PROCEDURALLY GENERATED ENVIRONMENTS FOR SIMULATING RSSI-LOCALIZATION APPLICATIONS

Sam Shue
Department of Electrical and Computer Engineering
University of North Carolina at Charlotte
9201 University City Blvd
Charlotte, NC, USA
slshue@uncc.edu

James M. Conrad
Department of Electrical and Computer Engineering
University of North Carolina at Charlotte
9201 University City Blvd
Charlotte, NC, USA
jmconrad@uncc.edu

ABSTRACT

Received signal strength indicator (RSSI) is often used in wireless localization applications as it attenuates as the signal propagates through the environment. Signal strength attenuation models allow distance to be estimated between transceivers and utilized in positioning techniques, such as trilateration and simultaneous localization and mapping (SLAM) algorithms. To evaluate these methods, it is costly and time consuming to construct hardware to physically test each implementation of an algorithm. However, simulating RSSI can prove difficult due the many environmental factors that impact attenuation, such as multipath interference. Simulating multipath effects require detailed information about the operating environment's properties (geometry, materials), and can be computationally expensive. This work describes a simulation method which procedurally generates RSSI values at given distances for wireless nodes utilizing collected data from a given environment type and a Markov chain. To demonstrate the effectiveness of this method, a range-only SLAM algorithm is simulated utilizing this environment.

Keywords: Received Signal Strength Indictor (RSSI), Distance Estimation, Location Tracking, Localization, Wireless Sensor Networks, Range-Only Simultaneous Localization and Mapping (SLAM), Markov chain.

1 INTRODUCTION

Localization is a common application of wireless sensor networks and robotics. Many sensors are used for estimating distances between devices, such as ultrasonic, LIDAR, and stereo vision (Lanzisera, Lin, and Pister 2006; Ansari, Riihijarvi, and Hahonen 2007; Honkavirta et al. 2009; Awad, Frunzke, and Dressler 2007). However, utilizing existing hardware already present on the device keeps hardware expenses down and saves space on footprint size. Wireless sensor networks can use the physical effects of wireless data transmission to estimate distance, such as signal attenuation, angle-of-arrival, or time difference of arrival of transmission. Angle-of-arrival and time difference of arrival techniques require specialized hardware, directional antennas and highly accurate clocks, respectively, and are not practical on commonly available wireless devices. Therefore, approaches which utilize signal strength decay are attractive to researchers. As the signal propagates through the air, it attenuates as a function of the distance, allowing mathematical models to exist which describe this process. This attenuation is expressed as the received signal strength indicator (RSSI) which is in negative decibel milliwatts (-dBm). However, RSSI distance estimation is difficult due the multipath effect. This occurs when a transmitting signal finds multiple paths to the receiver, adding together to form an unexpected amplification or attenuation of the signal, depending on the phases

SpringSim-CNS 2017, April 23-26, Virginia Beach, VA, USA

of the multiple paths. This effect is illustrated in Figure 1. This is especially apparent in indoor environments, where the multiple paths have similar signal strengths to the line-of-sight path. Models exist for outdoor and long-distance multipath effects, where the number of reflected paths are minimal, and the signal strength of the additional paths aren't as prominent as the line-of-sight path.

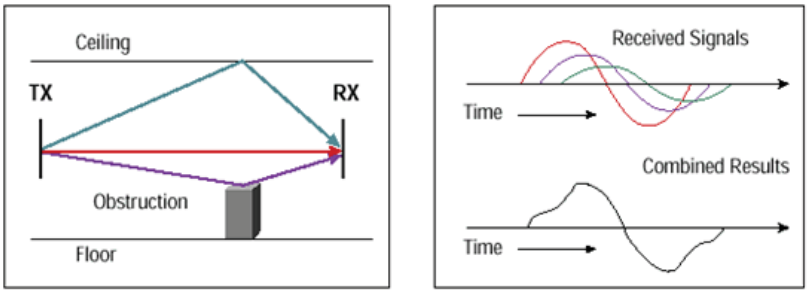


Figure 1: The effects of multipath on a received signal (CISCO 2016).

