Danny Turner
12/03/08
CSE 237a
Final Project Report

Wireless Location Detection for an Embedded System

Introduction

For my final project I implemented client side location estimation in the PXA27x DVK. The basic goal of client side location is for a device to be able to determine its physical location based on some type of direct observation. A GPS receiver is a common example of client side location detection. However, it was my goal to do detection inside of a building where GPS does not work. There are several motivating reasons behind indoor location detection. First, navigation inside of a building. This could either be a person attempting to find an store in a mall or perhaps a robot attempting to deliver an object to a room. One could also imagine having an extension to a web browser that fetched ads based on location. Also, by doing location detection on a client instead of on a central server or with external assistance we can better preserve privacy.

Since GPS does not work indoors we must use a different method of location detection. People have used RFID tags, proprietary technology, and 802.11 radios. In an attempt to reduce start up costs we will also use 802.11 radios since they are part of the infrastructure of most large buildings. Given we are using radio waves we must be able to determine the distance between know transmitters, access points (APs), and our client. GPS uses time difference of arrival (TdoA). Unfortunately, this requires very accurate time stamping of all arriving packets which simply is not available in todays common operating systems and wireless chipsets. A second approach would be to use angle difference of arrival (ADoA), but again todays embedded radios are unable to determine angle of arrival. Recently there has been work in directional antennas. But, these devices are physically large, power hungry, can only resolve to fifteen degrees. While ADoA might be a promising research field, it is not ready for embedded use. This leaves us with the received signal strength indicator (RSSI) values of a few known Access Points as measured by an 802.11 wireless Ethernet card. Luckily, radio waves in free space have exponential decay with exponents around three. All IEE 802.11 cards track the RSSI for each Access Point. The protocol uses this information to decide which AP to associate with. We however, will use these values for location detection purposes.

Related Work

To do client location we must take the detected RSSI values and use them to determine our location. This is usually done in one of a few ways. The original work assumed that one took a building, divided it into 1-2 Square Meter blocks, and recorded all the RSSI values for each AP at each block. Each measurement is called a fingerprint and the collection of measurements is called a fingerprint database. However, for a reasonably sized building this offline phase could take over 100 hours **[1]**. Then to determine location one could simply match their current fingerprint against the databse. While this allowed for high accuracy, as good as 1 meter **[3]**, searching could be computationally difficult. In a normal situation an exact match is not found and several heuristics are applied to determine the best fit. Further, the database could be much larger than the space available on an embedded platform. As a result, this technique has never been heavily deployed.

Eventually, people proposed schemes that relied only on the location of known access points or sniffers. These methods were known as calibration free, because they require no offline fingerprinting stage. But, because they work with less data, they are less accurate. At best they can locate a device

with-in three meters, but the 98 percentile is often as poor as fifteen meters. The reason for the poor performance of these calibration free algorithms appears to be the buildings themselves. Radio waves dissipate exponentially, through free space. However, most buildings are filled with walls, these walls cause the radio waves to dissipate much more rapidly than through free space. As a result the algorithms can not tell the difference between passing through walls and traveling a large distance in free space. While some work has been done in attempting to understand the layout of a building it requires heavy processing of CAD images of the building.

One example of a calibration free algorithm is simply to guess our location as the AP with the strongest RSSI. This makes a good baseline since any algorithm should work as well as the most naive method. In a reasonable deployment one can expect an average error of no more than 10 Meters. A second more complicated algorithm is trilateration **[2]**. The trilateration algorithm works in three stages. First, it collects data and picks the three AP's with the strongest RSSI values. Next it computes the distance to those APs using an exponential decay function that was computed offline. These curves were created using the Jigsaw infrastructure **[5]**. Finally, it performs trilateration.

A third type of calibration free algorithms is probabilistic **[4]**. These algorithms once again divide a building into a grid. Then for each AP it uses a model to predict a probability of being at a specific location in the grid. Finally, it combines the per AP probabilities to make a final location estimation. Clearly, all of the real work is done when we estimate the probability of being at a certain location. We determine the probability using a parameterized model. In the simplest case the model is parameterized with the average and standard deviation information from each off-line fingerprint. Then for a given location the model computes the expected RSSI and compares it to the real RSSI. As expected, the model most strongly predicts one to be at the locations where the expected and actual RSSI are the same and drops off exponentially as the error increases. It is also worth noting that the standard deviation is used to determine the amount of wiggle room are model allows.

Experimental Evaluation

The timeline for this project was over the course of five weeks. In the first two weeks I was able to install the wireless card and be able to take data in promiscuous mode. This task took over twenty-five hours. The reason this was so painful was because the platform runs a 4 year old version of the Linux kernel. When this version of the kernel was being developed there were limited 802.11 cards and few of them supported monitoring in promiscuous mode. I tried three different cards before finding one that supported monitoring in promiscuous mode. I also unsuccessfully, attempted to back port drivers, for an Orinoco wireless card, from a more recent version of Linux to the 2.6.9 kernel. Once I found a card that supported promiscuous mode, I needed it to provide me with RSSI values. Unfortunately, the drivers did not support this and Linux changed how wireless drivers are designed, making back porting too difficult. Eventually, I was able to hack the driver to provide me with RSSI values.

