

# Self-Organized Embedded Sensor/Actuator Networks for “Smart” Turbines

Nikolaus Correll, Christopher Cianci, Xavier Raemy and Alcherio Martinoli  
Swarm-Intelligent Systems Group (SWIS), École Polytechnique Fédérale de Lausanne

**Abstract**—Combining networks of static sensors with minimalist robotic swarms might enable a new generation of micro-machinery equipped with sensor and actuator networks for inspection, maintenance, and repair. In such a scenario, limited capabilities of swarm members (due to miniaturization or for economical reasons) might render deterministic algorithms unfeasible and require a self-organized approach.

Driven by a case study concerned with the autonomous inspection of a jet turbine engine, we identify three main research axes: development of appropriate hardware, modeling and design of self-organized dynamical systems, and synthesis of robot controllers to achieve a desired collective behavior, which is for instance provided by an human operator during runtime. We also present our developments of embedded communication systems for miniature robots, allowing for communication within the swarm and static nodes in the environment.

Such networks of static and mobile nodes might allow for sophisticated spatio-temporal coordination, which would otherwise require localization and navigation abilities that are unfeasible on miniature platforms.

## I. INTRODUCTION

Networked robotic systems can be understood as a new paradigm that aims at exploiting interaction between mobile robotic agents and sensor networks in the environment. Here, benefits of an ubiquitous sensor network (e.g., simultaneously sensing large areas) can be combined with actuation and sensing capabilities of a larger robotic platform (e.g., a robot for support of the elderly that uses sensory information from the environment to react on the human’s needs, or a sensor network providing information to first response teams in an emergency scenario [1]).

On the other hand, sensor networks might complement swarms of robots with otherwise limited capabilities: interactions between the sensor nodes, the environment, and the robot swarm might lead to self-organized spatio-temporal patterns that exhibit a desired behavior such as inspection, cleaning, surveillance or maintenance of the workspace or objects therein.

In recent years, research in sensor networks has seen tremendous growth, and led to the development of different hardware platforms, standards, and software tools, which are widespread and shared in academic and industrial research. On the other hand, robotics researchers are discovering the advantages of distributed robotic systems. While the focus of the sensor network community has been on developing systems that can be deeply embedded in the environment, i.e. being extremely small and power-economic, robotic research usually relies on existing technologies such as IEEE802.11b

(WLAN), which has found widespread use, but might be lead to scalability problems in large swarms of robots.

Networked robotic systems however require communication not only among robots, but also with environmental sensors, or other platforms for which a high-level protocol stack such as IEEE802.11b is unfeasible for one reason or the other (e.g., energy constraints).

Thus, a key challenge in developing feasible implementations of large scale networked systems are the developments of standards, as well as hardware, which is able to be deployed at different scales (with respect to the size of an individual unit) and yet able to communicate amongst each other. On the other hand, large scale distributed systems comprising extremely simple hardware, might require non-classical approaches for unit coordination. Here, self-organization is a promising new paradigm, which might yield extremely powerful and robust solutions on platforms with the above constraints.

### A. Smart Turbines

In order to minimize failures, jet turbine engines have to be inspected at regular intervals for evidence of internal distress such as cracking or erosion. This is usually performed visually using borescopes [2], a process which is time consuming and cost intensive. One possible solution for speeding up and automating the inspection process is to rely on a swarm of autonomous, miniature robots which could be released into the turbine while still on the wing [3]–[5]. Potentially, research in this direction might lead to the development of a new generation of “smart turbines” where a robotic swarm for inspection and repair will tightly interact with a network of sensors that monitored turbine health status in flight.

The jet turbine environment imposes drastic constraints on the robotic platform (e.g., miniaturization, only local communication), and therefore emphasizes a distributed approach. We started developing self-organized coordination algorithms [6], which will potentially exploit (local) radio communication for improving inspection performance [7], [8]. So far we concentrated on the mere coordination mechanism for inspecting the environment as fast as possible (in a simplified 2D environment, Figure 1), and did not implement interaction of the robot swarm and potential sensors embedded in the turbine yet. Such sensors could provide for instance directions to the robots, e.g. by providing information about engine status during operation prior to inspection [3], which would allow the swarm to organize in a decentralized fashion while providing some sort of crude localization facility. Also, so far only

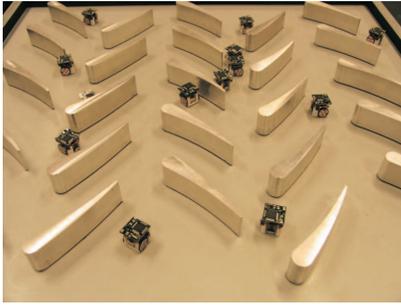


Fig. 1. A swarm of miniature robots *Alice* [9] performing boundary coverage of the blades in a jet turbine mock-up.

few attempts have been made [2] that would allow a human operator to correlate the sensory information collected by the individual agents with the geometry of the turbine. Finally, a sensor network could serve for transmitting mission control information from a human operator to the robot swarm (e.g. inspection of a certain part of the turbine).

