# ROBOMOTE: ENABLING MOBILITY IN SENSOR NETWORKS

*Karthik Dantu, Mohammad Rahimi, Hardik Shah, Sandeep Babel, Amit Dhariwal, Gaurav S. Sukhatme*

Dept of Computer Science,
University of Southern California,
941 W 37th Place,
Los Angeles, CA 90089.
Email: $(dantu|mhr|babel|dhariwal|gaurav)@usc.edu, hardik@gmail.com$

## ABSTRACT

Severe energy limitations, and a paucity of computation pose a set of difficult design challenges for sensor networks. Recent progress in two seemingly disparate research areas namely, distributed robotics and low power embedded systems has led to the creation of *mobile (or robotic) sensor networks*. Autonomous node mobility brings with it its own challenges, but also alleviates some of the traditional problems associated with static sensor networks. We illustrate this by presenting the design of the robomote, a robot platform that functions as a single mobile node in a mobile sensor network. We briefly describe two case studies where the robomote has been used for table top experiments with a mobile sensor network.

## 1. INTRODUCTION

Sensor networks hold the promise of revolutionizing our daily life by ubiquitously monitoring our environment and/or adjusting it to suit our needs. The benefits of this technology have been elaborated at length in the literature [1, 2, 3, 4]. The realization of such networks poses many challenges which are the subject of active research in the field. These include challenges stemming directly from the paucity of computation, storage and energy, systems challenges such as unattended long term function, routing and distributed computation (e.g., for localization, calibration, time synchronization), and finally, challenges associated with the dynamics and spatio-temporal irregularity of the environments within which these networks are expected to function.

As an illustration of the utility of autonomous node mobility within sensor networks we present the hardware and software design of the robomote, a robot platform that functions as a single mobile node in a mobile sensor network. We describe two case studies where the robomote has been used for table top experiments with a mobile sensor network. For design and usage details beyond those described in this paper, the reader is directed to the robomote website `http://robotics.usc.edu/~robomote`.

## 2. APPLICATIONS OF MOBILITY

***Controlled mobility*** enables a whole new set of possibilities in sensor networks. The ability to actively change location can be used to mitigate/solve many of the design challenges outlined above. Before we look at some of the advantages, we would like to clarify that controlled or robotic mobility is different from mobility in the context of Mobile Adhoc Networks (MANETs). MANET studies consider mobility as a given (e.g., as experienced by portable devices such as cell phones or laptops). They study the effects of mobility and assume no control over it. However, controlled or robotic mobility is the ability of a network to move intentionally, and without human assistance. We outline some of advantages of controlled mobility below.

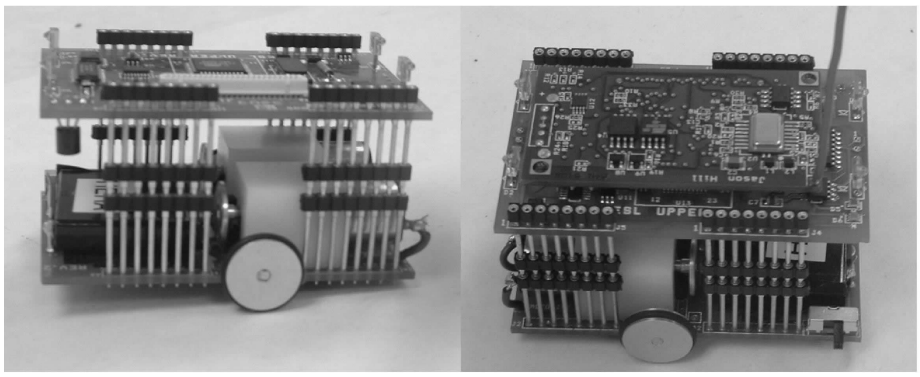
**Deployment:** Controlled mobility can be used to deploy sensors at optimal locations for monitoring. As an example see [5] wherein an algorithm for deploying a fully mobile sensor network has been proposed. We could conceive of a family of algorithms for deployments that have varying ratios of static to mobile sensor nodes that optimize various metrics like maximum coverage, required connectivity [6] etc.