Despite the multipath effect, RSSI ranging techniques can still prove effective in indoor environments. One of the more popular techniques is RSSI fingerprinting, where a signal strength map of an indoor environment is created beforehand, and as a mobile node moves throughout the map, the changes in RSSI are matched to determine the location. However, this technique requires extensive, tedious mapping prior to use, and is highly sensitive to changes in the environment which would affect the radiation pattern or the wireless transmitters. RSSI ranging can still be used in trilateration and simultaneous localization and mapping algorithms, however, some pre-filtering techniques are often required.

2 RADIO-PROPAGATION MODELS

Multiple models exist to describe the signal attenuation over distance. Presented here are two of the most common models.

2.1 Free Space Propagation Model

The free space propagation model describes the loss of signal-strength over distance in an open environment with line of sight between transmitters and antennae with omnidirectional radiation patterns. The received signal power, P_r, is expressed at distance, d, in meters as:

$$P_r(d) = \frac{P_t G_t G_\gamma \lambda^2}{(4\pi d)^2 L}$$

Where P_t is the transmitter's signal power, G_t and G_r are the transmitter and the receiver antennae gain in dB, respectively, L is the system loss, and λ is the wavelength in meters. The system loss doesn't relate to the transmission itself, but rather the environment in which it is operating. Usually a value of 1 can be applied to L (Xu et al. 2010; Elango, Mathivanan, and Pankaj 2011).

2.2 Log-Distance Path Loss Model

The log-distance path loss model is a general propagation model. It can be used in both indoor and outdoor environments. The log-distance path loss model provides a logarithmic attenuation model which has several parameters that can be tuned to make it fit nearly any environment (Xu et al. 2010). The RSSI (in dBm) for this model is expressed as:

$$RSSI = 10n \log_{10} d + A$$

Where n is the path-loss exponent, d is the transmission distance in meters, and A is the reference value, which is the RSSI at 1 meter away from the transmitter. This equation can be rearranged to be expressed in as distance for a given RSSI value:

$$d = 10^{\frac{RSSI-A}{10n}}$$

The path loss exponent, n, can be calculated for each environment by recording RSSI values at known distances and reverse solving for n. Typical values for n can be observed in Table I (Elango, Mathivanan, and Pankaj 2011, Mehra and Singh 2011).

Table 1. Sample path-loss exponent values for various environments.

Environment	Path-Loss Exponent (n)
Free Space	2.0
Urban Area Cellular Radio	2.7~3.5
In-Building LOS	1.6 ~ 1.8
Shadowed Urban Area Cellular Radio	3~5

The log-distance path loss model is the most versatile, as it can be configured for each environment and uses a reference value rather than requiring the transmission power and gain for each transmitter and receiver.

3 THE SIMULATION ENVIRONMENT

Implementing an entire sensor network to test localization algorithms can be costly and time consuming. Therefore, it is practical to simulate a testing environment prior to implementation. However, simulation of RSSI can be unreliable due to over-idealized signal strength data, which doesn't accurately portray the multipath effect, or it can computationally expensive from ray-tracing techniques and geometrically complex environments. Here we present a simulation technique which accurately portrays the RSSI signal behavior within an environment of interest, given the application is only concerned with signal behavior and not the geometry of the environment. This simulation technique involves training a discrete Markov chain to generate RSSI data over distance, utilizing actual collected RSSI data in the environment of interest. This generated data contains RSSI over distance vectors, which are then assigned to a specified number of rays emitting from the node's location. This forms a discretized radiation pattern for each node which serves as a lookup table given a certain distance and angle from the node. All figures and simulations presented in this work are generated using MATLAB 2016B.

3.1 Collection of RSSI Data

To generate RSSI data from the Markov chain, real RSSI-distance data must be collected. For the examples shown in this work, data was collected within the Energy Production and Infrastructure Center (EPIC) building on the campus of the University of North Carolina at Charlotte. Data was collected in various environment types, including large classrooms, hallways, and laboratories. An outdoor data set was also collected for a set with minimal multipath interference. This data was collected using XBee 802.15.4 radios and Digi 2.4 GHz Omnidirectional Dipole antennas with a 2.1 dBi Gain. The XBee radios were configured with boost mode disabled and power level set to 2 (1 dBm Gain). Each XBee module was interfaced with an Arduino Uno; one module being set to transmit, and the other configured to return the RSSI value of the received packet. Each node was placed on a 1 meter tall stalk and moved away from the transmitter at 0.25

meter increments up to 10 meters. At each increment, 5 readings where taken at 90 degree increments for a full rotation. Despite the antennas being omni-directional the radiation pattern is not perfectly symmetrical and the rotations attempt at capturing those variations. This data was recorded and placed within a spreadsheet. The simulation would then later use these spreadsheets to train the Markov chain. Figure 2 depicts an example of the collected RSSI-distance data.

