Report for Assignment 2

Group Member:
Rui Yang (467656)
Can Zhu (467711)

Answer for questions in 2.1.1

a) The mechanism behind WI-FI fingerprint based indoor localization is by measuring the received signal strength of possible multiple access points with the collected data in the data sets to estimate the location of the device. The parameters in fingerprint may include SSID and Mac address of the access points and the value of received signal strength.

b) The advantages of this method mainly include that relying on WI-FI infrastructures only, no requirements of knowledge about the info of access points and no requirements of any explicit user participations;

The main disadvantages would be:

1) Users are required to open WI-FI all the time, which consumes a lot of power energy;

2) Access point scanning will interrupt normal data communication;

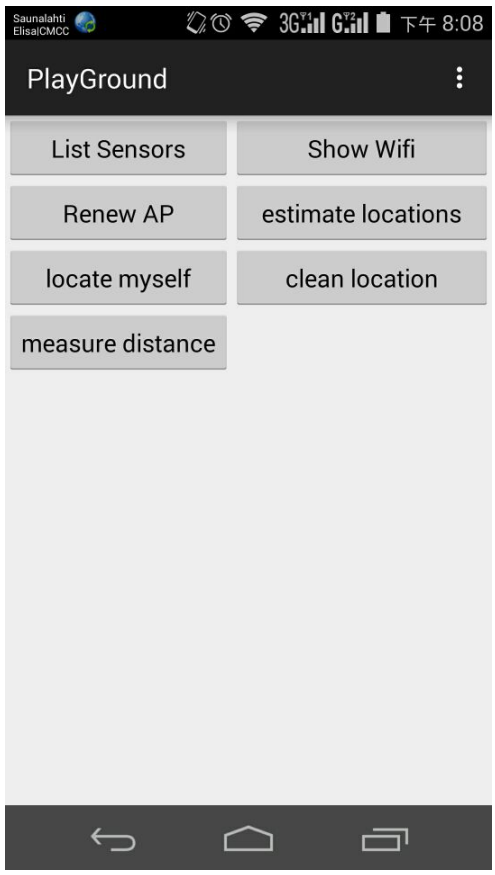
3) It may not work in the case of multi-floored buildings.

c) There are many commercial usages right now, such as Aeroscout, WhereNet, Newbury and Ekahau;

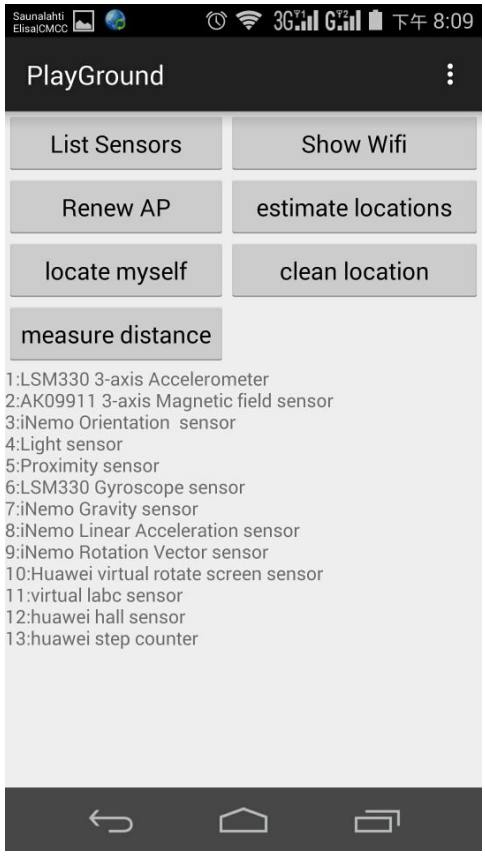
d) Unlike WI-FI, Bluetooth beacons can be more easily confined to a single room thus making it useful for localization. However, they are also likelihood. For example, both of them require pre-installed infrastructures. And during the process of localizations, Bluetooth and WI-FI are also required to be active, which consume a lot of energy. Speaking of the usage of ultrasounds in indoor localization, compared to WI-FI based localization mechanism, it does not require central management receiver computation and achieves much higher accuracy. And also unlike WI-FI having been widely embedded in most of smartphone nowadays, ultrasounds based localization has a smaller user group since a special hardware for this mechanism is needed whereas not widely installed.

