Bayesian Filtering Techniques for RSSI Based Indoor Localization

Ian Jacobsen

Zebra Technologies

August 18, 2016

1 Introduction

This paper was written during the week of August 15th, 2016 with intentions of outlining the project that I undertook for my internship with Zebra Technologies in the summer of 2016. The goal of this project was to develop a solution to the problem of indoor tracking and localization. A restriction that was placed on this project was that no additional hardware should be required for the implementation, resulting in a low-cost solution.
The approach taken was to utilize the received signal strength measurements from multiple WLAN access points to provide real time estimates of the current location of Zebra mobile computers. This paper will discuss the theoretical aspects of such a solution. Due to the short amount of time allocated to this project (10 weeks), the actual implementation into the Zebra devices was not investigated.

2 Applications

2.1 Navigation

One immediate application of this technology can be guiding persons throughout a building. For example, if a visitor has arrived at an office building for a meeting but is unsure of where the conference room is located, a navigation assistant may be used to guide the visitor. This may be useful in office buildings, hospitals, and large retail environments.

2.2 Locating Devices

If a device has gone missing in a warehouse environment, this technology can pinpoint what area the device is located in. In the case where the device has accidentally been left on a truck, the data will show the trajectory onto the loading dock, and it can then be narrowed down to which truck based on the time stamp of the trajectory.

2.3 Data Analytics and Product Placement

In a retail environment, if the customer is equipped with a *shopping assistant* (i.e. MC18), the path which the customer takes may be collected and stored for data analytics purposes. For example, it may be of interest which aisles are most frequented, which locations shoppers commonly pause at, and which areas have the least amount of foot traffic. This data can be used to sell

advertising space at a premium, to optimize product placement, or to aid in a deeper analysis of product sales.

3 Simulation Environment

3.1 Room Simulation and Access Point Placement

As a first step in the exploration of this method, a simulation environment was created in software. A room of dimension $100m \times 100m$ was created. Multiple access points were arbitrarily placed throughout the room. A grid partitioned into $4m \times 4m$ blocks was then superimposed over the room, and the position (x and y coordinates) of each node (corner at which the blocks connect) was stored in an array. The Euclidean distance between each node and each of the access points was then calculated and stored.

3.2 Simulated RSSI Values and the Log-Normal Pathloss Model

Received signal strength indicator (RSSI) values were simulated at each node in the grid according to a log-Normal pathloss model. Each node in the grid stores an RSSI vector with the same number of elements as there are access points. For example, if there are a total of four access points, each node in the grid will store an RSSI vector containing four RSSI values.

The dependent variable in the log-Normal pathloss model is the distance between the transmitter and receiver. There are two parameters in the pathloss model that are used to describe the environment in which the signal is propagating through. For this project, the RSSI values were generated according to the model that is proposed for modeling 802.11 2.4GHz signal propagation through office building environments in [3].

$$P_r(d) = P_{r_0} - 10\alpha \log(d) + \epsilon \tag{1}$$

4 RSSI Fingerprinting

4.1 Collection Process

In order to perform our proposed method, it is necessary to record a *fingerprint* of the space in which the tracking will take place. We first partition the space to be fingerprinted into a grid. The fingerprint is a database of typical

RSSI values for each of the access points at each node within the grid. We hope to characterize the propagation behavior of WLAN signals within the space, and store this information in a database that can be used as a sort of table lookup.

4.2 Averaging

To provide a more robust fingerprint database, we measure the RSSI vector at each node multiple times and average these values. The idea behind this averaging is to eliminate some of the RF nonidealities that may be present in a single measurement. This averaging process will be refered to as *offline averaging*.

4.3 Convolution Smoothing

Although the averaging removes some of the noise within our fingerprint database, we can obtain a smoother RSSI contour by additional processing techniques. In the simulated environment, performing convolution with a uniform kernel over each of the RSSI matrices that are produced by specific access points has greatly improved the smoothness of the RSSI contour. Other kernels were not explored, but this may be a parameter to vary when real data becomes available.

5 Observation Process

5.1 RSSI Vector Measurements

During online tracking, the mobile device must scan the network and store the RSSI that is associated with each of the detected MAC addresses. This information will be hashed and organized in a way such that access points with adequate coverage in an area can be used in the table lookup.

5.2 Norm Minimization and Table Lookup

After the online RSSI vector has been measured, a table lookup will be performed in order to get a *most likely* observation. This lookup will not be based on a specific key, but rather by minimizing the norm of the observed RSSI vector against each of the RSSI vectors in the fingerprint database. The returned value from the table lookup will be the position vector associated with this minimized norm. The goal is not to get an exact estimate of the

current position, but rather to use this *most likely* position as our observation vector, $\mathbf{Z_n}$.

