University of Miskolc
Institute of Information Science
Department of Information Technology

Comparison of WiFi RSSI Filtering Methods

Author
Bence Bogdándy
CE4RB0

Supervisor
Zsolt Tóth

2016.

Declaration of Authorship

I hereby certify that this thesis has been composed by me and is based on my own work, unless stated otherwise. No other person's work has been used without due acknowledgement in this thesis. All references and verbatim extracts have been quoted, and all sources of information, including graphs, have been specifically acknowledged.

Miskolc, April 15, 2016

Bence Bogdándy

Contents

Chapter 1

Introduction.

Providing a filter to accurately predict and remove noise from measurements is a very active research area. The Horus system showed that using time series instead of single values increases efficiency. Filters can be applied as preprocessing algorithm to a WLAN to make steady connection to an Access Point. Analysis of time series of WiFi RSSI measurements could lead to the development of an efficient client site filtering method for indoor positioning systems. There are various time series filtering methods in the literature. This paper focuses only on two simple solution which are based on time windowing. In the case that the two filters work efficiently, the process could be applied to networks with different filters to reach an optimal solution. Computational complexity also gives an important constrain of these filtering methods, because the client devices usually have limited computational capacity and battery life.

1.1 ILONA

The presented results are connected to the Indoor Positioning Research at the University of Miskolc. The ILONA System is web application for indoor positioning. It provides positioning functions for client. ILONA stands foor INdoor LOcation and NAvigation which is a web application created to perform indoor positioning and navigation tasks. It is made up by loosely coupled components such as `measurement, positioning, navigation` and `tracking`. This paper focuses on the data analysis and data mining of the ILONA System. A proper client side filtering method could increase the performance and the accuracy of the positioning service of the ILONA System.

Chapter 2

Related Works

This chapter presents the basic concepts of indoor positioning systems the most widely used technologies and gives a brief overview of the mathematical background of modeling and filtering time series.

2.1 Indoor Positioning

The first Indoor positioning systems were planned in the 1980s but they become popular in the last decade [13, 9]. IPS systems can be used in hospitals, malls, airports and offices. Indoor positioning is challenging due to the unique properties of the indoor environment. GPS is a very popular method of position, however it is not suited for such tasks because of its line of sight requirements and the multi path effect. Wireless LAN, Bluetooth, infrared, ultrasonic and radio frequency technologies are the most popular. Active Badge [15] was the first indoor positioning system which was based on infrared signals. Fingerprinting methods were first presented by the RADAR system [4]. RFID [14] based technologies has emerged in the last years. Hybrid Indoor Positioning Systems [5] has been created in the least few years. This paper connects to the ILONA System which is a hybrid indoor positioning and navigation framework. [17].

2.1.1 Indoor Positioning with Wireless LAN

Wireless LAN(WLAN) was designed to transmit data wirelessly over small distances, and connecting to the internet. WLAN also can be used for indoor positioning purposes [8]. WLAN is highly available since most devices can receive and process WLAN signals, smart phones can be programmed easily so they can implement client side filtering methods. It is also very cost-effective since the data received is highly accurate in most situations. Received Signal Strength Indication is measured by the device and it is determined for each available Access Points. WLAN positioning is based on the measurement of the signal strength value (RSSI). The user is linked to or multiple Access Points(AP) to determine user location at any given time. The User has to have a relatively stable and continuous connection. Unfortunately, the RSSI signal is not stable. It fluctuates over time even without any environmental impact. This fluctuation can be filtered at client side Electromagnetic signals are reflected [12] by most bodies, as it can be seen on Figure 2.1. A few

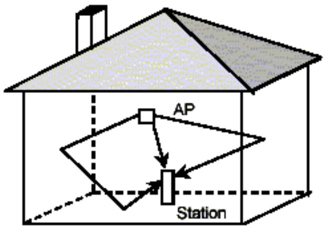

Figure 2.1: Indoor WLAN signal reflection [1]

people in a room can drastically alter the measurements, making it less reliable the more reflective bodies there are in the room.

2.1.2 Fingerprinting Methods

Although WLAN was not developed to be a positioning system, it is a popular candidate for indoor positioning systems. Most of the existing solutions are based on the fingerprinting technique [3, 10]. Instead of determining the distance between the AP and the user by the RSSI, it creates a radio map. Fingerprinting based indoor positioning methods require huge computational capacity, so they can be performed only on the server side.

