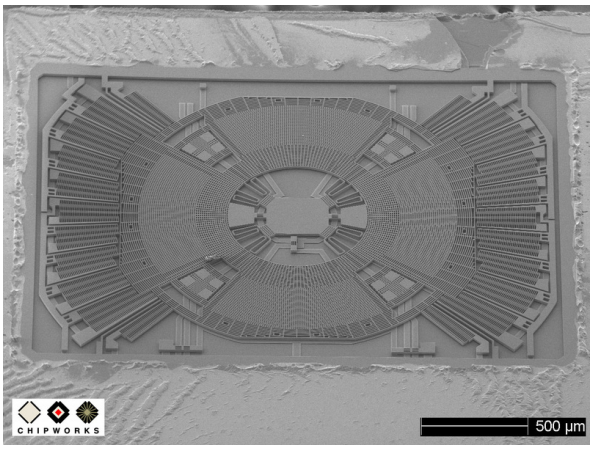

Figure 3.1: Image of ST LYPR540AH Tri-axis MEMS gyroscope, shot by a scanning electron microscope

3.3.1 Magnetometers

Magnetometer is an instrument used the measure the strength of magnetic field.

MEMS magnetometer sensors are either Lorentz-force based or solid-state Hall effect.

$$\mathbf{F} = q[\mathbf{E} + (\mathbf{v} \times \mathbf{B})]$$

Equation 3.3.1: Lorentz force

3.3.1.1 Magnetometer error and influence of environment

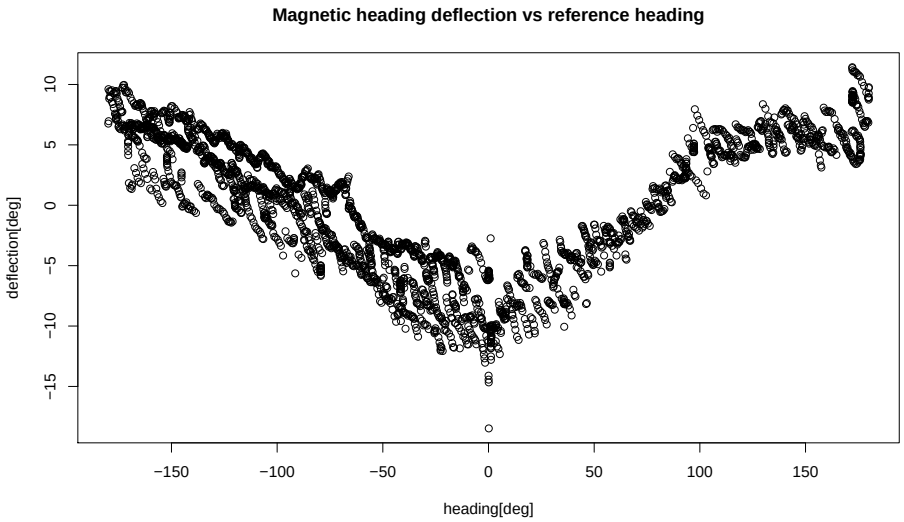

Figure 3.2: Magnetic heading deflection vs. reference heading

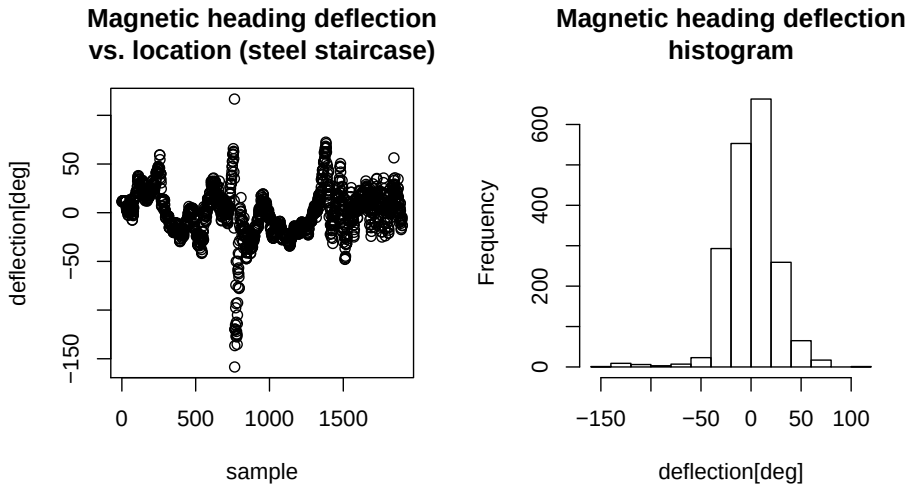

Figure 3.3: Magnetic heading deflection vs. location (steel staircase)

Chapter 4

Localization model

This chapter covers all the components of necessary for a localization system based on dead reckoning with step detection and absolute location estimates from WiFi signal strength measurements.

4.1 Motion measurements

4.1.1 Step detection

Other studies ([12], [8], [5]) dealing with the step detection by means of an accelerometer usually used a dedicated accelerometer unit placed on a suitable part of a body e.g. foot, hip or belt, which had considerable influence on the detection precision. We do not have the option of attaching an external unit to the handheld device, which means the accelerometer will pick-up all users hand motions, not only those caused by walking.

4.1.1.1 Human locomotion

Walking (also ambulation), according to Wikipedia (article "Walking") [**?**], is defined by “an ’inverted pendulum’ gait in which the body vaults over the stiff limb or limbs with each step.” The inverted pendulum motion is characteristic also for other gaits like running, differing mainly in the amount of take-off from the standing limb and causes a bobbing motion of the upper body and the connected limbs (hands). The bobbing motion, being universally present in all types of gaits, will be our main feature upon which to base the step detection methods.

4.1.1.2 Required sampling rate

According to the Nyquist–Shannon sampling theorem, required sampling rate for the detection of steps, under the assumption that the maximum walking speed is 3Hz, accelerometer must provide sampling frequence at least doubling the value, ie. 6Hz. This criterion is comfortably within the capability of our test device (Galaxy Tab), which provides rates as high as 100Hz.

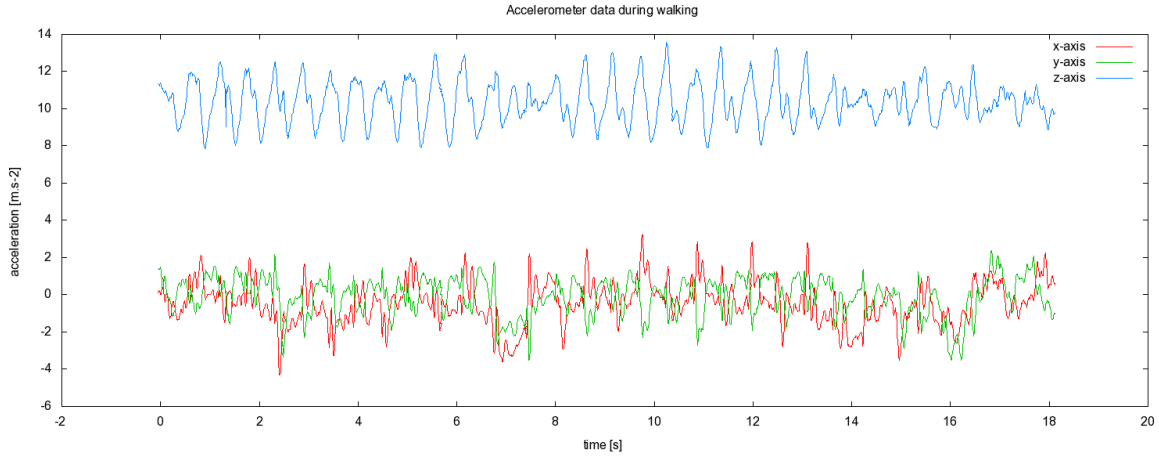

Figure 4.1: Accelerometer signal on the z-axis (vertical) during several steps measured at 100Hz sampling rate.

4.1.1.3 Signal processing tools

The following list contains digital signal processing filters [19] suitable for step detection application:

- Low-pass filter - removes high frequency components of the signal. This is important to avoid false steps when zero crossings or extrema detection method is used.
- Power threshold filter - computes signal power, which is used for identification of silence zones.
- Step duration change filter - Uses similar technique as fundamental frequency detection in human voice. It is probable that the duration of two succeeding steps doesn't change too much. Detected step candidate can be rejected e.g. when its duration is less then half of the duration of the previous step.
- Correlation comparison filter - computes similarity of two signals by computing their correlation.