### B. Self-Organization as Coordination Mechanism

Self-organization is emerging from the interplay of four ingredients: positive and negative feedback (e.g., amplification or saturation, respectively), randomness, and multiple interactions among individuals [10]. Given the properties above, self-organization can benefit from unreliable sensors and actuators (including communication devices), which create the necessary noise level for a balance between exploitation (performing the best possible action) and exploration (trying sub-optimal alternatives, which might nevertheless lead to superior overall performance).

Also, while self-organization might achieve less efficient coordination than other distributed control schemes, it can provide extremely high levels of robustness and can be applied to miniature robotic platforms such as those mentioned in this paper.

One of the major drawbacks of self-organization in an engineering context is its lack of analytical tractability of the—often emergent—collective behavior. We try to overcome this limitation by combining robust behavior-based control (e.g., [11]) at the individual level with probabilistic modeling [12] that allows us to calculate the analytic mean of arbitrary swarm performance metrics based on the (probabilistic) behavior of the individual agent.

### C. Related work

Exploiting a sensor network by a robotic platform and vice versa has been successfully implemented in a variety of case studies, for instance in [13] a environmental monitoring sensor network is used by a robot equipped with more sophisticated sensor to sample the micro-climate in a forest more efficiently. In [14] a sensor network is deployed by robot in order to efficiently explore an environment. Sensor networks have also been successfully used in hazardous environments [1], and showed to provide valuable information about hazard conditions to a response team. There are also implementations of

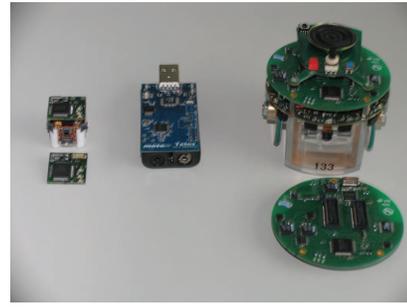


Fig. 2. From left to right: the *Alice* robot with its communication module, the *Telos* [17] platform, and the module with attenuation mounted on the *e-Puck* [16] robotic platform. Unmounted modules are placed in front of the robots.

miniature robotic platforms that have the potential to integrate with a static sensor network [15].

## II. MINIATURE NETWORKED (ROBOTIC) PLATFORMS

In this section we will briefly report on custom developed robotic platforms, which enable integration into large scale embedded sensor networks.

### A. Hardware

We developed embedded communication systems for the *Alice* [9] as well as the *E-Puck* [16] robotic platform. The *Alice* robot has a size of  $2\text{cm} \times 2\text{cm} \times 2\text{cm}$ , is endowed with a PIC micro-controller, 386 bytes of RAM, and four infrared sensors that allow for obstacle avoidance (3cm range), and low bandwidth local communication (6cm range). The *E-Puck* instead has a diameter of 7cm and is endowed with a more powerful processor (dsPIC), and additional hardware (sound generator, speaker, microphones, accelerometer, bluetooth module).

To turn both platforms into a networked robotic system, we constructed a radio board (as shown in Figure 2) with the requirements that it should be low power, as energy is usually the bottleneck on miniature robotic platforms, and that it operates on standardized protocols, so as to be interoperable with our other existing robotic and sensor network platforms running the open-source operating system TinyOS (see below). Figure 3 includes a block diagram illustrating the basic structure of the radio board, which is based on a modified version of the *Telos* (rev. B) [17] schematics provided by MoteIV. The processor is a Texas Instruments MSP430F169 with 2kB of SRAM and 60kB of flash memory (program storage), selected for its attractive energy consumption profile and the existence of a functional TinyOS port to its architecture. The physical radio is a Chipcon CC2420, an IEEE 802.15.4 and *ZigBee* compliant transceiver, which allows us to take advantage of the partial implementation of the IEEE 802.15.4 and *ZigBee* extensions already present in TinyOS. This enables communication between these radio modules and any of our other platforms (e.g., *MicaZ*).

