Toward Multi-Level Modeling of Robotic Sensor Networks: A Case Study in Acoustic Event Monitoring

Christopher M. Cianci, Thomas Lochmatter, Jim Pugh, and Alcherio Martinoli

***Abstract*—Modeling and simulation can be powerful tools for analyzing multi-agent systems, such as networked robotic systems and sensor networks. In this paper, it is shown concretely how instances of both these elements fit into a general methodology for *multi-level modeling,* providing insight into system dynamics. Use of the resulting general framework is illustrated through application to a specific sample case study involving a robotic wireless sensor network engaged in an acoustic detection task. We then compare and contrast the resulting family of models, highlighting explicitly the trade-off between realism and simplicity.**

I. Introduction

By the canonical definition, a sensor network is a system consisting of "many spatially distributed low-cost sensing nodes that collaborate with each other but operate autonomously, with information being routed to whichever node can best use the information." [1] Particularly as recent focus has shifted heavily toward *wireless* sensor networks and their potential to bring "spatially distributed collaboration" closer to "low-cost" ([2], [3]), the question of how to efficiently design and manage control of such networks is of ever increasing importance.

Many of the most common sensor network applications to date have been based upon the sampling of continuously available parameters (such as temperature, humidity, or other environmental factors, as in [4], [5]), which has allowed them to take advantage of extremely low duty cycles in the interest of extending network lifetime. However, in situations where the phenomena of interest are spatially and/or temporally unpredictable, the problem becomes slightly more complicated and the response of the network to environmental changes necessitates increased dynamism in behavior.

While there have been several attempts at modeling sensor networks for data filtering [6], [7], data prediction [8], network classification [9], [10], and system performance [11], all such work that we are aware of tends to focus on a single level of modeling for a very specific aspect of the system (typically either sensing or networking). Here we propose a slightly different approach: applying a statistical *multi-level* modeling methodology which allows us to capture the dynamics of the entire system together, at multiple levels of abstraction. This type of analysis has become commonplace in swarm-robotic systems [12], and should be equally applicable to sensor networks—which can also easily be considered multi-agent systems, just with slightly different constraints and capabilities.

C. Cianci, T. Lochmatter, Jim Pugh, and A. Martinoli are members of the Swarm-Intelligent Systems Group at the École Polytechnique Fédérale de Lausanne, CH-1015 Lausanne, Switzerland. {chris.cianci, thomas.lochmatter, jim.pugh, alcherio.martinoli}@epfl.ch

Sensor networks produce spatio-temporal monitoring data which can be considered both dense, compared with traditional monitoring and measuring techniques, and sparse, compared with the information gathered and used for the control of many multi-robot systems. A robotic sensor network has the potential to capitalize on benefits from the extremes of both systems: power management and explicit communication from the network and self deployment, reconfiguration, and collection afforded by the ability of the robots to self-locomote. Not all of these capabilities will be exploited here, but it is foreseen to leverage these possibilities in the future.

An element frequently encountered in many distributed systems is the concept of consensus [13], which is applied in mobile networks of autonomous agents [14], in stationary wireless sensor networks [15], and as bacterial quorum in natural systems [16]. And though some centralized solutions are occasionally used, fully distributed consensus mechanisms can be considered an expression of *Swarm Intelligence* (SI). While it is not the primary focus of this article, the ideal of simplicity in swarm-intelligent control will serve as an inspiration for the design of a concrete case study on which to illustrate the presented modeling framework.

The application of SI to distributed, real-time, embedded systems aims at developing robust task-solving methodologies by minimizing the complexity (including the intelligence) of the individual units and emphasizing parallelism, and self-organization.

From an engineering standpoint, the principle advantages of swarm-intelligent system design are four-fold:

- *scalability*, from a few to thousands of units;
- *flexibility*, as units can be dynamically added or removed without explicit reorganization;
- *robustness*, not only through unit redundancy but also through an adequate balance between explorative and exploitative behavior of the system;
- and *simplicity* (and low-cost) at the individual level, which also increases robustness.

Networked robotic swarms share several similarities with wireless sensor networks: they both consist of a large number of relatively simple nodes acting independently and interacting with each other. In this way, both can be considered potential candidates for swarm-intelligent methods of control.

The dynamics of a real-world sensor network deployment can be incredibly complex due to non-trivial, often noisy, interactions of individual agents with the environment and with each other. Furthermore, running a huge battery of

ROBOCOMM 2007, 14th – 16th Oct 2007, Athens, Greece.
Copyright © 2011 – 2012 ICST
ISBN 978-963-9799-08-0
DOI 10.4108/ICST.ROBOCOMM2007.2275

experiments on real hardware can be expensive and time-consuming. Models can help us to capture and understand these dynamics and give us the flexibility to explore certain aspects of the solution space with less effort. Naturally, there is a trade-off between the complexity of the model and its fidelity to the real system, but using a family of models can help us evaluate this trade-off by demonstrating directly comparable results on multiple levels of abstraction. This suite of models then becomes a powerful tool allowing us to bridge the gap between mathematical equations and reality. In many cases, it even allows us to generalize from one situation to another, or to a broader class of scenarios.

