A Simple Indoor Localization App

Mobile Computing Lab - Summer Term 2018

Christian Toth
Graz University of Technology
Graz, Austria
christian.toth@student.tugraz.at

Timur Cerimagic
Graz University of Technology
Graz, Austria
t.cerimagic@student.tugraz.at

I. Task Description

The goal of this project was to build an indoor localization app based on a particle filter [1] similarly to [2] to be tested and demonstrated on the first floor of Inffeldgasse 16, 8010 Graz. As a requirement for this task we were challenged to use either a Bayesian filter approach or a particle filter. Since we were not particularly fond of spending long periods on measuring signal strengths of many access points inside the building, we decided for the particle filter approach.

We separated our application into three main building blocks:

- the motion estimation
- the particle filter
- the graphical user interface (GUI).

In the remainder of this report we will elaborate on these building blocks. We will explain our motion estimation approach in Section II and its embedding in the particle filter in Section III. In Section IV we show examples of our GUI implementation and we conclude our results and experiences in Section V.

II. Motion Estimation

To achieve good convergence and accuracy in the position estimation by the particle filter it is crucial to estimate the users motion as accurately as possible. In the end, we used two sensors for the motion estimation, that are the accelerometer and the magnetometer. The accelerometer is used for the motion detection and step counting whereas the magnetometer is used to estimate the users heading angle relative to north (y-axis). All sensor readings were happening in the background on the one thread, which was constantly refreshing the buffer with new sensor data, so it can be used in the motion estimation. Multi-threading was fully utilized for this purpose, so we can have fresh data always ready for the usage in motion estimation. We estimate the motion as described above every 500 ms which proofed to be sufficiently often.

When detecting a users motion and do step counting, we firstly need to identify if the user is walking or idle. To do so, we monitor the signal power of the acceleration sensor readings and conclude that the user is idle, if this power falls below an empirical threshold value. If we have significant motion, that is, the signal power is above the idle threshold, we employ our frequency analysis system we developed for the activity monitoring app. If the amplitude of the fundamental frequency of the users motion is greater than again some empirical threshold, we consider the user to be walking with a step frequency given by the fundamental frequency estimated. During times of continuous motion we accumulate the step counts by multiplying the step frequencies with the observation time window as

$$stepcount_{k+1} = stepcount_k + f_0 * T_{obs}.$$

After the end of a walk (transition from moving to idle) we found that a correction of the step count by

$$stepcount = 0.5 + stepcount * 0.85$$

improves the accuracy significantly.

To estimate the users heading direction we compute the users heading in azimuth using the androids sensor API. However, due to environmental impairments (metal objects in the building) and device orientation changes due to the users movement, these estimates are rather noisy. We therefore take a median value of 501 estimates in each observation time frame when estimating the step frequency. At the end of a walk, we again take the median of all headings estimated during the sub-strides of the user which finally gives us acceptable accuracy.

III. Particle Filter

At the core of our application there is a particle filter. In a nutshell, it initially spreads a high number of particles uniformly all over the floor plan (see Figure 1). By walking around in the building and accurately estimating this motion we move the particles on the map. If any particles intersect any walls, we eliminate them and re-sample the remaining particles to converge to a position estimate of the user as the user walks around the building.

Since the particle filter itself is rather straight-forward to implement, the most critical part was to move the particles appropriately. From the motion estimator we get an approximate step count and heading direction after a movement sequence (a walk) has ended. We assume the stride length of the particles to be within a range of 0.5 meters to 1.2 meters according to [2]. Since there is some inaccuracy in our step counting mechanism and the users stride length may vary, we update

the particles with a stride uniformly distributed within $\pm 10\%$ of the estimated stride.

To account for the uncertainty in the heading estimate we move the particles in a direction drawn from a normal distribution centered at the estimated heading. We choose a standard deviation of 25 and we sample from this distribution for each particle individually.

The chosen uncertainty characteristics were found by testing in the building. Choosing too large uncertainties leads to slow convergence behavior, whereas too little uncertainties might fail to capture the true movement and thus may converge to wrong estimates. In the final version of the application we used 10000 particles to estimate our position.