**Adaptive Sampling:** One of the fundamental requirements from a sensor network is to sample its environment. It is possible to think of adaptive sampling strategies where spatial adaptation is by the use of mobility [7] since it is not possible to deploy sensors at all possible sampling locations a priori.

**Network Repair:** One of the main problems of adhoc deployment is the lack of guaranteed connectivity (or a certain level of connectivity) in the network. Such situations can be corrected by having a few mobile nodes which can actively move to desired locations and repair broken networks.

**Energy Harvesting:** Methods have been suggested [8] to use mobile sensor nodes to physically transport energy in the network from areas where it is available in plenty to other regions where energy availability is scarce. Using such techniques self-sustaining networks which rely on sources like solar energy could be built, thereby mitigating the energy constraint which is fundamental to sensor network design.

**Event Detection:** Detecting spatially and temporally spread environmental events is a fundamental function of a sensor network. An initial network deployment might not be the optimal deployment of nodes to detect events in time. We can correct this problem by having mobile nodes that move to regions of high probability of event detection over time. This arrangement could also be dynamic with mobile nodes repositioning themselves [5] as and when required.



(a) Robomote without the mote (b) Robomote with the mote

**Fig. 1**. The robomote

## 3. ROBOMOTE: HARDWARE AND SOFTWARE DESIGN

### 3.1. Introduction

The robomote was designed to be a tabletop platform for experiments in mobile sensor networks. An initial design was presented in [9]. This paper describes the second incarnation of the robomote. The primary design goals for the robomote are ease of deployment, and cost.

Keeping these in mind, the robomote is designed to be compatible with the popular mote platform ([10]). The robomote (Figure 1) consists of an Atmel 8535 microcontroller. This is a 8-bit AVR RISC MCU with 8k bytes of In-system programmable flash along with 512 bytes of EEPROM and 512 bytes of Internal SRAM. The microcontroller also incorporates various desirable features like programmable sleep modes and reprogramming capability. It has two motors, compass for heading and IR sensors. Each of these is described in further detail below. The robomote is complete with the addition of a mote. The mote is used as the master. All basic functionality of the robomote is exported to the mote via modular interfaces. We have also written TinyOS components for the mote to incorporate control of the robomote into TinyOS application. Just as the mote has a radio component and can send out packets, it now has an actuation component and can command motions as needed.

### 3.2. Hardware Architecture

#### *3.2.1. Internal Communication*

Communication between the robomote and the mote is via a serial interface and can achieve speeds of upto 19.2 kbps. There is a byte-to-byte reliability established via acknowledgements from the robomote for every byte received. This helps detect loss of commands and/or data. Each command consists of two bytes - a command and corresponding data. Some commands do not have any data to be passed and are padded (eg: get compass heading).

#### *3.2.2. Motion Control*

The wheels are driven using DC motors from Micromo at 6V and output power 1.41W. The efficiency of the motors is 71% with a no-load speed of 16,300 rpm and a no-load current of 30 mA. The motors are controlled using an H-bridge, made from discrete components, which utilizes Pulse Width Modulation (PWM) for its operation. The four signals which control the motors are PWM1, PWM2, Direction1 and Direction2. By changing the direction bits, the direction of the motors can be reversed. Motion control is triggered by velocity and distance commands from the mote. These commands are converted to the corresponding PWM and direction values. The gear ratio is 25:1. The robomote relies on odometry for movement from one location to another. The feedback uses IR TX/RX mechanism for sensing the number of ticks on the wheel. For one complete revolution of the wheel, 150 ticks are available for feedback control at a least count of 6 ticks (approximately 14 degrees). This is then fed back to the counters of the Atmel Microcontroller. A PI controller that compensates for odometry error has been constructed. It runs at a frequency of 2 Hz and uses the difference in ticks between the left and right wheels as an error signal. The controller outputs are sent as corrections to the PWM signals that are applied to the motors. This controller is bypassed for turn commands and calibration to avoid additional complexity.

