# SENSOR NETWORKS FOR SITUATION MANAGEMENT: A BIOMIMETIC MODEL

Kennie H. Jones, Kenneth N. Lodding and Stephan Olariu, Larry Wilson and Chunsheng Xin
National Aeronautics and Space Administration, Hampton, VA
Old Dominion University, Norfolk, VA
Norfolk State University, Norfolk, VA

## ABSTRACT

*Promises of ubiquitous control of the physical environment by massively-deployed wireless sensor networks open avenues for new applications in support of situation management. Recent sensor network research has concentrated on developing techniques for performing relatively simple tasks with minimal energy expense, assuming some form of centralized control. Unfortunately, centralized control is not conducive to situation management as it allows single points of failure and does not scale to massive size networks. We propose a new way of looking at massively-deployed sensor networks, motivated by lessons learned from the way biological ecosystems are organized. We demonstrate that in our model fully distributed data aggregation and integration can be performed in a scalable fashion where individual motes operate based on local information, making local decisions that are aggregated across the network to achieve globally-meaningful effects. This exemplifies the robust, fault-tolerant infrastructure required for successful situation management systems.*

## INTRODUCTION

Optimal situation management requires intelligent decisions based on fresh and timely information. Furthermore, many applications require high fidelity information that can only be acquired *in situ*. No new technology shows more promise for meeting these demands than sensor networks. Indeed, the small size and low cost of individual sensors make these networks ideally suited for massive and unobtrusive deployment, a most desired prerequisite of many situation management applications.

The recent flurry of research in sensor networks may be credited to the DARPA-sponsored *SmartDust* program whose goal was to make machines with self-powered sensing, computing and communication capabilities so small and inexpensive that they could be released into the environment in massive numbers [1]. These devices are called *motes* and serve as nodes in a sensor network [2,3]. As the motes are severely energy-constrained, they cannot transmit over long distances, restricting interaction to their immediate neighborhood.

In the past few years we have witnessed attempts at deploying small-scale sensor networks in support of a growing array of applications ranging from smart kindergarten [4,5,6], to smart learning environments [7], to habitat monitoring [8,9], to environment monitoring [10,11,12], to greenhouse and vineyard experiments [13,14], to forest fire detection [10,13], and to helping the elderly and the disabled [7]. These prototypes provide solid evidence of the usefulness of sensor networks and suggest the future will be populated by pervasive sensor networks that will redefine the way we live and work [15,16]. It is expected that in the near future, in addition to the examples above, a myriad of other applications including battlefield command and control, disaster management and emergency response, will involve sensor networks as a key mission-critical component

Current sensor networks are for the most part modeled after conventional networks under centralized control and involve a small number of motes. It is, therefore, not clear that they provide a credible approximation of the massive deployment envisioned by the proponents of sensor networks [15,17,18]. Rather than adapting conventional techniques of centralized computer control, new techniques dependent on local cooperation among network nodes will lead to self-sustaining communities of machines with emergent behavior that autonomously operate and adapt to changes in the environment. This evolution so parallels the development of life on Earth that living systems are likely to provide realistic models for sensor network design.

The main contribution of this work is to propose to look at sensor networks in a novel way, motivated by our belief that in order to scale to massive deployment, sensor networks can benefit from lessons learned from the way biological ecosystems are organized. Indeed in the presence of a massive deployment, sensor networks must behave as a community of organisms, where individual motes operate asynchronously and autonomously in parallel. We focus on fundamental characteristics of future sensor networks that are not demonstrated by current implementations, yet are imperative for optimal use in situation management. More specifically, we demonstrate that in such a model, fully distributed data aggregation and integration can be performed in a scalable fashion in massively deployed sensor networks, where individual motes operate based on local information, making local decisions that are aggregated across the network to achieve globally-meaningful effects.