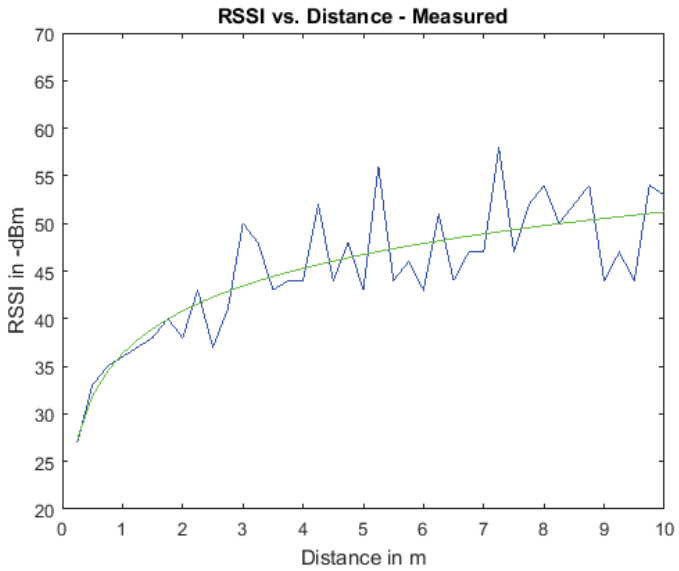


Figure 2: Collected RSSI data over distance in a hallway of UNC Charlotte's EPIC Building. The blue line represents the average RSSI value collected at each distance increment, and the green line is the calibrated log-distance model for this data.

3.2 The Markov Chain

Markov chains are procedural algorithms which determine the next state of a random variable from the current state. Each state has a probability associated with it to transition to a new state on the next iteration. (Bishop 2006)

For this simulation, each combination of discrete RSSI value and distance value is modelled as a state. Each state has some probability of transitioning to a state 1 distance increment higher than the current state (0.25 m in this example, because each RSSI measurement was taken at 0.25 meter increments. The probability of transitioning to any state with a distance lower than the current state or higher than 1 increment of the current state is 0. The probability of transitioning to the next state one distance increment higher is determined by how often a new RSSI value is observed after the RSSI value of the current state. Following the Markov chain from distance = 0.25 and a randomly selected initial RSSI value from the set of RSSI values at 0.25 meters, the chain generates an RSSI-distance data set that contains the multipath characteristics of the environment from which it was collected. Figure 3 graphically depicts the Markov chain sequence. This data is utilized as an RSSI look-up table for a certain distance from the transmitter. The limitation of this method is the maximum distance that can be accurately simulated is dependent on the maximum distance from the source data collected in the actual environment. Figure 4 depicts an RSSI vs. distance graph generated by a trained Markov chain.

