**2.0 Algorithm**

**2.1 Raw Data**

The raw data that we have collected are in the form of the RSSI readings from different stations and at each sample point we would collect about 6 RSSI readings from each of the base station as shown in Figure 2.1. From the data that we have collected, we found out that at some points that the RSSI from the same base station would deviate to up to 5 units of RSSI readings. That’s why we need to collect a reasonable many figure from each base station at the same point so that we could compute the average for prediction in the cleaning phase. However, there is another problem, that at the same point that some of the station’s reading are not very stable, aka, sometimes we will get readings from that base station and sometimes not. This is equivalent to a voting scenario, where if a user only votes for once, we are not confident that the voting is trustworthy enough. So, regarding this we will introduce some weight accordingly to better reflect the trustworthiness of each base station readings.

**2.2 Data Cleaning**

What we expect from the clean data is an array of readings from base station 1...5 and some weight corresponding to each of the base station w1…w5 as in the format below

(as shown in Figure 2.2):

*s1, s2, s3, s4, s5, x, y, w1, w2, w3, w4, w5*

Each base station reading Si is the average of the raw readings from each sample point data collection. x , y is the coordinates of that sample pont.The weight is computed as discussed above as the trustworthiness of the base station reading at a particular sample point according to the frequency their reading appears in that sampling point. For example, if we collected 6 times at a sample points, base station 3 have 2 readings out of the 6 times reading request and base station 1 have 6 readings out of the 6 times reading request, then the trustworthiness of base station 1 is 3 times the base station 3 since 3 = 6/2. These weights would be later used in calculation the Euclidian distance later on in the prediction phase.

We collect the test point in the same manner as described above and output is exactly as shown in Figure 2.2 with weights to each of the base station.

**2.3 K-nearest Neighbour Prediction**

Now we have the data in the format that we want with its weight, we can use K-nearest neighbour method to make the predictions based on the test data’s RSSI readings. We could use the following algorithms based on the Euclidian distance to find the K nearest sample points to predict the location:

For each of the test point, compare with each of the sample points and compute the Euclidian distance with each sample points and then pick k many sample points with the least distance with the test point. Then base on the k sample points we find the centre of the k points which is the predicted location of the test points. However, when calculation the Euclidian Distance the weight that mentioned before will be taken into consideration by multiplying the weight.

According to Figure 2.3 we could see that in this case k = 4 will gives us the best solution with error of 5.4 metres. Also, we have tried to take out the weight that we have mentioned before and the average error increase from 5.54 to 6.15 so we conclude that at this point with the weight and k = 4 would be the best solution for now.

**2.4 Discussion for the Error**

In the original paper, the error was to be around 2-3 metres, while currently the initial test of our experiment shows an average error of 5.54 metres. However, the outcome of this is expected for the following reasons:

1. Hardware Difference:

The experiment set up in the paper is using pc as base station and sending WIFI signal using wireless network card, while in our experiment we only use sensor tag as our base stations and using RSSI to send and receive signals. In terms of the transmission power and signal stabilities, wireless network card has far more advantages with stable power supply and signal strength. Since sensor tag runs on coin battery, it has very limited to send strong signals. Also, WIFI card could have much better abilities for signal penetration and deal with the signal to noise.

1. Algorithm Optimisation:

The current algorithm implementation is only minimum since our milestone one is focus on the rough replication of the original experiment namely to collect data and do a quick and dirty implementation. However, there are plenty of rooms to implement algorithm optimisation. For example:

-Further tweak on the weight when calculation the Euclidian Distance

-Use Kalman Filtering to Further improve the data aggregation

-Introduce another prediction method namely using known base station coordinates

and RSSI reading from the base station to estimate the distance from the top closest K base stations and use least square fit to find the predict location.

(This method could be tested independently first and then by adding some appropriate weight we could aggregate this method and the K nearest neighbour method together to make more accurate estimation).



**3. Review on project progress**

Intermediate Report

As scheduled, on the intermediate report day, we should be in the middle of our algorithm implementation. In fact, we are already at the end of the implementation, and ready to start our improvement phrase. As is showed in part 2, our outcome is acceptable considering our test is based on Sensor tag. For now, we are just slightly ahead of the schedule.

**4 Future Plan**

**4.1 Plan for algorithm improvement**

* Hardware Method
  + Using iBeacon to improve the accuracy

Since the Sensor Tag is not for commercial use, it may affect the accuracy of our project. One of our proposed possible improvement is to introduce iBeacon into our project. And we will do some test to see if there will be an obvious improvement or not.

* Software (Algorithm) method
  + applying different weight strategy

Sample point

Test point

**A**

**B**

Currently, our result is based on KNN, although the accuracy is acceptable, but there still some problems remain. One is that points is a certain area will get the same result, for example, in Figure 2, Point A and B will certainly get the same result (K = 4). It is due to the KNN strategy, obviously it can be improved. One possible solution is to get more sample points, but it is not efficient. So, the problem turns into how to improve the result with limited samples. One reasonable way is to add weight to the algorithm. Currently, we proposed two possible weight strategies.

1. Weight based on successful commination times.

It is clearly that we will not get the RSSI data once since the data may vary. There will be some package lost during the collecting. Usually the lager the distance, the more package lost will appear. And with the increase of distance, the RSSI data become more unreliable. Then, it is reasonable to give less weight to a point with high package lost. It is possible to use the package lose as a weight indicator.

1. Weight based on signal strength.

The signal strength is another promising indicator. The stronger the signal strength, the smaller the distance. Therefore, using the signal strength is an attractive method to make the predict more accuracy.

* + Other possible improvement algorithms

Besides the above mentioned, there are quite a few other methods that may contribute to our project. Like mentioned in 2.4, we can calculate the distance from the base station using RSSI and this makes it possible to get the position of the point.

We will try to apply various improvements if we have enough time. The final solution may not be limited to one improvement but a combination of them. It depends on the performance and remain time.

**4.2 Plan for others**

* Testing

The testing procedure has already begun, since it is not possible to check the accuracy without any test. But still we will execute an overall test after all the improvement complete.

* Output

We propose to show the point on the floor plan instead of giving a coordinate as an output.

In conclusion, we are confident that our project can achieve the initial target and may even go beyond it.