These methods have been quite successful in swarm-robotic systems, where the holistic approach to the system (including the environment, sensing, actuation, communication, etc.) presents an encouraging comparison to wireless sensor networks. On the other hand, current probabilistic models in swarm robotics which go up to the macroscopic level are often based on mean-field approaches and non-spatial metrics where the (continuously changing) actual locations and trajectories of the robots can be neglected if there is sufficient randomization of their positions over time and with repeated runs (due to noisy local interactions and basic, reactive navigation schemes). For specific classes of scenarios and objectives (e.g. aggregation [17], stick pulling [12], foraging [18]), particularly in swarm-robotic systems, this assumption makes sense, and allows for a drastic reduction in the number of states considered in the system; unfortunately, that is no longer the case when considering a sensor network, as nodes typically do not move continuously or rapidly, if at all, and therefore do not necessarily represent a well-mixed system over time and over repeated runs. Similar to further classes of scenarios where deliberative navigation schemes are used for the sake of system efficiency [19], this lesser degree of spatial randomness will eventually need to be treated differently at the macroscopic level.

The remainder of the article will be organized as follows: Section II will motivate the overarching structure of the multi-level modeling approach, and Section III will specify the details of a concrete case study and a corresponding behavioral controller that will be used as an example onto which we can apply the modeling framework. In Section IV, we explain how and why the module-based microscopic model is constructed and calibrated, and apply it to the presented case study. The higher-level agent-based models are shown and compared in Section V, further elucidating the similarities, differences, and relationships between the various levels. Some results are shown in Section VI, and their implications discussed in Section VII, highlighting the *generic* nature of this approach to modeling, despite the presence of the current illustrative case study.

II. Multi-Level Modeling Approach

Performing systematic experiments directly on the target hardware system can be cumbersome, costly, time-consuming, or even impossible for logistical reasons such as safety or availability. However, by demonstrating correspondence with higher abstraction layer representations of the system, we can gather and analyze information which may eventually be applied back to the design and control of the target system. In this context, simulation can therefore be a very useful tool for bridging the gap between theory and experiment. It is not intended to be a substitute for real experiments, but rather a supplement, allowing additional flexibility and diversity in the tests performed.

At the core of the multi-level modeling methodology is the trade-off between complexity and realism in the modeled representation of the system. Indeed, this relationship is implicitly present in any model, but part of the strength of the multi-level approach is that it not only treats this element explicitly, but provides a spectrum of different models presenting a clear mapping from one level to the next, demonstrating precisely the impact of this trade-off.

Models of lesser complexity, when used properly, can be quite desirable for any of several reasons; ease of manipulation, speed of execution, and aide to understanding underlying principles, for example. However, it is clear that care must be taken to ensure that the simplified model still faithfully represents the real system, or the work done with it will be of little use.

The real system can be seen as the basis of the hierarchy, as it must necessarily represent the ground truth on which all subsequent modeling levels will be built, and against which they will be eventually judged.[1] The following is a brief overview of the types of models that will later be described in more detail, from the most realistic and complex to the most abstract.

A. Module-Based Microscopic Model

The natural first step is to take a simulated model which is as realistic as possible, modeling not only the agents and their environment, but the individual modules which make up the agents and the environment, such as sensors, actuators and signal propagation.

B. Agent-based Microscopic Models

In contrast, agent-based models abstract away the internal details of the individual nodes, treating each as a simple, but nonetheless independent element. The environment can also be simplified in order to accelerate numerical implementation.

1) Continuous Spatial: The continuous model can be seen as a type of Monte-Carlo simulation, in which the agents and their environment are modeled in a continuous spatio-temporal framework.

2) Discrete Non-Spatial: It is often possible to analyze the system and derive independent expressions for the transition probabilities from the environment, allowing the Markov chains to be iterated over (in discrete time) *without* explicitly modeling the spatiality of the environment.

[1]Note that another option would be to incrementally validate models (i.e., the ground truth for level $n+1$ is level n), but this remains a different question for future study.

III. Case Study: Detecting Acoustic Events

In general, the problem of resource allocation is not limited to power management within the network, but extends also to the treatment of data and interrupts destined for the operator, outside the network. Various monitoring and detection applications naturally have a wide range of requirements regarding false positives and false negatives, and the relative severity of either occurrence.

For the purposes of illustration, we will consider a scenario in which false positives are particularly undesirable, as they may trigger the invocation of a costly (or otherwise resource-intensive) procedure. Such an environment places particular emphasis on measurement confidence, and we have constructed a simplistic collective decision algorithm accordingly, exploiting the multi-level modeling framework to carry out further systematic exploration of its behavior and perform some analysis.

Acoustic event detection has been selected as an example of a domain where the measurement target is unpredictable in space and time. Our treatment here will use acoustic events as an illustration, but may be straightforwardly applied to any modality which is localized in space and time.

In this system, we will attempt to increase measurement confidence by requiring a consensus among an arbitrary integer C different nodes that a significant event has occurred.

A. Experimental Setup

The physical aspects of the system and its environment can be described by a set of six parameters (as illustrated in Figure 1):

A	area of interest
N	number of available nodes
$D(N)$	distribution of available nodes
E	set of events occurring in the environment
$P_{det}(r, I_e)$	probability of detecting an event of intensity I_e at a distance r
$P_{com}(r, I_t)$	probability of message reception with intensity I_t at a distance r

Of these, A, N, and D may be directly and arbitrarily selected by the experimenters (though the precise realization of $D(N)$ may be perturbed by noisy factors beyond their control).