We started the step detection algorithm by implementing two moving average filters with different window sizes. The longer moving average (window size 0.2s which is 20 sample on 100Hz sampling rate) is used as an estimation of average value of G-force projection on the z-axis. The shorter moving average (windows size of 0.05s, 5 samples on 100Hz sampling rate) is a low-pass filter that removes the high frequency components from the signal. Candidate steps are detected as intersections of the two moving average signals.

The moving average method proved successful in tests and was correctly detecting steps with near 100% precision during sustained walking. The problem started when it detected

while standing still when the moving averages still intersected and detected false positives. Because of these, the algorithm was extended with a power threshold filter, that defines the walking vs. non-walking zones in the signal.

The algorithm still detects false steps due to acceration signal noise caused by manipulation with the held device, however we decided to handle these false steps by successive mathematical processing rather than trying to filter them out.

Power threshold has to be configured manually, no self-adjusting algorithm was employed.

4.1.2 Stride length estimation

During the implementation of the dead reckoning module, we noticed that the variability in step length has substantial effect on the localization precision. Through observation we deduced that there might be a dependency between the stride length and the walking frequency. Independent studies performed on other species than human supported the observation, however no suitable study was found so we decided to make our own measurements to support the hypothesis and to create an estimation model that we could include in our implementation.

4.1.2.1 Experiment setup

A path with fixed length (38.4 meters) was used to take the measurements. A human subject walked the path at a constant walking rate maintained by a metronome. Number of steps taken were counted. From the number of steps and the distance walked, the average stride length was computed.

4.1.2.2 Measurement results and derived model

The measurement results support our hypothesis and from the picture is clear that the frequency and step length have strong linear correlation and the measured data also show the longest step to be 40% longer than the shortest step which pinpoints the source of the inaccuracy in the step model. Tested subject reported that the walking pace was becoming uncomfortable under 80bpm and at and over 150bpm. Within the interval of, let's say, comfortable walking frequency, the linear dependency between frequency and stride length seems to hold.

We are aware of inconclusivity of a test conducted on a single person, where obviously more biomechanical parameters of the subject come into play, nevertheless we will still count with the assumption of the linear dependency of the stride frequency and length in our model.

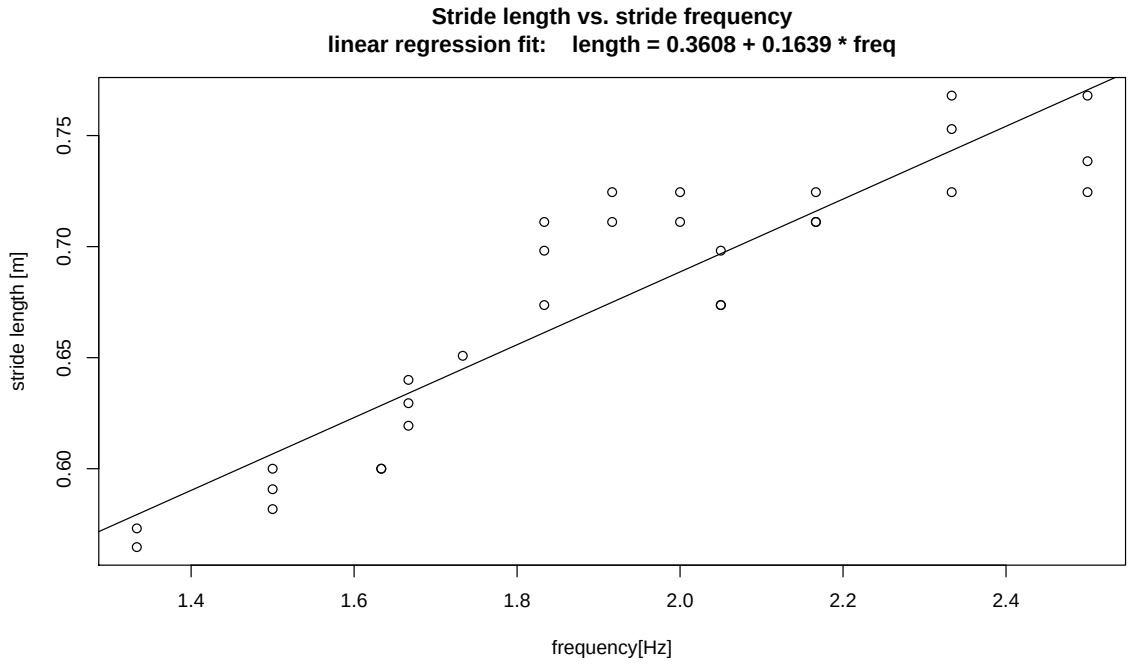

Figure 4.2: Dependency of stride length on stride frequency

4.2 Device orientation

4.2.1 Coordinate system

In the following text whenever we metion a device coordinate system, we mean the Android standard coordinate system defined relative to the device's screen [22]: "When a device is held in its default orientation, the X axis is horizontal and points to the right, the Y axis is vertical and points up, and the Z axis points toward the outside of the screen face. In this system, coordinates behind the screen have negative Z values."

4.2.2 Magnetic heading

The magnetometer provides device's heading relative to the Earth's magnetic pole. The problematic part is that indoor environments contain structural elements or electronic equipment that may have significant effect on magentic field and therefore on orientation measurements. For devices that are not equipped with other means of orientation sensing is the inaccurate magnetometer the only way. The responsibility to make orientation corrections has to be delegated on other components in the localization model.

4.2.3 Sensor Fusion

Although the magnetometer is subject to error, it is the only sensor that can provide the absolute north-bound heading. Gyroscope provide much precise orientation estimate derived from the angular rotation measurements, however it is subject to integration drift and will require recalibration after some time. Clever combination of these two sensors can diminish the deficiencies of both. Combination of sensors in this way is called sensor fusion.

4.2.4 Gyroscope orientation

4.2.4.1 Frames of reference

Sensor units are firmly attached to the smartphone. In order to express orientation of the device by means of sensor measurements we must transfer the measurements and the derived orientation from one frame of reference to another.

We adopt the definitions used in Woodman [20] and refer to "the navigation system's frame of reference as the *body frame* and to the frame of reference in which we are navigating as the *global frame*."

4.2.4.2 Coordinate system rotations

Gyroscope produces a sensor event with a measured angular rate vector of $\phi = (\phi_x, \phi_y, \phi_z)$. The size of the vector represents the total angular rate of the body frame of reference in the global frame. We use this vector to transform our base matrix by the axis and angle rotation. The angle θ equals to the the size of the vector divided by the $\theta = |\phi| = \sqrt{\phi_x^2 + \phi_y^2 + \phi_z^2}$. The normalized vector $\mathbf{u} = (u_x, u_y, u_z) = \frac{\phi}{\sqrt{\phi_x^2+\phi_y^2+\phi_z^2}}$ defines the axis of rotation. The transformation matrix for axis and angle rotation is:

$$\mathbf{R} = \begin{bmatrix} cos\theta + u_x^2(1 - cos\theta) & u_x u_y(1 - cos\theta) - u_z sin\theta & u_x u_z(1 - cos\theta) + u_y sin\theta \\ u_y u_x(1 - cos\theta) + u_z sin\theta & cos\theta + u_y^2(1 - cos\theta) & u_y u_z(1 - cos\theta) - u_x sin\theta \\ u_z u_x(1 - cos\theta) - u_y sin\theta & u_z u_y(1 - cos\theta) + u_x sin\theta & cos\theta + u_z^2(1 - cos\theta) \end{bmatrix}$$

Equation 4.2.1: Rotation matrix from axis and angle

4.2.4.3 Alignment to the global frame of reference

Relative angular rate sensor measurements are corrected by the absolute measurements from the magnetometer and accelerometer. Alignment is done separately for the accelerometer and magnetometer vector. Alignment vector coordinates are first converted to the global frame of reference coordinate system. Alignment is done by rotating the global frame of reference so that the global frame z-axis vector matches the acceleration vector and the global

frame y-axis vector matches the magnetic field vector's projection into the xy-plane. In attempt to mitigate the undesirable effects mentioned in Section 4.2.3). The mathematical tool for combining two distinct sensor inputs is called the *complementary filter*.

