

Sensor Coverage using Mobile Robots and Stationary Nodes

Maxim A. Batalin^a and Gaurav S. Sukhatme^a

^aUniversity of Southern California, Robotics Research Lab, Los Angeles, CA 90089, USA

ABSTRACT

We consider the dynamic sensor coverage problem in the absence of global localization information. In the regime where few sensors are available compared to the size of the space being explored, a successful strategy must effectively mobilize the sensors by mounting them on mobile robots. We present here an approach where mobile robots explore an uncharted environment, by deploying small communication beacons. These beacons act as local markers for preferred directions of further exploration. The robots never acquire a map of their surroundings, nor are localized, however they ensure timely coverage of all regions of the space by relying on the local instructions disseminated by the stationary communication beacons. Preliminary data from experiments suggest that our algorithm produces exploratory, *patrol-like* behavior, resulting in good spatial sensor coverage over time.

Keywords: Sensor network, static network, mobile network, mobile robots, coverage, exploration, patrolling

1. INTRODUCTION

Coverage¹ is the problem of arranging sensors in the environment, usually with the aim of detecting targets. The target-detection sensors are often mounted on mobile robots, thereby reducing the problem to one of robot positioning. Such a capability is of obvious use in the detection of unfriendly targets (e.g. military operations), monitoring (e.g. security), or urban search and rescue (USAR) in the aftermath of a natural or man-made disaster (e.g. building rubble due to an earthquake or other causes). Often, in such scenarios, global knowledge about the environment (e.g. a map) and robots (e.g. their locations) is missing or difficult to obtain. We focus on the design of 'minimalist' algorithms for the dynamic blanket coverage problem under the fundamental constraint that global knowledge about the environment (e.g. Global Positioning System or a map) is unavailable.

We consider the sensor coverage problem as a problem of sensor network deployment (static and/or mobile). We present a brief overview of our previous work that suggests several different ways of approaching the problem and concentrate the main discussion on the latest variant of the work. The first approach, described in,² deploys a mobile sensor network by dispersion and then applies local rules for sensor coverage maximization. The premise of this algorithm is that in order to achieve good coverage as a team, robots must 'spread out' over the environment, i.e. if robots are too close to each other, their coverage areas overlap resulting in poor overall coverage. This premise is loosely inspired by the diffusive motion of fluid particles. Thus, robots not only perform obstacle avoidance, but are mutually repelled by each other within the range of their sensors. The approach depends on the ability of a robot to distinguish another robot from other objects in its environment. In addition, the motion of every robot is guided by its perceived coverage area, that is individually, robots try to move in the direction of coverage maximization. This is a local greedy approach. We have shown² that the nature of the interaction needed between robots is very simple and the overall design is minimalist.

Our second approach,³ on the other hand, assumes that the coverage requirements are such that every point in the environment should be covered with certain frequency. The approach deploys static and mobile sensor networks and produces exploratory, *patrol-like* behavior. Robots deploy communication beacons into the environment to mark previously visited areas. These nodes act as local signposts for robots, which subsequently return to their vicinity. By deploying such (stationary) nodes into the environment, robots can make local decisions about their motion strategy. In the design of both systems we stress the general idea that local rules, minimalist and decentralized approach are the essential ingredients needed for creating reliable and