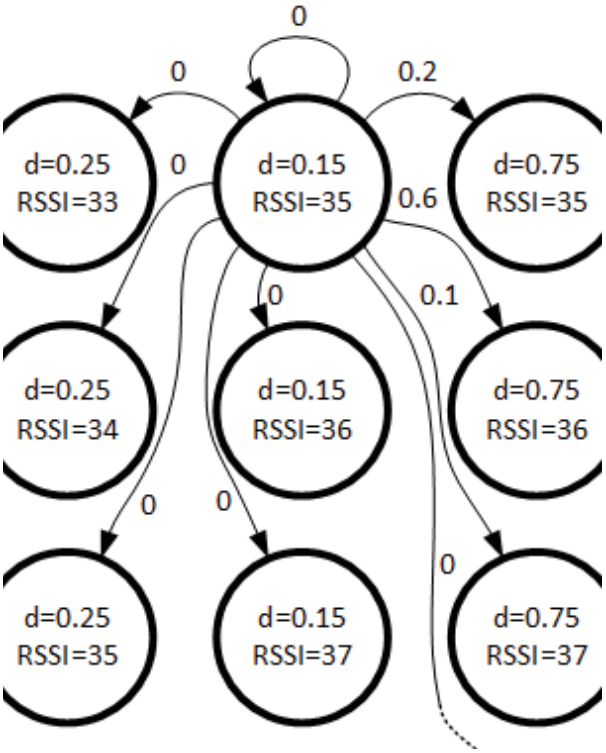


Figure 3: Example of RSSI-Distance Markov chain at state, "distance = 0.15, RSSI = 35". Each state is defined by a discrete RSSI/distance combination. The arrows and numbers indicate the probability of transitioning into the next state. As illustrated here, only states one distance increment above the current state have some probability of transitioning to. The size of the distance increment is determined by the resolution of the collected RSSI over distance data. The probability of transitioning is calculated based on how often an RSSI value was observed on the next distance increment during the data collection process.

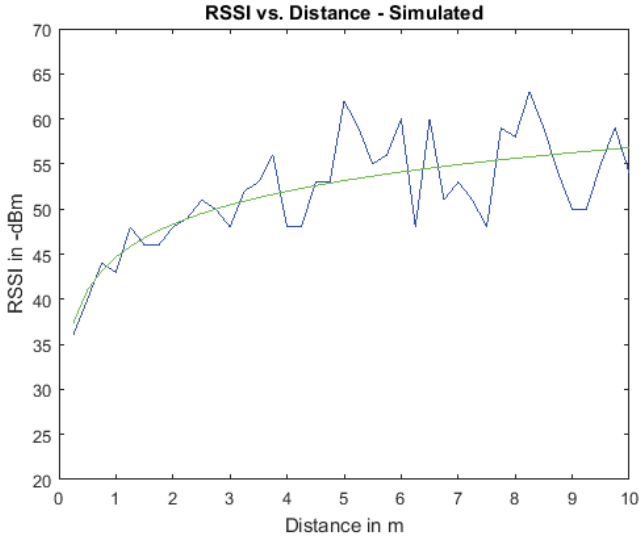


Figure 4: Simulated RSSI data over distance produced by a Markov chain trained on collected data from hallways. Notice the multipath similarities to the measured data collected in Figure 2. The blue line represents the average RSSI value collected at each distance increment, and the green line is the calibrated log-distance model for this data.

3.3 Determining RSSI Values

Within the simulation environment each node is assigned a certain number of rays where each ray is composed of a Markov-generated RSSI-distance vector. Given a receiver node is a certain position, distance and orientation, away from the node the RSSI value is determined based on distance from the node and the distance to the nearest rays using a bilinear interpolation method. Figure 5 depicts the ray configuration for a node generated with 8 rays.

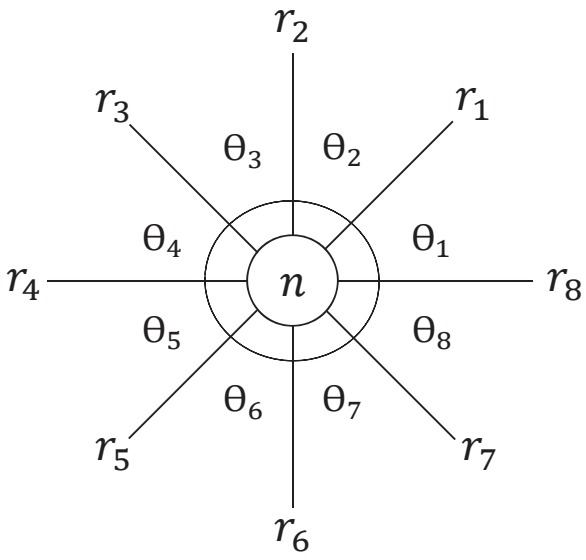


Figure 5: A Markov generated node with 8 rays. Each ray, r_n, corresponds to a generated RSSI-distance vector at an associated angle, θ.