Overall Android App description

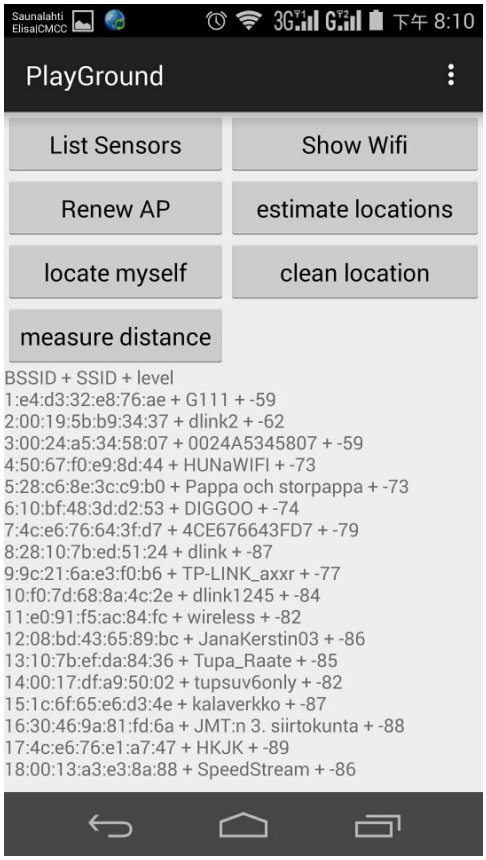
(Note: the picture given below is corresponding to the name below itself) The name of Android App for this assignment is PlayGround running on Android 4.4.2 in Huawei Ascend P6. The outlook of this App is showed in picture one. There are seven buttons in the layout panel. “List Sensors” button is to list the sensors that are available in current device. Picture two shows the result of listing sensors. “Show Wifi” button is to scan the nearby WI-FI access points and return the scanned result. Picture three presents the result of showing WI_FI.