It is not our claim that the examples presented here directly apply to any problem of situation management, but rather that the application of sensor networks to situation management requirements may benefit from our alternate approach. Particularly in adversarial scenarios, centralized control of in situ networks, with its complex infrastructure and single points of failure, may be quickly defeated. Our decentralized approach is much more robust and fault tolerant and would, therefore, be much more difficult to disable.

## EXAMPLE OF CURRENT SENSOR NETWORKS

An examination of current implementations reveals both successes and limitations to the promises of sensor networks. In 2002, a sensor network was implemented on Great Duck Island, Maine [19]. The initial application was to monitor the microclimates of nesting burrows, and, by disseminating the data worldwide, to enable researchers anywhere to non-intrusively monitor sensitive wildlife habitats. The sensor motes were placed in the habitat and formed a multi-hop network to pass messages back to a base station. The data was eventually passed by satellite to servers in Berkeley, CA, where it was distributed via the Internet to any interested viewer. The sensors periodically measured environmental factors and relayed the measurements to the base station. The largest deployment had 190 nodes up to 1000 feet from the nearest base station.

The interested reader can examine other implementations [18,20,21]. What was novel about these approaches is the small size of the sensors and their wireless networking allowing inexpensive and unobtrusive installation directly into the environment. However, these demonstrations used between 6 and 800 motes, thus, they do not approach the high fidelity information architecture advertised by proponents of sensor networks. Will the techniques used scale to massive numbers? In these designs, behavior is predetermined, its results collected, and otherwise managed by a central authority.

## AN ECOLOGICAL MODEL

We think of motes as organisms within a community. At birth (i.e., at deployment time) the motes are endowed with *genetic material*, containing, among others, an initial state and rules by which they interact with the environment. The state and the rules may change as the motes interact with the environment, reflecting their dynamic adaptation to conditions in their neighborhood. Additionally, the motes may remember and record their interaction with the environment by storing information in their limited on-board memory. Memory and its use to change state or rules are considered learning. Changing state conditions based on learning demonstrates some level of cognition.

One of the goals of this work is to demonstrate how learning and cognition can benefit the functionality of a sensor network. In particular, we are interested in using these attributes to enable local decisions based on local information that effect global results. Limiting decisions to localities is important for reasons of scalability and autonomy. Local decisions allow distributed control. In turn, distributed control through local decisions provides a natural redundancy affording fault tolerance – as some motes exhaust their energy budget and expire others will continue to make decisions.

Although genetic algorithms are a popular algorithmic paradigm, they rely in a crucial way on extremely fast computational speed to evaluate many random mutations of some genetic specification. While most of these new combinations will prove useless, or worse, harmful to the objective, the search is for the small percentage of mutations that prove beneficial. Consequently, while genetic algorithms may be useful for small-scale sensor networks, they are useless to large-scale systems. Indeed, the limited computational power of the motes would make the use of genetic algorithms prohibitively expensive. Furthermore, we view a mote as an individual organism. Just as with living organisms, successful changes in behavior or other capabilities must be based on experience and learning. Random changes would be highly likely to result in death (i.e., failure) of the mote and catastrophe for the network. Afterwards, there would be no chance to try another mutation.