The following equation describes the RSSI-value assignment given a non-integer valued (x,y) position relative to the node:

$$RSSI(\theta_1, d) = \frac{d_2 - d}{d_2 - d_1} RSSI_m(\theta_1, d_1) + \frac{d_1 - d}{d_2 - d_1} RSSI_m(\theta_1, d_2)$$

$$RSSI(\theta_2, d) = \frac{d_2 - d}{d_2 - d_1} RSSI_m(\theta_2, d_1) + \frac{d_1 - d}{d_2 - d_1} RSSI_m(\theta_2, d_1)$$

$$RSSI(\theta, d) = \frac{\theta_2 - \theta}{\theta_2 - \theta_1} RSSI(\theta_1, d) + \frac{\theta_1 - \theta}{\theta_2 - \theta_1} RSSI(\theta_2, d)$$

Where θ and d represent the angle and distance of the receiver node with respect to the transmitter node, and the sub-indexes of these variables represent the nearest discrete indexes generated by the Markov chain. The $RSSI_m$ function serves as a lookup table function for specified indexes. Depending on the number of rays specified, each ray has an angle value, θ, associated with it. Figure 6 graphically illustrates the bilinear interpolation process (MathWorks 2017).

Using this equation to determine the RSSI value, Figure 7 displays radiation patterns for nodes generated with Markov chains trained on various environment types.

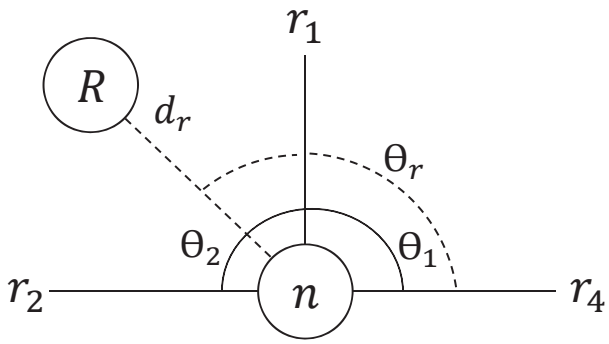
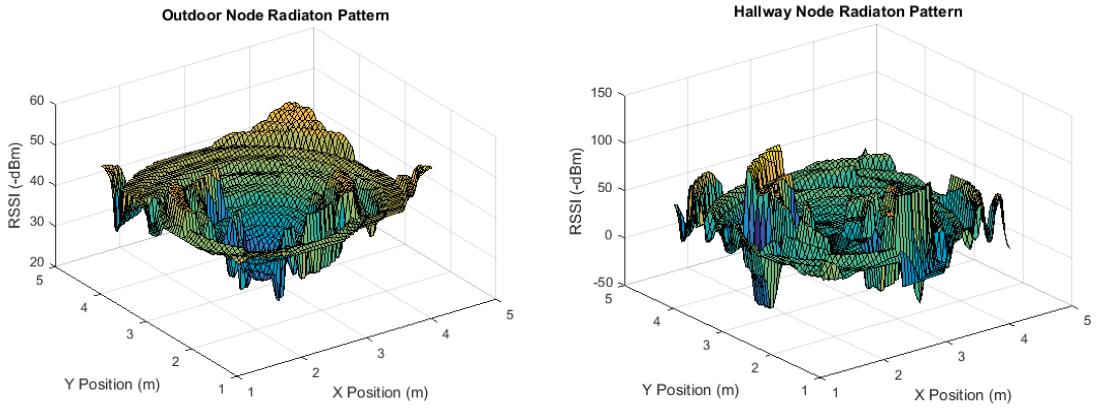


Figure 6: An Illustration of the bilinear interpolation process. The position of the node fuses two RSSI values from rays r_1 and r_2, which are the weighted average between two RSSI readings from each ray based on distance. The two RSSI reading are averaged again based on weights determined from distances to each angle, θ_1 and θ_2.



(A) (B)

Figure 7: Simulated radiation patterns from Markov chains trained on various environment data. Both figures have a node placed in the center of the graph, at (4,4) and the intensities along the z axis indicate the signal attenuation. The graph in (A) was trained on data collected in an outdoor environment, where the multipath interference was minimal, while the graph in (B) was trained on hallway data, which has very intense multipath fading effects.

4 SIMULATION OF RANGE-ONLY RSSI SLAM

To test the simulation method a range-only RSSI SLAM algorithm was implemented. This algorithm tracks the position and orientation of a mobile robot equipped with a wireless transceiver and the position of wireless nodes which act as landmarks for the correcting odometry errors from the robot by determining distance to each node. The algorithm uses an extended Kalman filter at its core to combine the odometry reading with distance estimations from the wireless nodes.