4.2.4.4 Complementary filter

Complementary filter is another name for a weighted average in digital signal processing jargon. This filter adds together two separate signals at a constant ratio β. The equation of the filter is Equation 4.2.2 [7], where θ_{rel} is the relative angle update, θ_{abs} is the absolute angle update.

$$\theta = \beta * \theta_{rel} + (1 - \beta)\theta_{abs}$$

Equation 4.2.2: Complementary filter equation

We want to be the relative sensor input from the gyroscope the main reference input, the β less than 0.5. The size of the β is chosen according to the formula Equation 4.2.3 [6], where T is the time constant of the low-pass filter, dt is the sampling period (e.g. 0.01s for 100Hz rate).

$$T = \frac{(1-\beta)dt}{\beta} \rightarrow \beta = \frac{dt}{T+dt}$$

Equation 4.2.3: Complementary filter time constant

For example, for the low-pass window of 5 seconds at 100Hz sampling rate we receive $\beta = \frac{0.01}{5+0.01} = 0.002$

4.3 WiFi Localization Methods

4.3.1 Chosen localization model

Triangulation is possible in environments where a propagation loss model can translate signal strength measurements into distance from the known locations of WiFi access points (APs). In the university buildings, location of WiFi APs is not centrally known, and even creating the database of locations would left us without a suitable propagation model. So instead, we decided go with the RSS fingerprint approach to use in our system.

4.3.2 RSS fingerprinting

Positioning techniques based on fingerprinting have two phases:

Off-line phase, fingerprint gathering during which a database of RSS fingerprints is created. An RSS fingerprint related to a location (x, y) is a vector of pairs of BSSID (Ba-

sic service set identification - unique Access Point ID) and the strength of received signal (in dBm).

On-line phase In the online phase, regular AP scans are done and resulting fingerprints are compared against the database of reference fingerprints. Location of the best matching candidate is returned as the estimated location based on a chosen metric.

4.3.3 Delaunay triangulation

The phase of gathering fingerprints can be simplified [4] by gathering smaller amount of fingerprints and interpolating the full map. The measured map is triangulated with the Delaunay algorithm. The triangulation algorithm splits the area into non-overlapping triangle mesh, where each location in the plane is uniquely assigned three vertexes of the triangle in which it is contained. At that point full linear interpolation of the map can be recreated from the barycentric coordinates of the location in the triangle. Care has to be taken when designing the initial fingerprint set to avoid inducing error with linear interpolation in places with larger signal gradient. This is the technique we used. We created basic fingerprint sets on the ground floor of the Faculty of Social Studies and on the 2nd floor of the Faculty of Arts library. Example of the interpolated map can be seen on Figure 5.6

4.3.4 Fingerprint metrics

4.3.5 Location finding methods

4.3.5.1 Nearest neighbor method

4.3.5.2 Probability map

4.4 Sequential Monte Carlo filtering

The key idea of the sequential Monte Carlo (SMC) filtering is modelling the state of a dynamic system by aproximating the posterior density function by a set of random samples of the state vector while sequence of noisy measurements are made on the system.[2]

The computation of the SMC requires two models [2]:

1. System model - model describing the evolution of the state with time
2. Measurement model - model relating the noisy measurements to the state

Filter processes events from the step detector (step event, length estimate) and a probability density function from the WiFi AP scans.

4.4.1 State model

First version of the state vector of the modeled system held only the hypothetical location coordinates of the pedestrian $\mathbf{state} = [x, y]$. We later extended the model also by heading deflection and stride length: $\mathbf{state} = [x, y, hdg, len]$. The extension enabled the application to gradually 'learn' these two parameters and improved localization precision. Especially heading correction caused a big improvement. During tests the user of the localization application often doesn't hold the smartphone in exactly the direction he or she walks. Usually a smartphone held in the right hand points slightly to the right and vice versa.

Any moving particle is subject to spatial constraints imposed by the building walls and other structures. Particles, whose hypothetical motion leads through impassable obstacles are removed from the set.

4.4.2 Measurement model

The state model is updated by two kinds of measurement events:

- Step events from the step detector
- WiFi location probability map

The step event accompanied with a heading in which the step was taken. The heading is a normally distributed random variable with configurable standard deviation.

The probability map is a grid-based probability density function that maps location (x, y) to a probability α. On the WiFi update event, particles that belong to grid point with coordinates (x, y) are removed with probability $1 - \alpha$.

4.4.3 Pedestrian Monte Carlo localization algorithm with map matching

The following code shows the update step of the localization filter:

```
particle state is a vector of (x, y, hdg, length)

void onStep(heading) {
   for all states in the particleSet do {

      # add measurement noise according to the respective distributions

      lengthWithNoise = length + lengthMeasurementNoise
      headingWithNoise = heading + headingMeasurementNoise

      # compute the new state, only the x, y coordinates are updated
      newX = state.x + state.lengthWithNoise * cos(state.hdg + headingWithNoise)
      newY = state.y + state.lengthWithNoise * sin(state.hdg + headingWithNoise)

      particleTrajectory = (state.x, state.y, newX, newY)
```

```
      # check for collision with walls
      collided = collisionModel.checkCollision(particleTrajectory)
      if (collided) {
         remove particle from the set
      }
   }

   # resample the set by multiplying randomly selected particles from the set
   resample(particleSet)
}
```

Chapter 5

Implementation of the Pedestrian Navigation System

The following chapter describes the environment for which the pedestrian navigation system is designed, internal design and interfaces of the system and the decisions we faced during its implementation.

5.1 Android

The system is implemented for the Android operating system as requested by the project assignment. Android is an opensource Linux-based operating system intended primarily for personal mobile communication devices such as smartphones and tablet computers. It is developed by the Open Handset Alliance, led by Google, and other companies [**?**]

Android consists of a kernel based on the Linux kernel, with middleware, libraries and APIs written in C an application software running on an application framework which includes Java-compatible libraries based on Apache Harmony. Android uses the Dalvik virtual machine with just-in-time compilation run Dalvik dex-code (dalvik executable). [22]

In the following text, we often mention Android activities. Android Dev Guide [22] defines an activity as "an application component that provides a screen with which users can interact in order to do something, such as dial the phone, take a photo, send an email, or view a map. Each activity is given a window in which to draw its user interface."

5.1.1 Android SDK and NDK

Two different frameworks can be used when writing an application for Android:

- SDK – Standard Development Kit
- NDK – Native Development Kit

Android SDK is a set of tools for writing and deploying Android applications written in Java. NDK is an extension of the SDK that lets developers write performance-critical portions in native code (in C or C++). It provides headers and libraries that allow developers to build activities, handle user input, use hardware sensors, access application resource. [21]

The initial discussions suggested that the programming code is written C/C++ with the Android NDK in order to allow for simpler portability to Apple iOS, but the Android developer guide itself discourages [http://developer.android.com/sdk/ndk/overview.html]

from writing native code, due to increased code complexity, unless some of the native APIs unsupported in the framework are required, so we decided to stay with the Java version of the program.

5.1.2 Target API Version

The application was written for the Android API 8 (Android V2.2). This decision is based merely on the API version available on the Galaxy Tab device on which this project was developed and tested.

5.1.3 Android Integration

We investigated ways how to inject our pedestrian localization component into the Android system framework, so that 3rd party components can register for reception of the location events. Currently Android doesn't support registration of proprietary location provider so a sibling LocationManager called PedestrianLocationManager was created. This class wraps the in-system LocationManager together with the implementation of all the pedestrian localization methods described in the previous chapter Chapter 4, "Localization model".

5.2 System Design

5.2.1 External Interfaces

The PedestrianLocalizationManager reuses the Android LocationManager interface, which serves for registering LocationListeners for location update events.

5.2.1.1 Android Framework Interfaces

The navigation system uses several Android system services as data sources for the localization algorithm:

SensorManager class for accessing device's hardware sensors. It registers SensorListeners for individual Sensors. The listeners are then periodicly updated when sensor values change.

WifiManager serves as a primary API for WiFi connectivity management. We use the class for starting and retrieving results of the access point scans.

LocationManager class for accessing the system location services. These services supply applications with updates of the device's geographical location.