2.1.3 WiFi Based Solutions

WiFi [6] based positioning popular due to the spread of mobile and smart phones in the early 2000's. However, GPS and AGPS are available in the modern phones, these technologies cannot be used in indoor environment. As more buildings, like hospitals, malls, etc. are designed with indoor WiFi Access Points in mind, it is only natural to try and use these APs to determine location.

2.1.4 Horus System

The Horus [16] system is location determining system based on the IEE 802.11b WiFi , and Bluetooth connection, as seen on 2.2. The system determines user location by the received RSSI values from the access points. RSSI values are sent automatically, so implementing the system is purely a software solution, requiring no modifications on the hardware.

2.2 Time Series Filtering Methods

2.2.1 Time Series

Time series [11, 7] is a set of observations (x), being recorded a specified time (t). It is stochastic in a sense that most of the time, measurements closer together have more importance towards each other. Time series are often plotted with line charts on a one

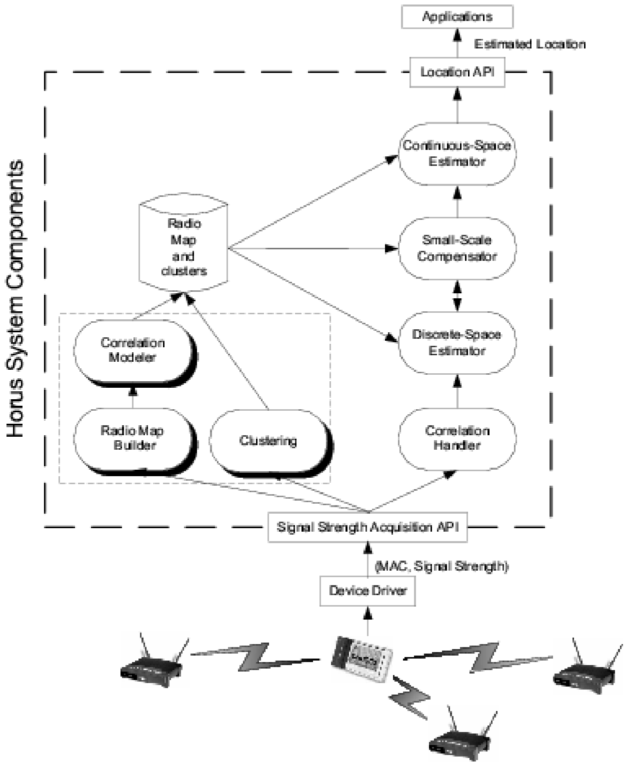

Figure 2.2: Horus System [2]

dimensional panel, as they represent the data fluctuations well. They are very frequently used in most domain of applied science.

2.2.2 Discrete Time Series

A discrete time series is one which the set T_0 of times at which observations are made is a discrete set, for example being recorded at fixed time intervals. It is called continuous, if the measurements are being recorded continuously over a set amount of time.

2.2.3 ARMA Models

The three main modelling variations of importance are the autoregressive(AR), moving average (MA), and the integrated (I) models. The ARMA modelling of wifi rssi will be the topic of a future research.

AR(p) refers to the autoregressive model of order p

$$X_t = c + \sum_{i=1}^{p} \varphi_i X_{t-i} + \varepsilon_t.$$

where $\varphi_1, \ldots, \varphi_p$ are parameters, c is a constant, and the random variable ε_t is white noise.

Some constraints are necessary on the values of the parameters so that the model remains stationary. For example, processes in the AR(1) model with $|\phi_1| \geq 1$ are not stationary.

The notation MA(q) refers to the moving average model of order q:

$$X_t = \mu + \varepsilon_t + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i}$$

where the $\theta_1, ..., \theta_q$ are the parameters of the model, μ is the expectation of X_t (often assumed to equal 0), and the$\varepsilon_t, \varepsilon_{t-1}, ...$ are again, white noise error terms.

The notation ARMA(p, q) refers to the model with p autoregressive terms and q moving-average terms. This model contains the AR(p) and MA(q) models,

$$X_t = c + \varepsilon_t + \sum_{i=1}^{p} \varphi_i X_{t-i} + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i}.$$

The general ARMA model was described in the 1951 thesis of Peter Whittle, who used mathematical analysis (Laurent series and Fourier analysis) and statistical inference.