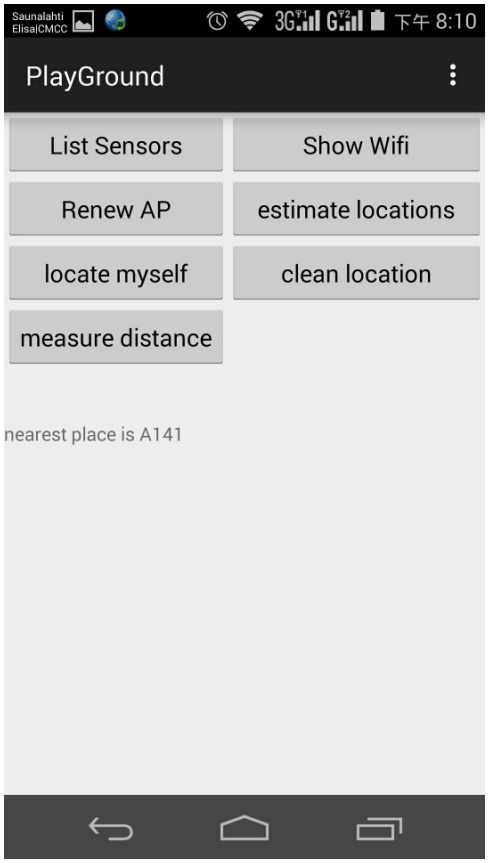
Picture One



Picture 2

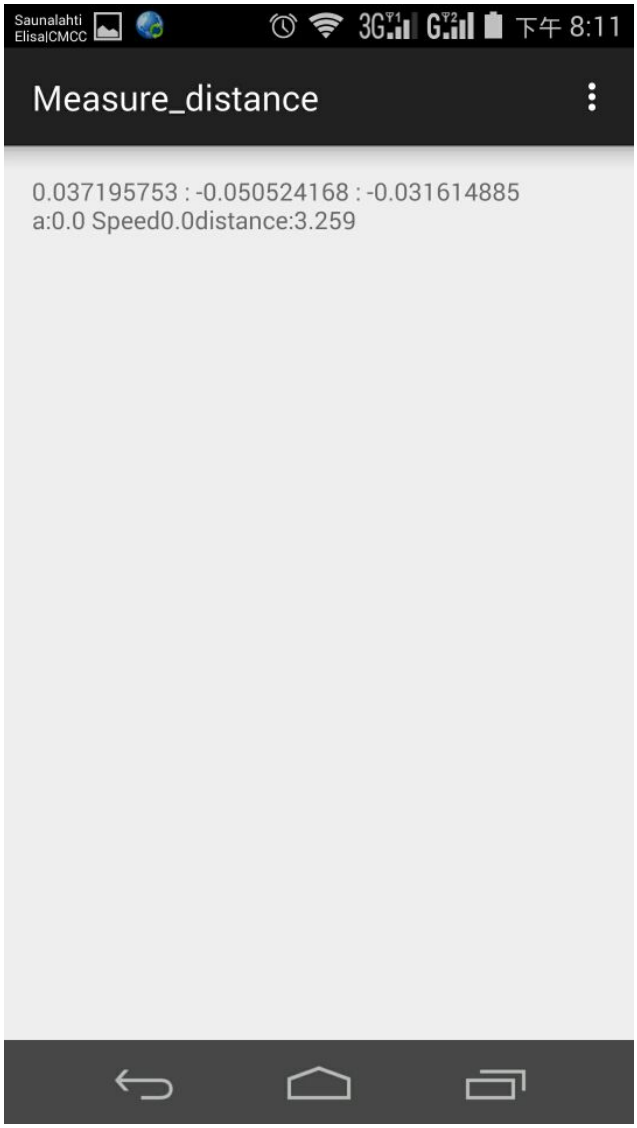


Picture Three



Picture Four

(Continue) “Renew AP” button is to renew the calibration data from the assets fold where the calibration files are stored. After clicking the “Renew AP” button, clicking “estimate locations” can show the estimated location based on the measurement data that we were given. The measurement data we chose is the random location 2 (A150). The output of estimating the location of the random location 2 is given in picture four. Button “locate myself” is real time localization function that calculating the current location of the device with referring to the six calibration locations. More detailed information could be found in “Description for 2.1.2 part C step Two”. And “clean location” is to clean current cached data for localizing the device. This should be done when the device is moved to another location. And the “measure distance” button is to calculating the distance between two reference locations. In order to properly run this measuring distance function, when starting from the initial location, the device should be hold horizontally and the initial speed should be 0. There is an example output of this function presented below in picture five. Estimated distance is given in textview.



Picture Five

Description for 2.1.2 part C step one:

The procedure to estimate the location contains two processes:

1) Collecting calibration data.
2) Comparing measurement data with the calibration data to estimate the location.

First of all, speaking of collecting calibration data, based on the measurement data set given in JSON format of 6 different locations, we characterize each position with the data structure of a list of {"Name_of_AP", "Average_of_Level" }. The "Name_of_AP" stands for the name of each access point and "Average of Level" for the average value of the total 15*3 measurements. So in general, each "location" contains the average level values of different access points.

Now, proceeding into comparing measurement data with the calibration data to estimate the location, we also parsed these measurement data into the same data structure of calibration data mentioned above. After that, we calculate the "distance" to different locations using the algorithm $(\sum_{i=1}^{n}(Si - Ci)^2)$ similar to Euclidean distance algorithm. The location with the smallest "distance" among six locations is the nearest location that we are about to find out. In the algorithm, n represents the number of scanned AP in measurement data set, Si for the average level value for no. i-th of APs in measurement data and Ci for the average level value for the same AP in calibration data. If the AP does not exists in the calibration data set, just set the value of Ci to -99.

Based on our observation about how does the number of measurement points affects the estimation, in general, the accuracy increases with the increase of the number of measurement points and this assumption only valid when the previous selected points are also exists in current chosen data sets. Otherwise this conclusion is not valid.

Description for 2.1.2 part C step two:

Most of time, our App returns the same location with both data. Sometimes due the fluctuation of the WI-FI signal and the direction that the device is hold, the signal strengths for different access points are different.

Description for 2.2:

We have implemented a prototype of distance measurer by using the accelerometer sensor (`Sensor.TYPE_LINEAR_ACCELERATION in android`). In order to calculate the distance, the initial speed of the device should be zero and the device should be hold in horizontally. The output from the accelerometer sensor is used to calculate the acceleration marked as A[3] for 3 directions. In order to eliminate the noise, a noise rate R, which is equal to 0.9, is given. Then

based on the algorithm,

1) current_speed = previous_speed + A * dT,
2) distance = previous_speed * dT + 1/2*A*dT*dT.

(dT is around 0.2 s)
we can calculate the distance traveled within dT time. And accumulating the distance from the start point to the total distance. Since there are many biases in this algorithm under the real environment, the accuracy cannot be guaranteed.

Summary

The total time estimation for this assignment is around 1 week. 4 days for developing App, 1 day for collecting data and 2 day for the report. This assignment overall give me a basic idea about how can we use the sensors to get the result that we want and the required efforts have been put into this assignment. In sum, this assignment is quite fair.