# Improving Wireless Simulation Through Noise Modeling

HyungJune Lee
Computer Systems Laboratory
Stanford University
Stanford, CA 94305
abbado@stanford.edu

Alberto Cerpa
School of Engineering
UC Merced
Merced, CA 95344
acerpa@ucmerced.edu

Philip Levis
Computer Systems Laboratory
Stanford University
Stanford, CA 94305
pal@cs.stanford.edu

## Abstract

We propose modeling environmental noise in order to efficiently and accurately simulate wireless packet delivery. We measure noise traces in many different environments and propose three algorithms to simulate noise from these traces. We evaluate applying these algorithms to signal-to-noise curves in comparison to existing simulation approaches used in EmStar, TOSSIM, and ns2. We measure simulation accuracy using the Kantorovich-Wasserstein distance on conditional packet delivery functions. We demonstrate that using a closest-fit pattern matching (CPM) noise model can capture complex temporal dynamics which existing approaches do not, increasing packet simulation fidelity by a factor of 2 for good links, a factor of 1.5 for bad links, and a factor of 5 for intermediate links. As our models are derived from real-world traces, they can be generated for many different environments.

## Categories and Subject Descriptors

I.6 [**Simulation and Modeling**]:

## General Terms

Experimentation

## Keywords

Sensor networks, wireless simulation

## 1. INTRODUCTION

Simulation is a critical part of developing, testing, and evaluating sensornet protocols and systems. Having complete control of the simulated environment allows us to run reproducible experiments, explore parameter spaces, and disambiguate causes of error or undesirable behavior. The inherent difficulty in developing robust sensornet codes has led many tools to focus on system dynamics through real-code simulation [1, 10, 15, 22].

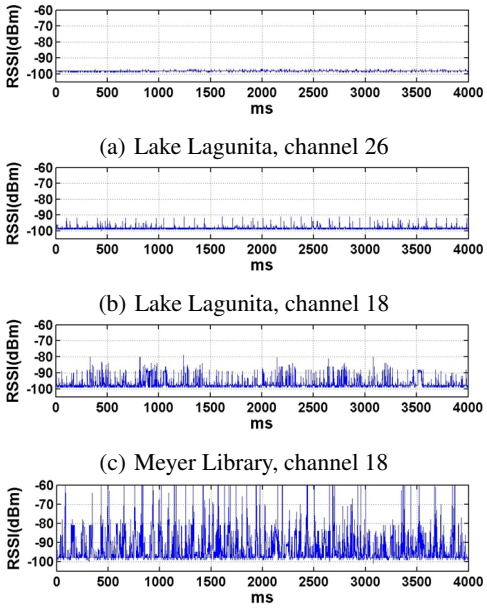
Very accurate system simulation allows users to test code paths. It does not, however, promise a representative execution environment. First and foremost, low-power wireless networks have many complex, rare, and difficult behaviors that protocols must address properly in order to be effective in practice [5, 6, 9, 21, 24]. Early studies noted that packet delivery rates are highly variable over distance [9, 24]. Many existing simulators have used the high-level packet delivery data from these experiments in their network models [10, 15]. This approach allows simulators such as TOSSIM and EmStar to have packet delivery behavior similar to the real world. However, as these simulators simulate loss rather than its causes, they are unable to easily or accurately model novel environments, concurrent transmissions, or variable packet sizes.

Recent investigations into low-cost radio hardware have distinguished how many different factors, such as hardware calibration, interference, and orientation affect packet delivery [20]. In particular, these and other results [16, 21] have verified that packet delivery follows a simple SNR curve. Furthermore, these studies have shown that the RSSI of received packets (the S of the SNR) is often very stable over long periods. Taken together, these observations point at the causes of temporal variations in packet loss and bursty connectivity. Hardware variations cause node pairs to have different SNR curves, but for any given pair the curve is precise. As RSSI is generally stable over short periods, it is reasonable to conclude that the missing piece of the RF simulation puzzle is the environmental noise. With that in mind, in the context of this paper, we term any RF energy generated by sources outside the control of a protocol designer *noise*, while *interference* is RF energy that can be controlled.

Unfortunately, simulating environmental noise is hard. Unlike hardware-based noise, which is typically modeled as additive white Gaussian noise (AWGN), environmental noise is often from packet-based devices. Section 2 shows how packet based noise appears as brief, strong, short-lived noise spikes which can be temporally correlated. To simulate this noise, we gather 1kHz noise traces using current 802.15.4 sensor node platforms and use these traces to generate statistical models of noise using three techniques, presented in Section 3: probabilistic sampling, closest-fit pattern matching (CPM), and a non-Gaussian random process. We simulate radio packet delivery with these noise models using an SNR/PRR curve. Whenever a simulated node receives a packet, it samples a noise reading from its model to determine the SNR and computes the packet delivery probability.

We implemented these three techniques as replacements of the TinyOS 2.0 TOSSIM simulator radio model. Section 4 evaluates how well the algorithms as well as wireless protocol simulators such as EmStar [10], TOSSIM 1.x [15], TOSSIM 2.x [4], and ns2 [2] simulate packet delivery dynamics for good, intermediate, and bad links. To capture temporal packet dynamics, we evaluate simulation accuracy using conditional packet delivery functions (CPDFs), which describe the probability a packet will be delivered successfully after $n$ consecutive failures or successes. We compare CPDFs using the Kantorovich-Wasserstein distance [11]. Our results indicate that existing techniques are sufficient for environments with


*IPSN'07,* April 25-27, 2007, Cambridge, Massachusetts, USA.




(a) Lake Lagunita, channel 26

(b) Lake Lagunita, channel 18

(c) Meyer Library, channel 18

(d) Meyer Library, channel 18, during nearby heavy 802.11 use

**Figure 1: 4 second 1kHz noise traces of 802.15.4 channel 26 and 18 measured at an outdoor park and in a library with dense 802.11b coverage. Noise sources in the 2.4GHz band are discrete but show significant temporal correlation.**

little noise from external transmitters, but for noisy environments CPM significantly outperforms all other approaches.

We have gathered noise traces for a variety of environments, including busy and quiet indoor office environments, outdoor areas with 802.11 connectivity, and outdoor environments with no interfering traffic (the Grand Canyon). Section 6 discusses the implications and limitations of our approaches as well as our planned directions of future work. Our results suggest that an effective route towards accurate wireless simulation is to simply *measure* a diverse set of environments and generate statistical models of them.

## 2. BACKGROUND

Accurately simulating wireless packet delivery is a long-standing challenge in sensornet research. Early studies used a unit-disc model, which defines transmission range as a simple disc of binary connectivity; nodes within a range $r$ successfully receive packets, while those outside $r$ do not. This model, while simple to implement and reason about, has little basis in reality. Experimental studies have shown that connectivity varies tremendously over distance [9, 24] and that many links fall into a "grey region" of intermediate quality.