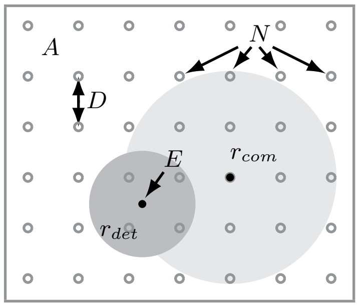

Fig. 1. Illustration of basic characteristics and parameters of the experimental setup. Nodes were spaced approximately 50 cm apart on a grid in a 3 by 3.5 meter area. (The r_{det} and r_{com} shown here are based on the Heaviside approximations that will be used for $P_{det}(r)$ and $P_{com}(r)$ in Section V-A. Obviously, other forms could be chosen for these functions, but they are more complicated to represent in this type of diagram.)

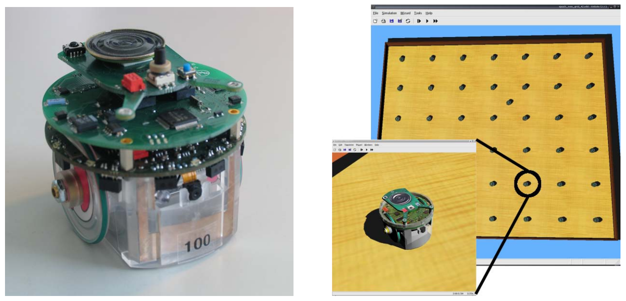

Fig. 2. LEFT: The e-puck, a small-scale experimental robotic platform. Shown here with the radio communication board stacked between the basic module and the jumper board, allowing the implementation of sensor networks and other networked robotic systems. RIGHT: The experimental setup described in Section III-A recreated in Webots, with plug-ins for acoustic and network dynamics. The 42 nodes arranged in a 6×7 grid remain stationary (fixed), while the event source wanders randomly around the arena emitting short acoustic events at 2 second intervals. The inset shows the simulated e-puck robot.

Characteristics of the desired target events E are specifiable, but the actual events (and their locations, times, etc. . .) are by definition unpredictable. The two probabilities $P_{det} and P_{com}$, while still indirectly controllable, are often inherent qualities of the physical agent being used, and should be determined empirically on the chosen hardware platform.

The experiments described here were performed on a fleet of e-pucks[2] (a miniature robotic platform recently developed at the École Polytechnique Fédérale de Lausanne, shown in Figure 2 left). The standard e-puck has a trinaural microphone array on-board, which was used in conjunction with a simple digital filter to detect acoustic pulses at approximately 3.6 kHz. It is also equipped with a small speaker, allowing it to emit sound. Additionally, the e-pucks have been fitted with a custom extension turret for short-range radio communication using a the subset of the 802.15.4 and ZigBee protocols present in TinyOS [20] (and are therefore fully interoperable with both MicaZ [2] and Telos [3]). The transmission power of the communication module is software-controllable, and passes through a custom attenuation circuit yielding effective ranges between approximately 10cm and 5m. More details about the radio turret used can be found in [21].

For the present study, we construct the default hardware system by distributing $N = 42$ agents over a regular grid[3] ($D = 50cm$ spacing) in a rectangular arena ($A = 3 \times 3.5$ meters). Each is equipped with a communication device (radio) and an acoustic sensor (trinaural microphone array). Acoustic pulses of a certain amplitude ($I_e = 8$, in the arbitrary units used within the firmware controller) are seen as events of interest, and are generated on this area at random locations by a 43rd agent unrelated to the established network in any way.

[2]http://www.e-puck.org

[3]Various strategies for automatic deployment have been explored in the literature, and for the sake of simplicity will not be treated here; these include mapping and monitoring an unknown indoor environment [22], [23], even distribution across an input target function [24], density-biased distribution based on sensor measurements [25], and using virtual pheromones [26].

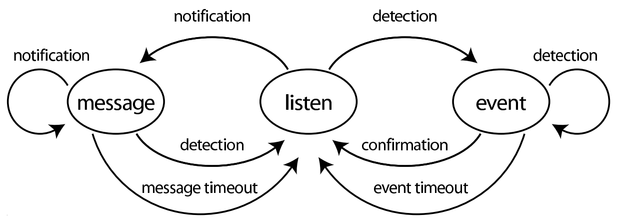

Fig. 3. Basic description of the individual controller algorithm as a finite state machine for $C = 2$. All nodes begin by "listening," and upon either detecting an "event" or receiving a "message," wait a short period of time for the complimentary signal before returning to the "listen" state either successfully (having observed a matched pair of the tow signals) or unsuccessfully (timeout, most likely indicating a false positive). Therefore, an event is only reported at the level of the entire network if detected by at least two nodes.

B. Control Algorithm and Parameters

Let us treat the system as follows. In our setup, we require an event to be detected by at least C nodes before it is reported. A node that perceives an event will announce the tentative detection to a subset of the network (the nodes within communication range; default being single-hop broadcast), and await confirmation in the form of similar messages generated by other nodes in the network. Thus, we have constructed a simple controller at the individual level which can be described by the finite state machine shown in Figure 3 for the case where $C = 2$. In this way, a successful detection can be defined as the reception of both an event and a confirmation within a given window of time.