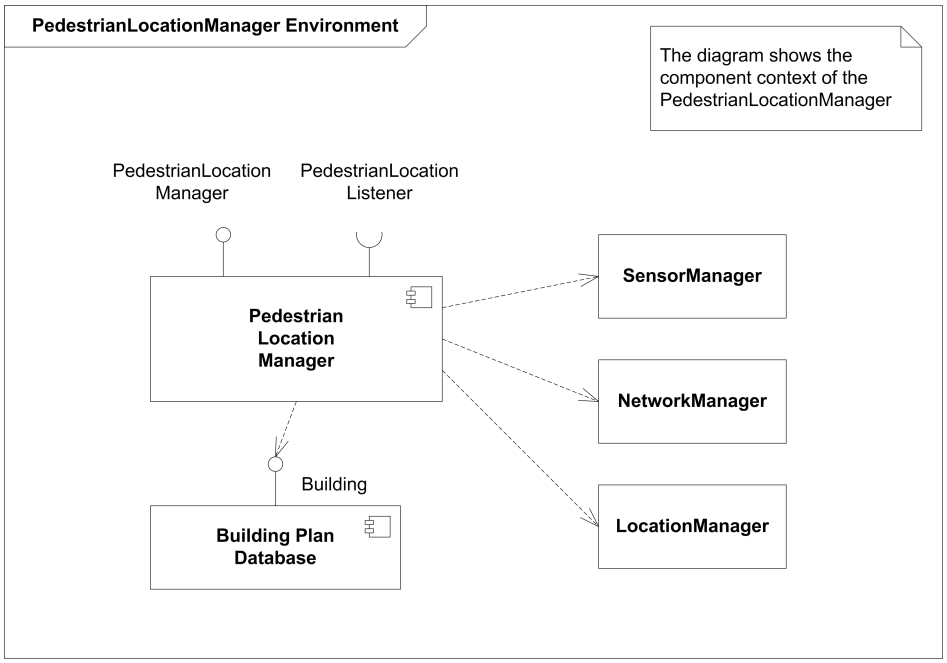

Figure 5.1: Context of the Pedestrian Localization Service

5.2.2 Intercomponent interfaces

PedestrianLocationManager components:

- Dead reckoning component
  - Compass (magnetic, gyroscopic, combined gyromagnetic)
  - StepDetector
  - StepLengthEstimator
- RssFingerprintController
- ParticlePosition

Compass module – receives data from the accelerometer, gyroscope and magnetometer sensors and combines them in estimation of coordinate system connected with the building.

Step detection module registers for accelerometer signal updates and generates step events.

DeadReckoning component combines the events from the Compass and Step Detector and produces relative position update events.

WiFi Controller – manages regular access point scans and constructs location probability density function.

Particle filter – registers for relative location updates from the dead-reckoning component and events from the WiFi interface and combines the two together in a posterior probability density function approximation.

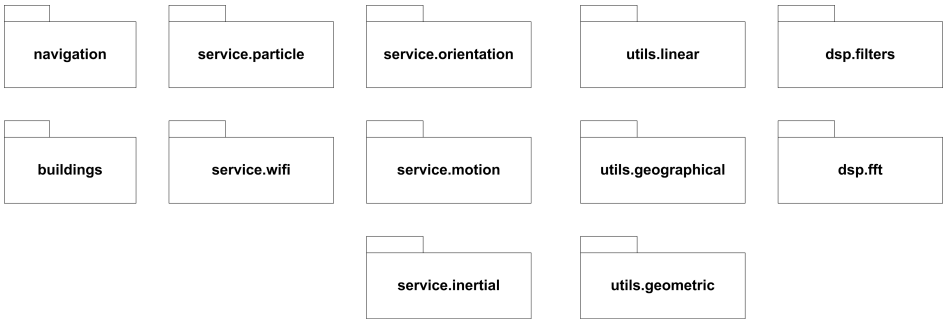

Figure 5.2: System packages

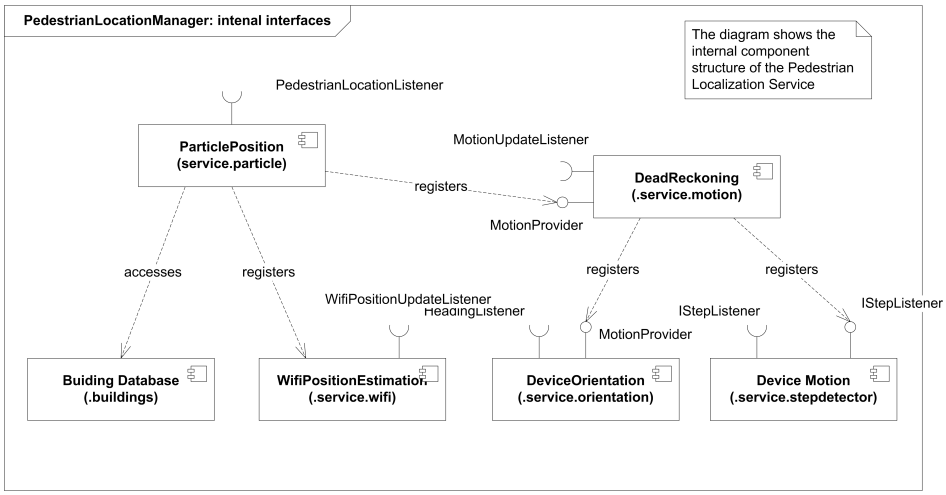

Figure 5.3: Pedestrian Localization Service

5.3 The Pedestrian Navigation Sandbox application

The installation .apk package is available on the attached CD. The application contains a set of demonstration activities that show functioning of individual system subcomponents:

- Pedestrian Navigation Prototype
- Precision Evaluation Activity
- Device Orientation & Motion Sensing

  - Step Detection
  - Acceleration FFT
  - Magnetic Compass Demo
  - Gyro Compass
  - Combined Gyro & Magnetic Compass
  - Acceleration FFT
  - Magnetic Compass Deflection Demo

- Radio Positioning Methods

  - WiFi RSS Map
  - WiFi RSS Scanner
  - Delaunay Triangulation Demo
  - Location Provider Demo

- Stochastic Position Computation Methods

  - Discrete Stochastic Model
  - Particle Filter Model

- Preferences

5.3.1 Pedestrian Navigation Prototype Activity

Pedestrian Navigation Prototype activity contains fusion of all the implemented navigation methods with a visualization of the estimated location.

5.3.1.1 Graphical interface

Activity view shows visualization of the user position in the building plan: Probability density function aproximated by the particle cloud with a piper on the average position, walls (red lines) and stairs (magenta lines) of the particle collision model and transition edges (cyan). Particle color changes between black for particles in the currently selected area/floor and gray for particles on other floors.

Figure 5.4: Pedestrian Navigation Prototype screen

5.3.1.2 User Controls

Activity GUI allows user to pan and zoom around the building plan by using slide and scale gestures.

Further user actions are accessible via the activity menu:

Select floor by selecting this item, user can make a selection of the currently displayed floor

Set position action to manually reset initial position of the particle filter. After this item is selected, user must set the new position by tapping a desired location in the building plan. New position of the particle filter is set to the coordinates of the tap on the currently selected floor.

Reset compass action to manually realign compass heading, if the integration drift or local magnetic field fluctuation cause unacceptable deflection. Reset behaviour differs for gyroscopic and combined gyromagnetic compass. Gyroscopic compass aligns to the device's coordinate system (i.e. north heading in the direction of the top of the screen, east heading to the right). Gyromagnetic compass aligns to match current gravitational acceleration and magnetic field vectors.

Settings action to open preferences activity

Pause/Resume action to pause or resume sensor data streams

Help action launches the instruction file in a viewer

5.3.2 Precision Evaluation Activity

This activity is a modified PedestrianNavigationPrototype with different controls used for testing the precision of the system. Graphical interface and all control elements apart from the menu remained the same. More on the precision evaluation testing in Chapter 6, "Precision evaluation".

5.3.2.1 User Controls

Select floor, Set position and Reset compass menu items remained the same and kept the same functinality. Settings, Pause and Help items were replaced by new three items:

Log attempt action writes a new evaluation log entry. The log contains the distance of the actual average location to the destination location in meters, the coordinates of the destination location and a timestamp of the log creation.

Pick next action randomly generates a new destination location.

Set destination action sets the destination location manually. Location coordinates are entered by tapping the screen.