In response to the observation that connectivity is more complex than what simple disc or RF propagation models (such as those used in ns2 [2]) can express, sensornet simulators have for the most part adopted an empirical approach. Rather than trying to model the underlying causes of RF connectivity, such as interference, noise, and RF propagation, an empirical approach merely recreates packet-level behavior. For example, TOSSIM takes inter-node distances and samples from a packet reception rate (PRR) distribution to determine the connectivity between a pair of nodes [15]. This simple approach can capture a large number of real-world complexities, such as link asymmetries and highly variable spatial connectivity. However, it also makes simplifying assumptions that do not hold in practice. First and foremost, this approach assumes that every link is independent (they are sampled independently from the distance distribution), while real networks tend to have "bad" nodes with poor connectivity. This simplification causes discrepancies between simulation and testbed experiments.

The EmStar system [10] avoids the independence problems of TOSSIM by having one of its radio models using PRR values measured in real-world networks [5]. This has the benefit of capturing effects such as poor receivers. The cost is that it can only simulate networks for which PRR has been measured. The EmStar and TOSSIM approaches assume that packet losses are independent (PRR does not change), but experimental results have shown that PRR varies significantly over time [5, 6].

Recent studies have begun to shed light on the underlying causes of the complex packet delivery behavior of real networks [20]. One important observation from these studies is that for a given node pair, there is a crisp SNR/PRR curve. Effects such as the reception grey region are caused by different pairs having different curves and signal strength variations. A hardware covariance matrix can capture these effects with reasonable accuracy [25].

Experimental studies of current sensornet platforms, such as the micaZ and telosB, have shown that signal strength is stable over short periods of time, but can have longer-term variations due to environmental conditions [16, 21]. However, computing PRR from an SNR curve requires the noise as well as the signal. Historically, the RF community has explored how to simulate signal propagation in tremendous detail [12, 19]. The underlying assumption in all of this work, however, is that the noise encountered is all additive white Gaussian noise (AWGN). If the spectrum is not shared with any other RF sources, then the signal propagation of simulated transmitters and AWGN can describe the entire channel. As sensornets often operate in unlicensed ISM bands, their spectrum is crowded with many conflicting transmitters. 2.4 GHz, the band used by micaz, telos, and imote2 nodes, is particularly crowded, as it is also occupied by 2.4 GHz phones, 802.11b/g, microwave ovens, and Bluetooth, all of which interfere significantly. Without these considerations, SNR-based simulation models are fundamentally limited in accuracy.

The hypothesis of this paper is that coming up with an efficient and effective model of environmental noise will allow a sensornet simulator to accurately model packet delivery using an SNR/PRR curve. We leverage the observations and advances of prior work to achieve this goal. From Zuniga *et al.*'s experimental work [25] we borrow the idea of hardware covariance matrices to govern the SNR curve of a node pair. From EmStar we borrow the idea of measuring real environments to derive a representative model. Once we have derived a per-node noise model, we plug it and the RSSI of a transmitter into a SNR curve to compute packet delivery probability. Simulating noise allows us to capture short-term connectivity variations, such as those caused by a large burst of 802.11 traffic.

The challenge in simulating 2.4GHz noise is that it does not follow a clean and elegant mathematical model. Because much of the interference is 802.11 traffic, it has a highly bimodal behavior: an 802.11 node is either transmitting or not. Instead of a Gaussian process or wave, transmissions are a discrete signal with highly variable temporal characteristics. Figure 1 shows four noise traces from different environments on 802.15.4 channel 18 and 26. Lake Lagunita at Stanford is almost free of 802.11b interferences and other noise sources. On the other hand, Stanford Meyer library has many 802.11b access points and so has severe 802.11b interference.

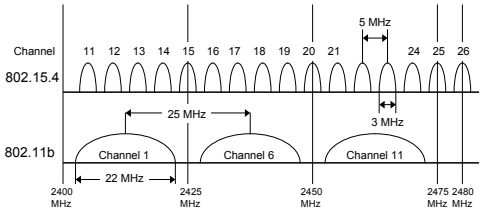



**Figure 2: 802.11b and 802.15.4 spectrum utilization. Channel 18 in 802.15.4 heavily overlaps with 802.11b channels, while channel 26 in 802.15.4 has no overlap with 802.11b spectrum.**

The periodic peak values in the plots are 802.11b beacon packets at a frequency 9.765Hz (0.1024s). The next section describes three approaches to statistically modeling 2.4GHz noise, and Section 4 evaluates how well these approaches reflect real-world behavior in comparison to commonly used simulators.

# 3. NOISE CHARACTERIZATION

This section describes three approaches to statistically characterize noise traces. In this paper we broadly define environmental noise as the RF interference produced by any unsimulated RF sources in the node's spectrum in addition to the thermal agitation of charge carriers in the electronic circuits and devices [14, 17].

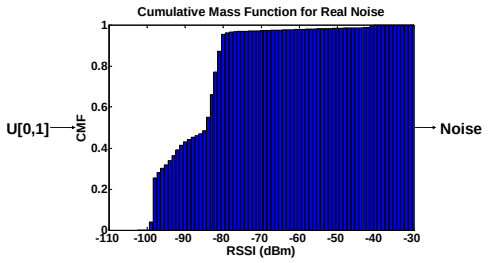
The first approach, *naive sampling*, generates a probability distribution of a noise trace and simply samples from this distribution. Naive sampling is fast and simple, but makes the assumption that noise samples are independent. The second approach, *closest-fit pattern matching* (CPM), computes the conditional probability distribution of noise values given $k$ previous noise readings. It generates a noise value based on the matching series and defaults to the mode when no measured series matches. The third approach uses a non-Gaussian random model with the *correlation distortion method* in order to describe noise as a random process. This can capture temporal dynamics, but is computationally expensive and has difficulty with signals that are highly non-Gaussian.

## 3.1 Measuring Noise

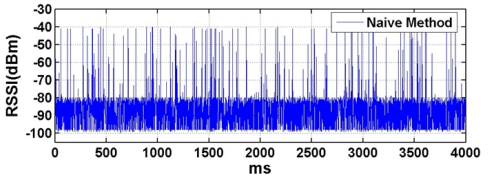
To measure environmental noise, we wrote a TinyOS application that samples RF energy at 1kHz by reading the RSSI register of the CC2420 radio. The register contains the average RSSI over the past 8 symbol periods (125$\mu$s). The application logs this data to flash for a fixed period of time ($3 * 2^{16}$ samples, so $\approx$ 197s). A PC application reads the data off of the mote. We sampled noise on different radio channels in a wide range of environments, including inside WiFi enabled buildings (Meyer Library at Stanford), in outdoor WiFi enabled areas (Lake Lagunita at Stanford), in outdoor quiet areas (Grand Canyon), and during controlled tests (a large HTTP download in Meyer Library).

