IN4254 Smart Phone Sensing-Final Report

Martyn Wong-4919793
TU Delft, 15 June 2019

Lim Yong Song-4985494
TU Delft, 15 June 2019

1 BASIC INFORMATION

Group Name: ceg
Phone Model: LG Stylus 3, Android Version 7.0 (API level 24)
Code used: In order to get the direction that we are facing in (Yaw) from the TYPE_ROTATION_VECTOR, we used the following code: https://stackoverflow.com/questions/14740808/android-problems-calculating-the-orientation-of-the-device

2 BAYES LOCALISATION

Data Collection

During data collection, we went to every cell (16) to collect 30 seconds of rssi data at a sampling rate of 1 sample per second. This was done on multiple weekdays at around 12-5pm (4) to collect a total of approximately 180 samples per cell, which is used for training. For testing, we collected rssi data per cell for 30 seconds on a separate day, at the same sampling rate, that is processed by the phone (online) to generate the confusion matrix files.

Data Processing/Filtering

After collecting the samples required, the rssi values were filtered to only the first 14 characters, since the last 3 characters just meant to different the same SSID just with different MAC address (to reduce number of unnecessary APs we need to check). Also, the rssi values were filtered into probabilities for each bssid and cell after smoothing (where each count is smoothed to minimum of 1 to prevent 0 probability). This process created a file that contains a bssid table that contains all the probability for every relevant rssi value from 0-99 of each cell.
To further filter out temporary access points, all incoming data (each bssid) are pre-checked against the trained data where only values that are within the file will be considered. This helps to invalidate unnecessary data points within the file itself while also disregarding any new APs during testing phase.
The data is calculated using the bayes formula in series where the posterior will be used as the prior when we run through all the bssid values of each sample/scan, which is able to provide upwards of 0.95 probability in cell prediction after just 1 iteration. Thus, the convergence was determined by looking solely at 1 sample (with all the APs detected in that sample), and the predicted cell would be the one that has the highest probability.

Radio Map

The radio map looks at a particular bssid and shows the pmf similarity of adjacent cells (cell 1 and cell 2), and difference of cells that are far apart (cell 16), which are used to help differentiate cell location.

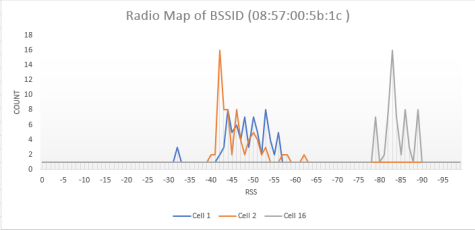

Figure 1: Radio Map

Discussion

Confusion matrix can be found on the last page (table 1) and it can be seen that all results were as expected.
GUI can be seen in Section 2 (figure 2).

Novelty

To ensure as much functionality and flexibility of the app can be done on the phone itself, we created functions to improve the app:

- Data for motion and RSSI values for each cell can be collected separately, and used for training or testing depending on what file name it was saved as (typed into the phone), where different files are combined as necessary [Note: Confusion matrix files can only be seen using a computer as no UI display is created for that]
- Additive smoothing (where 0 counts are defaulted to 1) was used so that we can retain the structure of the collected data, where it can be easily referenced in O(1) time, using arrays and hashmaps during testing/iterations while preventing zero-probability problems.

3 PARTICLE FILTER LOCALISATION

Motion model

Since our phone (LG Stylus 3), does not have a STEP_COUNTER or STEP_DETECTOR, we wrote a motion model that is based on our max-min KNN motion detection (ACCELEROMETER), TYPE_ROTATION_VECTOR and GYROSCOPE.

Firstly, to determine how much and when a person moves. We identified that an average stride of a 1.75m metre person is around 0.7m (0.4x of the height) and has an average step time of 0.5s. This allowed us to estimate that 1.4m would be moved every second that the person is in motion (based on our KNN model).

Secondly, to determine which direction and when a person is turning. We utilised the rotation vector and gyroscope sensors. By utilising the rotation vector (which takes into account accelerometer, magnetometer and gyroscope), we were able to determine the rough direction where the person was facing based on the Yaw. However, the rotation vector updated slowly (taking up to 10seconds to update to the correct direction) and was too dependent on keep Pitch and Roll constant to get accurate results. Thus, we only used it for the macro direction (to determine the correct starting direction) and added on the basic gyroscope for micro direction (to determine when a person is actually turning). The gyroscope helps to detect accelerations in the Z direction, where we set >1 to be turning 90 ° right, and <-1 to be turning 90 ° left.

Thirdly, to account for noise from motion, we made sure that particles that were re-sampled and re-spawned would be generated a random pixel diameter (60pix) around surviving particles more than the pixels per movement (56pix), so that some particles would still remain or even move back slightly to ensure a sufficiently spread cloud that is more resistant about noise.

