

# Sensor Networks Applied to the Problem of Building Evacuation: An Evaluation in Simulation

Constantine K. Christakos  
Cambridge, MA  
Email: dean@alum.mit.edu

**Abstract**—Use of ad hoc sensor networks as a decentralized control system has been proposed for a variety of applications. I propose a distributed sensor system to aid in path finding and apply such a system to the problem of building evacuation. In simulation, I design an ad hoc protocol to allow sensors to cooperate in an attempt to find the best path to an exit in a situation where paths may be blocked and the effective physical topology is dynamically changing. Integrating this sensor simulation with a pedestrian simulator, I show that when sensors direct pedestrians to travel in the direction of the best path to the exit, escape time is improved. Instructions from sensors based on local exchange of data about changing conditions results in performance close to the performance of decisions made by pedestrians based on if they had global knowledge of physical conditions before making their decisions.

## I. INTRODUCTION

This paper addresses the problem of sensor networks and effectors. By reducing the system behavior to simple operations that are executed by a great many sensing elements, I show that the overall network design can quickly reconfigure due to changing conditions.

I propose a method for solving the problem of redistributing data throughout the network to those who need it without the need to consult a central coordinator. I present the problem of pathfinding and, more specifically, building evacuation as an example application. Sensors spread throughout a building communicate over wireless channels to determine the direction of the nearest escape routes and reconfigure and redirect evacuees in the event of changing pathways due to congestion or disasters.

First, I present background on other wireless sensor network applications used in the past and make note of their architectures. Particular attention is given to other applications in which wireless communication has facilitated path finding. Section III examines methods of simple pathfinding through a graph and applies these methods to path discovery and routing through a building. Next, I implement a system that solves this problem using a sensor network simulator combined with a pedestrian simulator whose agents take instructions from the sensor nodes. Finally, I propose future directions to examine and test further interactions between sensors and effectors.

## II. BACKGROUND

Increased capabilities, intelligence, and displays in sensor nodes allows more complicated, fully-featured systems to be implemented on a small hardware platform. Developers of

Smart Dust[1] pointed to the trends of “complete systems on a chip, integrated low-power communication, and integrated low-power transducers,” as forces that made intelligent networks of sensors possible.

Early proposed applications of ad hoc sensor networks included environmental monitoring,[2], [3], which by their nature lend themselves to a centralized architecture in which data is aggregated back to a base station where it is examined and analyzed by inquirers.

Another application is traffic detection and control, using both inductive loop sensors[4], [5], microphones[6], or cameras[7],[8] to detect unusual incidents or count cars. The data from those sensors is aggregated back to a central base station, either directly [7],[8], [6] or via multi-hop protocols[5], which analyzes that data and sends signals to the drivers by changing traffic lights or electronic signs.

Sensor networks have been used to solve robotic path-planning problems, such as the pursuit-evasion game (PEG) in which an autonomous pursuer attempts to catch an evading agent. Solutions to the PEG are frequently based on centralized architectures, using sensors that report back to a central map builder and strategy planner, [9] and using sensors spread throughout the field which report to a central base station which actuates a camera to the evader’s location.[10] The pathfinding application of Li, De Rose, and Rus[11] uses a sensor network to direct a robot along the best path to a final destination point with sensors in “danger” areas sending out signals that the robots avoid moving towards.

My system specifically deals with providing instructions to people and estimates blocked passageways by assuming that sensors can count the number of pedestrians in a specific area. Pedestrian-counting is an active field, and applications using multiple types of sensors in consort, such as IR-sensors and thermal sensors, have been explored in this space[12], as well as the use of cameras[13], [14] to, in some cases, do a literal “head count” of pedestrians.

## III. PATHFINDING IN A GRAPH

Use of sensors as a decentralized control system mentioned above are used for controlling robots or redirecting traffic. I am interested in controlling and redirecting people and pedestrians through a building using actuators such as intelligent signs. A representation of the floorplan as a node-edge graph, such as in figure 1, provides a way for the sensor nodes to understand

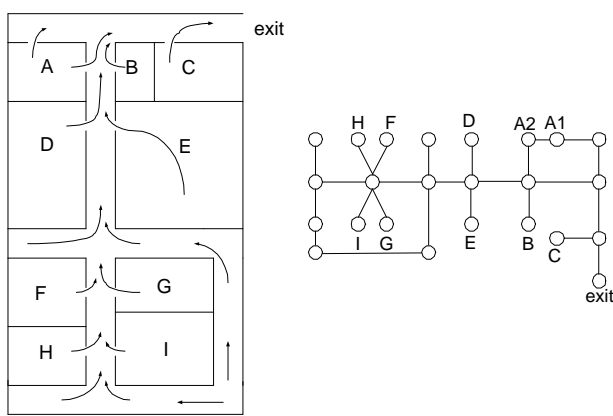


Fig. 1. **Left:** The paths that users will be directed to in order to reach the exit the fastest. **Right:** That layout as it corresponds to a graph, with individual rooms maintaining their same label