In order to enable large scale robotic experiments involving multi-hop communication in laboratory environments, we equipped the *E-Puck* module (which does not has the size constraints of the *Alice* platform) with a software selectable

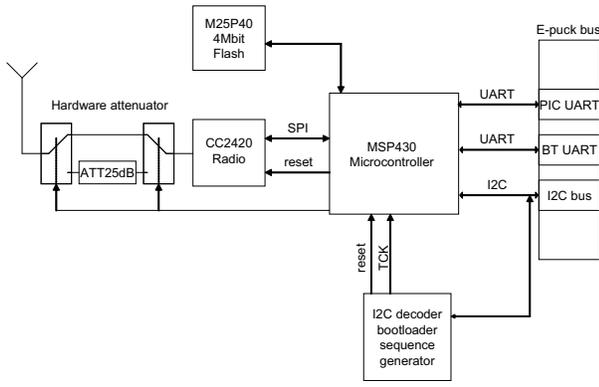


Fig. 3. Block diagram of the communication module for the *e-Puck* based on the *Telos* [17] platform. The *Alice* module does not provide I<sup>2</sup>C functionality.

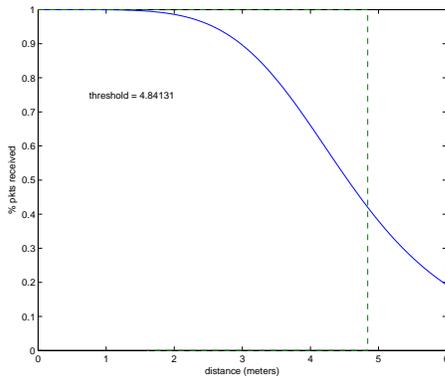


Fig. 4. Range measurements on the communication module with enabled attenuation.

custom attenuation circuit, which was placed between the transceiver and the SMD antenna (Antenna Factor ANT-2.45-CHP) for range reduction (note that this affects both reception and transmission), which complements the ability of the Chipcon radio to operate using different transmission power settings (from 10mW to 20mW). This property could also be desirable for providing a limited range (static) sensor network in our turbine mock-up (Figure 1), which measures only  $60\text{cm} \times 65\text{cm}$ .

For characterizing the transmission reliability of our communication modules we performed a series of tests using communication modules at different ranges (see [18] for details), from which we interpolated the probability of receiving a packet at a certain range. Results for enabled attenuation are given in Figure 4, results without attenuation are qualitatively similar while being distributed over larger ranges.

1) *Software control of the radio board*: Communication between the robotic platform and the communication module is enabled by the serial port of the TI MSP on the *Alice* platform, whereas the module with the attenuation circuit uses its I<sup>2</sup>C bus, which eases interfacing with other modules or robotic platforms. Appropriate primitives were then written and integrated into the radio module’s API for sending and

receiving of messages, as well as the modification of control parameters (such as the transmission power, etc.).

2) *Measurement of physical characteristics*: A number of tests have been run for ascertaining the performance and limits of the device. Preliminary measurements of power consumption indicate that when not in use, the modules draw less than 1.4mW, and with the radio on (ready to receive) and the processor under heavy load, approximately 76.2mW.

The modules achieve reliable communication of around 50m in an office environment (without attenuation), whereas the attenuation allows to tune the range between 0.5m and 4.8m [18].

### B. The TinyOS operating system

TinyOS [19] has been designed for wireless embedded sensor networks at the University of California, Berkeley, and is available open-source [20]. Its architecture is component-based, which allows rapid innovation and implementation as components can be developed and tested independently from each other. Functionality is then implemented by “wiring” components to each other. TinyOS’s component library includes network protocols, distributed services, sensor drivers, and data acquisition tools.

TinyOS’s execution model is event-driven allowing for low power operation, and enforces a programming paradigm which is well suited to distributed systems (notice that the robotic platform is usually driven by a reactive controller on the robot’s CPU).

Due to the open source characteristics and wide spread use of TinyOS in the research community, there exist various packages implementing high-level sensor-network functionality. While most of them have been developed for static sensor nodes, some might find application also for mobile nodes.

For instance, there exist different approaches for self-localization of sensors in the environment, which usually exploit the strength of the RF signal or time of flight measurements of an ultra-sound signal. In [21] accuracy of 2-3m is reported using the software *Motetrack*, and [22] proposes a dedicated sensor *Cricket* which allows localizing itself with respect to static beacons in the environment. Accuracy of ultra-sound time of flight measurement is usually higher than the range information provided by RF, while ultra-sound requires line-of-sight exposure and is more prone to interference than RF signals.