6 Particle Filtering

In this section I will discuss the basic theory of the particle filter, as well as the practical implementation and interpretation in the problem of indoor tracking. It is not my intention to provide a complete mathematical description of the techniques described, but rather to provide an interpretation to the problem at hand. Please refer to the references for a complete treatment on the subject of sequential Monte Carlo importance sampling.

6.1 Motivation

The particle filter is a robust Bayesian technique that can be used in state-space estimation problems where the environment is highly nonlinear and non-Gaussian. The particle filter can be used to provide an approximation to the posterior distribution of the hidden state vector. This posterior distribution is estimated recursively in time. An important concept to understand is that the hidden state is estimated indirectly by observing a related observable quantity. In this application, the hidden state is the position vector (x and y coordinates), and the observable quantity is the RSSI vector. The question that our particle filter provides an approximate answer to is:

Given our current observation and history of previous observations, where is our most likely current position?

The particle filter generates a ***cloud*** of ***particles***. Each particle represents a specific position vector, and each particle is assigned a unique ***weight***. The weights represent the probability density associated with each position vector.

6.2 Theory and Application

6.2.1 System Dynamics

To apply the particle filter, it is important that we first establish a basic understanding of the dynamics within the system that we are tracking. In the case of human motion, we can devise a ***motion model*** that is commonly used in tracking systems. We will use $\mathbf{X_n}$ to represent the two-dimensional position vector, $\mathbf{I_2}$ to represent the $2x2$ identity matrix, and $\mathbf{U_n}$ to represent

the perturbation at time n. The motion model that was used is a first order autoregressive function with a random vector term $\mathbf{U_n}$.

$$\mathbf{X_{n+1}} = \mathbf{I_2}\mathbf{X_n} + \mathbf{U_n} \tag{2}$$

We define $\mathbf{U_n}$ as:

$$\mathbf{U_n} = \begin{bmatrix} \psi_n \cos(\theta_n) \\ \psi_n \sin(\theta_n) \end{bmatrix} \tag{3}$$

where ψ_n and θ_n are two correlated Gaussian random variables. ψ_n represents the distance of the step taken at time n, which is modeled as a Gaussian. θ_n represents the angle of the step taken at time n, which is also modeled as a Gaussian.

6.2.2 Propagation

Once the motion model has been defined, we are ready to begin the particle filtering. In the first iteration of tracking, a cloud of uniformly distributed particles (two-dimensional position vectors) are generated. The uniformity in the initial step comes from the fact that we do not have any prior information, therefore we impose a uniform prior. After the first iteration, we begin to have a viable estimate of our hidden state vector $\mathbf{X_n}$. As per our understanding of the dynamics of the system, we feed our particles through the motion model to obtain an updated cloud of particles on each of the subsequent iterations.

6.2.3 Weight Update

After the observable quantity has been measured and we form our unfiltered estimate $\mathbf{Z_n}$, we calculate the weights for each particle.

$$w_n^i \propto w_{n-1}^i \frac{p(\mathbf{Z_n}|\mathbf{X_n^i})p(\mathbf{X_n^i}|\mathbf{X_{n-1}^i})}{q(\mathbf{X_n}|\mathbf{X_{n-1}^i}, \mathbf{Z_n})} \tag{4}$$

We select an importance distribution $q(\mathbf{X_n}|\mathbf{X_{n-1}^i}, \mathbf{Z_n})$ to be equal to the prior $p(\mathbf{X_n}|\mathbf{X_{n-1}^i})$ [1], which simplifies our weight update equation to be:

$$w_n^i \propto w_{n-1}^i p(\mathbf{Z_n}|\mathbf{X_n^i}) \tag{5}$$

We are now left with evaluating the likelihood of observing $\mathbf{Z_n}$. We apply a logarithmic transformation to obtain the kernel of the Gaussian, and shift to avoid numerical complications. These likelihood values scale to the weights associated with each particle. Mathematically, we have:

$$\log(w_n^i) = \log(\frac{1}{\sqrt{(2\pi)^2|\Sigma|}} e^{\frac{-1}{2}(\mathbf{Z_n}-\mathbf{X_n^i})^{\mathbf{T}}\Sigma^{-1}(\mathbf{Z_n}-\mathbf{X_n^i})}) \tag{6}$$

$$\log \tilde{w}_n^i = \log w_n^i - \max(\log w_n) \tag{7}$$

$$\overset{*}{w}_n^i = \frac{\tilde{w}_n^i}{\Sigma_i \tilde{w}_n^i} \tag{8}$$

6.2.4 Prediction

Once we have evaluated the weight that is associated with each particle, we can form an approximation to the posterior density of the hidden state vector. There are multiple approaches that can be taken to obtain the current estimate. The first approximation to the posterior can be defined by:

$$p(\mathbf{X_n}|\mathbf{Z_{1:n}}) \approx \Sigma_{i=1}^{N}(\overset{*}{w}_n^i \delta(\mathbf{X_n} - \mathbf{X_n^i})) \tag{9}$$