Author's e-mail: maxim@robotics.usc.edu

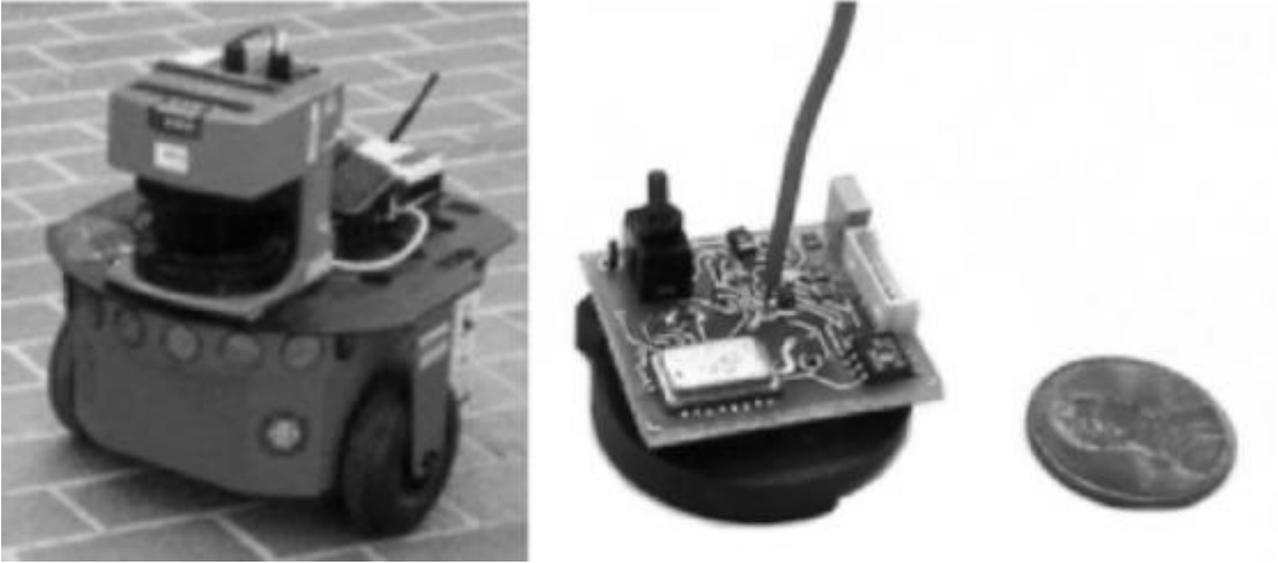


Figure 1. Hardware platforms: The Pioneer mobile robot and the mote beacon

robust coverage algorithms. The third approach, on which we focus here, is a variant of the second algorithm summarized above.

In this paper we briefly describe the approach and report on preliminary experimental validation of our approach. Figure 1 shows the two experimental platforms used - Pioneer 2DX mobile robot and a mote (used as a beacon).

2. RELATED WORK

The coverage paradigm was formulated by Gage¹ and divided into three groups of useful behaviors. Blanket (or Field) coverage, that aims to achieve a static arrangement of agents to maximize the detection rate of the targets in the sensor shadows. Barrier coverage, whose objective is to achieve a static arrangement of agents with the task of minimizing the probability of undetected target penetration through the barrier. Sweep coverage, that essentially represents a moving barrier coverage or can be achieved using random uncoordinated motion of agents (as shown in¹). Our first approach² proposed a solution to the Blanket coverage problem whereas the second³ and third approaches are proposed solutions to the sweep coverage problem.

The problem of dynamic coverage (or sweep coverage) is also related to the exploration problem in an unknown environment which has been studied by several authors.⁴⁻⁶ The frontier-based approach, described in detail in,^{4,5} concerns itself with incrementally constructing a global occupancy map of the environment. The map is analyzed to locate the ‘frontiers’ between the free and unknown space. Exploration proceeds in the direction of the closest frontier. The multi-robot version of the same problem was addressed in.⁷⁻⁸ discusses the problem of deployment of distributed sensors (robots) in the wireless adhoc network domain. In their setup, the communication ranges between the robots are assumed to be limited and the environment is assumed to be big enough so that the network connectivity cannot be maintained. A random-walk algorithm is used to disperse the robot network into the environment to support communication.

The approach presented in this paper differs from the above mentioned approaches in a number of ways. We use neither a map, nor localization in a shared frame of reference. Despite the similarity of the idea of dispersion, our algorithm differs from,⁸ since every robot performs local visibility maximization rather than a random walk. In our algorithm robots deploy a set of communication beacons into the environment in order to coordinate their motions to improve dynamic coverage. A basic assumption is that the beacon nodes that the robot deploys into the environment are small relative to the size of the robot and the environment.