IV. Graphical User Interface

The final part of the system is the GUI part where particle positions are represented on the floor plan. As the user stops, positions and weights of the particles are updated and the main UI thread refreshes the image. Particle positioning is done by matching the particles position in meter to pixel position using a fixed ratio. Considering that we downloaded the official floor plan, we had the ruler which we used to calculate pixels per meter. We had approximately 81 pixels per square meter. Figure 1 shows the initial GUI with uniformly distributed particles all over the map. Below, some motion information is printed for testing reasons.

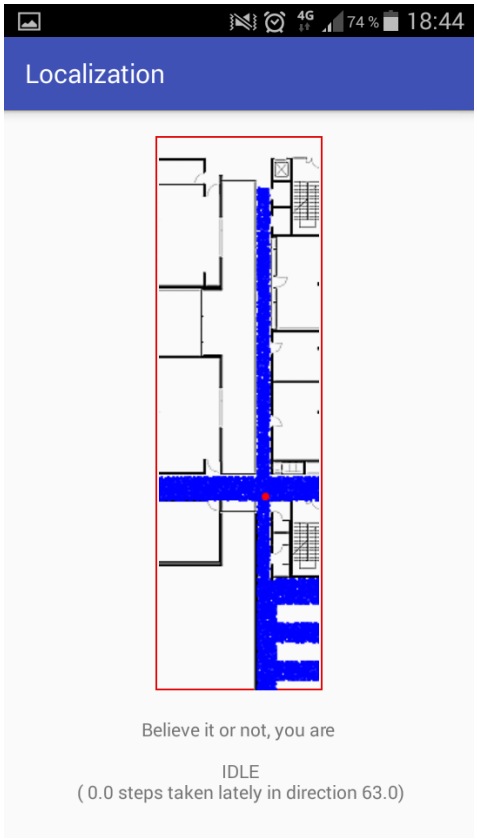

Fig. 1. Initial particle spread over the floor plan.

A possible particle movement for walking downwards and walking right from the initial users position is visualized in Figure 2 and Figure 3 respectively.

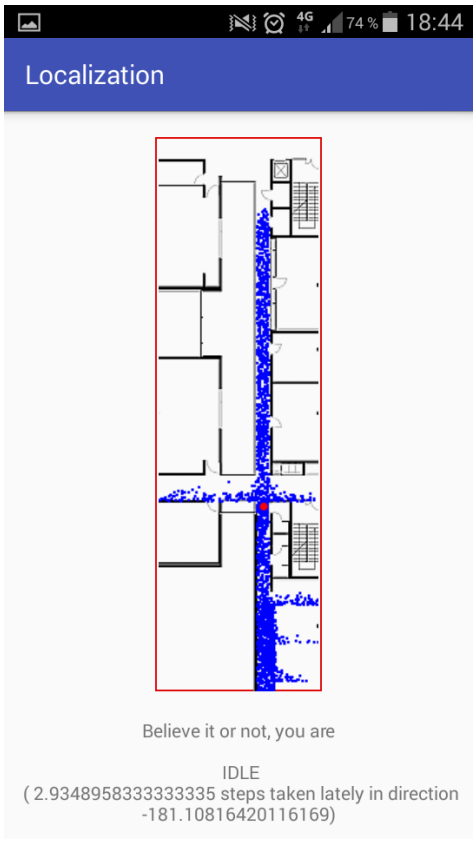

Fig. 2. Particle distribution after walking downwards from initial position.

Finally, Figure 4 shows a particle distribution where the particle filter actually converged to the users true position.

V. Summary

During testing the particles convergence within a 15- 20 meter walk, depending on the number of turns and starting position (e.g. we started inside the lab and after the initial 15-meter-long aisle and few steps to the left we converged). The biggest problem that we encountered was the building layout. Due to many metal frames and fences the phones' magnetometer was severely affected, resulting in a large uncertainty in heading estimates. Moreover, due to its shape (long straight aisle, few corners and turns) there is a lot of ambiguity in possible movement routes and thus the particle filter may take some time until convergence. However, the results were surprisingly positive and once converged, the particle filter gives accurate position estimation within approximately 2 meters.