#### *3.2.3. Compass*

The compass on the robomote is a 2-axis Honeywell HMC1022 IC. It is configured as a 4-element Wheatstone bridge. Each element is a magneto-resistive sensor which converts a sensed magnetic field to a differential output voltage. From this two analog readings are obtained, one that senses the Earth's North-South field and the other at 90 degrees to the first (i.e. West-East readings). The sensors are not absolute and must be calibrated at robot initialization and periodically throughout usage. This is done by periodically entering a calibration phase where the robomote makes a full turn to detect the maximum and minimum readings and sets them as reference. The compass can be run in two modes: free-running, and single conversion. We use it in a single conversion mode to conserve energy by avoiding usage when not required. When a request for a compass reading is obtained from the mote,

current is passed through the compass. This causes the compass to activate and the azimuth reading is obtained which is passed back to the mote.

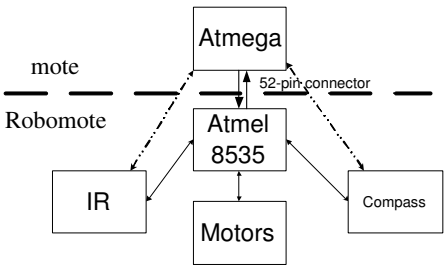



**Fig. 2**. Hardware architecture of the robomote

*3.2.4. IR Proximity Sensors*

The robomote has two infrared transmitters and one receiver on either end. Transmitters produce IR signals of 940nm wavelength which are modulated at 40 KHz over 1.5 KHz in order to reject the maximum amount of ambient light. The receiver has a viewing angle of 40 degrees with 80% sensitivity. The receivers can be selected by using a multiplexer and an interrupt is generated on the controller's external interrupt channel. These sensors are used to detect obstacles in the path of motion of the robomote. Obstacle avoidance runs in the free-running mode. If the received IR signal is above a threshold value, an obstacle is detected and the mote is signalled.

*3.2.5. Rechargeable Battery*

The robomote is provided with a rechargeable Lithium Ion battery and can be charged either by solar energy or by a DC wall charger. The wall charger uses LTC1734 chip from Linear Technologies. The solar charger uses the LTC1734L from Linear technology. For the solar charger, the minimum current supplied to the battery is 50mA, which can be increased by a NV trimmer potentiometer(DS1804). The battery itself is a Li-Ion prismatic rechargeable battery from RENATA. The typical capacity of the battery is 345mAh and it has a nominal voltage range of 3.7V. The mass of the battery is 10.1g. It is interesting to compare robomote

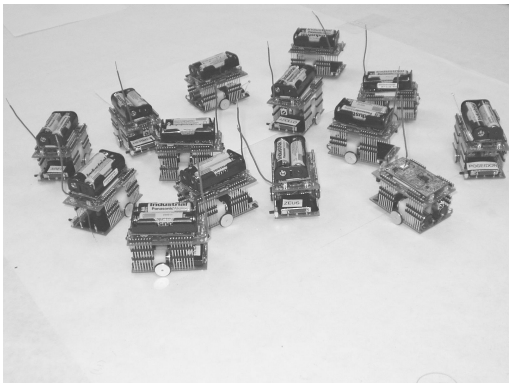


**Fig. 3**. A group of robomotes

**Table 1**. Power consumption of peripherals

| COMPONENT | POWER CONSUMPTION (in mW) |
|---|---|
| Both Motors On | 720 |
| Compass | 60 |
| Both LEDs | 44 |
| All external services off (MCU + MOTE_VDD + Leakage) | 36 |
| Mica2 Processor | 33 |
| Mica2 Radio Reveive | 29 |
| Mica2 Radio Transmit | 42 |

power consumption with the Mica2 mote as described in [11]. Note that actuation (physical movement) draws much more energy than computation or communication(Table 1). Hence, the energy consumption in increasing order is attributable to **Computation** $<$ **Sensing** $<$ **Communication** $<$ **Actuation**. This is an important design consideration to keep in mind while designing algorithms for mobile sensor networks.