5.3.3 Step Detection Activity

Step Detection Activity provides visualization of the step detection method and helps tuning the step detection parameters. Displayed signals are, from the top to bottom:

- cumulated signal power - the sum of power since the last crossing of the moving averages (see next point). On positive step detection, the power must reach over the red line (the threshold).
- two moving averages, with marked crossings - red circles represent step event candidates with insufficient cumulated signal power level. Green circles are reported step events.
- original acceleration signal on the z-axis

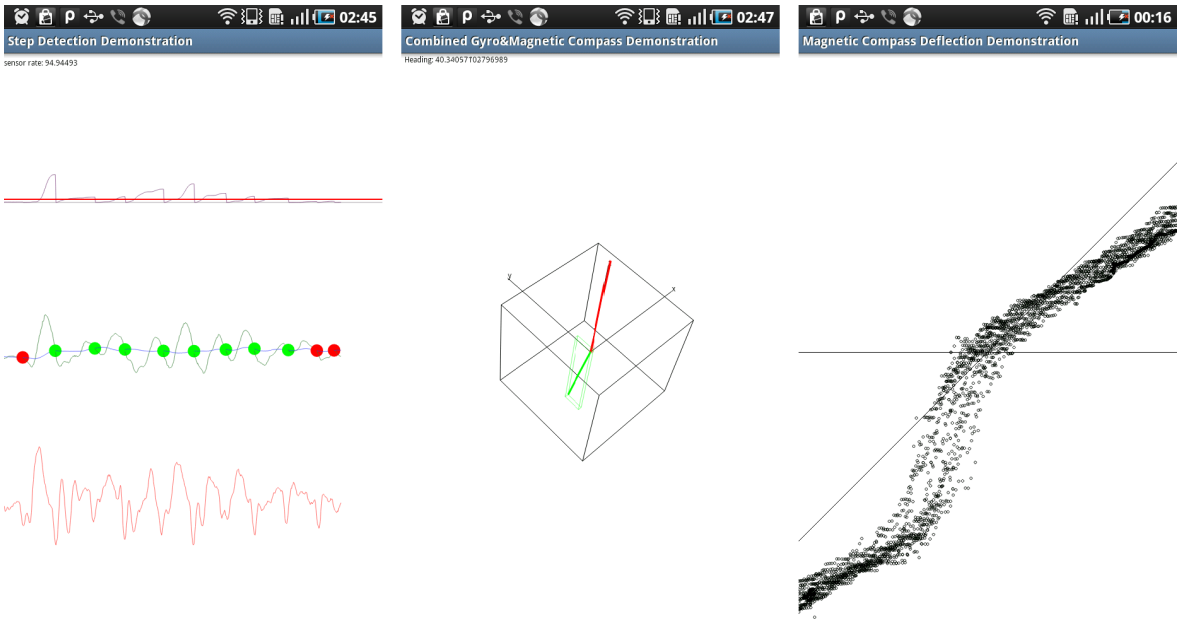

Figure 5.5: Orientation Demos Screenshot (from left to right): 1. Step Detection Activity, 2. Combined Gyro & Magnetic Compass Activity, 3. Magnetic Compass Deflection Activity

5.3.4 Acceleration FFT

Acceleration FFT activity displays the Fast Fourier transform of the input accelerometer signal on the z-axis. The tool was used to gather data for pattern matching of possible step events. In the upper part of the screen, black line is the original accelerometer signal on the z-axis, blue line is the computed FFT of the signal. Lower half of the screen is another graphical representation of the signal's spectrum.

5.3.5 Compass Demonstration Activities

MagneticCompass, GyroCompass, GyromagneticCompass activities serve as visualization and debugging tools for the three types of compasses. The MagneticCompass provides heading value supplied by the in-system Orientation sensor. GyroCompass and GyroMagneticCompass track devices orientation in 3d space by processing gyroscope, accelerometer and magnetometer sensors. Difference between the two gyro compasses is that the latter contains mechanism to align orientation to gravitational acceleration and magnetic field vectors, while the former doesn't and therefore is subject to integration drift. Amount of integration drift over time can be assessed by playing with the tool. The following text is relevant only for the gyroscopic versions of the activities.

5.3.5.1 Graphical interface

Both gyro compass activities show a world coordinate system orientation visualization represented by three orthogonal axis x, y, and z and a wire cube for better depth perception.

Visualization contains vectors acceleration (red) and magnetic field (green) with bounding boxes in the corresponding colors.

5.3.5.2 User Controls

Tapping the screen realigns the coordinate system. In the GyroMagnetic compasss activity, the speed of alignment for configured parameters can be observed.

5.3.6 Magnetic Compass Deflection Demo Activity

This activity was used to measure data for graphs of mangetic compass deflection in dependence on location and device orientation in Section 4.2. Procedure used for retreiving data for the graphs is also described there.

5.3.6.1 Graphical interface

The activity screen displays dependence between gyroscopic compass heading and magnetic compass heading, with gyro compass heading placed on the X-axis (horizontal line with zero in the center of the screen), and magnetic heading on the Y-axis (not displayed). The quadrant axis (the line across) represents a direct proportion and serves as a visual clue for assessing dependency between the two variables.

5.3.7 WiFi RSS Map

WiFi RSS Map activity provides visualization and recording capabilities for WiFi RSS fingerprints.

Works in three regimes: idle/viewing, location finding, recording. Location finding regime displays an updated location probability map on every finished RSS scan. In recording regime, a reference RSS fingerprint is recorded on a chosen location.

5.3.7.1 Graphical interface

Activity view of the interpolated fingerprint map on the building plan. Fingerprint RSS values are displayed on a square grid mesh with a blue-red color ramp.

5.3.7.2 User Controls

Activity view allows user to pan and zoom around the building plan by using slide and scale gestures.

Further user actions are accessible via the activity menu:

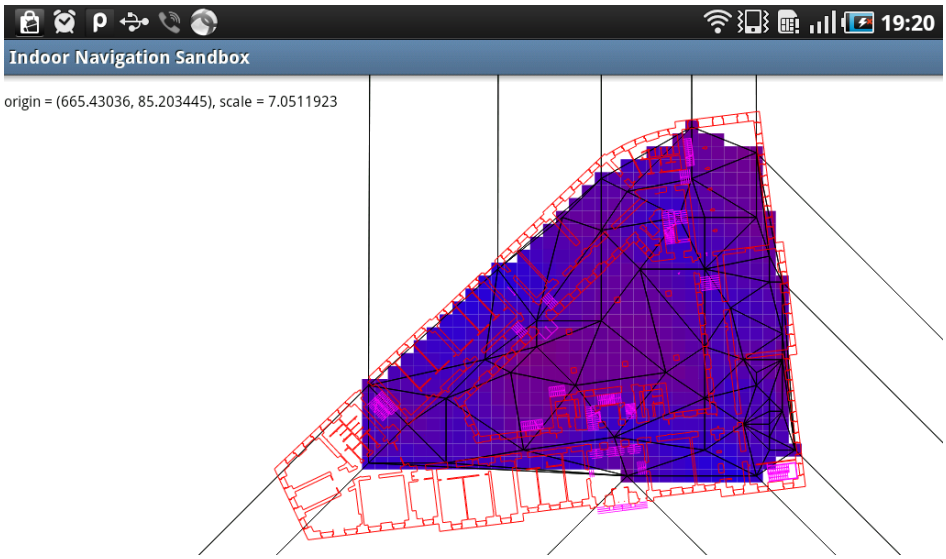

Figure 5.6: Interpolated RSS map

Select floor by selecting this item, user can make a selection of the currently displayed floor

Select BSSID Lets the user choose the network to be displayed or 'maximum'. If maximum is selected, the grid displays the value of the strongest signal present in that location.

Add fingerprint starts fingerprint recording

Find location Enters location finding regime.

Clear database Permanently deletes fingerprint database in the currently building area (floor).

5.3.8 WiFi RSS Scanner

An activity for inspection of the local access point RSS levels. This activity works in similar way to the WiFi RSS Map activity, with the difference that no map is shown and no location context is used for fingerprints, i.e. artificially generated coordinates are used when creating new fingerprints in the database. Scanned access-point RSS values are logged on the screen in textual form.

5.3.8.1 User Controls

Activity is controled solely by means of menu actions. Available actions are:

Add fingerprint by selecting this item, the activity enters the fingerprint recording state. Scanned access-points RSS values are saved and their average computed during the duration of the recording mode.