Considering that we hard-coded the wall constraints in the application, no problems arose from missing or incomplete constraints. However, in the case of more complex building it would be time consuming to hard-code all the walls and obstacles and therefore we see place for improvement for e.g.

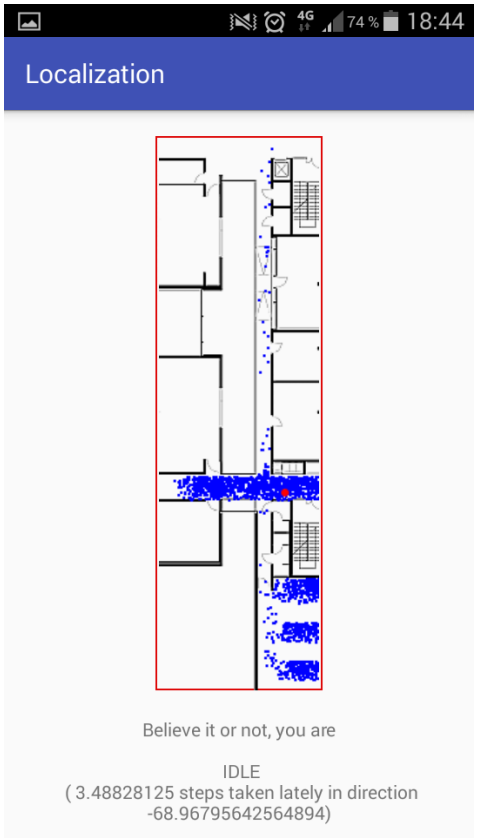

Fig. 3. Particle distribution after walking right from initial position.

image processing to generate wall constraints in an automated fashion.

From the hardware side, Samsung Galaxy S4 mini (2013) was able to deliver the background processing and the main UI thread update without significant latency (approximately 1.5 seconds per iteration). Place for improvement is seen in number of sensor readings per time frame and number of particles computed in the given space.

Our main conclusion is that particle filter is a powerful tool for indoor localization and it may outperform Bayesian approaches based on RSSI readings by far. However, computational effort and possible combination of both methods needs to be taken into consideration when dealing with large floor plans and difficult setups.

References

[1] M.S. Arulampalam ; S. Maskell ; N. Gordon ; T. Clapp, "A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking", IEEE Trans. Signal Processing, 2002.

[2] K.K. Chintalapudi ; V. Padmanabhan ; R. Sen ; K. Chintalapudi, "Zee: Zero-Effort Crowdsourcing for Indoor Localization", Mobicom, 2012.

[3] Oppenheim, Alan V. and Schafer, Ronald W. Discrete-Time Signal Processing. Prentice Hall Press, 2009.

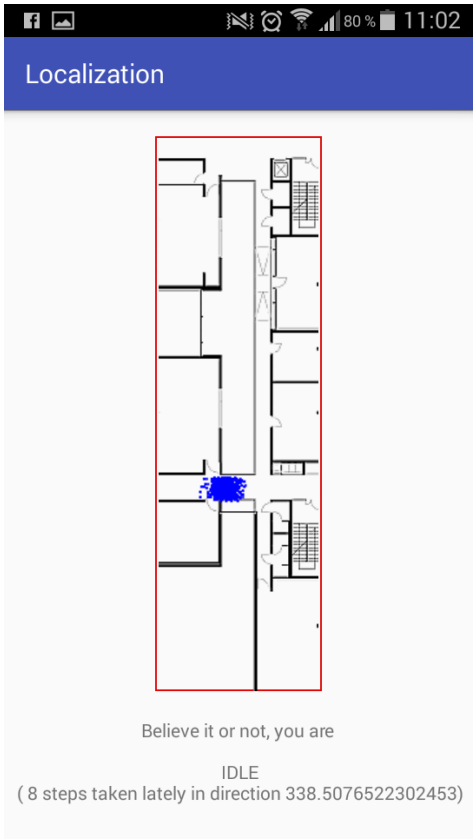

Fig. 4. Particle distribution after convergence.