### 3.3. Software Architecture

The mote is used as the master. Hence, the system software on the robomote has been designed to provide the basic functionality as modules on the mote. We have designed modules in TinyOS to allow easy access to robomote's functionality for sensor network experimenters. The TinyOS modules consists of two layers. There

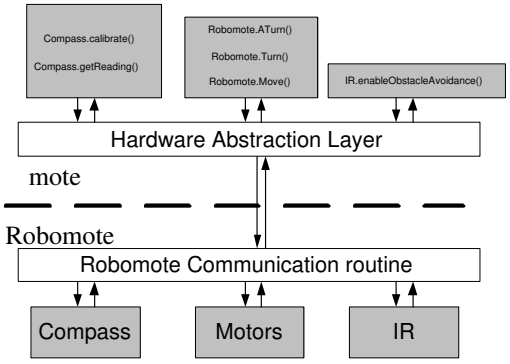



**Fig. 4**. Block diagram of software components

is a *Hardware Abstraction Layer* that handles the communication with the robomote. The second layer exposes the functionality of each individual feature (e.g., compass) of the robomote to applications in TinyOS. It is possible to include just the required components into existing TinyOS applications for ease of understanding and clear design. This is illustrated in Figure 4. Actuation is supported on the mote via three simple commands as shown in Fig. 5. Each command results in an event to notify the completion of the command. Similarly, functionality of the compass and IR are exposed to the mote via simple two-way interfaces as shown in Fig. 6.

command Robomote.Move(uint8_t distance)
– Move straight for distance cm.

command Robomote.Turn(uint8_t angle, uint8_t direction)
– Turn relative to current position in anti−clockwise(direction=1)/clockwise(direction=0) by angle degrees.

command Robomote.ATurn(uint8_t angle)
– Turn to absolute position angle degrees.

event result_t Robomote.MoveDone()
– Signify completion of the move command.

event result_t Robomote.TurnDone()
– Signify completion of the turn command.

event result_t Robomote.ATurnDone()
– Signify completion of the absolute turn command.

**Fig. 5**. Actuation Interface provided by robomote

command result_t Compass.calibrate()
– Calibrate the compass.
event result_t Compass.calibrateDone()
– Signal end of calibration of compass.
command result_t Compass.getData()
– Get the compass reading.
event result_t Compass.dataDone(uint8_t az)
– Signal end of compass reading.
command result_t IR.enableObstacleAvoidance()
– Enable signalling of obstacles.
event result_t IR.obstacle()
– Signal presence of obstacle.

**Fig. 6**. IR sensors and Compass Interface

## 4. CASE STUDIES - EXPERIMENTS USING THE ROBOMOTE AS A MOBILE SENSOR NODE

### 4.1. Detecting Level Sets of Scalar Fields using a Mobile Sensor Network

#### *4.1.1. Problem*

A static sensor network is deployed in a bounded area. The problem is to detect level sets(contours) of the sensed scalar field (e.g., isobars for pressure, isotherms for temperature). A single mobile sensor node is available in addition to the ensemble of static nodes.

#### *4.1.2. Algorithm*

We use a control law proposed in the context of sensor-based path planning [12] for this purpose. The control law has been adapted for the sensor network case. The mobile nodes queries static nodes in its neighborhood and computes the optimal location velocity using field gradient information from each neighbor. Simulation results show that the percentage of success in detecting the desired level set is above 80% for node degrees greater than 7-8 of the static sensor network. A simulation trace of the mobile node's path as it navigates towards the contour of interest is shown in Figure 7.