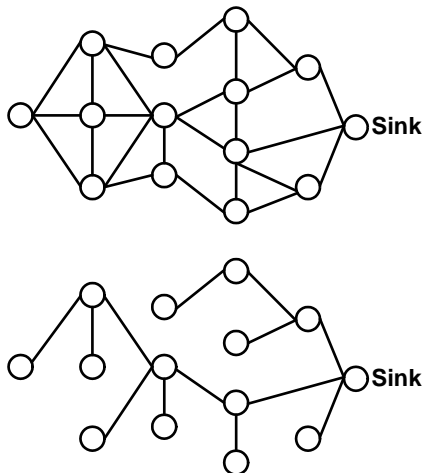


Fig. 2. A graph with an associated spanning tree rooted at the sink of that graph.

the available pathways as well as allows us to analyze the problem with greater rigor.

Let us first start with the most reduced form of an escape scenario—finding a path between a source and sink in a graph. If the graph,  $G$ , is fully connected and has  $n$  nodes, path lengths may vary from two nodes (directly from the source to the sink) to  $n$  nodes (a Hamiltonian path containing all nodes). In a graph of  $n$  nodes, the longest-length path will be  $n - 1$  steps.

Some simple analysis provides a starting point to count the number of paths that exist to the sink. A graph  $G$  can be represented as a spanning tree  $T$  rooted at the sink (Figure 2). In this tree, there are  $(n - 1)$  unique paths to the sink from any other node. So one can say that the number of unique paths to the sink is definitely  $\Omega(n)$ . All of the possible paths can be enumerated by creating a spanning tree of nested spanning trees. If there are  $(n - 1)$  unique paths to the sink using the spanning tree rooted at the sink, then each of those  $(n - 1)$  nodes in the spanning tree itself as a spanning tree rooted at itself with  $(n - 2)$  unique paths to that node, and

so on. This gives an upper bound of  $O(n!)$  on the number of unique paths to the sink in the graph. Clearly, exhaustively searching through such a large number of possible paths is time-consuming.

Finding a single shortest path in a graph is not difficult, as one can just use Dijkstra’s algorithm. However, quickly rerouting and finding a new path in the event that an edge is removed presents a challenge due to the large number of different possible paths in the graph; one wants to avoid the need to re-calculate an algorithm such as all-pairs shortest path, which has a cost of  $O(n^3)$ . Even just rerunning Dijkstra’s algorithm has a cost of  $O(n^2)$  and, even in the best case, a runtime of  $O(m + n \log n)$  using a Fibonacci heap[15]. Our goal is to see if one can recover from a failure in a graph.

#### IV. USING AD HOC NETWORK PROTOCOLS TO ROUTE PEOPLE, NOT PACKETS

I argue that routing people through a physical network is analogous to routing data in an ad hoc network, and thus the protocols for maintaining the physical paths will be similar. In a simplified case, one could make the assumption that packet hops are comparable to distance [11] when determining the distance to the exit from a node. Some early explorations of sensor network protocols assumed fixed-radius communication so that one could infer distance between two communication nodes or other geometric properties.[16], [17] In reality, radio transmission is a much more complex phenomenon which experiences fading, differences in signal strength based on obstacles, and asymmetric channel behavior between two points. Furthermore, radio communication between the nodes does not mean that there is a physical connection between two locations of those sensors. Empirical experiences with mote deployments in buildings [18] indicate that motes not physically connected to each other are still likely to have network connectivity and that the communication network is a much more highly connected network than the network of the physical floorplan.

Because using RF connectivity to infer physical topology is unreliable, my solution is to provide knowledge of the geographical layout to each sensor and allow the sensor to infer its own position within that map. The sensor node can then make decisions accordingly based on any additional data it receives from other sensors. I assume that sensors are able to detect conditions such as crowding in a corridor or a blocked passageway and can adjust their calculations about distance and routes based on this information.

Each sensor node is only aware of conditions in its immediate area and its estimated distance to the exit. It depends on data from the surrounding sensor nodes to get more information about the region. Ideally, the more sensor nodes that exist, the finer-grain and more distributed the dynamic path-data will become.

