Mobile Computing, Lab

Hannah Brunner, Markus Gallacher

June 17, 2019

1 Introduction

The aim of the course was to implement a smartphone app on Android, which utilizes and processes on-phone sensor data.

The source code is available on GitHub [1].

2 Implementation

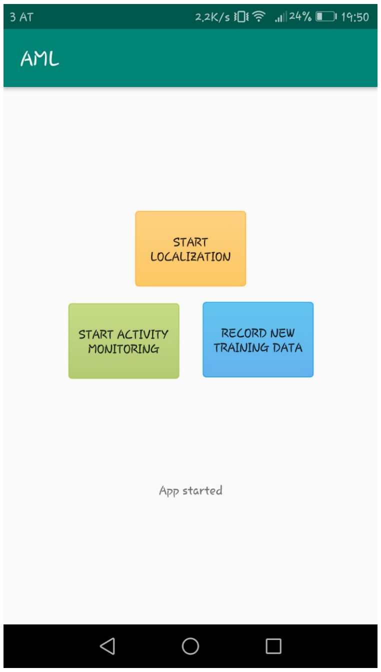
2.1 Main App

We combined both tasks (activity monitoring and localization) into a single app. It consists of four activities:

1. **MainActivity:** Shows the home screen, which appears at start of the app (see Figure 1) and contains buttons to start the desired activity.
2. **TrainActivity:** Records and stores accelerometer data associated with desired activity, which is used during activity monitoring (see Section 2.2.2).
3. **MonitorActivity:** Monitors and displays the user's current activity (see Section 2.2.3).
4. **LocalizationActivity:** Implementation of indoor localization using particle filter (see Section 2.3).

2.2 Activity Monitoring

The Activity Monitoring is split into two activities, one collects training data and stores it into a file locally. The other activity monitors the sensors and uses the kNN algorithm to classify the user's motion.



App started

Figure 1: Home screen

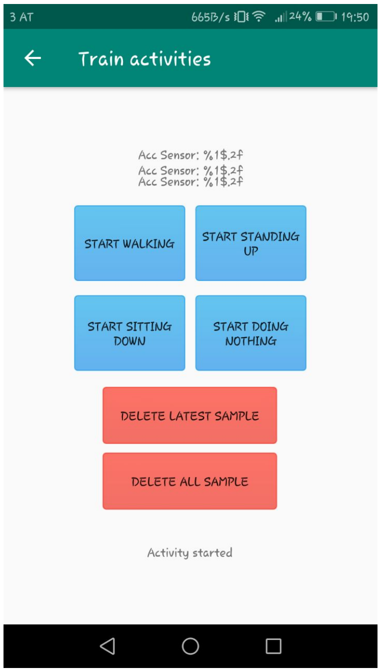
2.2.1 Background: Activity monitoring using KNN

2.2.2 Train Activity

This activity enables the user to collect training data for the monitoring activity. 1 of 4 activities are currently implemented, however any activity can be trained, only the label would mismatch at the moment.

The user needs to press one of the activities and perform the motion. The accelerometer data is sampled over a predefined window and the timestamp, mean of the x, y and z axis as well as the minimum and maximum of the euclidean distance is stores in a local .txt file. The most dominant frequency is also stored but it is currently not used as it lowers the recognition accuracy.

If a faulty measurement was recorded the user can delete the last entry or delete the whole .txt file.



Train activities

Acc Sensor: %1$.2f
Acc Sensor: %1$.2f
Acc Sensor: %1$.2f

Activity started

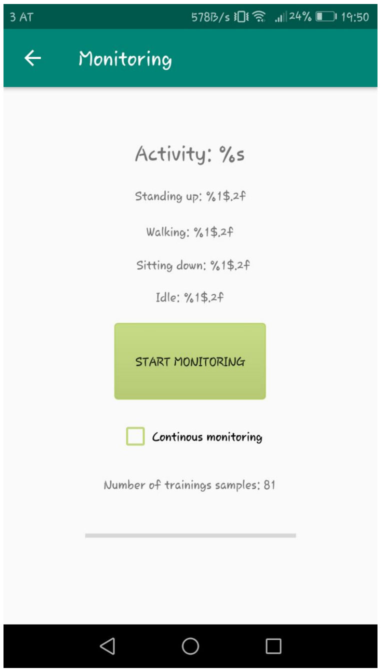
2.2.3 Monitoring

The second activity tries to classify the activity currently performed by the user. When the button is pressed, the accelerometer data is sampled with the same window size as the training data.

Then a kNN algorithm finds the nearest neighbours by calculating the euclidean distance of the averaged coordinates to each training record and selecting the k closest ones.

Then we classify the neighbours by weighting each label and adding the weights if a label occurs more than once. This gives us a probability of how certain we are of our match. The recognized activity is displayed together with the probabilities for each of our 4 predefined activities.

When the ”continuous monitoring” box is ticked, this prediction process (sample, kNN, classify) is repeated endlessly.



Activity: %s

Standing up: %1$.2f

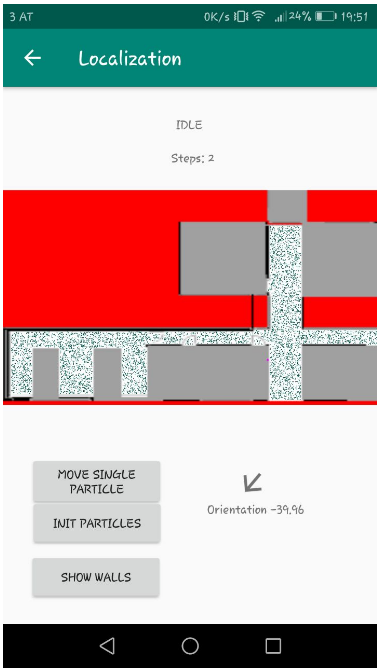
Walking: %1$.2f

Sitting down: %1$.2f

Idle: %1$.2f

□ Continous monitoring

Number of trainings samples: 81



IDLE

Steps: 2

Orientation -39.96

Figure 2: UI of localization activity

2.2.4 Limitation/Challenges

2.3 Localization

2.3.1 Background: Particle Filter

2.3.2 Implementation

GUI

As shown in Figure 2, the localization activity displays a map of the ITI building's first floor, text fields to report the users movement, an arrow indicating the user's current orientation and three buttons to

- move a single particle
- init particles and start localization
- show the walls of the building on the map (mainly debug purposes)

Movement detection

In order to detect the user's movement to create an appropriate movement model, we utilized the smartphone's build in accelerometer and magnetic sensor. The movement detection is implemented in a background service.

The user's orientation is retrieved using the Android method *getOrientation*. Since the value is quite inaccurate, the median value of the measurement while walking is used.

To detect if the user started respectively stopped walking, the

Particle Filter

2.3.3 Limitations/Challenges

References

[1] “https://github.com/hanuka24/actmonitoringlocalizationapp.git.”