Figure 1 shows 4 second periods from four gathered noise traces. These traces show three key characteristics of noise in the 2.4GHz band. First, noise tends to have discrete spikes, which are as much as 40dBm above the noise floor. These spikes typically but not always represent transmissions from copresent wireless packet networks. As Figure 2 borrowed from [21] shows, 802.11 shares spectrum with the 802.15.4 radios used in several sensor platforms. Second, many of these spikes are periodic. For example, 802.11b base stations transmit beacons every 0.1024s. Third, noise is temporally correlated: there are periods of activity and periods of quiet.

The rest of this section describes three approaches to modeling




(a) Simulating noise with naive sampling.




(b) Sample noise trace from naive sampling using heavy traffic Meyer trace.

**Figure 3: Simulating noise with naive sampling. By generating a uniformly distributed random variable in [0,1], a noise sample can be derived by filtering with CMF function of measured noise.**

| Bin | 1 | 2 | 3 | ... | 16 |
|---|---|---|---|---|---|
| RSSI(dBm) | $-102 \sim -98$ | $-97 \sim -93$ | $-92 \sim -88$ | ... | $-27 \sim -23$ |

**Table 1: Closest-fit pattern matching further discretizes noise values in order to shrink its state space.**

2.4GHz noise: naive sampling, closest-fit pattern matching, and the correlation distortion method.

## 3.2 Naive Sampling

Because copresent packet networks are discrete event sources, probabilistic sampling is a simple way to model noise. This approach works by computing the distribution of noise values and sampling from the distribution whenever a noise value is needed. This approach has the benefit that generating the model and taking samples from it is very fast.

Assuming that each noise sample is independent, simulating a noise trace can be reduced to generating random variables. Once a cumulative mass function (CMF) of target data is prescribed, the same distribution of simulated data can be achieved by filtering uniformly distributed random numbers as inputs by the inverse CMF in Figure 3. The probability mass function (PMF) of the simulated data is nearly identical to the target data.

While simple and fast, this method neglects crucial information such as time-dependence. Noise has temporal correlation, and making samples independent breaks this correlation. In theory, this means that if real noise has bursts of interference that cause bursts of packet losses, a naive sampling model may not be able to capture this behavior. On the other hand, it may be that this limitation ends up having minimal effects on the final simulation behavior. We therefore consider this approach to be a baseline measurement for noise simulation.

## 3.3 Closest-fit Pattern Matching (CPM)

Unlike naive sampling, which generates independent noise values, closest-fit pattern matching (CPM) uses a probability distribu-

| Noise | Mean | Std | Skewness | Kurtosis |
|---|---|---|---|---|
| Real Noise | $-97.1017$ | 2.9702 | 3.8350 | 23.2346 |
| Simul. Noise | $-97.6699$ | 2.0886 | 5.1972 | 49.3293 |

**Table 2: Statistical characteristics of real noise in a light Meyer trace and noise correspondingly simulated using the correlation distortion method.**

tion of noise values given $k$ previous noise values. One problem CPM faces is an exploding state space: if noise can take ≈60 values ($-100$ to $-40$ dBm), then CPM with a window of $k = 20$ has a state space of $60^{20}$, or $\approx 4 \cdot 10^{35}$. As our traces have only $\approx 2 \cdot 10^5$ samples, very few patterns will be populated. We therefore further discretize the RSSI values, as shown in Table 1.

Each data point in the CPM model is a PDF of the observed noise values given $k$ previous values. To calculate $n_t$, CPM samples from the PDF associated with $n_{t-1}, n_{t-2}, \ldots n_{t-k}$. If there is no PDF associated with this noise series, CPM samples from the most common PDF (the mode). CPM bootstraps from the measured trace: first $k$ noise values are simply the first $k$ samples from the real-world measurements.

In the degenerate case of $k = 0$, CPM is equivalent to naive sampling. There is a tradeoff in how large a $k$ is used. A large $k$ allows CPM to capture longer term periodicities. However, as the state space grows at $O(r^k)$ where $r$ is the number of discretized RSSI readings, but the number of samples does not increase, the probability that any sequence exists goes down exponentially. This is a basic overfitting problem: in the case where $k$ is the number of samples in the trace, then CPM will play back the trace exactly, which does not allow representative simulation.

Our CPM implementation uses a hashtable to store the CPM state space, where the key is a string concatenation of the noise values and the value is the PDF. Depending on the self-correlation of a given trace, the optimal $k$ value varies. For example, if noise values are completely independent, then a $k = 0$ will be best. We found that for the busy Meyer trace, a $k = 20$ provides a good tradeoff between being representative of the noise yet remaining non-deterministic, as determinism could lead to incorrect assumptions when testing protocols. We evaluate the effect of different $k$ values for the busy Meyer trace in the next section.

## 3.4 Correlation Distortion Method

The main cause of interference we observed, 802.11, has a non-Gaussian property as a result of its discrete traffic patterns. The tradeoff $k$ imposes in CPM raises a significant issue: much of the periodic noise spikes (e.g., 802.11 beacons) have very long periods. For CPM to be able to capture these beacons, for example, $k$ should be larger than or equal to 100. This large $k$ (100ms) makes the CPM state space very sparse. There is longer-term correlation in the noise trace, but CPM cannot effectively capture it. Our third approach addresses this limitation by using a non-Gaussian random process, which captures longer-term periodicities.

The core idea of the method is to transform non-Gaussian to Gaussian with the same auto-correlation or spectrum of the target. Expressing the relationship in terms of Hermite polynomials allows us to generate Gaussian random process by using spectral representation method. In the end, with the generated Gaussian process, the original non-Gaussian process can be achieved by using a transformation equation.

More formally, we apply the correlation distortion method [7, 8, 13] to generate a non-Gaussian random process with a prescribed auto-correlation function. We calculate the auto-correlation of the random process from a noise trace using mean-square (MS) ergodicity, assuming that the noise random process follows wide-sense stationarity. A non-Gaussian random process $x(t)$ has a nonlinear relationship with a Gaussian normal random process $u(t)$, i.e. $x(t) = g(u(t))$. In Eq. (1), the auto-correlation of non-Gaussian process in terms of that of Gaussian normal random process can be described as

$$R_{uu}(\tau) = \sum_{k=0}^{\infty} a_k^2 \rho_{xx}^k(\tau) \tag{1}$$

$$where\ a_k = \frac{1}{\sqrt{2\pi k!}} \int_{-\infty}^{\infty} g(\sigma u)\ exp(-\frac{u^2}{2}) H_k(u) du \tag{2}$$

$$H_k(u) = (-1)^k exp(\frac{u^2}{2}) \frac{d^k}{du^k} [exp(-\frac{u^2}{2})]. \tag{3}$$

In the above expressions, $\rho_{xx}$ is the normalized auto-correlation of the non-Gaussian process $x(t)$ and $H_k(u)$ is the $k^{th}$ Hermite polynomial. The Hermite polynomial is a classical orthogonal polynomial basis function. By Eq. 4, the auto-correlation of non-Gaussian process can be transformed into that of a Gaussian process.

$$R_{xx}(\tau) = \alpha^2 [R_{uu}(\tau) + 2\hat{h_3}^2 R_{uu}^2(\tau) + 6\hat{h_4}^2 R_{uu}^3(\tau)] \tag{4}$$

$$\hat{h_3} = \frac{\gamma_3}{4 + 2\sqrt{1 + 1.5\gamma_4}},\ \hat{h_4} = \frac{\sqrt{1 + 1.5\gamma_4} - 1}{18}$$

$$\alpha = \frac{1}{\sqrt{1 + 2\hat{h_3}^2 + 6\hat{h_4}^2}} \tag{5}$$

where $\gamma_3$ is skewness($3^{rd}$ order moment) of the process and $\gamma_4$ is kurtosis($4^{th}$ order moment) of the process.