This adds three more control parameters into the system description:

T_e	timeout after hearing an event, waiting for message
T_m	timeout after receiving a message, waiting for event
C	number of confirmations required for event acceptance

For the present setup, T_e and T_m can be selected as a function of the node spacing, the speed of sound, and the effective communication range. An event passing at the speed of sound will travel out of the area of interest in on the order of 10 milliseconds; therefore even accounting for some potential processing and sending delay, $T_e = T_m = 0.5$ seconds is more than sufficient.

In principle, C may be chosen arbitrarily, but there may be application specific instances in which certain values of C may be optimal, and others implausible (as a function of the field being monitored and the agents being used). Where necessary to further the example, we will continue to use $C = 2$, as above.

C. Preliminary Implementation in Hardware

The three-state controller described in Figure 3 was implemented to run in-situ on the individual nodes, and 42 nodes were arranged as shown in Figure 1. The event source is mobile, and wanders freely about the arena avoiding obstacles and emitting acoustic pulses, but does not interact in any other way with the observing network. Fifty events were generated at random locations within the area of interest, and the response of the network to each event was recorded.

IV. The Module-based Microscopic Model

The first abstraction layer we will consider is the *module-based microscopic model*: realistic simulation. While obviously a simplified version of the real world, this level still maintains as much realism as possible by preserving intra-node details, such as the individual sensors and sensing modalities, actuators, transceivers, etc. It bears reiteration that a *microscopic* model implies separate and independent computation for each of the individual agents; here, the *module-based* microscopic model further divides each agent into separate calculations for each of its constituent sub-systems (modules).

A. Identifying and Calibrating the Modules

Considering the chosen domain—a *robotic* sensor *network* engaged in the detection of *acoustic* events—at least three non-trivial modules will be necessary: the e-puck robot (with sensors and actuators), acoustic dynamics (speakers, microphones, propagation, and reflection), and the radio communication turret (OSI layers, channel emulation and noise, collisions, etc…).

Note that while the selection of modules is influenced somewhat by the problem domain being considered, we have not yet limited ourselves to any specific controller behavior; the module-based microscopic model is inherently generic, and can be used to explore and test any number of variations on the behavioral controller.

1) Robotic Node Dynamics—The e-puck in Webots: Beyond creation of the 3D model of the robotic platform in the Webots simulator [27], it is necessary to properly calibrate input and output responses to match those of the real hardware platform. The amount of wheel slip experienced by the virtual locomoting robot is determined by running odometric experiments on the real robot. The infrared proximity sensors are modeled as 3D cones, and the non-linear detection response and sensor noise closely match those observed in reality.

2) Acoustic Dynamics—Generation, Reflection, Fading, and Mixing: In an enclosed environment, sound propagation and perception can be highly influenced by the arrangement of environmental boundaries and physical obstacles. To accurately model these dynamics, it is necessary to extend simulation beyond simple source-to-receiver calculations. A framework was therefore created which runs in parallel to the Webots simulation and calculates sound dynamics. This framework uses a two-dimensional map of simulation walls, along with the locations of all microphones (receivers) and speakers (sources), to implement the Image-Source sound propagation algorithm [28]. The Image-Source algorithm was chosen as a good compromise in the trade-off between accuracy (the technique models reflection and attenuation but not diffraction) and computational speed. The algorithm works by creating virtual sound sources by reflecting actual sources across straight walls in the environment; the signal perceived by a receiver then becomes the sum of signals from all visible sources, both real and virtual. This method allows most of the algorithmic computation to be executed in the preliminary setup, while the calculation of receiver detection can be determined very

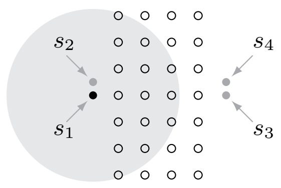

Fig. 4. Setup for experimental determination of the sensor and communication ranges through testing in the target environment. Nodes are spaced at 30 centimeter intervals for $P_{det}(r)$ and 60 centimeter intervals for $P_{com}(r)$.

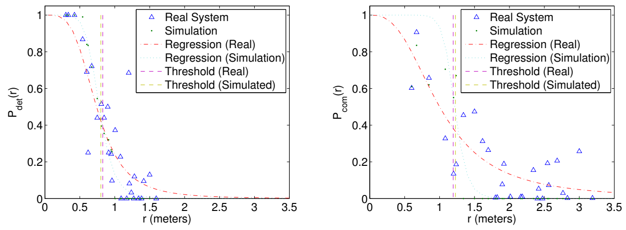

Fig. 5. The probabilities $P_{det}(r)$ of detecting an event (left) and $P_{com}(r)$ of successful communication (right) versus radial distance r to the robot. Sigmoidal regressions and approximate thresholds are shown with the measured data from both the physical system and the Webots simulation. The area left of the threshold (vertical line) is equal to the area under the curve; this value is used as a Heaviside approximation.

quickly, allowing for fast simulation. However, the original algorithm assumes that sound sources remain stationary, and because this is often not the case in Webots simulations, the algorithm was modified; instead of finding virtual sources in the preliminary execution, the algorithm calculates all possible sound reflection paths. At each step of the simulation, for every receiver, each source-path combination is checked to determine if it adds detectable input to reception. While this technique is computationally slower than the original Image-Source algorithm with stationary sources, it is significantly faster than recalculating virtual sources at every simulation step.