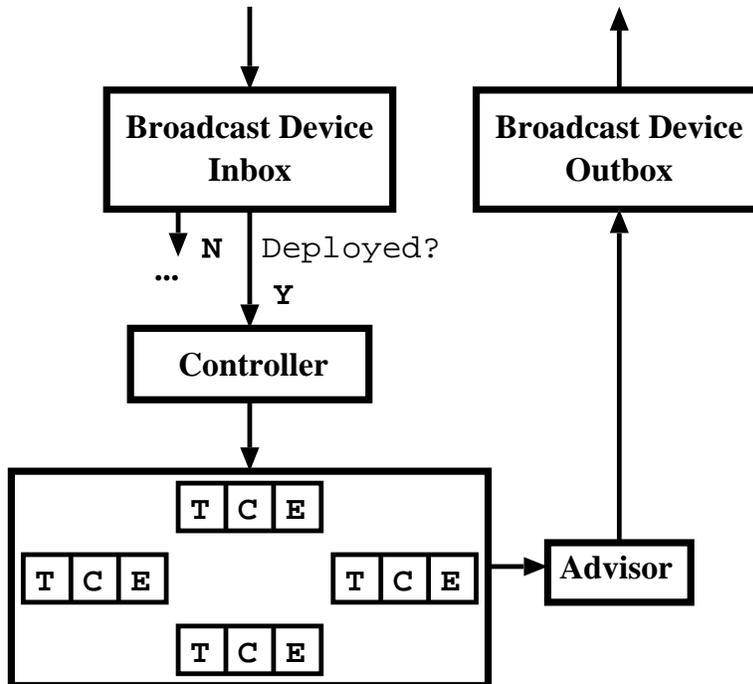


Figure 2. Communication Beacon Architecture

3. SYSTEM ARCHITECTURE

For corresponding description of architectures for the first two approaches we developed, the reader is referred to.^{2,3} The third algorithm, which we focus on here, uses three entities: the communication beacons, the mobile robots and the *Base Station* beacon (BSB) which is attached to every mobile robot and allows the robot to communicate with the communication beacons. The task of each communication beacon is to recommend a preferred exploration direction for robots within its communication range. Each beacon issues only a recommendation, robots combine this advice with local sensing to make a decision about which direction to actually pursue. Each robot is equipped with a 2D laser range finder with which it performs the sensor coverage task. Each robot is also equipped with the BSB and a number of communication beacons that it deploys into the environment to help with coverage.

As shown in Figure 2, each communication beacon consists of a *BroadcastDevice in/outbox*, *Controller*, *States block* and *Advisor block*. If the beacon has been deployed, messages are directed to the *Controller*, which parses them and updates the *States block*. The *States block* consists of four groups of states corresponding to the four directions (South, East, North, West). Note that these four directions represent an abstract notion of a set of any four perpendicular directions chosen for the environment. Every group has three states. The state T denotes whether a direction is OPEN or EXPLORED, C is a counter (if T is EXPLORED, then C counts the time since last update), and E is an extra field for network information propagation (direction of goal/home state, sensor readings propagation, etc.). The *Advisor block* computes the beacon's recommendation for the best action a robot should take if it is within the beacon's communication range. The computation of the recommendation is simple. All OPEN directions are recommended first (in order from South to West), followed by the EXPLORED directions with largest last update value (largest C value). The beacon's recommendation, generated by the *Advisor block*, is sent to the *BroadcastDevice outbox* for broadcast to nearby robots.

As shown in figure 3 the BSB consists of *Broadcast in/out*, *Parser* and *Serial Port in/out*. As mentioned earlier, the BSB serves as a "translator" between the mobile robot and communication beacons. The BSB is connected to the serial port of the mobile robot. Hence, whenever we say that the mobile robot sends a message