One limitation in the standard Hermite Model is that $\hat{h_3}$, $\hat{h_4}$, and $\alpha$ parameters have been calculated with the assumption of small deviations from Gaussian. Therefore, for non-Gaussians which deviate significantly, the method is not quite applicable. To reduce this problem, we applied modified Hermite models which were proposed by Tognarelli et al. [23], leading to improvement of performance in non-Gaussian simulation.

The correlation distortion method can generate noise data representative of a low-traffic 802.11b environment. This is because it can capture the long-term periodicities. We compared how well a simulated noise trace follows real noise behavior in terms of power spectral density corresponding to auto-correlation function, first-order PMF, mean ($1^{st}$ moment), standard deviation($2^{nd}$ moment), skewness ($3^{rd}$ moment), and kurtosis ($4^{th}$ moment). The power spectral density of simulated noise matches that of real noise. This means that time-correlated noise information, which could be a critical factor for consecutive packet failures, is successfully exploited. For the first-order PMF, our simulated noise closely follows the RSSI distribution of real noise, but it is not exactly same as the real one. The Jensen-Shannon distance between PMFs of real noise and simulated noise is 0.089. While the naive method in the above section can achieve the perfectly same first-order PMF, it fails to exploit time-correlated information. With a small difference of the first-order PMF, this approach achieves the sameness of auto-correlation between short-term noise data. Table 2 shows the mean, standard deviation, skewness, and kurtosis.

However, heavy-traffic 802.11b environments deviate significantly from Gaussian noise. The correlation distortion method is usually

| Model | Naive Sampling | CPM | Corr.Dist. |
|---|---|---|---|
| Running Time | 6 $\mu$s | 29.8 $\mu$s | 769 $\mu$s |

**Table 3: Mean execution time for each model to generate a noise sample. For CPM, $k = 20$.**

applicable to the environment of mediocre deviations from Gaussian. In Section 4, we compare the correlation distortion method to CPM and naive sampling for low- and heavy-traffic environments.

# 4. EXPERIMENTAL METHODOLOGY

We measure simulation accuracy by comparing conditional packet delivery functions (CPDFs). A conditional packet delivery function describes the probability that a packet will be received successfully given $n$ previous failures or successes. For example, the CPDF of node A to node B, $c_{AB}$, of 5 ($c_{AB}(5)$) is the probability that B will receive a packet from A after 5 consecutive failures, while $c_{AB}(-5)$ is the probability that B will receive a packet after 5 consecutive successes. If packet losses are independent, then the CPDF is for the most part uniform; if packet losses are bursty, then the CPDF is non-uniform.

We compare CPDFs using a rigorous theoretical measure, the Kantorovich-Wasserstein distance [11]. The Kantorovich-Wasserstein distance has been widely used in theoretical statistics and image signal processing applications to show the similarity of probability distributions. To calculate the Kantorovich-Wasserstein (KW) distance as our evaluation metric, we used open-source codes for the Earth Mover's Distance [18], which is equivalent to KW distance. Both quantify how much elements of two distributions would have to be shifted to make the two distributions equal. We do not use the Chi-squared test because CPDF values are not independent, and do not use the Kolmogorov-Smirnov test because CPDFs are not continuous functions.

We use the Kantorovich-Wasserstein distance of CPDFs rather than measuring the noise itself because of the difficulty of comparing noise traces. Because our goal is to generate a representative and reusable model of an environment's noise, rather than simply replay it, simulated noise will inherently differ from the measured noise. We found that comparing mathematical properties of simulated and real noise gave some indications that they might lead to similar packet behavior, but for almost every similarity measure between noise traces it is simple to create a degenerate case that is mathematically similar but behaves completely differently. We therefore measure similarity in terms of the behavior we seek to recreate: packet delivery.

We use the real noise trace as a baseline for measuring the accuracy of different simulation methods. This allows us to control all other variables in an experiment. To generate the baseline CPDF, we use the real noise trace against an SNR curve derived from CC2420 experiments, using a fixed signal strength with a fixed inter-packet interval (15ms). While the signal strength is fixed for each simulation model, it is not fixed across the models, as models assume different sensitivity thresholds or SNR curves. Instead, for each model we choose the signal strength that creates a desired PRR. This way, we can evaluate how good, bad, and intermediate links manifest in each simulation model, given a particular noise environment. This evaluation asks the critical question "What do good, bad, and intermediate links look like to a simulated node?"

For example, the default radio model of TinyOS 2.x's TOSSIM (TOSSIM2) simulator samples noise values from the uniform distribution $[m - r, m + r)$. Given a trace, we compute the mean and

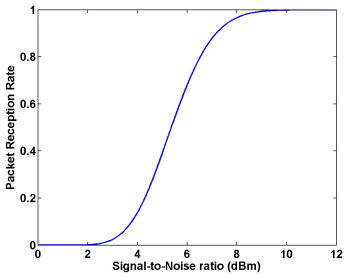



**Figure 4: The CC2420 SNR/PRR curve.**

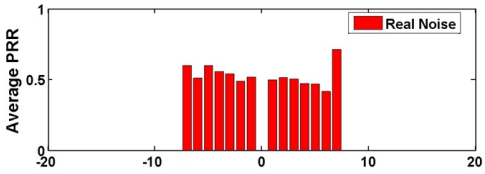



**Figure 5: CPDF of an intermediate link from low-noise Meyer trace of real noise. The X-axis [-20,20] is consecutive packet delivery successes (negative) or failures (positive), and the Y-axis is the PRR. Packet losses are nearly independent.**

variance of the noise values and use them as the mean and range of TOSSIM's RF model (this is not 100% accurate, but since noise does not follow a uniform distribution, we believe it to be a reasonable approximation). We then tune the signal strength until it has the desired PRR (e.g., 51% for an intermediate link, 90% for a good link). We do the same for the baseline: we tune the signal strength so that sampling from the PRR/SNR curve using the real noise trace has the same PRR. We measure PRR over a 195 second trace with an inter-packet interval of 15ms (135,000 packets).

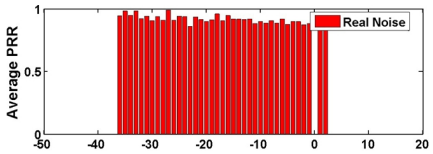
We evaluate the noise models as well as four simulators: EmStar's shadowing model with uniformly distributed random noise, TOSSIM's bit-error model [15], TOSSIM 2.x's gain model [4], and ns2's shadowing model with Gaussian random noise [2].