2.2.4 Filtering Methods

Filtering is the method of separating the noise from the signal, prediction of future values and control of current and future values.

2.2.5 Time Windowing

Time windowing is the process of allowing certain methods to stay idle for a certain amount of time before continuing to run or terminate. This certain amount of time depends on the predefined amount, and can be modified in the code. It is purely a software solution on most cases. Programs cannot andvence without any imput. Introducing a time window can make the program dynamic, because it can respond if it doesn't receive the needed values.

Chapter 3

Filtering Methods

The goal of the following filters are to reduce caused by multiple factors. These distractions are hard to avoid, and there is no surefire way to get rid of them. Filtering aims to solve this problem by creating an algorithm to reduce noise as much as possible, as simply as possible.

The unfiltered RSSI values, on shown on a Figure 3.1 and its descriptive statics can be found in Table 3.1.

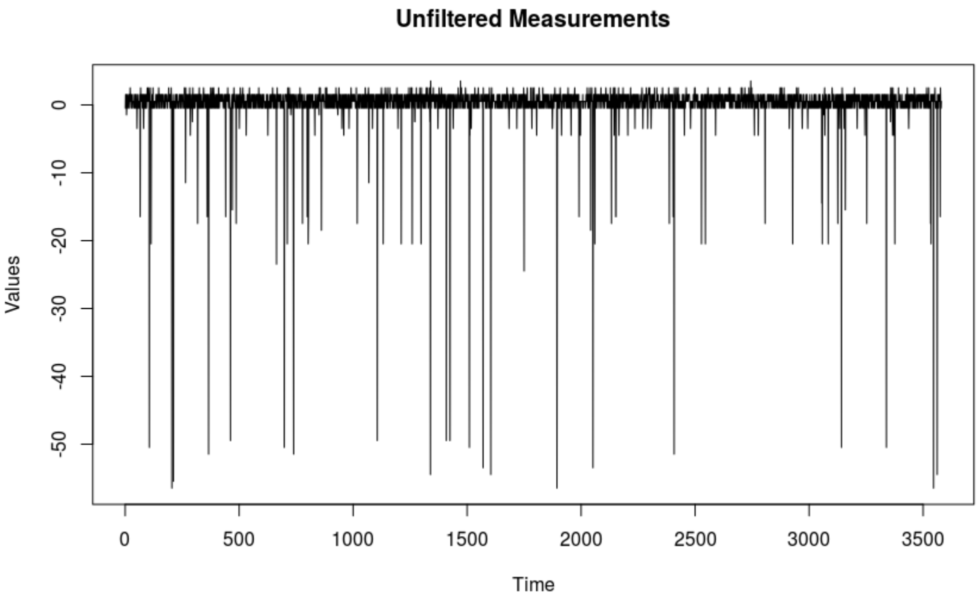

Figure 3.1: RSSI values

Table 3.1: Summary of the measurements

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
-87.00	-31.00	-30.00	-30.53	-29.00	-27.00

3.1 Formal Description

3.1.1 First Filter

The model of the first filter is the following $y = (y_1, y_2 \dots y_n)$ is the filtered vector, $x = (x_1, x_2, \dots x_n)$ is the starting value vector $n \in \mathbb{N}$ is the number of values in the vector and $m \in \mathbb{N}$ is the memory size.

$$y = (h(x, 1, m), h(x, 2, m) \dots h(x, m, m), f(x, m+1, m) \dots f(x, n, m)) \tag{3.1}$$

Where

$$h(x, i, m) = \frac{(\sum_{j=i}^{i+m-1} x_i)}{m} \tag{3.2}$$

And

$$f(x, i, m) = \begin{cases} x_i & \text{if} |x_i - x_{i-1}| < t \\ \overline{(x_i \dots x_{i-m})} & \text{if} |x_i - x_{i-1}| > t \end{cases} \tag{3.3}$$

The first part(h (x, i, m)):

The algorithm follows a simple pattern for the first 5 values. It takes the average from $x_i \dots x_{i+m}$ indexes to create the filtered value of $y_1 \dots y_m$. After the first m elements are calculated, it uses a different f function to calculate the rest of the elements of y.

The second part(f (x, i, m)):

After the first 5 elements, the algorithm's actions are determined by a t variable. If t is lesser than $|x_i - x_{i-1}|$, so the difference between the current and the last element exceeds the value of t.