Find location This action starts the location finding regime. Scanned access-point RSS values are compared to the fingerprint database and the closest match is printed on the screen.

Clear database This action renews the RSS fingerprint database. File for storing the fingerprints on the flash drive is deleted and recreated.

Help Action opens readme file in a viewer

5.3.9 Delaunay Triangulation Demo

Toy activity for familiarization with the Delaunay triangulation algorithm. The activity iteratively generates Delaynay triangulation from a set of points input by tapping the screen.

5.3.10 Location Provider Demo

Simple logging activity of events from the Android LocationManager. Logs both GSM and GPS location events, with additional info about GPS satellites. The activity was used to study LocationManager events streams in locations where laptop connection wasn't available.

5.3.11 Grid Stochastic Demo

Demonstration activity to show the basic functioning of a stochastic grid based location filter in space constrained by non-passable partitions. Tapping the screen generates a step event. The direction vector of the step is computed from the center of the screen to the coordinates where the screen was tapped. Length of the step is constant.

5.3.12 Particle Filter Demo

Demonstration activity to show the basic functioning of a particle filter, see Section 4.4 in space constrained by non-passable partitions. Steps are generated in the same way as in the previous demo (see Section 5.3.11). Length of the step is constant.

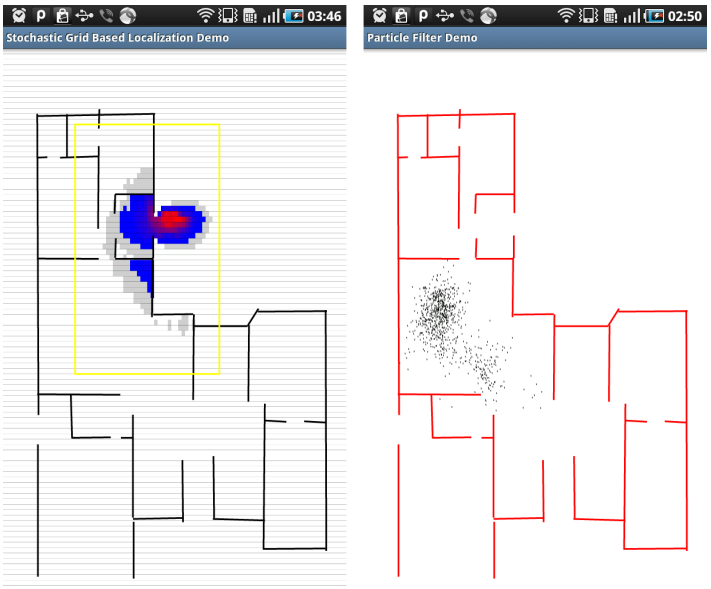

Figure 5.7: Grid and Particle Filter Activity Screen

5.3.12.1 Graphical interface

The graphical interface uses the same visual elements as the PedestrianNavigationPrototype demo (see Section 5.3.1).

5.4 Building Plan Data

The system was tested in real buildings thanks to kind cooperation of colleagues from the Geographical Information Systems department of the Masaryk University (GIS ÚVT MU) who provided vector plans of several university buildings from the university technological passport.

5.4.1 Technological passport of the Masaryk University

The maps from the technological passport (accessible from the Maps Muni portal <`http://maps.muni.cz`>, direct link <`http://gisweb.muni.cz/TechnologickyPasport/`>) contain information for management of university buildings. For navigation puposes only a fraction of the information was necessary so the provided plans had to be converted from the AutoDesk DWG format into a proprietory vector format containing only the parts of the building that can be used by the map matching models (i.e. walls, windows, stairs).

The building plans use the S-JTSK coordinate system, so a conversion to a common coordinate system is necessary.

Figure 5.8: DWG plan of the 1st floor, block B of the faculty of informatics

5.4.2 AutoDesk DWG format conversion

AutoDesk DWG is a closed proprietary format for storing multilayer vector data and their description, however tools capable of opening the files and working with the contained data exist. We used the jdwglib (see <`http://sourceforge.net/projects/jdwglib`>) java library to extract the relevant layers and store the contained Line objects into a text file that can be read directly from the Android OS.

5.4.3 S-JTSK coordinate system

In order to use geographical data from different sources, the location coordinates must be converted to the same coordinate system. In our case the data provided by the Android system framework are in the WGS84 system, whereas the building plans use the S-JTSK coordinates in Křovák's projection.

The WGS-84 system is defined by a set of land stations and positions of satellites of the GPS navigation system. [9]

The S-JTSK coordinate is used by the Czech and Slovak Katastrální úřad (Office for Surveying, Mapping and Cadastre) and the projection is adapted to bear the highest precision for location withins the former Czechoslovakia, which means that the coordinates are not universally available for the whole world. Nevertheless the fact that the coordinates are in meters meant significant advantage, so we decided to use the S-JTSK as the reference coordinate system. This raised the requirement to implement also the coordinate conversion from WGS84 to the S-JTSK. [3]

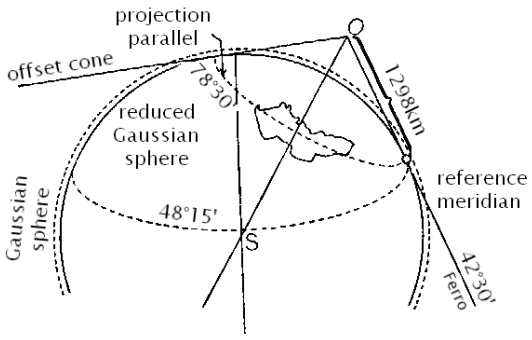

Figure 5.9: Schema of the Křovák's projection[3]

Hrdina 2006 [9] publishes a coordinate conversion routine from WGS84 to S-JTSK written in Pascal. We rewrote this conversion routine to Java and used it in our project.

Chapter 6

Precision evaluation

The precision of our localization system was tested many times during the implementation phase to find and remove bugs and tune the localization algorithms. This chapter describes the procedures and presents the final formal test results of the system's localization precision.

6.1 Localization precision tests

The target of the testing was to get idea of the system precision in conditions close to real usage of the device.

Since not all Android devices are equipped with a gyroscope, it seemed sensible to test also the precision of the system while using the magnetometer for orientation measurements. The following tests were executed:

- Dead reckoning tests of particle filter with combined gyro-magnetic compass with known starting position
- Dead reckoning tests w/ WiFi localization, with gyromagnetic compass and unknown starting position

6.2 Evaluation method

Other studies on the topic use precise localization method, most commonly a LIDAR (LIght Detection and Ranging, optical technology to measure distances from target objects by illuminating the target with laser pulses), to establish ground truth against which to compare developed method during the total duration of movement.

As we didn't have access to such equipment so we decided to measures localization error only at the end point of a randomly generated path. The arrival to the destination is estimated from the map and confirmed manually by the tester. The idea behind this kind of setup is that instead of measuring error between the computed and real position during the whole motion, error is measured only at randomly chosen points, which are at the same time the end-points of our test path.

6.3 Testing procedure

A set of random locations on the floor of the testing building was generated. There are of course locations in the building that are not accessible to unauthorized personnel so the locations of such places had to be removed from the set manually.

The tester equipped with a Galaxy Tab device and a specially modified PedestrianNavigationPrototype activity, see Section 5.3.2, was presented starting and end point of a path. The tester was advised to take the shortest route between the two points although taking a longer path was not detrimental for the test. Shortest path was preferred to have some sort of common criterion for all of the paths to prevent tester from (inadvertently) taking paths favoring the used algorithm.

The tester was given time to familiarize himself with the device and the application. Sufficient time was also spent tuning the settings of the localization method's parameters in attempt to find the optimum configuration.