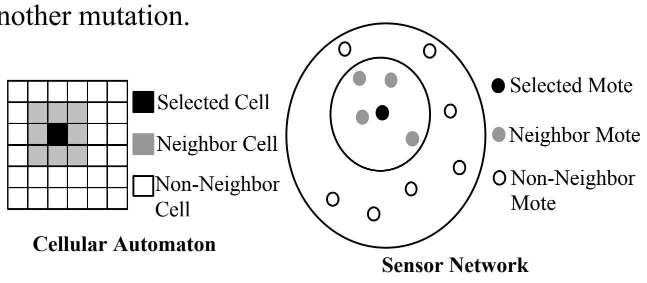

**Figure 1**: Illustrating the neighborhood of a cell

We use cellular automata as a viable model for massively deployed sensor networks operating as organisms in an ecosystem. A cellular automaton represents, in most ways, a distribution of sensor motes throughout a geographic region. As illustrated in Figure 1, eight neighbor cells surround each internal cell. Border cells have three or five neighbors. Neighbor cells represent those motes that can receive a transmission from a cell. Thus, the regularity of the grid represents a logical indication of physical proximity. Throughout this work we assume that each sensor has exactly eight neighbors. Visibly, the set of neighbors need not be limited to the eight adjacent cells. Specifying a neighbor radius greater than 1 increases the number of cells that can receive transmissions from the selected cell. A radius of two would include in addition to the eight adjacent cells, the 16 cells adjacent to these neighbors. One apparent limitation of this model is that the number of neighbors is fixed for a given radius; however, disabling some of the neighbors can change this.

## AGGREGATING SENSED DATA

There are, essentially, two ways in which data sensed by the motes can be aggregated. In a centrally-controlled network, data aggregation and integration is a two-stage process: in the first stage the motes forward the data collected to the sink. Some of the data may be fused en route but the final responsibility for aggregation rests within the sink. In the second stage, the aggregated result is broadcast back to the network. Though straightforward, this method does not scale well [17,22]. By contrast, in a truly distributed system, as is the case in an autonomous sensor network, the aggregation must be performed in-situ by the sensors themselves. To illustrate, suppose a distributed system has capabilities to both sample the temperature of its immediate environment and to set that temperature. The goal is for each mote to eventually obtain and maintain the global average by using local data only. As the values change, the process must be repeated. While a number of solutions to the aggregation problem have been proposed in the literature [7,17,22,23,24,25,26], they were either designed for sensor networks of small size or have a centralized flavor. One of the key contributions of this work is to show that fully distributed data aggregation can be performed in massively deployed sensor networks.

In the following, we first examine the issues of using a centralized approach for data aggregation in average calculation, and then propose our autonomous and distributed approach. As our cellular automaton is defined such that each cell can only transmit to its immediate neighbor, then in the centralized approach, data collected by the sink from each cell must pass through half the span of the grid on average. It is possible to minimize the number of transactions by aggregating values as they are routed to the sink, but this requires substantial infrastructure and coordination. In the second stage (broadcasting aggregated result), to conserve mote energy, a suggested approach is for the sink to broadcast a return message to all motes. But this method does not scale well as the power of the sink's transmission would have to grow with the size of the sensor network distribution and the transmission from the sink must be directly receivable by all motes (i.e., there can be no blockage). We assume a cost of flooding the computed average to all cells, which increases with the size of the grid. Besides transmission costs, there are additional problems with the centralized approach. Routing tables to reach the sink must be discovered and maintained. Disruption in these routes must be handled to assure messages arrive at the sink. Regardless of how this is done, this method is open to single points of failure.

Recently, it was noticed by Wadaa et al. [27] as well as by other workers that centrally-controlled sensor networks are prone to uneven energy depletion leading to the creation of energy holes in the vicinity of sinks. Specifically, [27] showed that by the time the motes close to the sink have expended all their energy, other motes in the sensor network still have about 80-90% of their original energy budget. This uneven energy depletion creates an energy hole around the sink, severely curtailing network longevity. Figure 2 illustrates the uneven energy depletion problem, where all black cells must pass messages to the sink through a single cell in the top row. In our cellular automaton model, the cells closest to the sink must relay messages from every cell in the grid and their energy budget will decrease rapidly relative to cells further away.

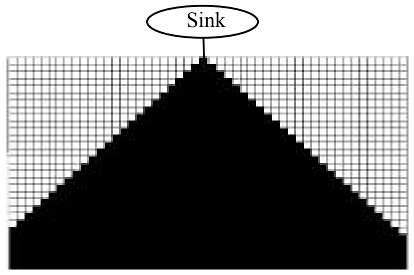

**Figure 2**: Illustrating the energy-hole problem

Our objective is for the sensor network to function as a community that will come to a consensus on some value across the network without any mote or any central authority having global knowledge of all mote values. The problem is for each mote to obtain and maintain the global value by iteratively using only data that is available locally.