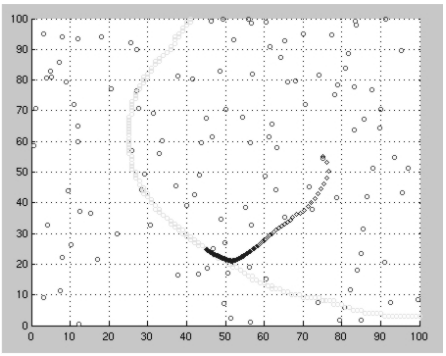



**Fig. 7**. Path of mobile node in a contour detection simulation

#### *4.1.3. Experiments*

We performed experiments on a testbed composed of static nodes (motes), a robomote, and an overhead vision system [13]. These experiments are performed to validate the algorithm in the presence of odometry error. The experiments consisted of a Matlab setup on a PC that simulated node deployment. The robomote would query the PC to read the sensor readings of its immediate neighbors. Based on it, motion control commands were generated in Matlab. These were executed by the robomote on the experimental testbed. This process was iterated until the robomote reached the desired contour (within a margin of error).

Twenty nodes were deployed in a 4ft by 8 ft area and radio range was set to 2.8 feet. The average node degree was approximately 8. Each experiment was repeated five times and the average result is shown in table below. The ratio of traveled distance to optimal is about 1.5. Success percentage for these trials were between 75% and 80%.

| *Optimal Distance* | *Traveled Distance* | *Ratio* |
|---|---|---|
| 5.5 | 8.3 | 1.509 |
| 5 | 6.1 | 1.22 |
| 5 | 7.6 | 1.52 |
| 3 | 5.5 | 1.83 |
| 5 | 7.8 | 1.56 |

**Table 2**. Experimental results on the robomote (averaged over five runs each)

### 4.2. Bacteria inspired robots for environmental monitoring

#### *4.2.1. Overview/Experiment Motivation*

The second case study we describe addresses the problem of locating and tracking a light source using the photo gradient generated by it using mobile sensor nodes (robomote) on a tabletop testbed. We implemented an algorithm based on biased random walk (details in [14]), modeled on taxis in bacteria, for tracking

gradient sources. Using gradient information and a rudimentary motion strategy, the mobile sensor nodes are able to track gradient sources analogous to the manner in which bacteria detect and track potential food sources using locally sensed gradients.

#### 4.2.2. *Experiments with robomote*

We carried out experiments on the robomote to validate our simulation results with actual mobile sensor nodes executing a biased random walk. We used a MICA mote to provide the control commands to the robomote using TinyOS. We used the two basic components `move` and `rotate` for controlling the robomote to carry out the biased random walk. We used a basic sensor board with a photo detector that could sense the light gradient generated by the light source. The position of the robomote on the testbed was tracked using an overhead vision system which captured image frames and passed these to a tracker for data analysis and storage.

Figure 8 shows the magnitude of the distance between the robomote and the light source as a function of time. The figure compares data from the robomote and simulation. The data from experiments agree with the results obtained from the simulation work presented in [14]. We carried out another set of experiments with two equal intensity sources present at the same time at opposite corners of the testbed and started the robomote at distances d (d = 25%,50% and 75% positions on the test bed). We started with only one source switched on initially at t = 0s. At t = 180s. we switched on the second source located at the other end of the testbed. This was followed by switching off the first source completely at t=435s. We repeated the experiment for different values of d (d = 25%,50% and 75% positions on the testbed)). The results (Figure 9) obtained using the robomote were in agreement with our simulation results. Benefits of Mobility Having briefly described two case studies, its worthwhile revisiting the advantages of using controlled mobility.

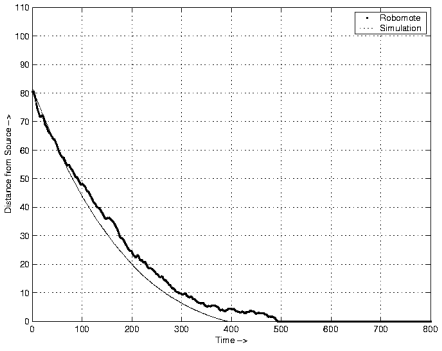



**Fig. 8**. Mobile sensor node moving towards a single source

#### 4.2.3. *Enables new capabilities*

Controlled mobility enables new capabilities. It is evident that both of the above applications would not have been possible without the use of mobility.