4.1 The Simulation Configuration

Before utilizing the simulated nodes an operating environment must be defined. Here, a robot is defined at a given position and given a pre-defined path to follow. Wireless nodes are randomly scattered within the environment at a user-specified amount. At each iteration of the simulation the robot takes a linear and rotational motion defined by the path vector. The difference between each motion is perturbed by a Gaussian noise and collected as odometric sensor data. RSSI values are then collected from each node, and passed to a pre-filtering phase to help reduce the multipath effect.

4.2 RSSI Pre-filtering

To reduce the multipath effect a pre-filtering method similar to the work described by Menegatti *et al*. was implemented. This method utilizes the linear displacement of the robot's odometry to generate an estimated RSSI value and combines that with the measured RSSI value to a particular node. If a robot moves some distance between RSSI measurements, the RSSI value cannot undergo an extreme change relative to the change in distance. Given the motion vector of the robot is relative the frame of the world or map, it is not known relative the node's position, however; the extreme cases can be considered. If a robot were to move 0.5 meters between readings the maximum expected reading would be the RSSI value associated with a 0.5 meter increase in distance and the minimum would be the RSSI value associated with a 0.5 meter decrease. Given a newly measured RSSI value and a displacement vector from odometery the filtered RSSI value is as follows:

$$RSSI_e = 10 * n * \log(d \pm u) + A$$

$$RSSI_p = \frac{RSSI_e + RSSI_m}{2}$$

$$d_f = 10^{\frac{RSSI_p - A}{10n}}$$

Where $RSSI_e$ is the expected RSSI at a given distance, d, plus or minus the displacement, u. If the change in RSSI is positive, the change is added, and if the change is negative, the change is subtracted. $RSSI_p$ is the predicted RSSI value found by averaging the expected RSSI with the measured RSSI. The filtered distance, d_f, is found by re-arranging the log-distance model and applying the predicted RSSI value. The filtered distance value behaves as a low-pass filter for RSSI values (Menegatti et al. 2009).

4.3 Range-Only SLAM Algorithm

The range-only SLAM algorithm utilizes estimated distances to landmarks to correct accrued odometric values. The algorithm presented here uses an extended Kalman filter to estimate the state of the system, that includes the pose of the robot and the coordinates of the wireless nodes, which serve as the landmarks for the system. The state vector, x, contains the following:

$$x = [x_r \quad y_r \quad \theta_r \quad x_1 \quad y_1 \quad \dots \quad x_n \quad y_n]^T$$

Where x_r, y_r, and θ_r denote the pose of the robot, and x_1, y_1 through x_n, y_n denote the positions of the nodes, up to n number of nodes (Menegatti et al. 2009).

The odometry data from the robot's motion is added to the filter as a control vector, u:

$$u = [\Delta D \quad \Delta\theta]^T$$

Where ΔD is the linear displacement of the robot, and $\Delta\theta$ is the rotational displacement of the robot between iterations.

The state transition function is a non-linear function, f(x,u), which contains the system dynamics to predict the next state based on control vector input:

$$f(x_{k-1}, u) = \begin{bmatrix} x_r + \Delta D * cos(\theta_r + \Delta\theta) \\ y_r + \Delta D * sin(\theta_r + \Delta\theta) \\ \theta_r + \Delta\theta \\ x_1 \\ y_1 \\ \vdots \\ y_n \\ x_n \end{bmatrix} = x_k$$

Where *k - 1* indicates the value of the last filter iteration, and *k* indicates the current state prediction. In order for the extended Kalman filter to update the covariance matrix, a linearized model of the system must be acquired. Hence, the Jacobian of *f* is taken, yielding the following:

$$F = \begin{bmatrix} 1 & 0 & -\Delta D * sin(\theta_r) & 0 & 0 \\ 0 & 1 & \Delta D * cos(\theta_r) & 0 & 0 \\ 0 & 0 & 1 & \vdots & \vdots \\ \vdots & \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

The observation function, h, is another non-linear function that transforms the state variables into the form of the measurement vector, z. The measurement vector contains the estimated distance value from the pre-filtering stage for the node/landmark currently being observed. Therefore, the h function must take acquire the distance between the robot's position and observed node.

$$h(x_k) = \sqrt{(x_r - x_i)^2 + (y_r - y_i)^2}$$

Where *i* indicates the index of the observed node. For the filter to update the covariance's of the measurement phase the linearized form of h must be obtained as well, giving the following Jacobian:

$$H = \begin{bmatrix} \frac{x_r - x_i}{\sqrt{(x_r - x_i)^2 + (y_r - y_i)^2}} & \frac{y_r - y_i}{\sqrt{(x_r - x_i)^2 + (y_r - y_i)^2}} & 0 \end{bmatrix}$$

These defined functions, Jacobians, and vectors, the extended Kalman filter is detailed in Algorithm 1.