We now give an informal description of our model. Assume a data value on which consensus is formed is known by each mote and obtained by some interaction with the environment. Each mote is assigned a *time period* and a *selection time* for action within that time period as genetic material. The time period is divided into one or more slots. The selection time assigned to a mote is one of these slots within the time period (e.g., one mote may be selected at slot 3, another at slot 18, but all motes will be selected at some time during a time period). Because the selection time for each mote is determined randomly, there is no guarantee that two or more motes will never be selected at the same time. Upon deployment, each mote starts its own clock. If two motes are at a neighbor radius greater than 3, they may execute the algorithm simultaneously (i.e., be selected), as the results of their calculations are independent of each other. If the radius is less than 3, the result of calculations is order-dependent. In this case, simultaneous transmissions will cause collisions, thus a MAC layer protocol is assumed to decide cellular execution order. To simulate this control, all motes (i.e. cells) selected at the same time slot are executed in random order. The order is randomized anew when they are selected next time. Thus, the selection time is fixed at deployment (being part of the genetic material), but the execution order in each time slot is a "function of the environment" and may change. Even with MAC layer negotiations, by this method each cell is acting asynchronously and autonomously. Because the time and cost of reaching a global decision is a function of the consensus value, selection time assignment, and execution order in each time slot, we run multiple executions varying all three parameters and average the results.

When a cell reaches its selection time within the time period, it executes the algorithm as follows. If its status is inactive, it does nothing. Otherwise, if its status is active, it begins a series of transactions. A transaction is either a request for information from a neighbor or a specification given to a neighbor. Transactions are significant because they require radio transmissions, typically the most costly activity of a mote. If a mote determines no action is required, it sets its status to inactive and will not participate again until it is reactivated by one of its neighbors. If a mote determines action is required, it will perform the necessary work resulting in some change to itself and/or its neighbors. During this process, if any neighbor is inactive, it may be reactivated as determined by the selected mote. The simulation continues until all cells are set to inactive. Thus, the simulation ends using only local information; no global control is required. As these local neighbor cells act autonomously and asynchronously yet cooperate with each other, act only on local environmental information, and remember information from one action that will affect a future action, we argue that this system demonstrates simple cognition.

## COMPUTING THE GLOBAL AVERAGE

In the following, we use our CA to simulate two functions: averaging a value and determining majority rule across the network. Figure 3 depicts an initial distribution in a cellular automaton with a 30 x 30 grid of cells showing a random distribution of values (for example, assume a color) in the range of [0,255]. The objective is to calculate an average color and set all motes to that value using only local information to determine local actions. When each cell is selected, it requests the color value of each of its neighbors. If all are equal, it inactivates itself. If not, it calculates an average for the neighborhood and sets itself and all neighbors to that value. Figure 4 illustrates the color change after 10 periods. Soon thereafter, the color differences are indistinguishable to the naked eye.

The converged average for the community is always equal to the average of the original distribution calculated prior to starting the algorithm. An interesting and valuable attribute of this algorithm is that the average color following each cell selection is also equal to the initial calculated average. Re-examining the process reveals the reasons for this state of affairs. A group of nine cells of different colors contributes to the average of the total grid. When these cells are averaged and set to the same average value for the group, they contribute exactly the same to the average of the entire grid as they did with differing values.

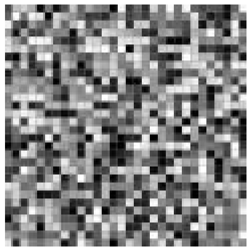
**Figure 3:** *Initial distribution of colors for color averaging*

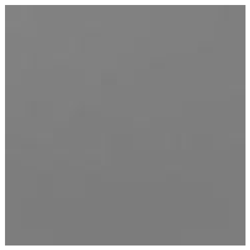
**Figure 4:** *Distribution of colors after 10 time periods*

A greater advantage of our decentralized approach is in the distribution of energy expenditure. The "funnel effect" of multi-hop routing required for the centralized approach described above will deplete the energy of cells much faster when their distance to the sink is shorter. In our decentralized approach, the workload is not only evenly distributed, but the workload required of an individual mote actually decreases as the grid size increases, as shown in Figure 5.