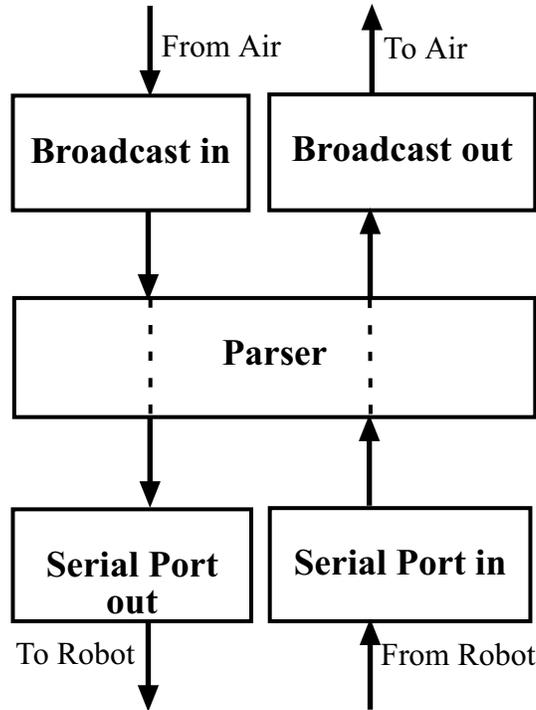


Figure 3. Base Station Beacon (BSB) Architecture

to beacons, it is actually being passed through the serial port to the *Serial Port in*, then to *Parser*, which verifies that the message packet has the correct format. Finally, the message is sent to the *Broadcast out* and broadcasts to nearby communication beacons. Alternately, whenever we say that the beacon sends a message to the robot, the message is first received by the BSB through *Broadcast in*, then checked for correctness in *Parser* and finally directed to the *Serial Port out* where the robot reads the message and acts accordingly. Note that in hardware we use the same device (a mote) for implementation of communication beacon and BSB, therefore, there is no need to create a special device from a hardware point of view.

Each mobile robot is programmed using a behavior-based architecture.⁹ Unlike our prior approach,³ one Laser and Compass are the only sensors used. Arbitration¹⁰ is used for behavior coordination. Priorities are assigned to every behavior *a priori*. As shown in Figure 4, there are four behaviors in the system: *Obstacle Avoidance*, *AtBeacon*, *DeployBeacon* and *SearchBeacon*. In addition to priority, every behavior has an activation level, which decides, given sensory input, whether the behavior should be in an active or passive state (1 or 0 respectively). Each behavior computes the product of its activation level and corresponding priority and sends the result to the Controller, which picks the maximum value, and assigns the corresponding behavior to command the Motor Controller for the next cycle.

One of the state variables that every robot keeps track of, is a reference to the last heard beacon. If this reference switches to a different beacon (i.e. the robot moved to the communication area of a different beacon), *AtBeacon* is triggered. *AtBeacon* analyzes data messages received from the current beacon broadcasts and orients the robot along the suggested direction. After the robot has been oriented in a new direction, it checks its laser readings for obstacles. If the scan does not return any obstacles, the robot proceeds in the suggested direction, while sending an update beacon message (this message updates the States block in Figure 2). If, however, the suggested direction is obstructed (something is in the way), *AtBeacon* sends a broadcast message updating the beacon with new information and requesting a new suggestion. *Obstacle Avoidance* has the same general implementation as described in earlier work.³ One addition, however, is that if the robot is avoiding an obstacle, then the laser scans the area right in front of it for obstacles. If this area contains obstacles then