Extended_Kalman_Filter($\boldsymbol{x_{k-1}, P_{k-1}, u_k, z_k}$)

Prediction Phase:

$x_k = f(x_{k-1}, u_k)$

$\boldsymbol{P_k = F_k P_{k-1} F_k^T + Q_k}$

Measurement Phase:

$y_k = z_k - h(x_k, u_k)$

$\boldsymbol{S_k = H_k P_k H_k^T + R_k}$

$\boldsymbol{K_k = P_k H_k^T S_k^{-1}}$

$x_{k+1} = x_k + \boldsymbol{K_k} y_k$

$\boldsymbol{P_{k+1} = (I - K_k H_k) P_k}$

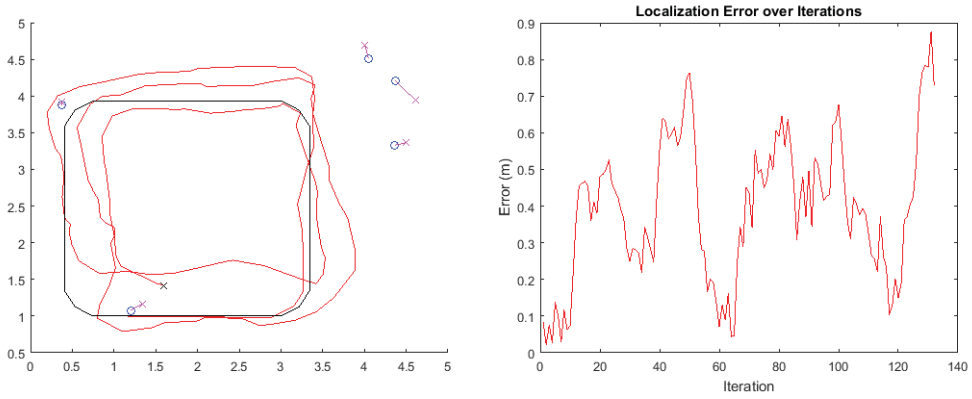
return $x_{k+1}, \boldsymbol{P}_{k+1}$

Algorrithm 1: Extended Kalman Filter. (Thrun, Burgard, and Fox 2005)

4.4 Simulation Results

The range-only SLAM algorithm presented above was implemented within the MATLAB simulation with near-true initial values. As the linearization of the EKF makes the state estimates susceptible to falling into local minima, the algorithm only performs reliably when state variables are initialized near their true values. (Menegatti et al. 2009) The robot traverses a pre-defined rectangular path three times and wireless node/landmarks are randomly placed throughout the environment. Figure 8 shows the results of this algorithm.

The simulation results are similar to what would be expected to be observed employing this type of localization algorithm within a highly reflective indoor environment, in which the nodes' radiation patterns where trained. The average localization error for the robot's position after multiple simulations within a highly reflective environment was ~ 0.75 meters and node locations where ~ 0.5 meters.



(A) (B)

Figure 8: Range-only EKF SLAM simulation using wireless nodes as landmarks. Graph (A) shows the true path of the robot in black, and the estimated path from the filter in red. The robot's final estimated position is indicated by the black 'x'. The blue circles are the randomly assigned node locations, and the magenta 'x's are the estimated positions of the nodes. All units are in meters. Graph (B) shows the error of the estimated robot position throughout its trajectory.

5 CONCLUSION

This work presents a procedurally generated simulation environment which generates RSSI values within a certain environment type based on a trained Markov chain for testing RSSI-based localization algorithms. While realistically simulating multipath effects on a wireless signal requires gratuitous amounts of processing and detailed information about the environment, such as geometry and material properties and their effects on radio waves, the method presented here statistically generates multipath interference on an RSSI signal over distance which well models the interference effects observed in the actual environment. Localization algorithms which do not utilize environment geometry can be tested using this simulation method without the concern of providing over-idealized RSSI data. The range-only SLAM algorithm performs as expected given the technique and environment type.