#### 4.2.4. *Distibuted Nature*

One of the perceived design constraints of sensor networks is their distributed nature. In our first case study, the mobile node queries only its neighbors. This scheme reduces the communication overhead among static sensor nodes. Querying neighbors depends only on the density of the deployed network and not on the total nodes deployed. Thus such algorithms are scalable in comparison to techniques where all or some fraction of the data from the network needs to be pulled out. Note however that actuation has an energy cost. The tradeoffs between energy saving by reducing communication to energy expense due to actuation make for more detailed study on a per case basis.

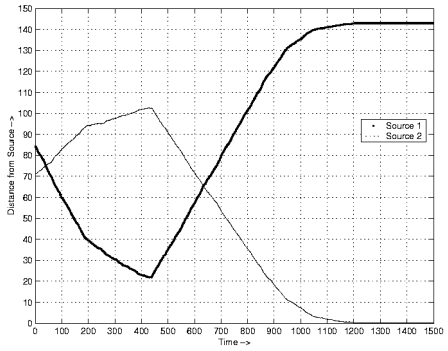



**Fig. 9**. Mobile sensor node trajectory with multiple sources

#### 4.2.5. *Deployment and Coverage*

Another design constraint of sensor networks is the problem of deployment and coverage. Most sensor networks are deployed to monitor events in their environment. Due to a variety of factors, the pattern of occurances of these events might vary in time and space. This requires the sensors to redeploy themselves suitably. Consider the first case study. Assume that the level set being sought was the event of interest. Upon execution of the algorithm, the mobile nodes would redeploy themselves around the event of interest. This is an example of dynamic redeployment of the sensor network adapting to changes in the environment.

#### 4.2.6. *Limited Resources*

One other constraint of sensor networks is limited resources. The algorithms used in both the case studies above retain minimal state. Their operation works in two steps - query sensors, and actuate. In this we draw inspiration from a legacy of reactive robotics literature where thin connections between sensing and actuation have been shown to be surprisingly robust and powerful [15].

In the two case studies presented here, there is no retention of state. Hence, memory utilization is minimal, and computation is simple enough to run easily on the mote hardware. Thus these two case studies illustrate that it is possible to use mobility with limited resources to implement useful applications.

As can be seen, there are numerous benefits of using controlled mobility in sensor networks. However, there are many challenges also. These are elaborated in the next section.

## 5. CHALLENGES IN ENABLING MOBILE SENSOR NETWORKS

**Adaptive Localization:** There has been a lot of research in the area of probabilistic localization in robotics off late [16]. Incorpo-

rating mobility into sensor networks would need distributed lightweight implementations of such algorithms to implement localization in sensor networks.

**Coverage:** Maximizing coverage [17] in sensor networks using static and mobile nodes has received some attention. However, there has not been much work on mobile sensor networks and how they could be used to adapt networks by varying coverage dynamically.

**Massive Reprogramming:** Massive reprogramming of sensor networks is one of the envisioned problems [18]. It is possible to consider solutions using mobile nodes that travel across the geography of the sensor network, reprogramming parts of it.

**Distributed Calibration:** Another hard problem in sensor networks is calibrating the sensors, particularly when the sensors used are cheap and erroneous. We can think of having a calibrated sensor on a mobile node and the mobile node covering the area of sensor node deployment calibrating the nodes in its neighborhood.

**Network Repair:** An interesting area of work is that of network repair. As mentioned earlier, it can be imagined that a few mobile nodes can be used to repair static networks by positioning themselves at hotspots or points of disconnection. However, moving the mobile nodes expends energy and there is scope for study of the tradeoff. This also makes for interesting theoretical study for optimal algorithms to move the mobile nodes.

## 6. CONCLUSIONS

We presented the robomote, a mobile robotic testbed for mobile sensor network experiments. We also presented two case studies where the robomote was used to experimentally validate algorithms designed for next generation mobile sensor networks. Finally, we enumerated some of the potential issues that need to be resolved in order to enable mobility in sensor networks.

## 7. REFERENCES


[1] Joseph Kahn, Randy Katz, and Kris Pister, "Next century challenges: Mobile networking for smart dust," in *Proceedings of Mobile Computing and Networking*. ACM.