To determine $P_{det}(r)$ in the target environment, we arranged 28 nodes on a 30 centimeter grid near an event source (as shown in Figure 4), and performed two tests of 50 generated events each at four different source positions (in order to minimize the effects of orientation and self-interference). The percentage of successfully detected events at each position was recorded, and the combined estimation of $P_{det}(r)$ is shown in Figure 5 left.

3) Network Dynamics—The OMNeT++ Webots Plug-in: Not unlike the sound propagation just described, realistic representation of the radio communication channel also presents a complex modeling challenge. For the radio transceiver, an 802.15.4/ZigBee module was developed for the OMNeT++ [29] network simulation engine, which was wrapped as a plug-in to webots; positions and instructions are passed from Webots to OMNeT++, which then handles the channel coding and fading signal propagation, and in turn notifies Webots

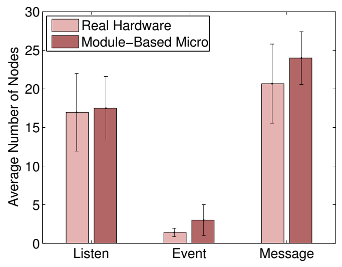

Fig. 6. Average number of nodes in each state in response to an event. Results and standard deviation for the physical system compared with the module-based microscopic simulation.

when messages should be received at other agent locations.

Similar to the setup described for estimating the detection range above, an analogous battery of tests measuring the effective communication range was performed in the same configuration (Figure 4), but using 60 centimeter spacing to cover a larger area, and the eight tests consisted of 100 messages each. Figure 5 right shows the results estimating $P_{com}(r)$ for a fixed transmission power of 0.5 mW.

B. Correspondence Between the Physical System and the Module-Based Microscopic Model

As can be seen in Figure 5, the individual modules that make up the module-based simulation exhibit responses which are quantitatively very close to the observations taken from the real system. Having demonstrated correspondence, it is now reasonable to perform additional simulations of the target system for analyzing its behavior.

Figure 2 right shows the setup described in Section III-A reproduced in the embodied simulator Webots, enhanced with the aforementioned plugins modeling the propagation of acoustic signals and the wireless network dynamics. The same experiment was performed for 500 events, with similar results, as shown in Figure 6.

This is only a very basic method of analyzing the system dynamics. It is naturally also possible to define specific spatial or non-spatial metrics. For instance, we could have asked our model to predict the number of events detected by C nodes and reported versus the number of events generated (non-spatial) or a mapping of message propagation around specific event detections (spatial). Some examples of such metrics will be shown below.

C. Additional Exploration Using the Module-based Microscopic Model

Another part of the stated reasoning for modeling is to facilitate further exploration of the parameter space. The existence of the model allows us to manipulate and re-run the experimentation quickly and easily.

1) Discriminating Between Two Different Event Sources: As the cited goal of this case study was to reduce the impact of incorrect detections on the network (false positives), let us now introduce a second source of what we will call "undesirable" events into the area of interest. This additional event source also moves randomly in A, and emits acoustic pulses identical to the target source, but at a lower intensity (proportional to I_e: $\frac{I_e}{I_u} = \{0.5, 0.75, 0.95\}$).

2) Performance Metric: In order to have a quantitative method of reporting system performance, we define a metric function M as the number of desirable events successfully detected and the number of undesirable events successfully ignored, in relation to the actual number presented:

$$M(\alpha, \beta) \quad = \quad \alpha \frac{E_{det}}{E_{tot}} + \beta \left(1 - \frac{E_{fp}}{\max(E_{fp}, E_{tot})} \right) \quad (1)$$

where E_{det} is the number of events reported, E_{tot} the total number presented, and E_{fp} the number of false positives reported. The coefficients α and β may be balanced according to the severity one wishes to associate with either term, so long as they sum to one for normalization.

Results using this metric will be presented and compared with additional modeling layers in Section VI.

V. Using Higher Levels of Abstraction: Agent-Based Microscopic Models

By the same principle of abstraction and correspondence, we can further distill the system down to the interactions between its key parameters by considering an *agent-based microscopic model*. At this level, we consider one copy of the controller state machine (Figure 3) for each agent in the system, and the interactions between them in a further simplified environment (a *spatial* model; intentionally eliminating intra-node hardware module details).

A. Continuous Spatial Model

The obvious first step is to keep the same logical system construction from the module-based version, and simply remove the modules. For this case study, then, the event generation becomes a Monte Carlo simulation on a continuous space (there is no real need for the moving robot), the acoustic events simply propagate at the speed of sound for a fixed radial distance, and radio messages are delivered instantaneously to other agents in a fixed radial range. Details such as acoustic reflection and message collision are completely neglected at this level; the model is simpler and faster, but less realistic. By comparing the results of adjacent modeling layers, we can assess directly how much less realistic, and determine whether or not it will have an appreciable impact on the usability of the results.

The physical setup described in Section III-C was recreated in matlab, with each node characterized by its position in the x-y plane. Heaviside (step function) approximations were used for the functions measured in Section IV-A (the vertical dotted lines in Figure 5 represent the cutoff, and correspond to an area under the step function equal to that under the sigmoid).

This simulation was then run for 1,000 events, the results of which will be shown as part of the comparison in Section VI.