## 4.1 Noise Sampling

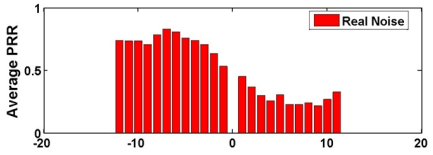
We used our noise sampling TinyOS application to gather data from a wide range of environments and 802.15.4 channels. Figure 1 showed four example traces. We also collected noise traces from the Grand Canyon in Arizona, Gates Hall at Stanford, and in the middle the Great Salt Desert. In the Grand Canyon and Great Salt Desert we observed no 2.4GHz noise besides AWGN; in Gates Hall we observed noise similar to Meyer Library.

## 4.2 Implementation

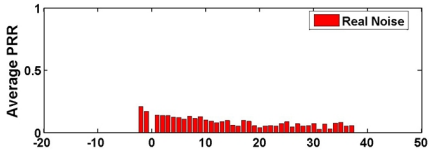
To evaluate the effectiveness of our models, we implemented each one as a replacement for the standard packet simulation engine of the TOSSIM simulator of TinyOS 2.0 [3]. The implementations all use a combined path-loss and shadowing model for signal propagation. In the rest of this section, when we refer to the TOSSIM 2.0 simulation approach, we mean the default one included in the TinyOS distribution. Naive sampling keeps a single probability distribution of noise values. CPM uses a hashtable to efficiently query for a particular distribution to sample from. The correlation distortion method requires 1,024 data points of power spectral density information. Each of our implementations computes noise values at




(a) Good Link (PRR = 90%)




(b) Intermediate Link (PRR = 51%)




(c) Bad Link (PRR = 11%)

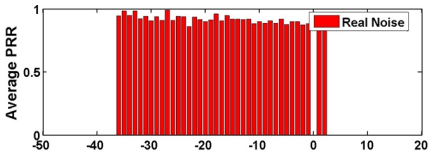
**Figure 6: Conditional packet delivery functions for a good, intermediate, and bad link using the heavy use Meyer library trace. The X-axis is the consecutive packet successes (negative) or failures (positive), and the Y-axis is the PRR. Losses in a good link are independent, losses in a bad link are slightly correlated, and losses in intermediate links are highly correlated.**

1024Hz and uses its combined path-loss and shadowing model to determine packet delivery success or loss using the signal-to-noise ratio curve in Figure 4. In order to separate out the effects noise have on packet delivery from the effects it has on media access, we disabled CSMA at the transmitter in all experiments (its noise is always below the clear channel threshold).

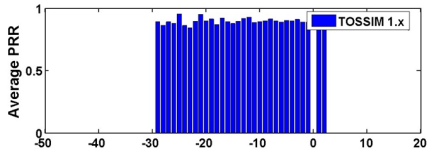
We measured the running time of each of our simulation models, shown in Table 3. We measured these values under Cygwin using `gettimeofday(2)` on a Fujitsu S6000 laptop with a 1.6GHz Pentium M processor. Both the naive sampling and CPM approaches are very fast; we do not expect them to be a significant bottleneck in packet-level simulation. The correlation distortion method, in contrast, introduces significant delays. Because our noise traces are 1kHz samples, we simulate noise at a 1kHz granularity. Currently we simulate noise as a continuous stream (take every sample). For large simulations with bursty traffic patterns this approach is inefficient, We are currently exploring ways to avoid this cost (e.g., after $n \cdot k$ unsampled periods, revert to the mode distribution).

# 5. EVALUATION

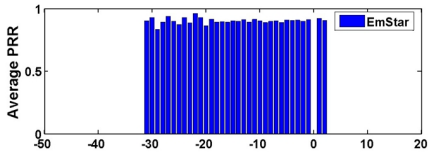
We generated many traces with a variety of signal strengths in order to measure packet delivery behavior for good, bad, and intermediate links. For the most part, low-rate traffic and quiet environments behave in a simple fashion: packet losses due to noise are independent. For example, packet losses from the Grand Canyon trace would be due to AWGN and the SNR curve, both of which cause independent losses rather temporally correlated bursts of loss. In low-rate conflicting traffic environments, clock skew as well as differing intervals between conflicting sources and sensor nodes make periodic losses possible but highly unlikely.



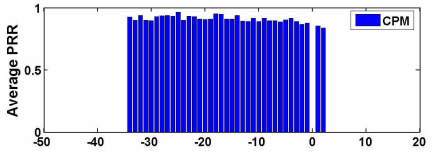

(a) Real Noise



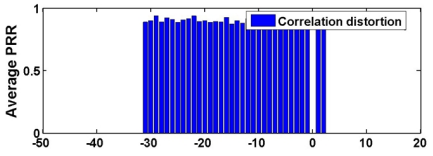

(b) TOSSIM 1.x



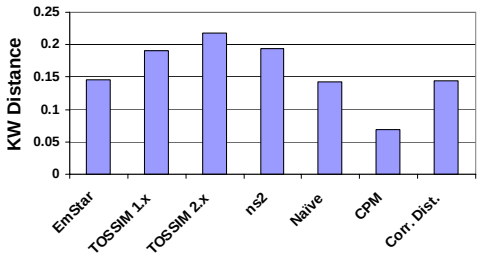

(c) EmStar




(d) CPM

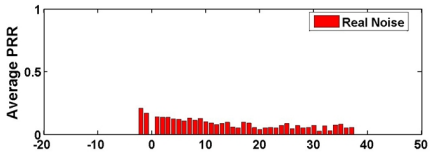



(e) Correlation Distortion

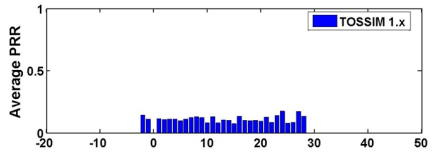



(f) KW distance of all approaches.