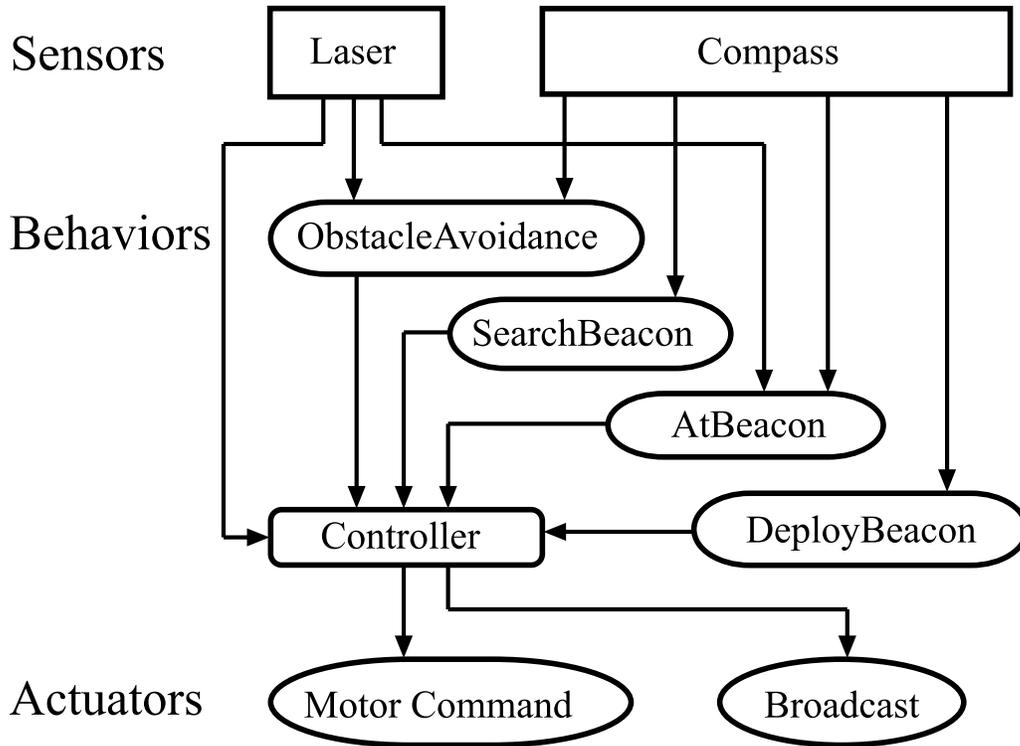


Figure 4. System Architecture

the direction of the obstacles is compared against the direction that the current beacon suggests. If the two directions approximately match, the message requesting a new direction suggestion from the beacon is sent. Therefore, unlike our previous work,³ both *AtBeacon* and *Obstacle Avoidance* can request a new direction suggestion, which allows system to avoid conflicts between *AtBeacon* and *Obstacle Avoidance*. *SearchBeacon* is triggered after *AtBeacon* chooses and positions the robot in a certain direction. The task of *SearchBeacon* is to travel a predetermined distance. *DeployBeacon* is triggered if the robot does not receive a "suggestion" message (i.e. a recommended direction to traverse) from any beacon after a certain timeout value. In this case the robot deploys a new beacon into the environment.

The approach to the coverage problem taken by us in prior work² is that the robots repel each other by detecting each other visually and computing the direction 'away' that would maximize the spread or dispersion of robots into the environment. When not repelling, each robot applies local rules that allows it to maximize local coverage. Several techniques are used for breaking the problems of local minima/maxima.

The fundamental idea of the present algorithm is simple. A robot explores as long as there are OPEN regions left. If all regions are EXPLORED, then the robot picks the direction, which was least recently explored (has the least C value). As discussed in Section 3, decisions of which direction to explore next are made by beacons. The robots, however, may alter those decisions if real world observations (through laser data analysis) diverge from the beliefs of beacons.

As mentioned previously in the present approach the robot relies only on its Compass, whereas previously^{2,3} we have used odometry to compute distance traveled as a cue to decide when to deploy a beacon. In the present approach a new beacon is deployed only if the communication link between the BBS and previously deployed beacon breaks, which signifies that the robot might be out of range of the communication beacon and the network is disconnected.



Figure 5. Snapshots of the experimental validation: a robot deploys beacons into the environment

4. PHYSICAL EXPERIMENT

The experimental results^{2,3} of our first two approaches were documented previously. These were conducted in simulation, using the Player/Stage engine^{11,12} and the following list of sensors: two 180° field of view planar laser range finders positioned back-to-back (equivalent to a 2D omnidirectional laser range finder), color camera, vision beacons. All robots were equipped with wireless communication, compass and the odometry. In addition, the laser was used as a sensor for coverage estimation and obstacle avoidance at the same time. In other words the laser range finder represents a unified sensor for the purposes of safe navigation and coverage estimation.

The present approach uses the same principles as the second algorithm, but minimizes the set of sensors to one 180° field of view planar laser range finder and a compass. The main goal of this approach is to show that the coverage and deployment task can be accomplished with a limited set of sensors.