The average of the last 5 x value is put into the current y element. If the previous condition is false, then the current x element is used as the filtered value.

3.1.2 Second filter

The second one's mathematical model: $y = (y_1, y_2 \dots y_n)$ is the filtered vector, $x = (x_1, x_2, \dots x_n)$ is the starting value vector $n \in \mathbb{N}$ is the number of values in the vector and $m \in \mathbb{N}$ is the memory size. The second filter's algorithm is much like the first one, except t is calculated after every iteration:

$$t = \sqrt{\frac{1}{m} \sum_{j=i}^{i-m} (x_j - \overline{x})^2} \tag{3.4}$$

The second part of the algorithm is different:

The f function determines the elements by calculating a t threshold that changes dynamically through the function. It is always calculated by taking the standard deviation of the last 5 elements of the unfiltered vector. After the algorithm knows the value of t, it checks if the difference between the current element and the last element is greater than this value.

In the case that it is, it takes the average of the last m elements of the X vector and puts it in the y_i element.

If it's lesser then the t value, then x_i is imply put into y_i.

3.2 R Implementation

The implementations of the mathematical descriptions were done in R script. The two filters are similar in nature, although offering different results. The end results differ mildly depending on the choosing of threshold and memory size. Choosing those two variables is key to an optimal filtering.

3.2.1 First filter implementation

The first part of the algorithm (h(x, i ,m)) can be seen on Figure 3.2, while the second part (f(x, i ,m)) can be seen on Figure 3.3

```
while(j<=memsize)
{
  i=1;
  while(i<=memsize)
  {
    a[i]<-Measurement$Signal[i+j-1]
    i=i+1
  }
  FilteredMeasurement$Signal[j]<-mean(a)
  j=j+1
}
```

Figure 3.2: The implementation of h(x,i,m).

The diagram of the first filter with the threshold and memory size both chosen as 5 can be seen on Figure 3.4, and the summary of the statistics of the filtered measurements can be seen on Table 3.2

Table 3.2: Summary of the measurements after the first filter

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
-47.00	-31.00	-30.00	-30.21	-29.00	-27.00

```
while(x<=nrow(Measurement))
{
    if((abs(Measurement$Signal[x]-Measurement$Signal[x-1]))>thr)
    {
      i=1;
      while(i<=memsize)
      {
        a[i]<-Measurement$Signal[x-memsize+1+i]
        i=i+1
      }
      FilteredMeasurement$Signal[x]<-mean(a)
    }
    else
    {
      FilteredMeasurement$Signal[x]<-Measurement$Signal[x]
    }
  x=x+1
}
```

Figure 3.3: The implementation of f(x,i,m)