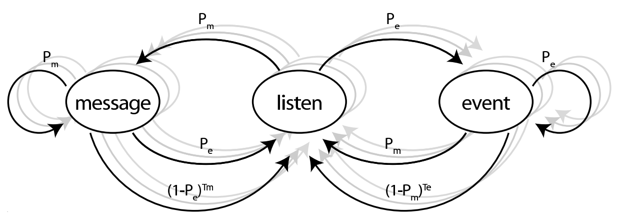

Fig. 7. The deterministic responsive controller from Figure 3 modified with transitions labeled by the probability of encountering the associated stimulus, yielding a probabilistic finite state machine (PFSM).

B. Discrete Non-Spatial Model

Recognizing that the finite state machine representation of the behavioral controller can be considered a Markov chain, yet a further simplification can be to treat it as such (see an example in Figure 7), essentially reducing the system to a 1-dimensional time-discrete synchronous simulation. In order to do this, transition probabilities will need to be determined, either analytically or by estimation/extraction from the spatial microscopic simulations where a closed form solution is not possible. This yields a system in which each agent becomes independent of the others, and actual location is immaterial (similar to neglecting trajectories in a mobile robotic system), hence the loss of spatiality.

1) Probability of Event Arrival: As we are considering the response to a single event independently of other events (for the time being, we will require that events be disjoint in the time granularity of the system), we immediately condition the following on the reception of an event. That is, given that an event has occurred, at an unknown random location in A, the probability P_e that it is detected at a certain node should correspond roughly to the geometric ratio of the detection area to the total area:

$$\frac{A_{det}}{A} = \frac{\pi r_{det}^2}{A} = \frac{\pi (0.8287)^2}{3 \cdot 3.5} = 0.2055 \quad (2)$$

where r_{det} corresponds to the discontinuity in the Heaviside function. This conceptual relationship can be seen in the illustration provided in Figure 1. This is an over-estimate, though, due to the fact that the outermost nodes are only $d = 0.25m$ from the border, and therefore their detection zones extend beyond the arena. However, the average effective detection area can be approximated by removing the $N_{border} = 26$ circular segments (A_{seg}) which extend over the boundary.

$$\begin{aligned} A_{seg} &= \left(r_{det}^2 \arccos\left[\frac{d}{r_{det}} \right] - d\sqrt{r_{det}^2 - d^2} \right) \quad (3) \\ &= 0.6708 \\ \hat{A}_{det} &\approx \frac{N \cdot A_{det} - N_{border} \cdot A_{seg}}{N} = 1.7422 \quad (4) \\ P_e &\approx \frac{\hat{A}_{det}}{A} = 0.1659 \quad (5) \end{aligned}$$

Confirmation by Monte Carlo simulation of 100,000 generated events yields $P_e \approx 0.1656$.

2) Probability of Message Arrival: P_m is a bit more complicated, as it must be conditioned not only on the arrival of an event, but additionally on the detection of that event at another node in the network, within a certain proximity so as to allow for communication. The probability of a node being able to communicate with the node in question can be calculated geometrically using the radius of communication, as was done above with the sensory radius. Multiplied by the total number of nodes, this gives an estimate of the number of nodes within communication range. Together with the fraction of these which are in the "Event" state (and therefore sent a message), this tells us what the probability should be that any given node receives a message:

$$\begin{aligned} P_m(k) &\approx \frac{\hat{A}_{com}}{A} \cdot N \cdot \frac{N_E(k)}{N} = \frac{\hat{A}_{com}}{A} \cdot N_E(k) \quad (6) \\ &= \frac{3.4648}{3 \cdot 3.5} \cdot N_E(k) = 0.3300 \cdot N_E(k) \end{aligned}$$

where k is the iteration, and kT the time (T is the sampling interval). Notice the dependence on time; as such, we will not be able to use average values in execution, but they may still give us some confirmation that our derivation is not entirely unfounded. In the spatial microscopic model of Section V-A, the average percentage of nodes that received a message ($N_M(k)/N$) was 0.5987, while our theoretical equation (using the average value of $N_E(k)$ from the simulation) gives 0.5623.

VI. Results and Comparison of Modeling Layers

Here we can finally see, in Figure 8, the output of each modeling level side-by-side for the source discrimination experiment described in Section IV-C. The 'undesirable source' was assigned an intensity I_u proportional to that of the target source I_e ($\frac{I_u}{I_e} = \{0.5, 0.75, 0.95\}$) and $C = 2$. All models were run 20 times; for 100 (module-based), 1,000 (continuous spatial), and 10,000 events (discrete non-spatial), respectively. Figure 8a shows $M\left(\alpha = \frac{1}{2}, \beta = \frac{1}{2}\right)$, an even balance between the two contributing terms of the metric; the lower plots show the extreme cases $M(1,0)$ and $M(0,1)$.

All three models reflect similar trends, despite substantial differences in computational complexity (approximately an order of magnitude in execution time between each). More experiments are needed to identify subtle effects of different modeling design choices on a given metric related to this case study.

VII. Remarks, Conclusions, and Future Work

Here we have shown the application of a multi-level modeling methodology to a robotic wireless sensor network tasked with the reliable detection of acoustic events. Clear correspondence has been demonstrated at each transition, allowing experiments to be performed at a simpler level with only minimal loss of quality in the results, and maintaining generality and applicability to the target system. This is a basic formulation, with plenty of avenues open for refinement, but the essence of the multi-level framework is that it places overt and primary emphasis on maintaining a view of the whole system by providing an intuitive and incremental process for building descriptions and abstractions of the system at several levels. Once the system has been fully constructed, further modifications can always be made to improve the correspondence between layers and the interaction between parameters, which the framework allows us to do much more effectively and with greater confidence that the analysis is both correct and complete.