For simplicity’s sake, I use a reactive protocol based on the Destination-Sequenced Distance Vector (DSDV) [19] protocol. For a single-resource/exit building evacuation scenario, DSDV

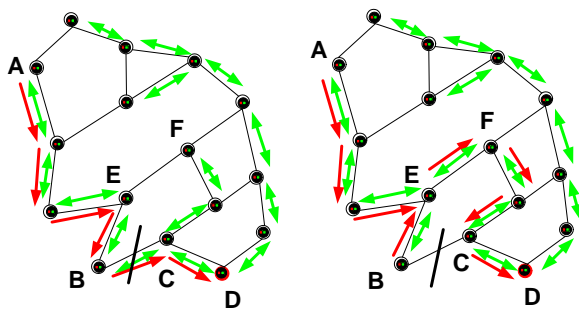


Fig. 3. **Left:** The network communication relationships (double-arrowed lines) and path to the exit (single-arrowed lines) before a blockage is detected. **Right:** The new network communication relationships and new path to the exit after a blockage is detected.

is a more effective method of communication and coordination. My implementation is not a pure DSDV protocol, as DSDV concerns itself primarily with network routing. I am concerned with geographical routing, so what I have created is a modified protocol based on DSDV that determines communication routes based on the physical layout of the sensors, in this case a Geographical-DSDV.

The model for the actions of the individual nodes in simulation is as follows: at each node on the map, there is a single mote with sensors on each edge. Motes forward on their inferred distance to the exit based on the results from their sensors on each edge and distance data they receive from their neighbors. No wireless node need know the entire path between it and the exit, only the number of hops from there to the exit and which link will lead that way. Knowledge of the entire path is *distributed*. In the event that conditions change due to a blocked passageway, the mote that senses the blocked passageway recalculates its distance estimate to the exit node which is forwarded on to its neighbors, which update their information and make decisions about which mote to direct pedestrians towards. Continued network communication between the two sensors connected by that blocked passageway is still possible.

An example of this process of reconfiguration is in Figure 3. The motes between node A and exit at node D direct pedestrians to the next step along the way to the destination. However, when there is a physical blockage between node B and node C, the network will reconfigure to set up new communication relationships between motes at nodes E and F and direct pedestrians down the new path.

It is important to show that the system can reconfigure itself efficiently. Because the system uses DSDV as its routing protocol, the number of messages sent per node is inextricably linked to the amount of time it takes for the system to recover from any changes. The number of messages per node that occur during an event is a reflection of the speed with which the system reacts. Network communication is the dominating factor in power consumption, rather than power consumed by the CPU.[20]

To measure network performance, I created a simulation

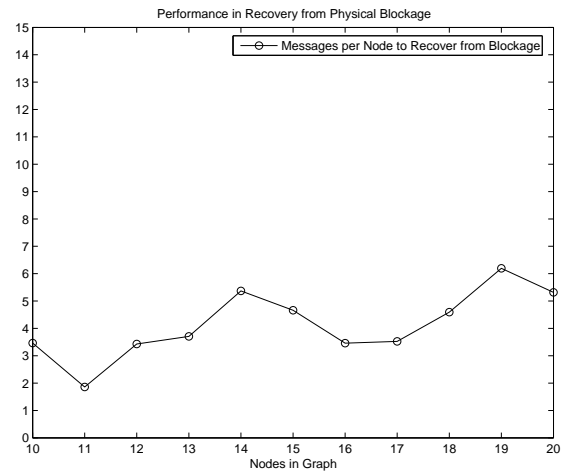


Fig. 4. Number of messages per node needed to recover from a detected physical blockage, according to graph size.

in the TinyOS simulator, TOSSIM[21], induced a “blocked passageway” detected by the sensing elements of a simulated mote, and measure the number of messages per node required for the motes to reconfigure themselves. As can be seen in figure 4, the number of messages to recover from a blocked passageway remains relatively steady, even as the size of the graph doubles from 10 to 20 nodes. This indicated that a dynamic system in which sensor nodes needed to efficient route around physical blockages was a practical solution.

## V. EVALUATING PERFORMANCE

To understand how effective a sensor network simulation is in solving a problem such as building evacuation, I create a model of sensor interactions with physical agents, bridging the world of sensor simulations and the world of physical simulations, specifically pedestrian simulations.