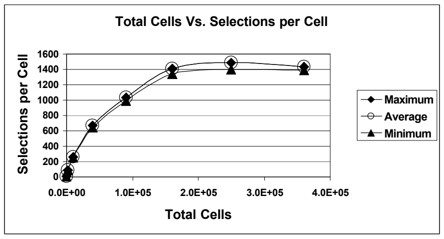

**Figure 5:** *Illustrating the minimum, average, and maximum number of cell selections*

As stated earlier, some applications require a close agreement of common value, while others may tolerate a much larger divergence. We call the latter case "good enough computing" and show that, in such cases, a distributed consensus on a common value range can occur quickly with relatively few transactions. As an example, a 200 x 200 grid begins with a color distribution depicted in Figure 6. All 256 colors are represented in a fairly even distribution with a standard deviation of 73.74, 49.6% of colors above the average plus a tolerance of 0.5, and 49.95% of colors below the average minus this tolerance.

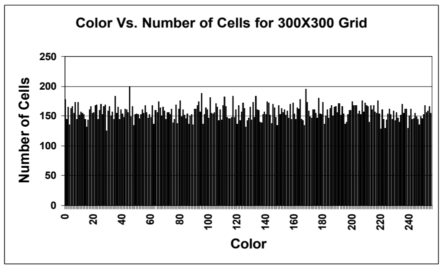

**Figure 6:** *Initial distributions of colors*

However, as shown by the visual display of the simulation, the algorithm comes close to the solution very quickly. The simulation takes 271 time steps to come to a solution where all cells are within the specified tolerance. By the 86th time step, there are only 3 colors represented. The standard deviation is within the tolerance of 0.5. No color is more than 0.56 above the average plus the tolerance and no color is more than 0.42 below the average minus the tolerance.

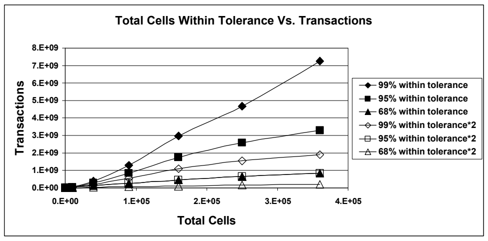

**Figure 7**: *Percentage of cells within tolerance*

Figure 7 shows that, like the total transactions required for a solution, solutions with a large percentage of cells within the tolerance also asymptotically slope towards zero as the grid size increases. Most importantly, solutions up to 95% within the tolerated average reach that asymptote quickly with few transactions regardless of grid size.

## EFFECTING THE MAJORITY RULE

Often there is a requirement for members of a community to reach consensus on a binary choice (e.g., *on* or *off*). In our algorithm, a selected cell determines the major frequency of its neighborhood and sets all in the neighborhood to that value. Figure 8 depicts an initial distribution in a cellular automaton with a 30 x 30 grid of cells showing a random distribution of binary values, represented by the colors black and white. Figure 9 shows the color change at the end of four iterations. Soon thereafter, all cells are the same color indicating the major frequency is determined. For larger grids, interesting patterns display, as groups of colors appear to move around the grid while cells on group boundaries compete for majority value.

Figure 10 shows that the time required for consensus escalates quickly for smaller grid sizes but after 300x300, the acceleration slows and then stabilizes for massive grid sizes. Figure 11 confirms that the same is true for the required number of transactions, an indicator of energy expenditure.

As with our approach to averaging, the workload is not only evenly distributed, but the workload required of an individual mote actually decreases as the grid size increases, similar to that shown in shown in Figure 5. Each time a cell is selected, it initiates transactions resulting in energy expense. Thus, the fewer the selections, the smaller the energy consumed. As in Figure 5, the cost in energy does not escalate for larger grids.