ACKNOWLEDGMENTS

The authors would like to thank Bryan Hollis, Colin Stewart, and Sam Knittel who helped gather RSSI measurements in various environment used for training the RSSI-distance Markov chain. The authors would also like to thank Jeremy Sabo and Benjamin Rhoades for their help creating the visuals found in Figures 3, 5, and 6.

REFERENCES

Lanzisera, S.; Lin, D.T.; Pister, K.S.J., "RF Time of Flight Ranging for Wireless Sensor Network Localization," in Intelligent Solutions in Embedded Systems, 2006 International Workshop on , pp.1-12, 30-30 June 2006

Ansari, J.; Riihijarvi, J.; Mahonen, P., "Combining Particle Filtering with Cricket System for Indoor Localization and Tracking Services," in Personal, Indoor and Mobile Radio Communications, 2007. PIMRC 2007. IEEE 18th International Symposium on , pp.1-5, 3-7 Sept. 2007

Honkavirta, V.; Perala, T.; Ali-Loytty, S.; Piche, R., "A comparative survey of WLAN location fingerprinting methods," in Positioning, Navigation and Communication, 2009. WPNC 2009. 6th Workshop on , pp.243-251, 19-19 March 2009

Awad, A.; Frunzke, T.; Dressler, F., "Adaptive Distance Estimation and Localization in WSN using RSSI Measures," in Digital System Design Architectures, Methods and Tools, 2007. DSD 2007. 10th Euromicro Conference on , pp.471-478, 29-31 Aug. 2007

Shue, S.;Conrad, J. M.. 2016. Reducing the effect of signal multipath fading in RSSI-distance estimation using Kalman filters. In Proceedings of the 19th Communications & Networking Symposium (CNS '16). Society for Computer Simulation International, San Diego, CA, USA, , Article 5 , 6 pages.

"Omni Antenna vs. Directional Antenna", Available: http://www.cisco.com/c/en/us/support/docs/wireless-mobility/wireless-lan-wlan/82068-omni-vs-direct.html

Xu, J; Liu, W; Lang ,F.; Zhang, Y.; Wang, C., "Distance Measurement Model Based on RSSI in WSN," Wireless Sensor Network, Vol. 2 No. 8, 2010, pp. 606-611.

Elango, S; Mathivanan, N and Pankaj, G. "RSSI Based Indoor Position Monitoring Using WSN in a Home Automation Application." *Microprocessors and Microsystems*. 2011; 11(4): 14-19.

Mehra, R.; Singh, A., "Real time RSSI error reduction in distance estimation using RLS algorithm," in Advance Computing Conference (IACC), 2013 IEEE 3rd International , pp.661-665, 22-23 Feb. 2013

Bishop, C., 2006. Pattern Recognition and Machine Learning (Information Science and Statistics). Springer-Verlag New York, Inc., Secaucus, NJ, USA.

"Nearest Neighbor, Bilinear, and Bicubic Interpolation Methods." Retrieved January 8, 2017. Available: https://www.mathworks.com/help/vision/ug/interpolation-methods.html.

E. Menegatti, A. Zanella, S. Zilli, F. Zorzi and E. Pagello, "Range-only SLAM with a mobile robot and a Wireless Sensor Networks," 2009 IEEE International Conference on Robotics and Automation, Kobe, 2009, pp. 8-14.

Thrun, S; Burgard, W; Fox, D. 2005. Probabilistic Robotics (Intelligent Robotics and Autonomous Agents). The MIT Press.

AUTHOR BIOGRAPHIES

SAM SHUE is PhD student at the University of North Carolina at Charlotte. His research interests involve robotic control, localization and mapping, wireless sensor networks, and signal processing. His email address is slshue@uncc.edu.

JAMES M. CONRAD received his BS degree in computer science from the University of Illinois, Urbana, and his MS and PhD degrees in computer engineering from North Carolina State University. From 1984 to 1990 he was a software engineer with IBM, from 1992 to 1995 he was an Assistant Professor at the University of Arkansas, and from 1997 to 2003 he was a software engineer and project manager with Ericsson and Sony Ericsson. Since 2003 he has been an Associate Professor and Professor with the Electrical and Computer Engineering Department at the University of North Carolina at Charlotte. He is the author of eight books and more than 140 articles in the areas of embedded systems, robotics, parallel processing, and engineering education. His email address is jmconrad@uncc.edu.