The field of pedestrian simulation is a very lively one, including applications to evaluate the effects of escape panic[22] and dealing with large crowds over large outdoor areas.[23] For my purposes, I created a simply pedestrian simulator based on Helbing’s escape panic simulator[22], specifically focused on the issue of the delays caused by pedestrian crowds attempting to escape through a doorway. As can be seen in figure 5, pedestrian crowding through a doorway causes delays as the movement of pedestrians is constrained by other pedestrians also attempting to escape through taht same doorway.

I integrated a simple pedestrian simulation with the TinyOS simulator. Pedestrians are seeded in the nodes—representing rooms—in the graph with the goal to reach the destination node. Without the aid of the motes in the TinyOS simulator, the pedestrians choose the exit from a node over an edge to the next node that represented the best path to the destination. If too many pedestrians crowd that exit from the node onto the next edge, the pedestrian considers that exit “blocked” and choose a random exit for a few steps, seeking an alternate

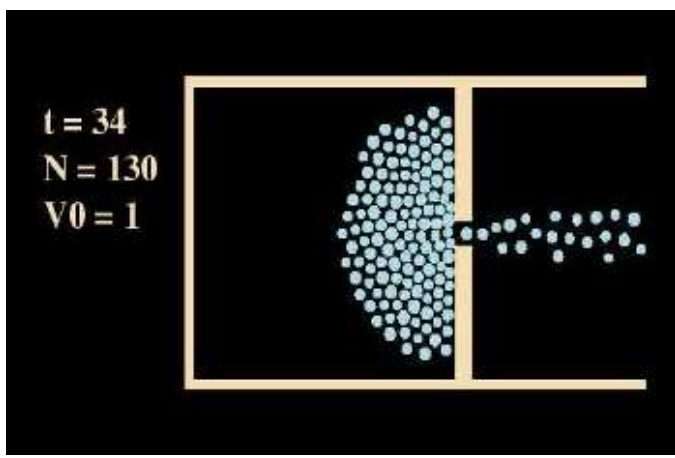


Fig. 5. A simulation of pedestrians escaping through a room in the Panic simulator.

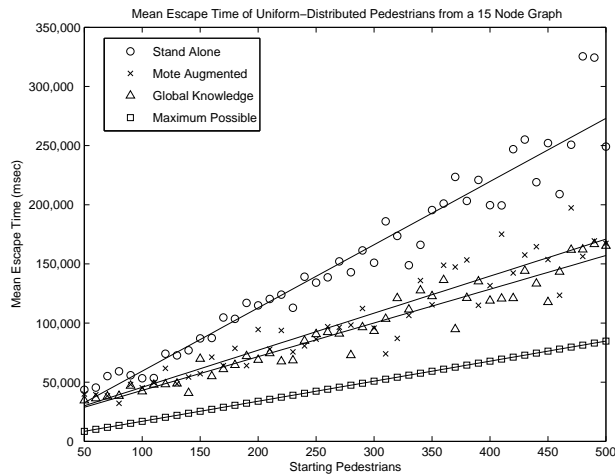


Fig. 6. A comparison of the mean escape times for the Movement Simulator in the stand-alone case and mote-augmented case for uniform distributed pedestrians

path. With the aid of the motes, when an exit is blocked by too many pedestrians, the motes find a new alternative path for the pedestrian to take and direct him in that new direction. Pedestrians take instructions from the intelligent, dynamically-changing motes which are looking for the best path to the destination node and paths that bypass the crowds.

I ran several simulations of escape scenarios. To get an idea of how escape times with the pedestrians' knowledge augmented by the motes compared to the stand-alone case, I kept the number of nodes fixed at 15 while increasing the initial number of pedestrians. I ran these simulations with both uniform and power law distribution of pedestrians among the nodes.

The first set of experiments examines the results of pedestrian escape performance for the case of uniform distributed pedestrians (Figure 6). The results are stark. As one can see, the performance in the mote-augmented case results in consistently improved performance. The mote-augmented case