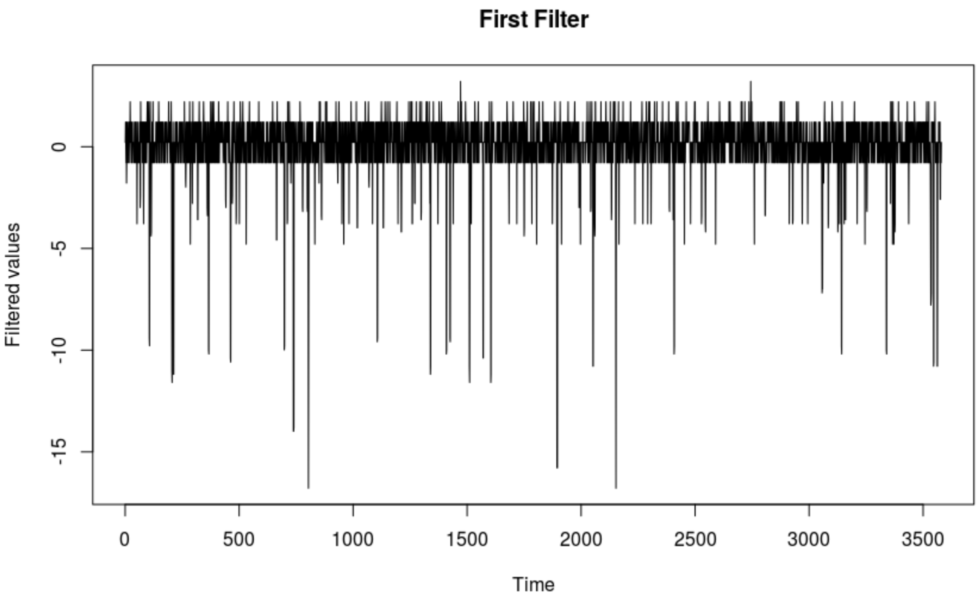

Figure 3.4: Values with the first filter

3.2.2 Second filter implementation

Since the first part of the algorithm is analogous with the first one, it will not be included in this section. The second part of the implementation can be seen on Figure 3.5

```
while(x<=nrow(Measurement))
{
  thr<-(sd(a))*3
  i=1;
  while(i<=memsize)
  {
    a[i]<-Measurement$Signal[x - memsize + i]
    i=i+1
  }

  if((abs(Measurement$Signal[x]-Measurement$Signal[x - 1]))>thr)
  {
    FilteredMeasurement$Signal[x]<-mean(a)
  }
  else
  {
    FilteredMeasurement$Signal[x]<-Measurement$Signal[x]
  }
  x=x+1
}
```

Figure 3.5: The implementation of f(x,i,m).

The diagram of the filtered time series can be seen on Figure 3.6, and the summary of the statistics can be seen on Table 3.3

Table 3.3: Summary of measurements after the second filter

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
-54.00	-31.00	-30.00	-30.07	-29.00	-27.00

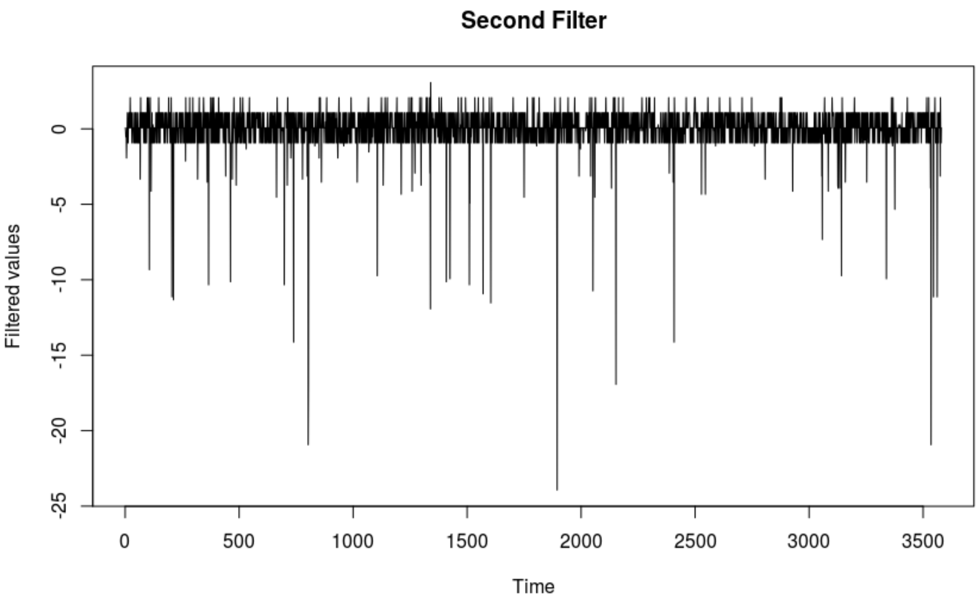

Figure 3.6: Values with the second filter

Chapter 4

Evaluation

4.1 Test Environment

The measurements were recorded in the University of Miskolc's Institution of Information Science, Department of Information Technology. The recorded values were taken with 4 identical smart phones with Android operating system. Two of the tests were recorded without any disturbance, just the natural reflective surfaces in the room, and two were recorded with people present in the room. The latest showed a lot of inconsistencies on the recorded values, further increasing the need for a filter on the preprocessed data. As noise grows, the filters value increases, as it can be smoothed out the naturally occurring jumps on the diagram. The following sample is taken from the data set used to test the filters on:

```
Time,SSID,MAC,Signal
2016/02/02.09:43:33,IITAP1,E0:5F:B9:0C:71:27,-39
2016/02/02.09:43:33,IITAP1-GUEST,E2:5F:B9:0C:71:27,-39
2016/02/02.09:43:33,KRZ,0:18:E7:DE:A3:90,-48
2016/02/02.09:43:33,LABOR,00:14:C1:33:A0:78,-72
2016/02/02.09:43:33,GEIAKFSZ,F8:66:F2:AD:E6:91,-76
2016/02/02.09:43:33,doa207,14:CC:20:57:3D:0C,-67
2016/02/02.09:43:33,dd,00:19:E0:65:E4:F2,-84
2016/02/02.09:43:33,doa208,F8:66:F2:AD:E6:79,-81
2016/02/02.09:43:35,IITAP1,E0:5F:B9:0C:71:27,-40
```