6.4 Test results

6.4.1 Dead reckoning tests of particle filter with gyromagnetic compass with known starting position

Figure 6.1: Generated points and the transitions between them on the ground floor of FSS

Measurements were taken on the ground floor of the Faculty of Social Studies. 75 reference measurements were made (Figure 6.1). Position was lost in two cases (2.5%). One other measurement was had error distance of 66 meters, and was also removed, because such a high error was probably caused due to faulty user input. Median error distance of

the measured sample is 2.3 meters. Plots of the distribution are in the Figure 6.2

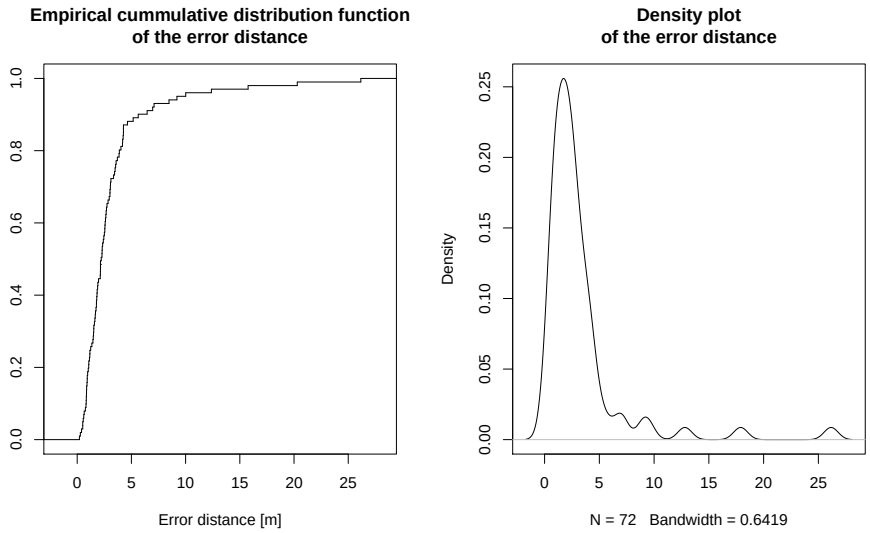

Figure 6.2: Distribution of error distance in the sample set

6.4.2 Dead reckoning tests of particle filter with gyromagnetic compass with unknown starting position, and WiFi localization

Measurements were taken on the ground floor of the Faculty of Social Studies. We made 63 reference measurements (Figure 6.3). Median error distance is 3.6 meters. Plots of the distribution are in the Figure 6.4

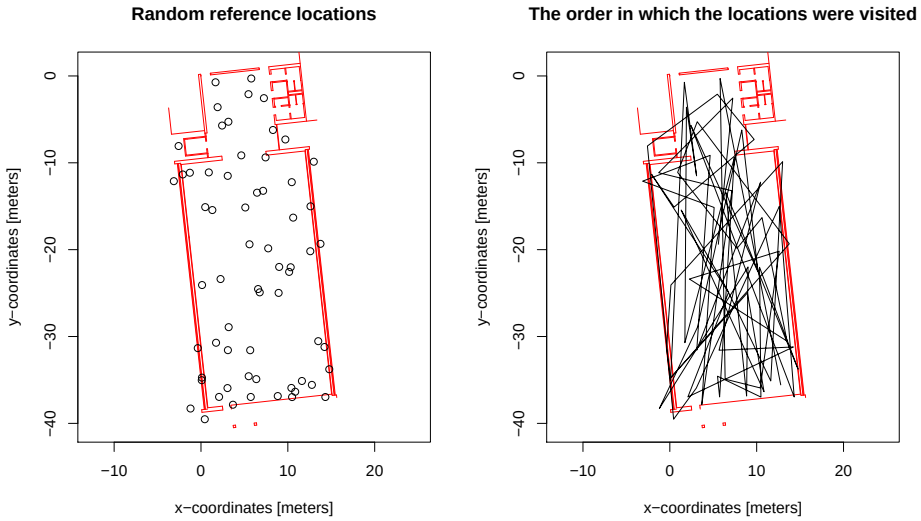

Figure 6.3: Generated points and the transitions between them in the FF Library, 3rd floor

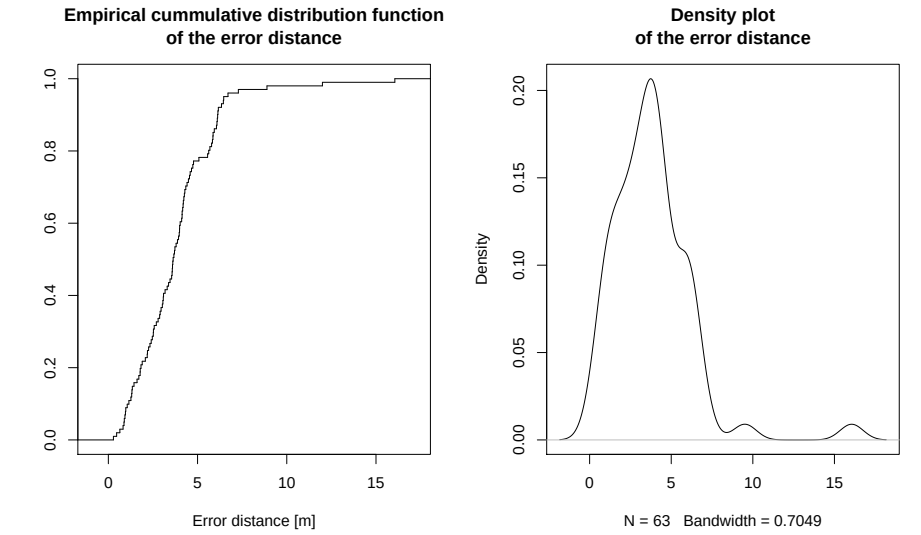

Figure 6.4: Distribution of error distance in the sample set

Chapter 7

Conclusion

7.1 Summary

In this thesis project, we first investigated available methods for indoor navigation with the use of chosen sensors present in state-of-the-art mobile communications devices.

Following the analysis we implemented a system for pedestrian navigation based on combined dead reckoning and absolute localization methods. Dead reckoning uses step detection filter and the device orientation is measured by combining sensor signals from gyroscope, magnetometer and accelerometer.

Two possible stochastic models were considered. Grid based method seemed promising but the computation load requirements during early implementation phases proved too high for a handheld device in real time condition, so the model was abandoned. The final implementation uses a particle filter together with a wall collision model for building map matching.

Wifi signal strength is used for the initial location lock and repeated location correction. We used the RSS fingerprint methods for the computation of the location probability density function, which is added to the one of the particle filter.

The system supports transitions between areas (can be used either for transitions between adjacent buildings or transitions between floors), and the computational model had to be extended to handle walking stairs in a special way.

7.2 Results

The system was tested in the buildings of the Masaryk University in Brno with the use of converted maps from the MU's technological passport. Evaluation test of the particle filter dead reckoning method measured the median error distance of 2.3 meters, 90th-percentile is at 5.6 meters. Particle filter test with WiFi localization updates, however in unconstrained space of a library, scored median error distance of 3.6 meters, 90th-percentile at 12.7meters. Major effect in these two results are caused by missing spatial constraints in the latter case. Used WiFi localization method didn't have significant effect on the dead reckoning position estimate in the university buildings where the system was tested. The particle filter had difficulties converging to the correct location on the floors of FSS after the WiFi initial location estimate at the start of navigation or when the particle filter position was lost. Our test results of the WiFi localization precision, differ significantly from the localization precision of

other WiFi localization studies, that promise localization precision of around 5 meters. The cause of inaccuracy of our method may be in using wrong algorithm for the conditions we face in university buildings. We also noticed during the phase of fingerprint collection that the areas are usually efficiently covered with WiFi access points with few areas where several APs with good signal overlap. This is of course critical factor for WiFi localization and without fulfilling this, high precision in WiFi localization can't be expected.

7.3 Future work

The implemented system, even though its accuracy proved insufficient for the use with augmented reality application, forms a basis upon which new more precise methods can be implemented. As further studies in this field show, methods based on processing an image stream from smartphone camera can provide sub-meter accuracies, which is the scale of precision necessary for indoor augmented reality applications. The camera based method can be used in cooperation with step detection based navigation to increase the accuracy in places where visual navigation doesn't provide good results or with a possible benefit of computational load decrease on the carrier platform.

Worth mentioning is that the application caught the interest of colleagues from the GIS department and there is a potential of creating an indoor room-to-room pedestrian navigation application for university students and employees. However before doing so, we suggest further precision and acceptance tests of the implemented localization methods with wider public.