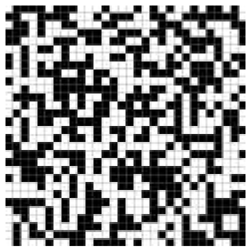

**Figure 8:** *Initial distribution of colors for majority rule*

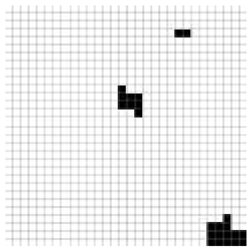

**Figure 9:** *Distribution of colors after 4 time periods*

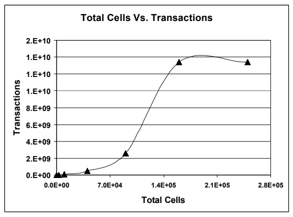

**Figure 10**: *Time required for majority rule*

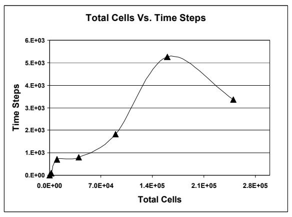

**Figure 11:** *Transactions required for majority rule*

In some cases, it may be sufficient for most motes to be set to the majority value. As depicted in Figures 8 and 9, this algorithm has a similar property to averaging, as more cells change to the majority value very rapidly. Figure 12 shows that about 70% of the cells have the majority value with very few time steps and, furthermore, this time requirement stabilizes for larger grid sizes. Figure 13 shows that the number of transactions continues to increase for a final solution, but the cost of 70% stabilizes for larger grid sizes. This again is a case of "good enough computing", and where applicable, it can substantially reduce the cost in time and energy of the calculation.

If carried to completion, this algorithm will always result in the community agreeing on a single switch value. However it suffers from the same problems described by

Mitchell, et al. and Epstein [29,30]. In some cases, particularly when there are large blocks within the initial distribution of the same color and especially if the distribution is close to equal, the final configuration may have a color opposite from the initial majority color. This is possible if initially some cells with the majority color are mistakenly turned to the minority color because its neighborhood has predominantly minority colored cells. This will incorrectly flip-flop the percentage of majority and minority color cells.

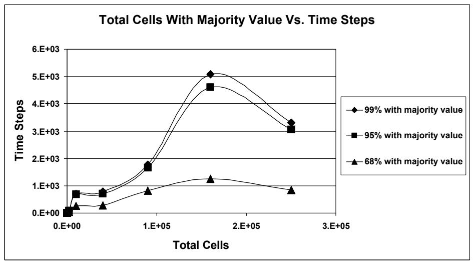

**Figure 12**: *The stabilizing time requirement for percentage of cells with majority value*

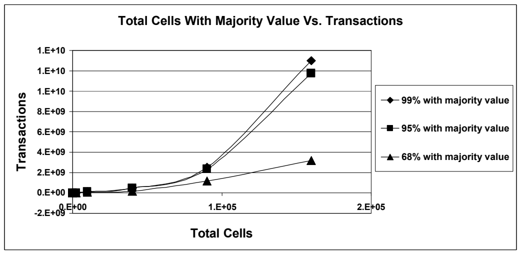

**Figure 13:** *The stabilizing transaction requirement for percentage of cells with majority value*

We solve this problem by using averaging to incrementally calculate the majority frequency. Instead of keeping an integer value for the switch, we maintain a real value. All cells are initially assigned genetic material for the switch value of 0.0 or 1.0. The algorithm proceeds as before, but now, instead of the selected cell calculating a binary frequency for the neighborhood, it calculates the average of all neighbors and itself. This real value is stored as the switch value. Whenever the binary value of the switch is required this value is assessed as (Value > =0.5) = 1.0 or (Value < 0.5)=0.0.

This works because of the property described for averaging. If the initial distribution has a majority of 1s, then the initial average will be greater than or equal to 0.5 and if the majority is of 0s, the initial average will less than 0.5. Because the average remains constant throughout the process, the majority at the end will be the same as the majority at the beginning.