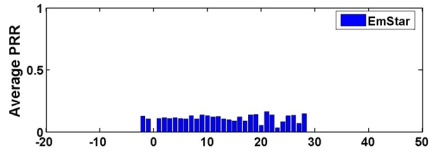
**Figure 7: CPDFs of a good link using TOSSIM 1.x, EmStar, CPM, and Correlation Distortion approaches and the KW distance of all CPDFs from the real noise CPDF. The x-axis [-50,20] is consecutive packet delivery successes (negative) or failures (positive), and the Y-axis is the PRR. These results are from the busy Meyer noise trace.**



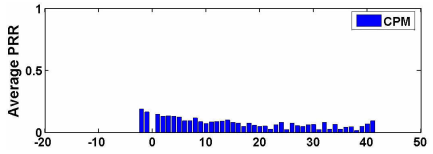

(a) Real Noise



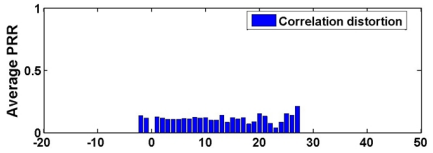

(b) TOSSIM 1.x



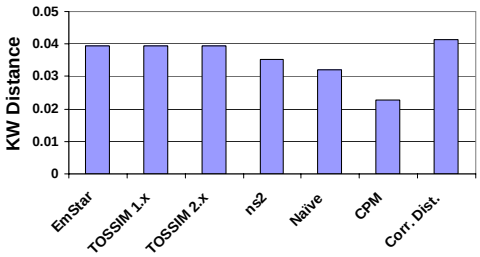

(c) EmStar




(d) CPM

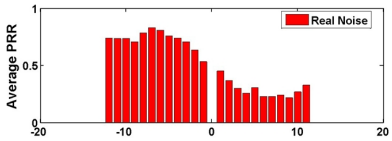



(e) Correlation Distortion

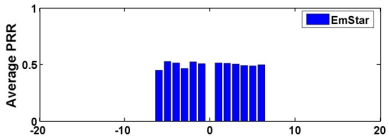



(f) KW distance of all approaches

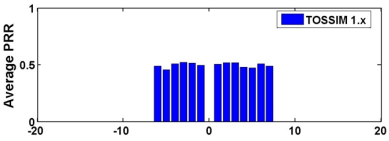
**Figure 8: CPDFs of a bad link using TOSSIM 1.x, EmStar, CPM, and Correlation Distortion approaches as the KW distance of all CPDFs from the real noise CPDF. The x-axis [-20,50] is consecutive packet delivery successes (negative) or failures (positive), and the Y-axis is the PRR. These results are from the busy Meyer noise trace.**



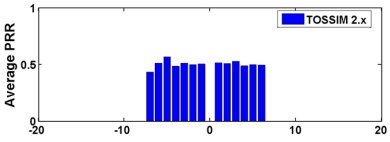

(a) Real Noise



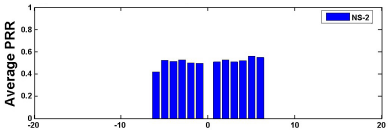

(b) EmStar



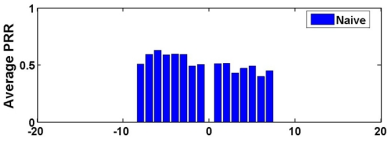

(c) TOSSIM 1.x



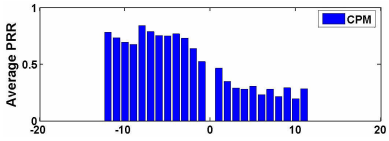

(d) TOSSIM 2.x



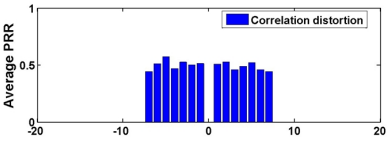

(e) ns2




(f) Naive Sampling




(g) CPM




(h) Correlation Distortion

**Figure 9: CPDFs of an intermediate link from the busy Meyer trace. The X-axis is consecutive packet delivery successes (negative) or failures (positive), and the Y-axis is the PRR.**

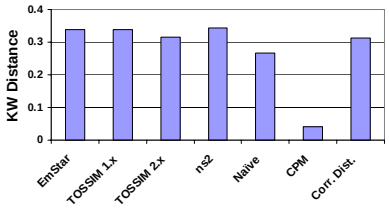



**Figure 10: Kantorovich-Wasserstein distance of the CPDFs of all simulation approaches from the real noise CPDF for the intermediate link in Figure 9.**

Figure 5 shows that for an intermediate link, packet losses are independent with respect to consecutive packet losses. This means that low 802.11b traffic does not lead to bursty packet errors and the temporal effects are negligible in a low-traffic 802.11b environment. Therefore, other simulation methods do not capture the differences between these two types of environments.

Traces taken in a busy 802.11 environment, however, behave differently. Figure 6 shows CPDFs for a good, intermediate, and bad link generated from the busy Meyer trace in Figure 1. Despite the temporal correlation in noise, packet behavior in good and bad links is for the most part independent. In the case of a good link, this is due to the fact that the packet transmission interval (15ms) is not a factor of the large noise spikes, which are governed by TCP and HTTP timing. In the case of a bad link, there are many long bursts of loss caused by the web traffic, creating a long tail over which PRR degrades slightly. For an intermediate link, there is a 3-fold difference in the loss rate after 6 delivery successes and 6 delivery failures.

Figure 7 shows how different simulation approaches capture the dynamics of a good link. We show CPDFs of a subset of existing approaches due to space limitations. Because losses have little correlation, all simulation approaches perform reasonably well. However, at high PRRs, slight variations can significantly change the CPDF. The real noise trace has up to 36 consecutive packet delivery successes, while TOSSIM and EmStar only reach 29 and 32 respectively. In contrast, CPM reaches up to 35. Figure 7(f) shows the Kantorovich-Wasserstein distance of the CPDFs of our three approaches as well as both versions of TOSSIM, ns2, and EmStar. CPM has the lowest KW distance (0.0692) by a factor of 2 over the next best, naive sampling. Every approach had an effectively identical PRR over the 130,000 packets of the 195s interval.

Figure 8 shows how different simulation approaches capture the dynamics of a bad link. We show CPDFs of a subset of the existing approaches due to space limitations. Again, the different simulation approaches all perform reasonably well. However, CPM is able to capture short-term trends well enough to capture PRR degradation as losses increase. Figure 8(f) shows the KW distance of the CPDFs of our three approaches and sensornet simulators. CPM has a KW distance of 0.0227, which is the lowest by a factor of 1.5 over the next best, naive sampling.

As Figure 6 shows, intermediate links are more complex than their good and bad counterparts. Unlike the comparatively flat CPDFs of good and bad links, an intermediate link can have a huge variation in PRR. This behavior supports the common observation that intermediate links are the difficult ones for networking algorithms such as link estimators. They are therefore the most interesting and important to simulate. Given the simplicity of other cases, we focus on intermediate links for the rest of the evaluation.

Figure 9 shows the CPDFs of an intermediate link based on real noise as well as using EmStar, TOSSIM 1.x, TOSSIM 2.x, ns2, naive sampling, closest-fit pattern matching, and the correlation distortion method. For real noise in an intermediate link, the PRR decreases as the number of consecutive packet losses increases. This represents the burstiness of the noise in this class of environment. One packet loss indicates that the node is likely encountering a packet burst, and therefore the PRR decreases for a reasonable period. The PRR values in response to packet successes indicate the probability of encountering a burst of losses. The PRR values given consecutive losses are non-zero because of 802.11b timing; 802.15.4 packets can transmit in between 802.11b/TCP timers.