However, when running the baseline algorithm from the section above I discovered a problem. My hacked driver in promiscuous mode only hears wireless traffic from the nearest AP. I believed this was happening for one of two reasons. One possibility was that the other Access Points are in 802.11g mode and are therefore ignored by my 802.11b card. The second is that the driver or firmware is discarding these packets. To test this theory I had my laptop broadcast in 802.11b and see if the platform detects my laptop. Since my laptop was detected, I knew it the first problem. Because of this I needed to collect data from a device that could understand 802.11g. However, I am unaware of any such cards for our platform and experience leads me to believe that 802.11g cards actually draw more power than the PCMCIA slot is able to deliver. As a result I collected data using an external device.

I collected RSSI values in seven different locations on the third floor of EBU3b. These locations can be seen as the black numbers in the image below. I then performed the baseline algorithm and trilateration. The results from the base line were as expected each device correctly locates itself as the

nearest AP. The results for the trilateration are the numbers in red. While I ideally wanted to also run a probabilistic algorithm I was unable to make one run on the platform in my limited amount of time.

From the illustration we can see that locations 1, common area, and 4, copy room worked the best. They had 3M and 4.5M errors respectively. Location 2, embedded lab, also had less than 10M meter error, making three of our points better than the base line. However the remaining locations had much worse errors as can be seen in Table 1.

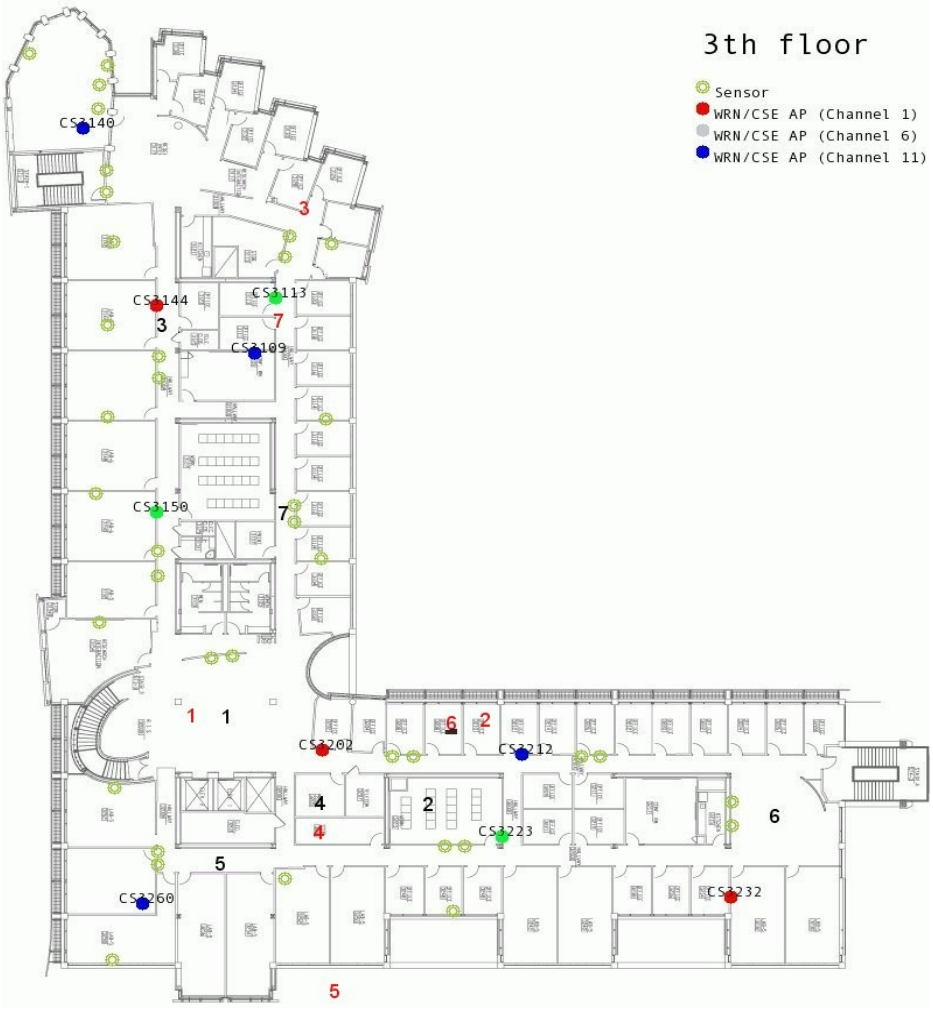
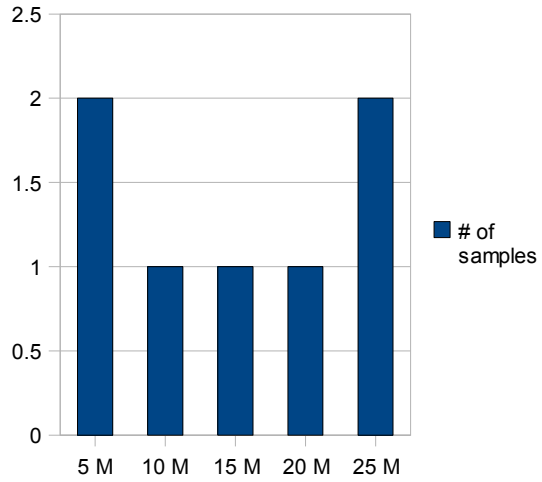


