Poster Abstract: Emuli - Model Driven Sensor Stimuli for Experimentation

Najla Alam, Thomas Clouser, Richie Thomas and Mikhail Nesterenko*
Department of Computer Science
Kent State University
Kent, Ohio, 44242 USA
{nalam,tclouser,rthomas,mikhail}@cs.kent.edu

ABSTRACT

We describe Emuli — a method of replacing sensor data with a network-wide model of stimuli events. Sensor readings are generated on demand from the modeling data stored at each device. This approach allows for both repeatable and variable experimentation with a network of physical devices for existing and planned sensing modalities. We illustrate the approach with (i) a light sensor and (ii) a hypothetical range sensor used in a tracking application.

Categories and Subject Descriptors: C.4 Performance of Systems: Measurement Techniques

General Terms: Measurement, Performance, Experimentation

Keywords: Sensor Stimuli Modeling

1. INTRODUCTION

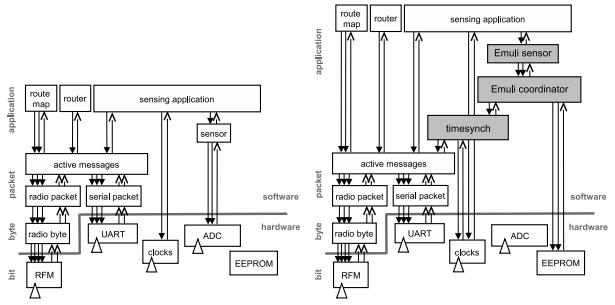
Repeatable experimentation is critical when evaluating low-level components or high-level applications for wireless sensor networks. This is confounded by the very nature of wireless sensor networks, which are highly dependent on their environment to provide external stimuli. Without the means to fully control the environment it becomes difficult to draw strong conclusions from experimental results.

To achieve repeatability, Emuli provides control over the sensor networks' perception of the environment. Through modeling, the stimuli of interest can be represented compactly at each device in the network. By using a model-based instead of a trace-based approach [2, 3]; there are no timing-dependent anomalies or restrictions, inconvenience or peril of collecting trace recordings (consider a biotoxin sensor), nor even the need for the sensing modality to exist. Emuli replaces a standard ADC component (sensor) of TinyOS [1] with a component that utilizes node specific representations of the modeled stimuli to generate sensor readings on demand (see Figure 1). That is, other than a rewiring, the application is unaware the readings are not being generated by the hardware ADC.

We illustrate the use of Emuli by modeling a light sensor and a *hypothetical* range sensor.

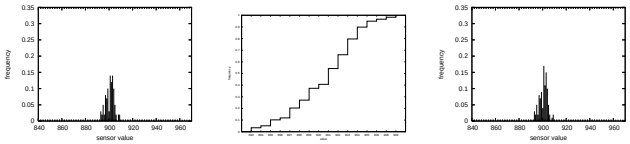
2. LIGHT SENSOR EMULATION

We demonstrate Emuli's statistical modeling capabilities by collecting light sensor readings, compiling the cumulative distribution function (CDF) of the readings and storing the resulting function on a mote in a tabular form. When the application requests a reading, Emuli generates a psuedo-random number using the RandomMLCG component and returns the associated value stored in the CDF (see Figure 2).



(a) original application (b) app. with Emuli

Figure 1: Applying Emuli to an example sensor application.



(a) actual (b) CDF (c) Emuli

Figure 2: Statistically modeled light sensor.

3. RANGE SENSOR EMULATION

The *hypothetical* range sensor determines the distance from the mote to the target within its range. We arbitrarily fix the sensor range to 32 meters. For the results shown, the modeled target moves in a zigzag pattern with speed of 3 meters per second across a surface. We compute the environment model for 5 motes. The modeled track of the target and mote positions are shown in Figure 3.

To generate range sensor readings, each mote stores the information about segment(s) of the target track that are within the range of its sensor (c.f. Figure 4). These segments are computed offline from the simple physical model and loaded onto each device. To coordinate these distributed events, we use FTSP [4] to time synchronize the motes. To optimize the calculations and storage, a segment is represented by two points: the closest to the mote and

*This research is supported in part by NSF Career Award CNS-0347485.

SenSys'08, November 5–7, 2008, Raleigh, North Carolina, USA.
ACM 978-1-59593-990-6/08/11.

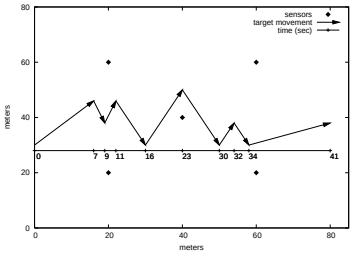


Figure 3: Target track and mote placement.