The TinyOS community provides also various packages for implementing multi-hop routing. For instance *Collection* [23] provides a best-effort, multihop delivery of packets to the root of a tree, while *Dissemination* [24] allows for distributing data to every node in the network. The company *MoteIV* maintains the open-source software *Boomerang* [25], which allows reliable mesh networking at reduced duty cycles for reduced power consumption (although at cost of reduced bandwidth).

### C. Lessons learned

Experimenting with various static sensor network platforms showed that energy consumption is still a major bottleneck

when deploying sensor networks that rely on extensive communication, processing, or both. This is in particular the case for the *Alice* platform whose battery is only providing 80mAh and available current might not be sufficient for supporting wireless communication, sensing, actuation, and data processing at the same time. In particular, while the emission power of the Chipcon radio is variable, listening to the channel continuously consumes energy on the same level as during full emission.

Being component based, TinyOS allows indeed for rapid prototyping of applications, whereas adapting it to custom developed hardware is cumbersome, mainly due to the lack of documentation. Although this problem seems to be vanishing with the release of TinyOS 2.0, the developing community might greatly benefit from joint efforts of the robotics and sensor net researchers.

Finally, experimenting with *real* robotic platforms—at least those with size, energy, and computational constraints as described above—showed that sensor and actuator noise as well as unreliable communication (Figure 4) should be taken into consideration for algorithmic design, as it is the case with the self-organized paradigm.

### III. ENGINEERING SELF-ORGANIZED ROBOTIC SYSTEMS

A paramount question in designing self-organized robotic systems is the relation between individual and collective behavior. We approach this problem by modeling the swarm and interactions among its members at various levels of abstractions. Given a distributed system with its practically infinite parameter space, ranging from the individual’s controller, its morphology to features of the environment that influence the swarm, we try to identify key parameters that allow us to describe a particular metric of interest with sufficient accuracy. Following the principle of parsimony (Occam’s razor), we gradually decrease the level of detail at different model abstraction levels, allowing for drastically decreasing the experimental/simulation time at each abstraction step.

At the lowest abstraction level we consider realistic, embodied simulation, which faithfully reproduces body morphology, sensor features and placement, as well as physical constraints of the robots and the environment in a 3D multi-unit simulator Webots [26]. Webots allows for capturing intra-unit details such as body morphology, spatial characteristics (e.g., sensor aperture, range), and noise (e.g., amplitude, distribution). Results obtained with Webots can be considered to come very close to those observed with a real system [12], [26]. More recently, we also implemented a realistic network simulator OmNet++ into Webots [18] that allows taking physical properties of simulated communication channels into account.

At a higher level, we consider multi-agent simulation, where some properties of a real system are intentionally replaced by average quantities. For instance, the agent’s speed together with its sensorial range and the morphology of an object are abstracted by a constant probability for encountering this object at every time step. We refer to this abstraction level as microscopic level. On the microscopic level the state of an

individual agent and the probability to change its state are represented by a Probabilistic Finite State Machine (PFSM) [12], [27]. Results are then obtained by simulating the ensemble of PFSMs, one for each robot.

At the highest abstraction level, we describe the system using rate equations following a mean field approach [12], [28]. Similar to population models commonly used in biology [29], the entire swarm is captured by a set of difference equations, which we refer to as the macroscopic level. We note here that a mean-field approach does not necessarily imply the number of individuals in the swarm to be huge, but rather predicts the mean state of the system over a large number of experiments. This approach has led to qualitative and quantitative agreement between model predictions and real robot experiments, for instance in experiments concerned with distributed manipulation [12], collective assembly of objects [30], or the inspection scenario considered in this paper [6], [27].

### IV. HUMAN-SWARM INTERFACES

In order for humans or high-level agents to interact with a swarm as a whole without bothering about the individual control of its members, techniques have to be developed which provide this synthesis—which is the inverse of the modeling process described above—automatically. In the inspection case study for example, individual agents form a network of sensors whose data needs to be fused and presented to the user as reading of a single sensor (i.e. information about turbine health). On the other hand, tasks have to be defined in terms of swarm rather than individual behavior, raising the need for synthesis methodologies for generating individual behavior out of a given complex behavior at collective level. For instance, a task might be defined in terms of the sensor coverage to be achieved, leading to closed-loop control based on the actual sensor coverage of the swarm (Figure 5).