[2] Deborah Estrin, Ramesh Govindan, and John Heidemann, "Embedding the Internet," *Communications of the ACM*, vol. 43, no. 5, pp. 39–41, May 2000, (special issue guest editors).

[3] Greg Pottie and William J. Kaiser, "Wireless integrated network sensors," *Communications of the ACM*, vol. 43, no. 5, pp. 551–8, May 2000.

[4] National Research Council Staff, *Embedded Everywhere: A Research Agenda for Networked Systems of Embedded Computers*, National Academy Press, 2001.

[5] Andrew Howard, Maja Mataric, and Gaurav S. Sukhatme, "Self-deployment algorithm for mobile sensor networks," *Autonomous Robots - Special Issue on Intelligent Embedded Systems*, vol. 13, no. 2, pp. 113–126, 2002.

[6] Sameera Poduri and Gaurav Sukhatme, "Constrained coverage for mobile sensor networks," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Apr. 2004.

[7] Maxim Batalin, Mohammad Rahimi, Yan Yu, Duo Liu, Aman Kansal, Gaurav Sukhatme, William Kaiser, Mark Hansen, Gregory J Pottie, Mani Srivastava, and Deborah Estrin, "Call and response: Experiments in sampling the environment," in *Proceedings of 2nd Annual Conference on Sensors and Systems (Sensys 2004)*, Baltimore, MD, USA., November 2004, ACM.

[8] Mohammad H. Rahimi, Hardik Shah, Gaurav S. Sukhatme, John Heidemann, and Deborah Estrin, "Energy harvesting in mobile sensor networks," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Taipei, Taiwan, September 2003.

[9] Gabriel T. Sibley, Mohammad H. Rahimi, and Gaurav S. Sukhatme, "A tiny mobile robot platform for large-scale sensor networks," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Washington DC, USA, May 2002.

[10] Jason Hill, Robert Szewczyk, Alec Woo, Seth Hollar, David Culler, and Kris Pister, "System architecture directions for networked sensors," in *Proceedings of Architectural Support for Programming Languages and Operating Systems-IX*, Cambridge, MA, USA., Nov. 2000, ACM.

[11] Joseph Polastre, Robert Szewczyk, Cory Sharp, and David Culler, "The mote revolution: Low power wireless sensor network devices," in *In Proceedings of Hot Chips 16: A Symposium on High Performance Chips*, August 2004.

[12] Howie Choset, Ilhan Konukseven, and Alfred Rizzi, "Sensor based planing: A control law for generating the generalized voronoi graph," in *IEEE International Conference in Advanced Robotics*, 1997.

[13] Mohammad Rahimi, Rohit Mediratta, Karthik Dantu, and Gaurav Sukhatme, "A testbed for experiments with sensor/actuator networks," Tech. Rep. IRIS-02-417, Institute for Robotics and Intelligent Systems, 2002.

[14] Amit Dhariwal, Gaurav S. Sukhatme, and Aristides A. Requicha, "Bacterium-inspired robots for environmental monitoring," in *IEEE International Conference on Robotics and Automation*, April 2004.

[15] Rodney Brooks, "A robust layered control system for a mobile robot," *International Journal of Robotics and Automation*, vol. 2, pp. 14–23, 1986.

[16] Sebastian Thrun, Dieter Fox Wolfram Burgard, and Frank Dellaert, "Robust monte carlo localization for mobile robots," *Artificial Intelligence (AIJ)*, 2001.

[17] Maxim Batalin and G. S. Sukhatme, "Coverage, exploration and deployment by a mobile robot and communication network," in *Proceedings of the 2nd International Workshop on Information Processing in Sensor Networks*, Palo Alto Research Center (PARC) Palo Alto, CA, USA, April 2003, pp. 376–391.

[18] Jonathan Hui and David Culler, "The dynamic behaviour of data dissemination protocol of network programming at scale," in *Proceedings of second annual conference in Sensors and Systems (Sensys)*.