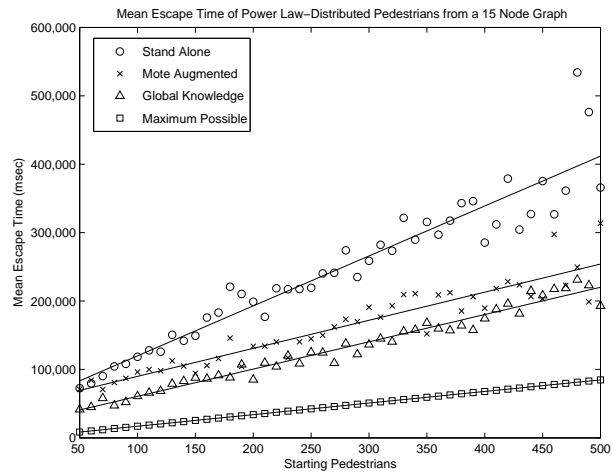


Fig. 7. A comparison of mean escape times for the Movement Simulator in the standalone case and mote-augmented case for power law-distributed pedestrians

is compared to the “Global Knowledge” case in which each pedestrian, when faced with a decision, chooses the fastest route to the exit at that given moment. Note that this is not a global optimum, simply the result if every pedestrian makes what would be the “best” decision whenever the pedestrian is presented with a choice. One can see that the decisions provided by the motes—facing issues such as delay, dropped packets, and decisions marking a path as “blocked” based only on the existence of more than 10 pedestrians in a queue—compares quite well when compared to the “Global Knowledge” case, which faces none of the communication issues and makes more accurate estimations of delay with respect to reaching the exit. In the latter case, the pedestrian has all knowledge of the conditions in the entire layout at the moment the pedestrian makes the decision about where to turn. As a baseline, this is compared to the mean escape time when pedestrians escape at their maximum possible bandwidth. In the case of power-law distributed pedestrians, in which there is more crowding, the results also reflect a consistent improvement (Figure 7), and the results show a striking improvement, given that the effects of crowding are felt much more acutely in the case of power-law distributed pedestrians versus the uniform distributed case. In the power law distributed case, variance appears to collapse with the augmented instructions (Figure 8), indicating that crowding relief—and thus less random behavior—contributes to the improved performance.

## VI. CONCLUSION

Augmenting the decisions of pedestrians with dynamic instructions provided in response to changing conditions provides a distinct advantage over the static instructions “posted” instructions that one might find in a typical building evacuation scenario. Using just a simple network protocol to repair breaks in a spanning tree, the system can function effectively and has performance quite close to an “ideal” system in which all

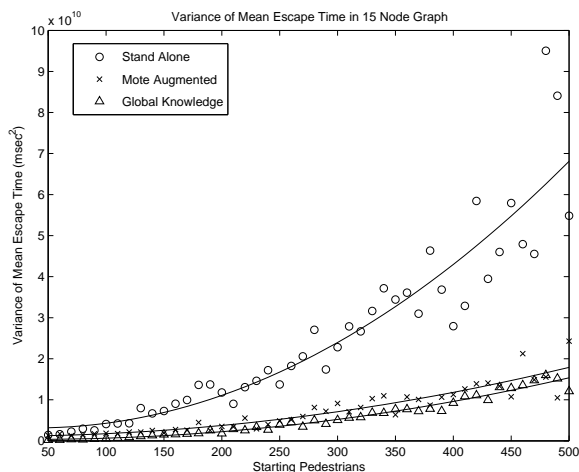


Fig. 8. A comparison of mean escape time variance for the Movement Simulator in the standalone case and mote-augmented case for power law-distributed pedestrians

agents had enough information at any given moment to make the best possible decision.

Certainly this provides a compelling case for experiments using physical deployments, rather than simple simulation. Potential applications other than the physical world involve traffic flow management or crowd flow management, such as in an amusement park. These latter two examples create more complicated issues because while the example of building evacuation is an inherently “damped system”—agents leave the system, never to return—traffic and pedestrian flow has agents entering the system as well as leaving. Such a dynamic system could be explored with more full-featured simulations. In addition, this simulation does not account for communication or other interactions between the pedestrians as well. Since matters of crowd control and escape panic has a large psychological component, work needs to be done examining how sensor networks can influence the decision-making and communication processes of individuals—particularly when changing instructions from sensor nodes provide instructions that could be seen as contradictory when observed over a fixed time interval.

#### ACKNOWLEDGMENTS

The author would like to thank Andrew Lippman, Joseph Paradiso, Samuel Madden, and Matthew Welsh for providing input and support that contributed to this work.

#### REFERENCES

[1] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. E. Culler, and K. S. J. Pister, “System architecture directions for networked sensors,” in *Architectural Support for Programming Languages and Operating Systems*, 2000, pp. 93–104.

[2] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk, and J. Anderson, “Wireless sensor networks for habitat monitoring,” in *WSNA '02: Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications*. New York, NY, USA: ACM Press, 2002, pp. 88–97.

[3] A. Wheeler, “Tephranet: Wireless, self-organizing network platform for environmental science,” Master’s thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, Cambridge, MA, 2001.