We have conducted a physical experiment using the robot platforms mentioned in Section 1. For the purposes of the experiment, each beacon's transmission power was lowered to allow 1500mm radio range. However, due to the noise in beacon's radio signal (especially with such a low transmitting power) the actual radio range varied greatly. The experiment yielded positive results, that is the environment was covered with static sensor network and despite the noise in the system the robot was able to navigate using the deployed network. Two screen shots of the experiment are shown on Figure 5. This preliminary validation is encouraging and future experimental work will be designed to study the behavior of the algorithm under varying conditions.

5. DISCUSSION AND FUTURE WORK

This paper gave an overview of three algorithms which aim to solve the problem of sensor coverage by deploying a sensor network (static and/or mobile). If the task requirement is that every point should be covered with certain frequency. then the first algorithm performs well in the case of the environment saturated with robots. On the other hand, if the number of robots is very small compared to the size of the environment, then the second and third approaches do better, since the exploration is preferred to static sensor maximization.

It is also important to note that all three mentioned algorithms are minimalist, reliable and decentralized (destroying one robot does not collapse the performance of the algorithm, given that there is at least one more

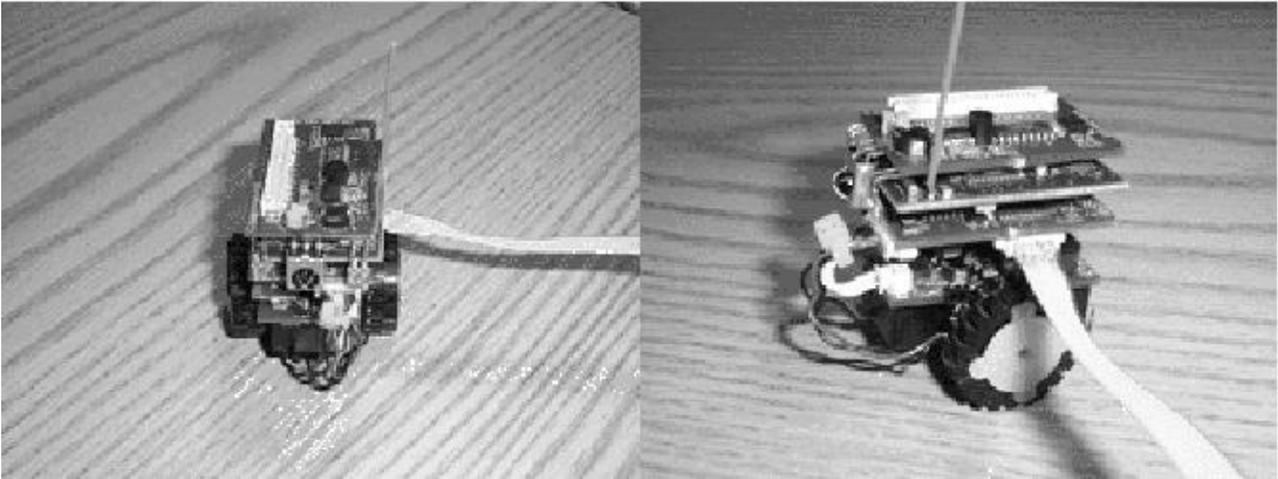


Figure 6. The Robomote platform

robot left). Algorithms two and three rely on the deployment of beacons into the environment as support infrastructure which the robots use to solve the coverage problem. Robots explore the environment, and based on certain local criteria, drop a beacon into the environment, from time to time. In addition, algorithm three requires dropped beacons to be connected and parameters of algorithm two can be set to provide the same outcome. As a result, a static sensor network is deployed, which can be used for numerous applications other than coverage (remember the E field in communication beacon - Figure 2). Note also that the deployed network is connected. Therefore, in general, any network application that requires information propagation along a certain path in the deployed network can be implemented. In addition, the communication ranges of beacons do not have to be the same for the algorithm to work. Therefore, in future work we plan to extend the algorithm to a case where the communication radius would be tuned according to the structure of the environment. In other words, if the environment does not have any obstacles, then the radio strength can be at its maximum. On the other hand, if the environment is highly obstructed, then it would be beneficial for the performance of the coverage algorithm to drop beacons at smaller ranges to allow a finer exploration.