Figure 4.1: Data set sample

4.2 Comparison

The filters have clearly shown that their usage have increased the accuracy and steadiness of the WiFi signals. These results mean that the usage of preprocessing filters can be used to steadily improve positioning. The comparison of the diagrams

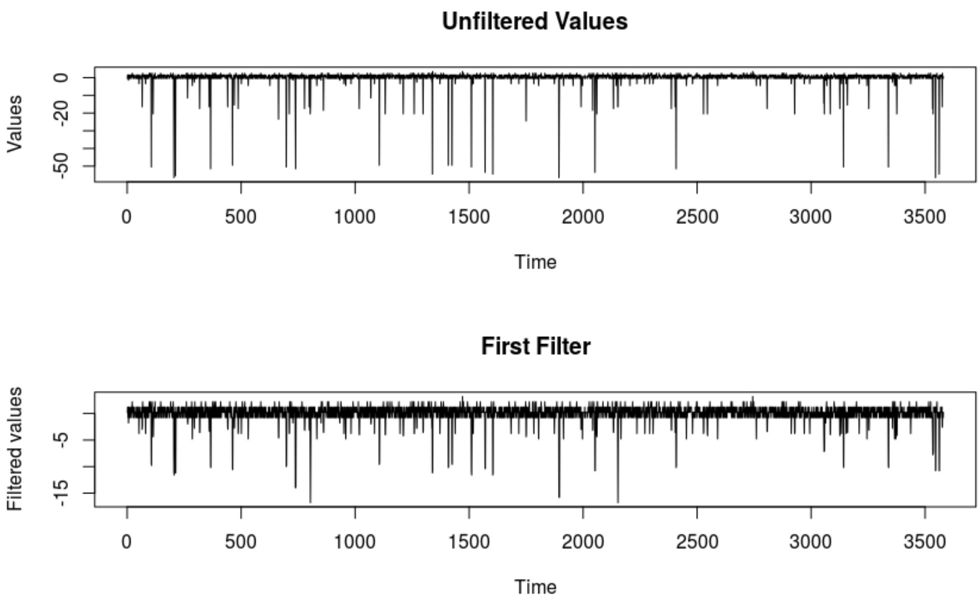

Figure 4.2: Comparison of the values of the raw and the values after first filter

As seen on the Figure 4.2 the filter marginally increases the continuity of the series. It dynamically reduces the range of numbers in which the series works in. Please note that the original, unfiltered series' lowest value was exceeded -50, where with the first filter it only goes down to around -15.