The major frequency calculation shares many of the same properties as the averaging technique described above. The requirement for cell selections to calculate the majority rule is similar to that shown in Figure 5 for averaging. Again, the energy requirement per cell stabilizes for larger grids.

## CONCLUDING REMARKS

We see the full potential of sensor networks only reached when there are massive numbers of heterogeneous motes acting asynchronously and autonomously, yet cooperating in a way that their local actions, based on local information, combine to affect a functional and sustainable network interacting with the environment. This is how living systems have evolved so successfully. Individual organisms operate by a combination of innate rules (i.e. genetics) and learned behavior in a local niche. The combination of the actions of the individual organisms results in a multifunctional, sustainable ecosystem.

In this work we have demonstrated a function completed by a sensor network working as a community: autonomous motes functioning asynchronously cooperative to achieve a common goal. The function is carried out without centralized control and without any mote needing to know all information known within the community. We have also shown that the goal can be closely approached with few costs in time and resources compared with the much more costly final answer

The ever-increasing technology curve is going to continue to escalate the capabilities available for situation management. It is untenable that these tools can be timely managed and analyzed under centralized control. Methods such as we have described here will be necessary for the large systems to operate as a community where many functions can be facilitated by autonomous, asynchronous sub systems.

## References

[1] J. M. Kahn, R. H. Katz, and K. S. J. Pister, “Next century challenges: Mobile support for Smart Dust”, *Proc. ACM MOBICOM*, pp. 271-278, Seattle, WA, August 1999.

[2] B. Warneke, M. Last, B. Leibowitz and K. Pister, “SmartDust: communicating with a cubic-millimeter computer”, *IEEE Computer*, 34(1), pp. 44-55, 2001.

[3] V. V. Zhirnov and D. J. C. Herr, “New frontiers: self-assembly and nano-electronics”, *IEEE Computer*, 34(1), pp. .34-43, 2001.

[4] S. Park, I. Locher, A. Savvides, M. B. Srivastava, A. Chen, R. Muntz and S. Yue, “Design of a wearable sensor badge for smart kindergarten”, *Proc. 6th International Symposium on Wearable Computers*, Seattle, WA, October, 2002.

[5] K. Ryokai and J. Cassell, StoryMat: "A play space for collaborative storytelling", *Proc. CHI'99*, October 1999.

[6] M. Srivastava, R. Muntz and M. Potkonjak, "Smart Kindergarten: Sensor-based wireless networks for smart developmental problem-solving environments", *Proc. ACM MOBICOM*, Rome, Italy, July 2001.

[7] D. Estrin, R. Govindan, J. Heidemann and S. Kumar, "Next century challenges: Scalable coordination in sensor networks", *Proc. MOBICOM*, Seattle, WA, August 1999.

[8] J. Polastre, R. Szewcyk, A. Mainwaring, D. Culler and J. Anderson, "Analysis of wireless sensor networks for habitat monitoring", in Wireless Sensor Networks, Raghavendra, Sivalingam, and Znati, Eds., Kluwer Academic, 2004, pp. 399-423.

[9] R. Szewczyk, J. Polastre, A. Mainwaring, J. Anderson and D. Culler, "An analysis of a large scale habitat monitoring application", Proc. 2nd ACM Conference on Embedded Networked Sensor Systems, Nov. 2004.

[10] D. M. Doolin and N. Sitar, "Wireless sensors for wild re monitoring", Proc. SPIE Symposium on Smart Structures & Materials (NDE 2005), San Diego, California, March 6-10, 2005.

[11] D. Estrin, D. Culler, K. Pister and G. Sukhatme, "Instrumenting the physical world with pervasive networks", Pervasive Computing, 1(1), pp. 59-69, 2002,.

[12] K. Martinez, J. K. Hart and R. Ong, "Environmental sensor networks", IEEE Computer, 37(8), pp. 50-56, 2004.