Although spatial dispersion of robots in the environment has been studied ([31], [32] and references therein) it usually requires localization of the robots in the environment, which is unfeasible for robots with limited capabilities. Here, an embedded network of static sensors (at known locations) might be used for providing crude localization information to the robots and bias their dispersion in the environment. Notice that this coordination mechanism still requires probabilistic modeling techniques, but considerably eases the transition from individual robot control to a human understandable collective metric.

### V. CONCLUSION

Endowing the workspace of minimalist robotic swarms with static sensor networks might allow for generating spatio-temporal patterns at collective level, which were otherwise unfeasible on platforms with limited or none localization and navigation capabilities.

Information provided by the network could be either sensory information for guiding the swarm (for instance for odor source localization, surveillance, or inspection as in the case

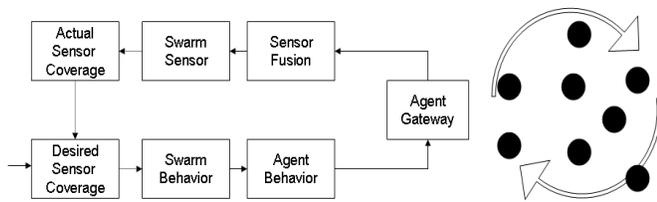


Fig. 5. The swarm is controlled using high-level metrics (here: desired sensor coverage) from which individual robot controllers can be synthesized. Feedback is provided to the operator by appropriate fusion of sensory information.

study in this paper), but can also be injected by a human operator for biasing the swarm's spatial distribution and behavior in a probabilistic fashion. By this the sensor network provides an elegant vehicle for bridging the gap between high-level human mission control and low level reactive control on individual agent level.

Further research is however needed to develop appropriate modeling tools that help understanding design and optimization of large scale distributed systems whose interactions are probabilistic to a large part.

#### Acknowledgments

The authors would like to thank Jonas Fritschy, Peter Brühlmeier, and André Badertscher for their help with developing the radio modules hard- and software. All authors are sponsored by a Swiss NSF grant (contract Nr. PP002-68647).