Illustration 1: Trilateration

While it is not exactly clear what causes such dramatic errors there are at least two factors that must be considered. First, that the off-line calibration data was inaccurate. Several factors can cause the RSSI values to be distorted. A device like a cell phone may have been causing additional RF interference. Also since people are mostly water, a group of people moving between an access point and the off-line fingerprint location could also distort the data. A second reason for the poor performance may be that the fingerprint in each location was not representative. The fingerprint could have been distorted in the same ways as the calibration data. Additionally, it is possible that the locations I chose to test may have had a significantly fewer or greater number off walls between the known APs than the calibration points. This would cause the fitting curves to decay too rapidly or slowly to be a proper fit, thus resulting in incorrect distance estimations and therefore incorrect tirlateration.

Table 1: Trilateration Error



Future Work

Much work remains in the field of embedded client location detection. The clear goal is of course to improve the overall accuracy of the system. One possible way might be by creating new algorithms. However, a more promising direction might be combining several existing algorithms. For example, starting with trilateration and then doing the probabilistic technique only on the the small area that trilateration suggests. A second idea might be to add additional structure to the problem. For example a reasonable assumption is that people spend most of their time either walking in straight lines or standing still. Given this assumption one could imagine building a post-processing system that uses past predictions to improve a current prediction. If one was thought to be standing still it might discard large jumps and average small variations. Or if the systems thought one were moving it could push noisy points towards the most probable line. A third option especially relevant to an embedded platform would be to query the user for input. For example if a user walked into a building the location detection system would display its current guess. If the guess was wrong the user could use a touch screen to point to the correct location. With this information the system should be able to improve all prediction in the general area by offsetting them by the error amount.

Conclusion

The overall goal of this project was to accurately perform client side location on an embedded platform. There have been several techniques for location detection systems proposed in the literature. We decided to focus on system which relied on commodity 802.11 hardware rather than a more expensive alternative. At the same time we also limited ourselves to using the RSSI values instead of TDoA or ADoA. Given these constraints we were left with three major algorithms, fingerprint matching, trilateration, and probabilistic methods. In the end we decided that fingerprint matching's offline phase was too time consuming to pressure. We also were unable to code a probabilistic method for our embedded platform in the short time frame. While we did have some trouble collecting data with our platform in the end we were able to run two algorithms. The performance of the naive algorithm was exactly as expected. The performance of the trilateration algorithm on the other land left something to be desired. Originally we wanted to determine our location to within less than 7 meters.

We did meet this goal in some location, yet failed in many more. While it is not exactly clear why we failed more often than we succeeded we did have two major ideas. First, we believe that transient events, cell phone emitting RF noise and groups of meandering people, are partially to blame. However, we also recognize that there is also a structural component to the error, in our case the walls. It is simply impossible to use a specific set of offline calibration points that have the same number of walls between APs and points as the majority of the other locations in the building. This is probably a significant reason why the fingerprinting algorithms perform so much better on non-embedded platforms. While we have also suggested avenues of future work, it is not clear that an embedded device will be able to more accurately locate itself than the nearest AP in the near future.

Citations

[1] Bahl, P.; Padmanabhan, V.N., "RADAR: an in-building RF-based user location and tracking system ," *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE* , vol.2, no., pp.775-784 vol.2, 2000
URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=832252&isnumber=18009

[2] Gwon, Y. and Jain, R. 2004. Error characteristics and calibration-free techniques for wireless LAN-based location estimation. In *Proceedings of the Second international Workshop on Mobility Management &Amp; Wireless Access Protocols* (Philadelphia, PA, USA, October 01 - 01, 2004). MobiWac '04. ACM, New York, NY, 2-9. DOI= http://doi.acm.org/10.1145/1023783.1023786

[3] Elnahrawy, E.; Xiaoyan Li; Martin, R.P., "The limits of localization using signal strength: a comparative study," *Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004. 2004 First Annual IEEE Communications Society Conference on* , vol., no., pp. 406-414, 4-7 Oct. 2004
URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=1381942&isnumber=30129

[4] de Moraes, L. F. and Nunes, B. A. 2006. Calibration-free WLAN location system based on dynamic mapping of signal strength. In *Proceedings of the 4th ACM international Workshop on Mobility Management and Wireless Access* (Terromolinos, Spain, October 02 - 02, 2006). MobiWac '06. ACM, New York, NY, 92-99. DOI= http://doi.acm.org/10.1145/1164783.1164799

[5] Cheng, Y., Bellardo, J., Benkö, P., Snoeren, A. C., Voelker, G. M., and Savage, S. 2006. Jigsaw: solving the puzzle of enterprise 802.11 analysis. *SIGCOMM Comput. Commun. Rev.* 36, 4 (Aug. 2006), 39-50. DOI= http://doi.acm.org/10.1145/1151659.1159920