All simulation models except CPM have PRRs that are independent of consecutive packet delivery failures or successes: the CPDF converges to the average PRR value regardless of error bursts. CPM captures the short-term temporal effects, showing the same behavior as real noise. Figure 10 shows the Kantorovich-Wasserstein distance of each CPDF with the real noise trace. CPM significantly outperforms all other simulation methods, with a KW distance of 0.0402. The second best is naive sampling, with a KW distance of 0.266: and CPM's KW distance is lower by a factor of 5. CPM captures the effects of the real noise much better than any other method.

Figure 5 shows that for an intermediate link, the CPDF does not show the same short-term effects under light 802.11b traffic as it does under heavy 802.11b traffic. The packet losses are independent with respect to the number of consecutive packet losses. Low 802.11b traffic does not lead to bursts of packet errors: the temporal effects are negligible in low-traffic 802.11b environment.

Overall, the correlation distortion method has mediocre performance. The advantage of the correlation distortion method is that it can accurately capture occasional spikes, such as those observed in the good link. Bursts of high noise, however, are too non-Gaussian for it to capture well. Unfortunately, occasional spikes generally appear as independent packet losses to timing differences, and so the expressive power of this approach turns out to have very little benefit in practice: on good links, naive sampling and EmStar perform just as well.

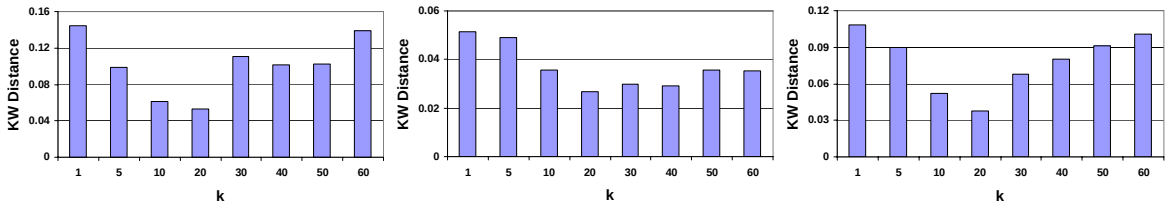
Of the three techniques we proposed, CPM performs best. In our experiments, we set $k = 20$, packets are sent every 15ms and the noise sampling rate is 1kHz. This means that there will be $\approx 15$ samples between two packet transmissions: the noise at one packet transmission is never in the historical window of more than one transmission. CPM can capture bursts that span multiple inter-packet intervals, however, because the values it does consider are still dependent on those outside its window. Consider, for example, if CPM has a historical entry of this form:

$$PDF(8,8,8,8,8,8,8,8,8,8) = \{0.02 : 1, 0.98 : 8\}$$

That is, given 10 consecutive noise readings of 8, 2% of the time CPM will produce a noise value of 1 and 98% of the time CPM will produce a noise value of 8. Once a run of 8s begins, the expectation is that it will last for 50ms (50 samples). In practice, CPM histories are much more complex, but the principle still holds.

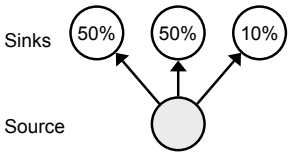
## 5.1 Varying $k$

All of the CPM results in Figures 7-11 use a $k = 20$, which we noted was the best value for the busy Meyer trace. Figure 11 shows how varying $k$ for the good, intermediate and bad link affects the KW distance from the real noise trace. For intermediate and good links, $k$ has a pronounced effect on the accuracy: the KW distance of $k = 20$ is approximately 40% of the KW distance of a very high

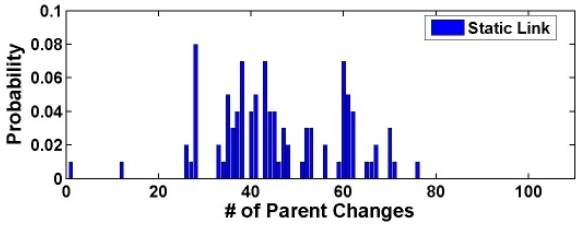



(a) Good link. (b) Bad link. (c) Intermediate link.

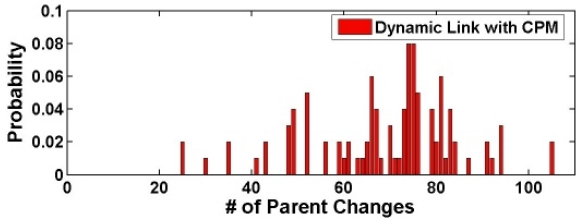
**Figure 11: Effect varying $k$ has on KW distance from CPDF of the Meyer busy noise trace. Setting $k = 20$ produces the best results for all three link classes.**




(a) Topology.




(b) Parent changes with static link qualities (independent packet loss).




(c) Parent changes with CPM (correlated packet loss).

**Figure 12: Number of next hop route selection changes for a four-node topology using CPM and static link loss rates. The percentages on the sinks are the PRRs: there are two intermediate links and one bad link. The bursty losses of CPM affect link estimators and higher-level protocol behavior.**

or low $k$. Additionally, in good and intermediate links, $k = 60$ as roughly the same accuracy as $k = 1$.

For a bad link, $k = 20$ also has the highest accuracy, but the accuracy does not degrade nearly as significantly with an increasing $k$ as it does with the good and intermediate link. The fact that $k = 20$ is the most accurate for the busy Meyer trace does not mean it is the right value for all traces. One area of our future work is to determine the optimal $k$ values for a wide range of environments in order to gain insight on its relationship to the traffic patterns of packet-based interference sources.

## 5.2 Effect on higher-level protocols

Finally, we evaluated how correlated packet losses affect higher-level protocols. In simulation, we set up a simple 4-node topology, shown in Figure 12(a) where the source node transmits to one of three sink nodes. The source has intermediate links to two of the sinks and a bad link to the third. We ran the standard TinyOS 2.0 tree collection layer such that the three sinks were all base stations. The collection layer's link estimator therefore decides which of the three sinks to transmit to. The TinyOS 2 collection layer uses data as well as control traffic to estimate link quality, such that it reacts to bursts of losses.

In each trial, the source sent 100,000 packets as quickly as it could. In terms of the TinyOS code, the application called `send()` in the `sendDone()` handler. We counted how many times the link estimator caused the routing layer to change the next hop node. We ran 100 trials each for two configurations: CPM links derived from the busy Meyer trace and static links where packet losses were independent (as is the case in all other simulation methods). As with the CPDF experiments, noise spikes at the transmitter were always below the clear channel assessment threshold in order to remove MAC effects.

Figure 12 shows the results. On average, a network simulated with CPM has 50% more parent changes (69) than one with static links (46). Additionally, the minimum number of parent changes observed in the CPM case is much higher than in the static case. This shows that the bursty losses of CPM can affect the behavior of higher-level protocols and therefore higher-level simulation results.

# 6. DISCUSSION AND CONCLUSION

This paper takes a step forward in simulating packet delivery by modeling difficult noise signatures from measurements. Rather than depend on a simplified and abstract view of an environment, the models strive to recreate the behavior of a real network. This allows us to simulate a particular network, rather than a fictional one. However, modeling noise as we have presented here has three simplifying assumptions; relaxing each assumption is in and of itself a complete research topic which we plan to explore in the future.