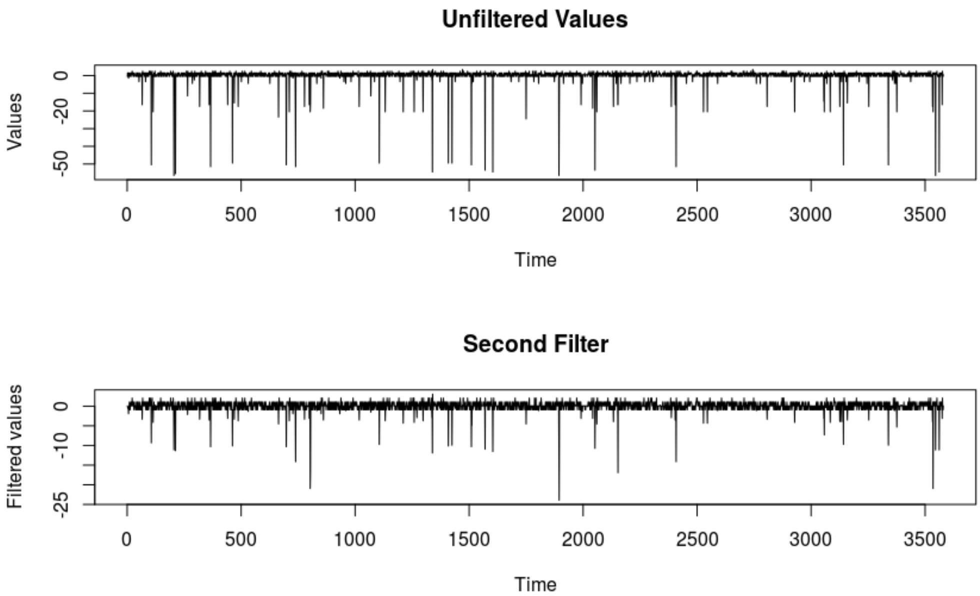

Figure 4.3: Comparison of the values of the raw and the values after second filter

The second filter as seen on Figure 4.3 is similar in nature. It reduces the range of numbers by more then half of the original values. It has a more complex algorithm, but it is less effective than the more simple, first filter, where the threshold isn't computed dynamically. It may be more useful in environments where filtering has to be done on unknown data sets.

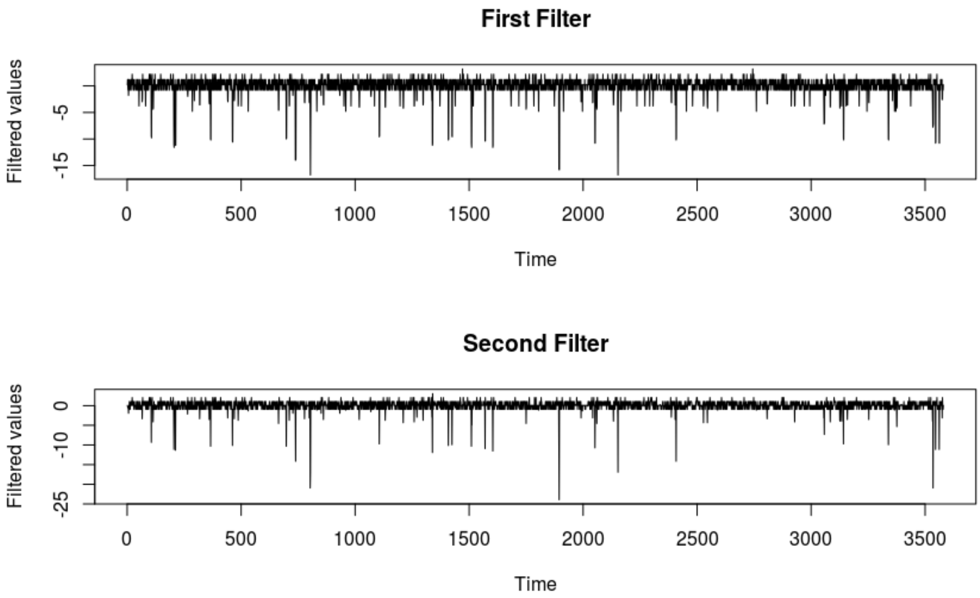

Figure 4.4: Comparison of the values of the two filters

4.3 Suggestion

The first filter works well if you can specify the threshold amount well. It can marginally reduce the range, thus increasing the efficiency and accuracy of WiFi RSSI measurement in an indoor positioning environment. The second filter is more dynamic in nature, as it can work on any data set without setting a certain threshold. Although less effective, it may be more practical to use in real life situations. More filters can be developed in the future to increase effectiveness. Computational complexity also has to be taken in mind. The first and second algorithm are quite simple, and could be run on most devices, but as effectiveness grows, so could complexity, rendering more and more devices unable to use preprocessing filters.

Chapter 5

Summary

The project began with ILONA, a hybrid Indoor Positioning System that uses WLAN. The need of improving accuracy came up mainly because the measured data were often unusable due to inconsistencies in the RSSI values. The solution could not be done on the hardware side, so it had to be done by a piece of computer code. We started out with a big mass of data, which were recorded for the purpose of testing. The data clearly showed points of time where the connection was disturbed, and the RSSI values were significantly altered compared to the neighboring measurements. The goal was to create a method that would smoothen out these values, keeping the connection steady. Having consistent and accurate RSSI values are essential in indoor positioning, as the program calculates position based on these values.