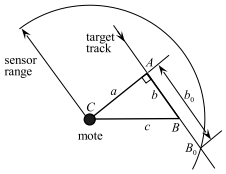


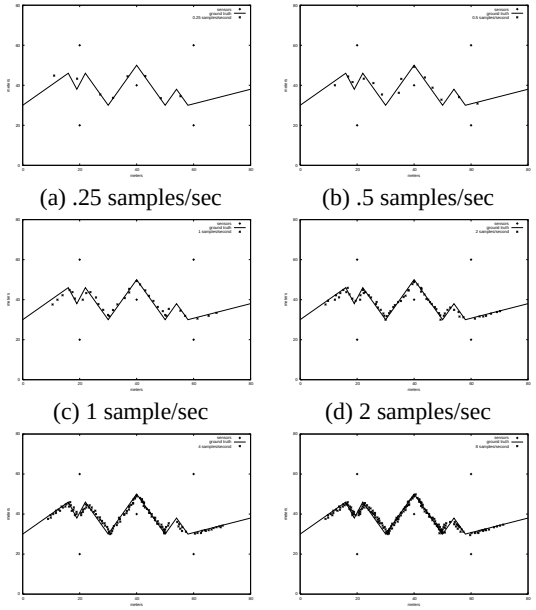
Figure 4: Target distance computation with Emuli

the furthest sensed (i.e. the intersection of the track and the sensor range). Note that the track of the target through the sensing field of individual mote may be represented by multiple segments. This is especially so if the target changes direction within the sensing range.

For the track segment $|A, B_0|$, the mote stores the following data: times t_a and t_{b0} when the target is at the endpoints of the segment, target speed s, and distances $a^2 = |C, A|^2$ and b_0. Note that if $t_a < t_{b0}$ then the target moves from A to B_0. Assume that this is so. Times are stored as integers, distance and speed – as floating point numbers. To compute distance c of the target at time t, the Emuli component first determines whether t is within the time interval of this segment. If it is, Emuli computes the distance $b = (t - t_a)/s$. The actual distance c to the target is computed as $c = \sqrt{a^2 + b^2}$. Emuli also handles special cases for the target passing exactly over the mote or just touching the sensing range.

To demonstrate the operation of range sensing simulation with Emuli, we implement a simple trilateration application. The trilateration requires target distance measurements from three motes. We compute intersection points of the circles whose radius are these measurements. We select two arbitrary pairs of intersection points and draw lines through them. The target location was at the intersection of these two lines. In our triangulation application, motes report their timestamped target distance measurement to the base station. Offline, for each distance measurements, we select two other closest in time measurements and compute the target location.

To showcase Emuli, we vary the sampling rate of our application while using the same model. The results are shown in Figure 5. The precise track emulated by Emuli is known. This allows us to compare the results computed by tracking application to the modeled "ground truth". In essence this comparison evaluates the performance of the Emuli-modeled range sensor. The data analysis is shown in Figure 6. Figure 6a shows the difference in distance returned by range sensor of Emuli and the actual distance of the emulated target from the mote at the time the measurement is taken. This difference stays roughly the same across sampling rates. The error can be attributed to time synchronization error and floating point rounding errors. Figure 6b shows the time differences between the three distance measurements used for trilateration. Recall that our trilateration algorithm selects three measurements at



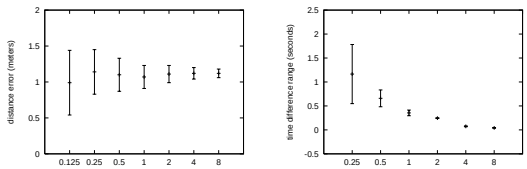
(a) .25 samples/sec (b) .5 samples/sec

(c) 1 sample/sec (d) 2 samples/sec

(e) 4 sample/sec (f) 8 samples/sec

Figure 5: Analysis of tracking measurements.

separate motes that are the closest in time. As the sampling rate increases, the time between measurements at separate motes decreases.



(a) distance differences (b) time differences

Figure 6: Analysis of tracking measurements.

4. TOWARDS COMPREHENSIVE SENSOR EMULATION

Our preliminary results indicate that Emuli's model-based approach for sensor emulation is feasible. Emuli sensor modeling can be enhanced with relative ease to represent more realistic sensors: for example, by combining the deterministic and probabilistic components. Emuli can be of particular service to the community in the form of a library of models for common modalities that wireless sensor network designers can experiment with.

5. REFERENCES

[1] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister. System architecture directions for networked sensors. *SIGPLAN Not.*, 35(11):93–104, 2000.

[2] D. Jia, B. Krogh, and C. Wong. TOSHILT: Middleware for hardware-in-the-loop testing of wireless sensor networks, 2005.

[3] L. Luo, T. He, G. Zhou, L. Gu, T. F. Abdelzaher, and J. A. Stankovic. Achieving repeatability of asynchronous events in wireless sensor networks with envirolog. In *INFOCOM*. IEEE, 2006.

[4] M. Maróti, B. Kusy, G. Simon, and Ákos Lédeczi. The flooding time synchronization protocol. In *SenSys*, pages 39–49, New York, NY, USA, 2004. ACM.