Approaches two and three assume that the number of beacons available to a robot is infinite (even though in practice, of course, only a finite number is needed/used). We plan to extend our algorithm with switching from exploration mode (search for OPEN directions) to patrolling mode (follow the EXPLORED directions) in case the robot runs out of beacons. In future work we also plan to exploit the deployed static sensor network for other behaviors. One example is recovery, when after being deployed, every robot uses the network to return to "home base", the task that is considered to be difficult in coverage problems with limited or no global information. The propagation of information through the network could also dramatically increase performance of the coverage algorithm itself (e.g. by dynamically adjusting the beacon drop-off distance).

We deployed a static sensor network (the motes) and mobile sensor network (the Pioneer robots). In the future experiments, instead of motes we plan to use motes on the mobile platform (Robomotes)¹³ shown in Figure 6, which would allow us to broaden the spectrum of applications the algorithm can be applied to.

ACKNOWLEDGMENTS

This work is supported in part by DARPA grant DABT63-99-1-0015 and NSF grants ANI-0082498 and IIS-0133947. We thank Srikanth Saripalli for assistance with the physical experiments.

REFERENCES

1. D. W. Gage, "Command control for many-robot systems," in *the Nineteenth Annual AUVS Technical Symposium*, pp. 22–24, (Huntsville, Alabama, USA), 1992.
2. M. A. Batalin and G. S. Sukhatme, "Spreading out: A local approach to multi-robot coverage," in *Proc. of 6th International Symposium on Distributed Autonomous Robotic Systems*, (Fukuoka, Japan), 2002.
3. M. A. Batalin and G. S. Sukhatme, "Multi-robot dynamic coverage of a planar bounded environment," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (Submitted)*, 2002.
4. B. Yamauchi, "Frontier-based approach for autonomous exploration," in *In Proceedings of the IEEE International Symposium on Computational Intelligence, Robotics and Automation*, pp. 146–151, 1997.
5. B. Yamauchi, A. Schultz, and W. Adams, "Mobile robot exploration and map-building with continuous localization," in *In Proceedings of the 1998 IEEE/RSJ International Conference on Robotics and Automation*, 4, pp. 3175–3720, 1998.
6. A. Zelinsky, "A mobile robot exploration algorithm," in *IEEE Transactions on Robotics and Automation*, 8(2-3), pp. 707–717, 1992.
7. W. Burgard, D. Fox, M. Moors, R. Simmons, and S. Thrun, "Collaborative multirobot exploration," in *Proc. IEEE ICRA*, 2000.
8. A. F. Winfield, *Distributed sensing and data collection via broken ad hoc wireless connected networks of mobile robots*, vol. 4, pp. 273–282. Springer, 2000.
9. M. J. Mataric, "Behavior-based control: Examples from navigation, learning, and group behavior," *Journal of Experimental and Theoretical Artificial Intelligence, special issue on Software Architectures for Physical Agents* 9(2-3), pp. 323–336, 1997.
10. P. Pirjanian, *Behavior Coordination Mechanisms - State-of-the-art*. PhD thesis, University of Southern California, October 1999.
11. B. P. Gerkey, R. Vaughan, K. Stoy, A. Howard, G. Sukhatme, and M. Mataric, "Most valuable player: A robot device server for distributed control," in *IEEE/RSJ Intl. Conf. On Intelligent Robots and Systems (IROS)*, (Wailea, Hawaii), 2001.
12. R. Vaughan, "Stage: a multiple robot simulator," Tech. Rep. IRIS-00-393, Institute for Robotics and Intelligent Systems, University of Southern California, 2000.
13. G. T. Sibley, M. H. Rahimi, and G. S. Sukhatme, "Robomote: A tiny mobile robot platform for large-scale sensor networks," in *IEEE International Conference on Robotics and Automation ICRA2002*, (Washington DC, USA), 2002.