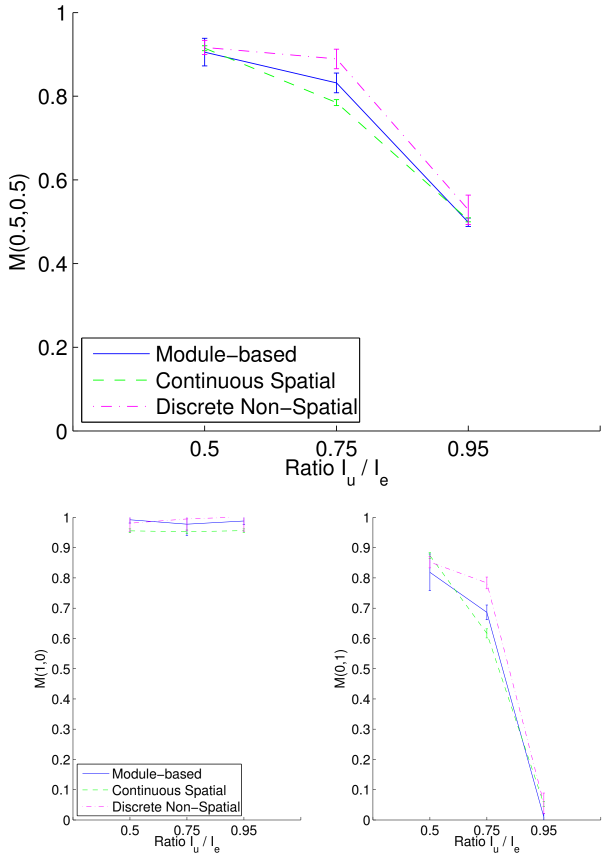

Fig. 8. Comparison of results from all three of the modeling levels presented, for the source discrimination experiment described in Section IV-C. Mean and standard deviation over 20 runs of 100, 1,000, and 10,000 events for the module-based, continuous spatial, and discrete non-spatial models, respectively.

Continuing the work presented here, an ongoing effort is being made to add a "discrete spatial" level between the continuous spatial and discrete non-spatial models shown here, using the known properties of the signal propagation (acoustic and radio) and analytical geometry to identify regions of space covered by a given number of nodes and their relative areas. Combined with a variable quantization of the field on which the events are generated, we can further increase performance and simplicity (which also aids understanding) in comparison to the continuous model, but without sacrificing as much detail as the non-spatial model, a feature which is particularly desirable when used with a spatial metric. This construction is providing further insight into the effects of varying the node spacing in relation to the sensing and communication ranges, and how to intelligently specify the number of nodes participating in the consensus (C).

Other obvious additions that we are currently studying include the adaptation of the models to deal with nonlinear systems in more a general way, particularly those involving spatial metrics. Eventually, we would also like to explore the possibility of applying a macroscopic model as well, incorporating all of the system dynamics into a single, concise representation, neglecting even the individuality of the agents, as previously done in swarm robotics systems.

VIII. Acknowledgments

The physical experiments described in this paper were facilitated by the invaluable hardware expertise provided by Xavier Raemy, whose assistance was greatly appreciated. Christopher Cianci, Jim Pugh, and Alcherio Martinoli are currently sponsored by a grant from the Swiss National Science Foundation (Contract Numbers PP002-68647 and PP002-116913). Thomas Lochmatter receives support from the Mobile Information and Communication Systems National Center of Competence in Research (NCCR-MICS). The authors would also like to thank Olivier Michel, Yvan Bourquin, and Alexei Kounine, without whose assistance the modification and integration of the realistic network simulation engine OMNeT++ into Webots^{TM}would not have been possible.

References

[1] C. Y. Chong and S. P. Kumar, "Sensor networks: Evolution, opportunities, and challenges," *Proceedings of the IEEE*, vol. 91, no. 8, pp. 1247–1256, August 2003.

[2] J. L. Hill and D. E. Culler, "Mica: A wireless platform for deeply embedded networks," *IEEE Micro*, vol. 22, no. 6, pp. 12–24, November/December 2002.

[3] J. Polastre, R. Szewczyk, and D. Culler, "Telos: Enabling ultra-low power wireless research," in *Information Processing in Sensor Networks: Special track on Platform Tools and Design Methods for Network Embedded Sensors (IPSN/SPOTS)*, Los Angeles, CA, USA, April 2005.

[4] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson, "Wireless sensor networks for habitat monitoring," in *ACM International Workshop on Wireless Sensor Networks and Applications (WSNA)*. Atlanta, GA, USA: ACM Press, September 2002, pp. 88–97.

[5] R. Szewczyk, E. Osterweil, J. Polastre, M. Hamilton, A. Mainwaring, and D. Estrin, "Habitat monitoring with sensor networks," in *ACM Conference on Embedded Networked Sensor Systems (SenSys)*, J. A. Stankovic, A. Arora, and R. Govindan, Eds. Baltimore, MD, USA: ACM Press, November 2004, pp. 214–226.