[13] D. M. Doolin, S. D. Glaser and N. Sitar. "Software Architecture for GPS-enabled Wildfire Sensorboard", TinyOS Technology Exchange, University of California, Berkeley CA, February 26, 2004.

[14] Intel Research, "New Computing Frontiers – the Wireless Vineyard", http://www.intel.com/technology/techresearch/research/rs01031.htm, accessed April 17, 2005.

[15] K. A. Delin and S. P. Jackson, "The sensor web: a new instrument concept", Proc. SPIE Symposium on Integrated Optics, San Jose, California, January 2001.

[16] P. Saffo, "Sensors, the next wave of innovation", Communications of the ACM, 40(2), pp. 93-97, 1997.

[17] I. F. Akyildiz, W. Su, Y. Sankarasubramanian, and E. Cayirci, "Wireless sensor networks: A survey", Computer Networks, 38(4), pp. 393-422, 2002.

[18] D. Lammers, "Embedded projects take a share of Intel's research dollars", EE Times, August 28, 2001. Retrieved April 5, 2004, from http://today.cs.berkeley.edu/800demo/eetimes.html

[19] Mainwaring, A., Polastre, J., Szewczyk, R., & Culler, D., "Wireless Sensor Networks for Habitat Monitoring", Intel Research, IRB-TR-02-006, June 10, 2002, 2002 ACM International Workshop on Wireless Sensor Networks and Applications. Retrieved April 5, 2004, from http://www.greatduckisland.net

[20] UCB/MLB 29 Palms UAV-Dropped Sensor Network Demo, 2001, University of California, Berkeley. Retrieved April 5, 2004, from http://robotics.eecs.berkeley.edu/~pister/29Palms0103

[21] Chen, M., Majidi, C., Doolin, D., Glaser, S., & Sitar, N., "Design and construction of a wildfire instrumentation system using networked sensors", Network Embedded Systems Technology (NEST) Retreat, Oakland California. Retrieved April 5, 2004, from http://firebug.sourceforge.net

[22] D. Culler, D. Estrin and M. Srivastava, "Overview of sensor networks", *IEEE Computer*, 37(8), pp. 41-49, 2004.

[23] I. Chatzigiannakis and S. Nikoletseas, "A sleep-awake protocol for information propagation in smart dust networks", Proc. IEEE International Parallel and Distributed Processing Symposium, (IPDPS'03), Nice, France, April 2003.

[24] S. Olariu and Q. Xu, "A simple self-organization protocol for massively deployed sensor networks", Computer Communications, to appear, 2005.

[25] K. Sohrabi, J. Gao, V. Ailawadhi and G. Pottie, "Protocols for self-organization of a wireless sensor network", IEEE Personal Communications, 7(5), pp. 16-27, 2000.

[26] K. Sohrabi, W. Merrill, J. Elson, L. Girod, F. Newberg and W. Kaiser, "Methods for scalable self-assembly of ad hoc wireless sensor networks", IEEE Transactions on Mobile Computing, 3(4), pp. 317-331, 2004.

[27] A.Wadaa, S. Olariu, L. Wilson, M. Eltoweissy and K. Jones, "Training a wireless sensor network", Mobile Networks and Applications, 10, pp. 151-167, 2005.

[28] S. Olariu, A. Wadaa, L. Wilson and M. Eltoweissy, "Wireless sensor networks: leveraging the virtual infrastructure", IEEE Network, 18(4), pp. 51-56, 2004.

[29] M. Mitchel, J. Crutchfield, and R. Das, "G1.15: Computer Science Application: Evolving Cellular Automata to Perform Computations", in Bäck, T., Fogel, D., and Michaelewics, Z. (Eds.), Handbook of Evolutionary Computation. Oxford University Press, 1997.

[30] J. Epstein, "Learning To Be Thoughtless: Social Norms And Individual Computation", Center on Social and Economic Dynamics Working Paper No. 6, revised January 2000. Forthcoming in Computational Economics.