Implementation of Map

With the floor plan being 14.3x72m, we created a map that was 572x2880pix (where each metre was scaled to 40 pixels). This ensures that both X and Y direction are equally mapped. The map was created on the canvas using black surrounding walls with their X, Y values coinciding with the mapped floor plan, while the cells themselves are created in green (as seen in figure 2) .

Particle Filter Implementation

Each particle (a total of 5000) is spawned randomly within the map when the particle filter mode is selected. The particles are then moved (56 pixels) every time there is a motion, which is detected at every 1s (2 steps) intervals. During this movement, all particles are re-sampled and any collisions with the wall will result in that particle being re-spawned 60pixels near a randomly chosen surviving particle. This ensures that the cloud can converge but not too quickly.

During this wall collision check, we also wrote in a secondary check that checks how many particles are colliding with each cell (in green) and save the count in an array-list. The cell with the highest number of particles within it would be the cell that we are predicted to be in.

Results

By following a unique path until we reach convergence, the app has an almost 100% accuracy, when we walk at a normal pace and only turn at 90° rotation intervals (especially when turning out of a room).

When walking too slowly or too fast, sometimes cell prediction can defer within 1 cell difference. Cell 16, 13 and 11 are also more affected by minor movement errors (for example, when turning too fast and the motion model thinks it is rotating and walking at the same time), due to the walls between them only being 10 pixels wide while particle re-spawns in a 60 pixel radius.

When rotating too much (i.e not at 90° intervals), there would be much more errors as we only allow 90° interval turns per second. The reason why we chose 90° intervals was because it was difficult to differentiate between rotation differences in degrees based solely on the Z acceleration. On top of this, refreshing the interval faster than 1 second results in 1x 90° turn being logged as 180° (x2 90°) at times because the gyroscope readings does not refresh fast enough.

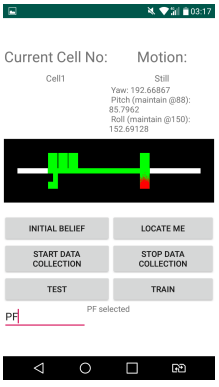

Figure 2: Particle Filter after convergence

	P.1	P.2	P.3	P.4	P.5	P.6	P.7	P.8	P.9	P.10	P.11	P.12	P.13	P.14	P.15	P.16
A.1	30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A.2	0	25	1	4	0	0	0	0	0	0	0	0	0	0	0	0
A.3	0	0	30	0	0	0	0	0	0	0	0	0	0	0	0	0
A.4	0	2	0	28	0	0	0	0	0	0	0	0	0	0	0	0
A.5	0	0	0	2	22	6	0	0	0	0	0	0	0	0	0	0
A.6	0	0	0	0	0	27	2	0	0	0	0	0	0	0	0	0
A.7	0	0	0	0	0	0	29	1	0	0	0	0	0	0	0	0
A.8	0	0	0	0	0	0	0	30	0	0	0	0	0	0	0	0
A.9	0	0	0	0	0	0	0	1	23	3	0	2	0	0	0	0
A.10	0	0	0	0	0	0	0	0	5	22	0	3	0	0	0	0
A.11	0	0	0	0	0	0	0	0	0	1	30	0	0	0	0	0
A.12	0	0	0	0	0	0	0	0	0	2	0	27	0	0	0	0
A.13	0	0	0	0	0	0	0	0	0	0	0	13	13	0	3	0
A.14	0	0	0	0	0	0	0	0	0	0	0	0	0	30	0	0
A.15	0	0	0	0	0	0	0	0	0	0	0	9	0	0	21	0
A.16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	30

Table 1: Confusion Matrix for Bayes filter (A= actual, P= predicted)

4 NOVELTY FOR PARTICLE FILTERS

- To improve direction detection, we used both the base rotation vector and gyroscope to determine a macro (to get initial facing direction) and micro (to log any rotations) direction, which is also mentioned in Section 3 Subsection: Motion Model.
- To improve processing time, we made use of the loop for checking for wall collision to also check which cell a particle is in. This allowed us to not require additional data structures (other than an array to store particle count) and also no additional processing time to reiterate through every particle/cell in other methods which can add up, since we are refreshing every second.

5 INDIVIDUAL WORKLOAD

	Martyn Wong	Lim Yong Song
GUI		X
Data collection	X	
Data handling		X
KNN model	X	X
Bayes AP filtering	X	X
Bayes Calculation	X	X
PF MAP	X	X
PF Motion Model	X	X
PF Rotation	X	X
PF Resampling	X	X

6 ATTENDANCE/ CHALLENGES

- We attended all the lectures (6) and most of the lab sessions (3).
- The most challenging parts were data collection and handling, especially when it comes to data processing where we need to refine the data.

7 POSSIBLE FUTURE DIRECTIONS

- Attempt more types of smoothing/filtering processes for the Bayes portion to see which is better.
- Get a more accurate direction detection algorithm working.
- Get a more targeted walking motion detection algorithm working.