[6] S. Subramaniam, T. Palpanas, D. Papadopoulos, V. Kalogeraki, and D. Gunopulos, "Online outlier detection in sensor data using non-parametric models," in *International conference on Very large data bases (VLDB)*. Seoul, Korea: VLDB Endowment, September 2006, pp. 187–198.

[7] C. Guestrin, P. Bodik, R. Thibaux, M. Paskin, and S. Madden, "Distributed regression: an efficient framework for modeling sensor network data," in *International symposium on Information processing in sensor networks (IPSN)*. Berkeley, CA, USA: ACM Press, 2004, pp. 1–10.

[8] D. Chu, A. Deshpande, J. M. Hellerstein, and W. Hong, "Approximate data collection in sensor networks using probabilistic models," in *International Conference on Data Engineering (ICDE)*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 48–59.

[9] R. Jurdak, C. V. Lopes, and P. Baldi, "A framework for modeling sensor networks," in *Workshop on Building Software for Pervasive Computing (OOPSLA)*, Vancouver, Canada, October 2004.

[10] S. Tilak, N. Abu-Ghazaleh, and W. Heinzelman, "A taxonomy of wireless micro-sensor network models," *SIGMOBILE Mobile Computing and Communications Review*, vol. 6, no. 2, pp. 1559–1662, 2002.

[11] C.-F. Chiasserini and M. Gareto, "Modeling the performance of wireless sensor networks," in *Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*. IEEE Press, March 2004, pp. 231–242.

[12] A. Martinoli, K. Easton, and W. Agassounon, "Modeling of swarm robotic systems: A case study in collaborative distributed manipulation," *The International Journal of Robotics Research*, vol. 23, no. 4-5, pp. 415–436, 2004.

[13] J. Turek and D. Shasha, "The many faces of consensus in distributed systems," *IEEE Computer*, vol. 25, no. 6, pp. 8–17, 1992.

[14] R. Olfati-Saber and R. M. Murray, "Consensus problems in networks of agents with switching topology and time-delays," *IEEE Transcations on Automatic Control*, vol. 49, no. 9, pp. 1520–1533, 2003.

[15] R. Olfati-Saber and J. S. Shamma, "Consensus filters for sensor networks and distributed sensor fusion," in *IEEE Conference on Decision and Control, European Control Conference (CDC-ECC)*. IEEE Press, 2005, pp. 6698–6703.

[16] J. C. March and W. E. Bentley, "Quorum sensing and bacterial cross-talk in biotechnology," *Current Opinion in Biotechnology*, vol. 15, no. 5, pp. 495–502, October 2004.

[17] W. Agassounon, A. Martinoli, and K. Easton, "Macroscopic modeling of aggregation experiments using embodied agents in teams of constant and time-varying sizes," *Autonomous Robots*, vol. 17, no. 2-3, pp. 163–191, 2004.

[18] K. Lerman, C. Jones, A. Galstyan, and M. J. Mataric, "Analysis of dynamic task allocation in multi-robot systems," *The International Journal of Robotics Research*, vol. 25, no. 3, pp. 225–241, March 2006.

[19] N. Correll, S. Rutishauser, and A. Martinoli, "Comparing coordination schemes for miniature robotic swarms: A case study in boundary coverage of regular structures," in *International Symposium on Experimental Robotics (ISER)*. Springer Tracts in Advanced Robotics, July 2006, p. to appear.

[20] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, "System architecture directions for network sensors," in *Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, Cambridge, November 2000.

[21] C. M. Cianci, X. Raemy, J. Pugh, and A. Martinoli, "Communication in a swarm of miniature robots: The e-puck as an educational tool for swarm robotics," in *Simulation of Adaptive Behavior (SAB-2006), Swarm Robotics Workshop*, Rome, Italy, October 2006, Lecture Notes in Computer Science (2007), vol. 4433, pp. 103–115.

[22] V. Kumar, D. Rus, and S. Singh, "Robot and sensor networks for first responders," *PERVASIVE Computing*, pp. 24–33, October 2004.

[23] A. Howard, L. Parker, and G. Sukhatme, "The SDR experience: Experiments with a large-scale heterogeneous mobile robot team," *Int. J. of Robotics Research*, 2006.

[24] L. Chaimowicz, N. Michael, and V. Kumar, "Controlling swarms of robots using interpolated implicit functions," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, Barcelona, Spain, April 2005, pp. 2487–2492.

[25] M. Schwager, J. McLurkin, and D. Rus, "Distributed coverage control with sensory feedback for networked robots," in *Robotics: Science and Systems*, Cambridge, MA, USA, June 2006.

[26] D. Payton, R. Estkowski, and M. Howard, "Compound behaviors in pheromone robotics," *Robotics & Autonomous Systems*, vol. 44, pp. 229–240, 2003.

[27] O. Michel, "Webots: Professional mobile robot simulation," *Journal of Advanced Robotic Systems*, vol. 1, no. 1, pp. 39–42, 2004.

[28] J. B. Allen and D. A. Berkley, "Image method for efficiently simulating small-room acoustics," *Journal of the Acoustic Society of America*, vol. 65, no. 4, pp. 943–950, 1979.

[29] A. Varga, "Software tools for networking: "OMNeT++"," *IEEE Network Interactive (2002)*, vol. 16, no. 4, July 2002.