#### REFERENCES

- [1] V. Kumar, D. Rus, and S. Singh, "Robot and sensor networks for first responders," *PERVASIVE Computing*, October 2004.
- [2] K. Martin and C. Stewart, "Real time tracking of borescope tip pose," *Image and Vision Computing*, vol. 10, no. 18, pp. 795–804, July 2000.
- [3] G. Hunter, "Morphing, self-repairing engines: A vision for the intelligent engine of the future," in *AIAA/ICAS Int. Air & Space Symposium*, 2003.
- [4] J. Litt, E. Wong, M. Krasowski, and L. Greer, "Cooperative multi-agent mobile sensor platforms for jet engine inspection - concept and implementation," in *IEEE Int. Conf. on Integration of Knowledge Intensive Multi-Agent Systems*, 2003, pp. 716–721.
- [5] E. Wong and J. Litt, "Autonomous multi-agent robotics for inspection and repair of propulsion systems," in *AIAA 1st Intelligent Systems Technical Conf.*, 2004.
- [6] N. Correll and A. Martinoli, "Collective inspection of regular structures using a swarm of miniature robots," in *Int. Symp. on Experimental Robotics (ISER)*. Singapore: Springer Tracts for Advanced Robotics (STAR), Vol. 21, June 2006, pp. 375–385.
- [7] —, "Modeling and analysis of beacon-based and beaconless policies for a swarm-intelligent inspection system," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, Barcelona, Spain, April 2005, pp. 2488–2493.
- [8] —, "Comparing coordination schemes for miniature robotic swarms: A case study in boundary coverage of regular structures," in *Int. Symp. on Experimental Robotics (ISER)*, Rio de Janeiro, Brazil, July 2006, to appear in Springer Tracts of Advanced Robotics (STAR) (2008).
- [9] G. Caprari and R. Siegwart, "Mobile micro-robots ready to use: Alice," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, Edmonton, Alberta, Canada, August 2005, pp. 3295–3300.
- [10] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm Intelligence: From Natural to Artificial Systems*. SFI Studies in the Science of Complexity, Oxford University Press, New York, NY, USA, 1999.
- [11] R. Arkin, *Behavior-Based Robotics*, 2nd ed. The MIT press, Cambridge, MA, USA, 2000.
- [12] A. Martinoli, K. Easton, and W. Agassounon, "Modeling of swarm robotic systems: A case study in collaborative distributed manipulation," *Int. J. of Robotics Research*, vol. 23, no. 4, pp. 415–436, 2004.
- [13] R. Pon, M. Batalin, J. Gordon, A. Kansal, D. Liu, M. Rahimi, L. Shirachi, Y. Yu, M. Hansen, W. Kaiser, M. Srivastava, G. Sukhatme, and D. Estrin, "Networked infomechanical systems: A mobile embedded networked sensor platform," in *IEEE/ACM Int. Conf. on Information Processing in Sensor Networks (IPSN-SPOTS)*, 2005, pp. 376–381.
- [14] M. Batalin and G. Sukhatme, "The analysis of an efficient algorithm for robot coverage and exploration based on sensor network deployment," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, Barcelona, Spain, April 2005, pp. 3489–3496.
- [15] K. Dantu, M. Rahimi, H. Shah, S. Babel, A. Dhariwal, and G. Sukhatme, "Robomote: Enabling mobility in sensor networks," in *IEEE/ACM Int. Conf. on Information Processing in Sensor Networks (IPSN-SPOTS)*, Los Angeles, CA, USA, April 2005, pp. 404–409.
- [16] (2006) E-Puck: The EPFL educational robot. [Online]. Available: <http://www.e-puck.org>
- [17] J. Polastre, R. Szewczyk, and D. Culler, "Telos: Enabling ultra-low power wireless research," in *IEEE/ACM Int. Conf. on Information Processing in Sensor Networks (IPSN-SPOTS)*, Los Angeles, CA, USA, April 2005.
- [18] C. 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 *Proc. of the SAB 2006 Workshop on Swarm Robotics*. Rome, Italy: To appear in Lecture Notes in Computer Science (2007), September/October 2006.
- [19] 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)*, 2000.
- [20] (2006) Tinyos Community Forum—An open-source OS for the networked sensor regime. [Online]. Available: <http://www.tinyos.net>
- [21] K. Lorincz and M. Welsh, "Motetrack: A robust, decentralized approach to RF-based location tracking," in *Proceedings of the Int. Workshop on Location and Context-Awareness (LoCA 2005) at Pervasive 2005*, 2005.
- [22] N. Priyantha, A. Chakraborty, and H. Balakrishnan, "The cricket location-support system," in *Proc. of the Sixth Annual ACM Int. Conf. on Mobile Computing and Networking (MOBICOM)*, 2000.
- [23] (2006) Tep 199: The collection protocol. [Online]. Available: <http://www.tinyos.net/tinyos-2.x/doc/html/tep119.html>
- [24] (2006) Tep 118: The dissemination protocol. [Online]. Available: <http://www.tinyos.net/tinyos-2.x/doc/html/tep118.html>
- [25] (2006) Boomerang — low power reliable mesh-networking. [Online]. Available: <http://www.moteiv.com/>
- [26] O. Michel, "Webots: Professional mobile robot simulation," *Journal of Advanced Robotic Systems*, vol. 1, no. 1, pp. 39–42, 2004.
- [27] N. Correll and A. Martinoli, "Modeling and optimization of a swarm-intelligent inspection system," in *Int. Symp. on Distributed Autonomous Robotic Systems (DARS)*. Toulouse, France: Springer Distributed Autonomous Systems VI, 2006, pp. 369–378.
- [28] K. Lerman, A. Martinoli, and A. Galystan, "A review of probabilistic macroscopic models for swarm robotic systems," in *Proc. of the SAB 2004 Workshop on Swarm Robotics*, Santa Monica, CA, USA, 2005, pp. 143–152, Lecture Notes in Computer Science Vol. 3342, Springer-Verlag, Berlin.
- [29] S. Camazine, J.-L. Deneubourg, N. R. Franks, J. Sneyd, G. Theraulaz, and E. Bonabeau, *Self-Organization in Biological Systems*, ser. Princeton Studies in Complexity. Princeton University Press, 2001.
- [30] A. Martinoli, A.-J. Ijspeert, and F. Mondada, "Understanding Collective Aggregation Mechanisms: from Probabilistic Modelling to Experiments with Real Robots," *Robotics and Autonomous Systems*, vol. 29, pp. 51–63, 1999.
- [31] M. Hsieh and V. Kumar, "Pattern generation with multiple robots," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, Orlando, FL, USA, May 2006, pp. 2442–2447.
- [32] 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.