Another approach leads to the windowed MAP estimate, where the window size was chosen to be 3. In this approach, the positions of the 3 highest weighted particles are averaged and used at the current estimate. The windowed MAP estimate consistently performed the best in the simulated environment, but the margin between the other estimates was small.

6.2.5 Resampling

The resampling function is a crucial step in the operation of the particle filter. The purpose of the resampling step is to remove particles with negligable weights, meanwhile keeping the number of particles constant. This means that particles with larger weights will continue to be chosen and propagated through the motion model. It is common for the resampling function to return multiple instances of the same particle. The addition of the random vector in the motion model will correct the multiple instances of the same particle. The resampling function that was used in this project was translated from the psuedo-code provided in [1]. The steps within the resampling algorithm are as follows:

1. Construct the sample cumulative distribution function of the M weights, $F(\cdot)$
2. Draw a uniform random variable on the interval $[0, \frac{1}{M}], u_1$
3. For each j within M particles, create a threshold, $u_j = u_1 + \frac{j-1}{M}$
4. For each j, find the index i in the cumulative distribution function where $F(i) > u_j$

5. For each j, assign the new particle $\mathbf{X^j_{new}} = \mathbf{X^i_{old}}$
6. Distribute an equal weighting to each of the M new partciles, $\mathbf{X_{new}}$
7. Return the lists of new particles, and their associated weights

7 Additional Time-Series Filtering

Although the motion model is an autoregressive process, it is of low order and hence there are high frequency components that result in jumps in the estimated trajectory. To counter this, an additional autoregressive-moving average (ARMA(3, 1)) filter was applied to the particle filter estimates to provide a more stable estimate of the hidden state vector. A complete investigation of the best ARMA filter was not performed, but simple experimentation resulted in a reduced error in simulation. As a next step, these coefficients should be optimized on real data.

8 Future Considerations

8.1 Incorporating Additional Information

To improve our estimate we should consider the incorporation of readily available sensor data (accelerometer, gyroscope, etc.) into our model. All modern mobile computers contain these sensors, so we would not be using all of our resources if this information was simply ignored.
It would also be interesting to consider how we can incorporate this RSSI based technique with existing indoor locationing technologies, such as SNAP and Bluetooth low energy.

8.2 Floor Plans

If the floor plans for the area that is of interest to us is available, we may be able to use this to our benefit. Points of interest within the area (such as office locations, restrooms, etc.) may be marked, and we can assign probabilities to the different paths throughout the room. The probability models may be collected from the devices themselves during online useage (i.e. heat maps of frequented paths).

8.3 Dynamic Fingerprinting

It is important that we take into consideration that the RSSI fingerprint is dynamic and may change throughout different times of the day, as well as more significant changes due to the access points themselves (i.e. replacing access points with different models, changing the transmit power, or relocating them). One way to handle these situations is to have multiple fingerprint databases that correspond to different time periods throughout the day (i.e. in an office setting, during working hours there will be more signal attenuation due to the number of people in the area). Another possibility is to construct a device that can navigate through the room to measure RSSI values and look for significant changes against the fingerprint database.

References

[1] M. Sanjeev Arulampalam, Simon Maskell, Neil Gordon, Tim Clapp. *A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking*. IEEE Transactions on Signal Processing, Vol. 50, No. 2, February 2002.

[2] Arnuad Doucet, Simon Godsill, Christophe Andrieu. *On Sequential Monte Carlo Sampling Methods for Bayesian Filtering*. Statistics and Computing (2000) 10, 197-208

[3] Daniel B. Faria *Modeling Signal Attenuation in IEEE 802.11 Wireless LANs - Vol. 1*

[4] Arnuad Doucet, Nando de Freitas, Kevin Murphy, Stuart Russell. *Rao-Blackwellised Particle Filtering for Dynamic Bayesian Networks*

[5] Zhu Nan, Zhao Hongbo, Feng Wenquan, Wang Zulin. *A Novel Particle Filter Approach for Indoor Positioning by Fusing WiFi and Intertial Sensors*