[4] D. I. Robertson and R. D. Bretherton, “Optimizing networks of traffic signals in real time—the scoot method,” *IEEE Transactions on Vehicular Technology*, vol. 40, no. 1, February 1991.

[5] A. N. Knaian, “A wireless sensor network for smart roadbeds and intelligent transportation systems,” Master’s thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, Cambridge, MA, 2000.

[6] S. Chen, Z. P. Sun, and B. Bridge, “Automatic traffic monitoring by intelligent sound detection,” in *Proceedings of the IEEE Conference on Intelligent Transportation Systems*, November 1997, pp. 171–176.

[7] S. Kamijo, Y. Matsushita, K. Ikeuchi, and M. Sakauchi, “Traffic monitoring and accident detection at intersections,” in *Proceedings of the IEEE International Conference on Intelligent Transportation Systems*, October 1999, pp. 703 – 708.

[8] R. Gangisetty, “Advanced Traffic Management System on I-476 in Pennsylvania,” in *Proceedings of the IEEE International Conference on Intelligent Transportation Systems*, 1997.

[9] H. J. Kim, R. Vidal, D. H. Shim, O. Shakernia, and S. Sastry, “A hierarchical approach to probabilistic pursuit-evasion games with unmanned ground and aerial vehicles,” in *40th IEEE Conference on Decision and Control*, 2001, pp. 634–639.

[10] B. Sinopoli, C. Sharp, L. Schenato, S. Schaffert, and S. Sastry, “Distributed control applications within sensor networks,” in *Proceedings of the IEEE, Special Issue on Sensor Networks and Applications*, August 2003.

[11] Q. Li, M. D. Rosa, and D. Rus, “Distributed algorithms for guiding navigation across a sensor network,” in *MobiCom '03: Proceedings of the 9th Annual International Conference on Mobile Computing and Networking*. New York, NY, USA: ACM Press, 2003, pp. 313–325.

[12] K. Hashimoto, K. Morinaka, N. Yoshiike, C. Kawaguchi, and S. Matsueda, “People count system using multi-sensing application,” in *Proceedings of the International Conference on Solid State Sensors and Actuators, TRANSDUCERS '97*, vol. 2, Chicago, USA, June 1997, pp. 1291–1294.

[13] D. B. Yang, H. G.-B. anos, and L. J. Guibas, “Counting people in crowds with a real-time network of simple image sensors,” in *Ninth IEEE International Conference on Computer Vision (ICCV '03)*, vol. 1, 2003.

[14] V. Kettner and R. Zabih, “Counting people from multiple cameras,” in *Proceedings of the IEEE International Conference on Multimedia Computing and Systems*, vol. 2, 1999, pp. 267–271.

[15] J. A. Hugh, *Algorithmic Graph Theory*. Prentice Hall, 1990.

[16] R. D. Poor, “Hyphos: A self-organizing wireless network,” Master’s thesis, Massachusetts Institute of Technology, Program in Media Arts and Sciences, Cambridge, MA, 1999.

[17] R. Nagpal, “Programmable self-assembly: constructing global shape using biologically-inspired local interactions and origami mathematics,” Ph.D. dissertation, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, 2001.

[18] “The Intel Mote laboratory testbed,” <http://astragalus.lcs.mit.edu/labdata/labdata.html>.

[19] C. Perkins and P. Bhagwat, “Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers,” in *ACM SIGCOMM'94 Conference on Communications Architectures, Protocols and Applications*, 1994, pp. 234–244.

[20] G. Anastasi, A. Falchi, A. Passarella, M. Conti, and E. Gregori, “Performance measurements of motes sensor networks,” in *MSWiM '04: Proceedings of the 7th ACM international symposium on Modeling, analysis and simulation of wireless and mobile systems*. New York, NY, USA: ACM Press, 2004, pp. 174–181.

[21] P. Levis, “Tossim: Accurate and scalable simulation of entire tinyos applications,” in *Proceedings of the First ACM Conference on Embedded Networked Sensor Systems (SenSys 2003)*, November 2003.

[22] D. Helbing, I. J. Farkas, and T. Vicsek, “Simulating dynamical features of escape panic,” *Nature*, vol. 407, pp. 487–490, 2000.

[23] C. Gloor, D. Cavens, E. Lange, K. Nagel, and W. Schmid, “A pedestrian simulation for very large scale applications,” A. Koch und P. Mandl (Hrsg.): *Multi-Agenten-Systeme in der Geographie*, 2003.