First, by modeling each node's noise traces independently, these models ignore the fact that noise is spatially dependent. If node A hears a noise spike, nearby node B will hear it as well. In one formulation, this means that node A's noise value not only depends on its prior noise values but also the noise values of its nearby neighbors. Capturing these dependencies requires information on where the noise sources are. Another formulation is infer noise sources from correlated measurements and simulate those sources.

Second, while packet-based noise changes are abrupt and therefore contribute to short-term changes in SNR and correlated losses, there are also longer-term changes due to gradual RSSI trends [16, 21]. Concurrently simulating both phenomena – brief noise spikes and long-term RSSI swings – would allow simulation to accurately capture both long-term and short-term dynamics. Furthermore, CPM only handles short-term noise bursts; characterizing longer-term noise trends (busy and quiet periods) would allow longer-running simulations that address another level of dynamism.

Finally, all of our results are based on a single (albeit dominant) low-power radio technology, and we have not observed all forms of 2.4GHz interference. Microwave ovens and analog 2.4GHz devices, for example, produce relatively long (seconds-minutes) periods of high interference, while Bluetooth's frequency hopping undoubtedly has complex and interesting dynamics. Evaluating our approaches in other ISM bands (e.g., the 433 and 915 MHz CC1000 radio on the mica2 platform) would better establish whether or not they are general or particular to the crowded 2.4GHz band.

Our experimental results demonstrate that using an SNR curve with a closest-fit pattern matching noise model can significantly increase wireless packet delivery simulation accuracy. Furthermore, we can easily generate CPM models from real noise traces, allowing tools to effectively represent real-world environments in simulation. This shifts the focus of simulation from hypothetical network configurations to capturing real-world behavior based on real-world data. This allows us to quantify simulation accuracy, allowing us to avoid the pitfall of simulation results which do not reflect the real systems we are trying to improve.

## Acknowledgements

This work was supported by generous gifts from the Intel Corporation and DoCoMo Capital, a fellowship from the Samsung Lee Kun Hee Scholarship Foundation, the National Science Foundation under grant #0615308 ("CSR-EHS"), a Stanford Terman Fellowship, the California Institute for Energy and Environment (CIEE) under grant #CR0601A, and the Center for Information Technology Research in the Interest of Society (CITIRS). We would like to thank Eddie Kohler and our shepherd, Bhaskar Krishnamachari, for their help in improving this paper.

## 7. REFERENCES

[1] Sensor network emulator/simulator/debugger. http://www.cshcn.umd.edu/research/atemu/.

[2] The Network Simulator - ns-2. http://www.isi.edu/nsnam/ns/.

[3] TinyOS 2.0. http://www.tinyos.net/tinyos-2.x/.

[4] TOSSIM 2.x. http://www.tinyos.net/tinyos-2.x/.

[5] A. Cerpa, N. Busek, and D. Estrin. Scale: A tool for simple connectivity assessment in lossy environments. Technical Report 0021, Sept. 2003.

[6] A. Cerpa, J. L. Wong, M. Potkonjak, and D. Estrin. Temporal properties of low power wireless links: Modeling and implications on multi-hop routing. In *Proceedings of the Sixth ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC'05)*, 2005.

[7] D. Conner and J. Hammond. Modeling of stochastic system inputs having prescribed distribution and covariance functions. In *Applied Mathematical Modeling*, volume 3, 1979.

[8] R. Deutsch. *Nonlinear Transformations of Random Processes*. Prentice-Hall, 1962.

[9] D. Ganesan, B. Krishnamachari, A. Woo, D. Culler, D. Estrin, and S. Wicker. An empirical study of epidemic algorithms in large scale multihop wireless networks. UCLA Computer Science Technical Report UCLA/CSD-TR 02-0013, 2002.

[10] L. Girod, T. Stathopoulos, N. Ramanathan, J. Elson, D. Estrin, E. Osterweil, and T. Schoellhammer. A system for simulation, emulation, and deployment of heterogeneous sensor networks. In *Proceedings of the 2nd international conference on Embedded networked sensor systems (SenSys)*, pages 201–213, New York, NY, USA, 2004. ACM Press.

[11] C. Givens and R. Shortt. A class of wasserstein metrics for probability distributions. In *Michigan Math. J.*, volume 31, pages 231–240, 1884.

[12] H. Hashemi. The Indoor Radio Propagation Channel. *Proceedings of the IEEE.*, 81(7), July 1993.

[13] G. Johnson. Constructions of particular random process. In *Proceedings of the IEEE*, volume 82, pages 270–285, 1994.

[14] J. Johnson. Thermal agitation of electricity in conductors. *Physics Review*, 32(97), 1928.

[15] P. Levis, N. Lee, M. Welsh, and D. Culler. TOSSIM: Simulating large wireless sensor networks of tinyos motes. In *Proceedings of the First ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2003.

[16] S. Lin, T. He, J. Zhang, G. Zhou, L. Gu, and J. A. Stankovic. Atpc: Adaptive transmission power control for wireless sensor networks. 2006.

[17] H. Nyquist. Thermal agitation of electric charge in conductors. *Physics Review*, 32(110), 1928.

[18] Y. Rubner, C. Tomasi, and L. J. Guibas. A metric for distributions with applications to image databases. In *Proceedings of the 1998 IEEE International Conference on Computer Vision*, pages 59–66, 1998.

[19] S. Y. Seidel and T. S. Rappaport. 914 MHz path loss prediction models for indoor wireless communications in multifloored buildings. *IEEE Transactions on Antennas and Propagation.*, 40(2), Feb 1992.

[20] D. Son, B. Krishnamachari, and J. Heidemann. Experimental study of concurrent transmission in wireless sensor networks. In *Proceedings of the Fourth ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2006.

[21] K. Srinivasan, P. Dutta, A. Tavakoli, and P. Levis. Understanding the causes of packet delivery success and failure in dense wireless sensor networks. In *Technical report SING-06-00*, Stanford, CA, 2006.

[22] B. L. Titzer, D. K. Lee, and J. Palsberg. Avrora: scalable sensor network simulation with precise timing. In *IPSN '05: Proceedings of the 4th international symposium on Information processing in sensor networks*, page 67, Piscataway, NJ, USA, 2005. IEEE Press.

[23] M. Tognarelli, J. Zhao, and A. Kareem. Equivalent statistical cubicization: A frequency domain approach for nonlinearities in both system and forcing function. In *Journal of Engineering Mechanics, ASCE*, volume 123, 1997.

[24] J. Zhao and R. Govindan. Understanding packet delivery performance in dense wireless sensor networks. In *Proceedings of the First International Conference on Embedded Network Sensor Systems*, 2003.

[25] M. Zuniga and B. Krishnamachari. Analyzing the transitional region in low power wireless links. In *First IEEE International Conference on Sensor and Ad hoc Communications and Networks (SECON)*, 2004.