The data was processed and represented in R as a time series. Two formulas using time windowing were created and implemented as filters. R Studio was used as a development environment. The first filter requires 2 inputs, a threshold and a memory size. Choosing these inputs optimally is crucial to the algorithm, as the result depends heavily on them. The second filter is much like the first one, except the threshold is generated dynamically. This can lead to a less effective, albeit more independent algorithm.

After the algorithms were complete, and working optimally, diagrams were created to better represent data, and to evaluate the results of the filtering. The results clearly show the improvement over using raw values. The range of values is heavily reduced and the diagrams are smoothened out significantly. Significantly improving the examined data set, it can be clearly stated that the end result is successful.

Using filters can lead to a more steady and accurate set of RSSI values . More complex algorithms can be developed in the future to further improve functionality. The two filters in this paper are easy to understand, and simple to implement. Their computational requirement isn't high. The program only work with a few integers and real numbers to calculate optimal values. Presumably, as the algorithms change, so will their computational requirement, although they should remain relatively simple to be able to use as a preprocessing program constantly.

Bibliography

[1] 802.11g starts answering wlan range questions. http://www.eetimes.com/document.asp?doc_id=1271877. [Online; accessed 02-April-2016].

[2] The horus wlan location determination system. https://www.usenix.org/legacy/publications/library/proceedings/mobisys05/tech/full_papers/youssef/youssef_html/index.html. [Online; accessed 08-April-2016].

[3] Properties of indoor received signal strength for wlan location fingerprinting.

[4] Paramvir Bahl and Venkata N Padmanabhan. Radar: An in-building rf-based user location and tracking system. In *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 2, pages 775–784. Ieee, 2000.

[5] Artur Baniukevic, Christian S Jensen, and Hua Lu. Hybrid indoor positioning with wi-fi and bluetooth: Architecture and performance. In *Mobile Data Management (MDM), 2013 IEEE 14th International Conference on*, volume 1, pages 207–216. IEEE, 2013.

[6] Yongguang Chen and Hisashi Kobayashi. Signal strength based indoor geolocation. In *Communications, 2002. ICC 2002. IEEE International Conference on*, volume 1, pages 436–439. IEEE, 2002.

[7] James Durbin and Siem Jan Koopman. *Time series analysis by state space methods*. Number 38. Oxford University Press, 2012.

[8] Frédéric Evennou and François Marx. Advanced integration of wifi and inertial navigation systems for indoor mobile positioning. *Eurasip journal on applied signal processing*, 2006:164–164, 2006.

[9] Yanying Gu, Anthony Lo, and Ignas Niemegeers. A survey of indoor positioning systems for wireless personal networks. *Communications Surveys & Tutorials, IEEE*, 11(1):13–32, 2009.

[10] Ville Honkavirta, Tommi Perälä, Simo Ali-Löytty, and Robert Piché. A comparative survey of wlan location fingerprinting methods. In *Positioning, Navigation and Communication, 2009. WPNC 2009. 6th Workshop on*, pages 243–251. IEEE, 2009.

[11] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45, 1960.

[12] Mikkel Baun Kjærgaard, Martin Wirz, Daniel Roggen, and Gerhard Tröster. Mobile sensing of pedestrian flocks in indoor environments using wifi signals. In *Pervasive Computing and Communications (PerCom), 2012 IEEE International Conference on*, pages 95–102. IEEE, 2012.

[13] Hui Liu, Houshang Darabi, Pat Banerjee, and Jing Liu. Survey of wireless indoor positioning techniques and systems. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 37(6):1067–1080, 2007.

[14] Lionel M Ni, Yunhao Liu, Yiu Cho Lau, and Abhishek P Patil. Landmarc: indoor location sensing using active rfid. *Wireless networks*, 10(6):701–710, 2004.

[15] Roy Want, Andy Hopper, Veronica Falcao, and Jonathan Gibbons. The active badge location system. *ACM Transactions on Information Systems (TOIS)*, 10(1):91–102, 1992.

[16] Moustafa Youssef and Ashok Agrawala. The horus wlan location determination system. In *Proceedings of the 3rd international conference on Mobile systems, applications, and services*, pages 205–218. ACM, 2005.

[17] Richárd Németh Judit Tamás Zsolt Tóth, Péter Magnucz. Data model for hybrid indoor positioning systems. *Production Systems and Information Engineering*, 7(1):67–80, 12 2015. HU ISSN 1785-1270.