Bibliography

[1] Apostolyuk, V. and Varshavsky, A. and LaMarce, A. and de Lara, E.: *Theory and design of micromechanical vibratory gyroscopes*, MEMS/NEMS Handbook, Springer, 2006, Vol.1, pp.173-195., 2006. 3.2.2

[2] Arulampalam, M. and Maskell, S. and Gordon, N. and Clapp, T.: *A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking*, IEEE Transactions on Signal Processing, Vol. 50, No. 2, February 2002. 4.4

[3] Čada, V.: *Přednáškové texty z Geodézie, section 2.3. Coordinate systems*, <http://gis.zcu.cz/studium/gen1/html/ch02s03.html>, online, visited May 25, 2012. 5.4.3, 5.9

[4] Chen, A. and Harko, C. and Lambert, D. and Whiting, P.: *An Algorithm for Fast, Model-Free Tracking Indoors*, Mobile Computing and Communications Review, Volume 11, Number 3, 2006. 4.3.3

[5] Cho, S.: *MEMS Based Pedestrian Navigation System*, The Journal of Navigation (2006), 59, pp. 135–153, The Royal Institute of Navigation, 2006. 4.1.1

[6] Colton, S.: *The Balance Filter*, <http://web.mit.edu/scolton/www/filter.pdf>, slides online, visited May 23, 2012, 2007. 4.2.4.4

[7] Esfandyari, J. and De Nuccio, R. and Xu, G.: *Solutions for MEMS sensor fusion*, Solid State Technology, volume 54, issue 7, 2010. 4.2.4.4

[8] Frank, K. and Krach, B. and Catterall, N. and Robertson, P.: *Development and Evaluation of a Combined WLAN & Inertial Indoor Pedestrian Positioning System*, 4th International Symposium on Location and Context Awareness, ION GNSS, Savannah, Georgia, USA, 2009. 4.1.1

[9] Hrdina, Z.: *Transformace souřadnic ze systému WGS-84 do systému S-JTSK*, České vysoké učení technické v Praze, Praha, 1997. 5.4.3, 5.4.3

[10] Kaemarungsi, K. and Krishnamurthy, P.: *Modeling of Indoor Positioning Systems Based on Location Fingerprinting*, IEEE INFOCOM 2004, 2004.

[11] Le, M. and Saragas, D. and Webb, N.: *Indoor Navigation System for Handheld Devices: Bachelor degree project*, Worcester Polytechnic Institute, 2009.

[12] Libby, R.: *A simple method for reliable footstep detection on embedded sensor platforms*, 2008. 4.1.1

[13] Lukianto, C. and Sternberg, H.: *Overview of Current Indoor Navigation Techniques and Implementation Studies*, Bridging the Gap between Cultures, Marrakech, 2011.

[14] Martin, E. and Vinyals, O. and Friedland, G. and Bajcsy, R.: *Precise Indoor Localization Using Smart Phones*, Proceedings of the ACM International Conference on Multimedia (ACM Multimedia 2010), Florence, Italy, pp. 787-790, 2010.

[15] Miller, L.: *Indoor Navigation for First Responders: A Feasibility Study*, National Institute of Standards and Technology, 2006.

[16] Navarro, E. and Peuker, B. and Quan, M.: *Wi-Fi Localization Using RSSI Fingerprinting*, California Polytechnic State University.

[17] Otsason, V. and Varshavsky, A. and LaMarce, A. and de Lara, E.: *Accurate GSM Indoor Localization*, UbiComp 2005, LNCS 3660, pp. 141–158, Springer-Verlag Berlin Heidelberg, 2005.

[18] Ševčík, J.: *Rozšířená realita na mobilní platformě Android a její aplikace v knihovnictví, Master's thesis*, Masaryk University, Brno, 2011. 1, 1.2

[19] Smith, S.: *The Scientist and Engineer's Guide to Digital Signal Processing*, 1997-2006. 4.1.1.3

[20] Woodman, O.: *An introduction to inertial navigation, Technical Report No. 696*, Computer Laboratory, University of Cambridge, United Kingdom, 2007. 4.2.4.1

[21] A, A.: *What is the NDK?*, <developer.android.com/sdk/ndk/index.html>, online, visited May 23, 2012. 5.1.1

[22] A, A.: *Android Developer's Guide*, <http://developer.android.com/guide/index.html>, online, visited May 23, 2012. 4.2.1, 5.1

[23] Serra, A. and Carboni, D. and Marotto, V.: *Indoor Pedestrian Navigation System Using a Modern Smartphone*, MobileHCI 2010, Lisboa, 2010.

[24] Serra, A. and Dessi, T. and Carboni, D. and Popescu, V. and Atzori, L.: *Inertial Navigation Systems for User-Centric Indoor Applications*, 2010 NEM Summit Proceedings, 2010.

Appendix A

File formats

This section covers the definitions of the building layer formats used by the Indoor Navigation Sandbox application

- Building definition DTD
- Walls, stairs format
- Transition edge format
- RSS Fingerprint file

A.1 Building package DTD

```
<!ELEMENT area ( walls, stairs, transitions, rss ) >
<!ATTLIST area name CDATA #REQUIRED >
<!ATTLIST area originX #IMPLIED >
<!ATTLIST area originY #IMPLIED >

<!ELEMENT building ( area+ ) >

<!ELEMENT rss EMPTY >
<!ATTLIST rss format CDATA #REQUIRED >
<!ATTLIST rss grid CDATA #REQUIRED >
<!ATTLIST rss path CDATA #REQUIRED >

<!ELEMENT stairs EMPTY >
<!ATTLIST stairs format CDATA #REQUIRED >
<!ATTLIST stairs path CDATA #REQUIRED >

<!ELEMENT transitions EMPTY >
<!ATTLIST transitions format CDATA #REQUIRED >
<!ATTLIST transitions path CDATA #REQUIRED >

<!ELEMENT walls EMPTY >
<!ATTLIST walls format CDATA #REQUIRED >
<!ATTLIST walls path CDATA #REQUIRED >
```

A.2 Walls and stairs file format

Walls and stairs use simple text format with walls and stairs represented by lines. Lines are separated by newlines and the starting and finishing point coordinates (x1, y1, x2 and y2) are separated by a sequence of spaces.

```
line-set     : LINE_DECL . NEWLINE . line-set
             | LINE_DECL

LINE_DECL    : COORDINATE
             . SPACE+
             . COORDINATE
             . SPACE+
             . COORDINATE
             . SPACE+
             . COORDINATE

COORDINATE is a signed floating-point-number [+-]?[0-9]+[.][0-9]+
SPACE is a space (0x20) character
NEWLINE is a linefeed (0xA) character
```

A.3 Area Transition Edge file format

Transition edge serve as gateways between two adjacent areas. They are represented by lines that contain a tuple of areas they connect.

```
edge-set     : EDGE_DECL . NEWLINE . edge-set
             | EDGE_DECL

EDGE_DECL    : LINE_DECL . SPACE+ . AREA_ID

AREA_ID is a unique alphanumeric identification of an area.
```

Appendix B

Attachments

The following attachments were uploaded in the Information System thesis archive.

- source codes and the Android installation package of the Indoor Navigation Sandbox application.
- building plans converted to the format usable by the navigation sandbox for the following buildings:
  - Faculty of Informatics, Botanická 68a, Brno, 602 00
  - Faculty of Arts, Building F - The Library, Arna Nováka 1, Brno, 602 00
  - Faculty of Social Studies, Joštova 10, Brno, 602 00

Appendix C

Instructions for the pedestrian localization demos

The Pedestrian Localization Prototype and Wifi RSS Map activities require some time configuring to work well. Please check the following list to make sure all conditions of are fulfilled:

1. Download and install the `IndoorNavigationSandbox.apk` installation package on your Android device.
2. Download the `indoor_navigation_data.tgz` and unpack on the external SD card drive of your device.
3. Start the IndoorNavigationSandbox application.
4. If the above tarball was unpacked into the root of the sdcard drive (check the `indoor` directory exists), building plans should load fine.
5. Go to preferences, last item in the list, and choose one of the buildings from the "Building plan" preference list.
6. If after starting 'Pedestrian Navigation Prototype' activity, building plan of the chosen building doesn't show, set the 'Root data directory' preference manually in the Preferences. Use full absolute path to the indoor directory including the 'indoor' name. E.g. `/mnt/sdcard/indoor`
7. Next step is tuning the step detection settings. Open the Device Orientation and Motion / Step Detection demo. And check that steps are detected correctly.
8. If they are not, maybe the power threshold setting needs adjusting. Check Section 5.3.3 to get familiar with the Step Detection demo, and adjust the 'Signal power cutoff threshold filter' preference in the Preferences.
9. In order to localize you position well, hold your device flat, display facing to the sky, and the top side of the screen in the direction of walking.