

# Scalable Control of Distributed Robotic Macrosensors

Brian Shucker and John K. Bennett

Department of Computer Science  
University of Colorado at Boulder  
{shucker, jkb}@cs.colorado.edu

## Abstract

This paper describes a control mechanism by which large numbers of inexpensive robots can be deployed as a distributed remote sensing instrument, and in which the desired large-scale properties of the sensing instrument emerge from the simple pair-wise interactions of its component robots. Such sensing instruments are called *distributed robotic macrosensors*. Robots in the macrosensor interact with their immediate neighbors using a virtual spring mesh abstraction, which is governed by a simple physics model. By carefully defining the nature of the spring mesh and the associated physics model, it is possible to create a number of desirable global behaviors without any global control or configuration. Properties of the resulting macrosensor include arbitrary scalability, the ability to function in complex environments, sophisticated target tracking ability, and natural fault tolerance. We describe the control mechanisms that yield these results, and the simulation results that we have used to evaluate its efficacy.

## 1. Introduction

Robots are currently used in a broad range of remote sensing applications. The use of robotic platforms to gather information provides several advantages: robots can be used in hazardous situations without risk of injury to humans; robots are capable of reaching areas that are infeasible to reach with a human observer; and robots can be inexpensive to operate. In many domains, e.g., space exploration and military reconnaissance, robots are an effective tool for discovering and tracking targets of interest.

Advances in integration, actuator design and power management have resulted in the availability of mass-produced, inexpensive robotic components. It is now possible to build small, autonomous robots in large numbers and at relatively low cost. The resulting potential to deploy robotic sensors on a large-scale creates the opportunity to explore a new type of remote sensing. Hundreds, or thousands, of robots can potentially be deployed to cover and explore an area, record data, and track targets of interest. However, while it is now relatively simple to deploy large numbers of robotic sensors, the coordination and control of the activities of these robots presents a number of challenging problems, including scalability, autonomy, coverage, flexibility of deployment, fault-tolerance, and security.

We have developed a control algorithm for distributed robotic macrosensors (DRMs) to address these concerns. DRMs consist of large numbers of robots whose individual “instinctive” behavior directs them toward a common goal. The properties of the macrosensor emerge as a result of simple local interactions between individual component robots. By defining these interactions carefully, we create a scalable macrosensor with sophisticated overall behavior. The resulting properties of these macrosensors include the following:

1. Arbitrary scalability - A single macrosensor may contain any number of robots, from a single robot to tens of thousands of robots.
2. Automatic operation - deployment and operation is fully automatic; that is, there is no global control (or even global knowledge from the standpoint of the control algorithm).
3. Starting state independence - The macrosensor can deploy from an arbitrary starting state. For example, robots may or may not start together in a cluster.
4. Exploration - Prior knowledge of the target environment is not required. The macrosensor explores unknown areas automatically, mapping the environment as it extends its deployment.
5. Coverage - The macrosensor deploys in such a way as to efficiently cover the area of interest, regardless of the number of robots deployed.
6. Target tracking and mapping - The macrosensor tracks moving targets within the area of interest. Targets may be discrete (such as a vehicle or person), or diffuse (such as a chemical cloud or fire).
7. Fault tolerance - The macrosensor's overall capabilities degrade gracefully with the loss of an arbitrary number of robots.
8. Extendibility - The macrosensor's capabilities increase gracefully with the addition of new robots of the same or different type.
9. Security - Compromise of individual robots, even in significant numbers, results in little gain for an adversary.

By comparison, centralized control mechanisms do not scale adequately, and existing distributed control mechanisms have serious practical limitations, especially with respect to complex environments and unknown initial distributions. Many existing control methods do not allow new robots to be added easily to a system that is already deployed. To our knowledge, robotic tracking of diffuse targets (as opposed to tracking of discrete objects) has not been addressed previously. Security has been addressed in the related context of sensor networks[6], but existing sensor network security protocols are still in the research stage, and do not address all of the relevant issues pertaining to mobile robots. Our work is intended to overcome these limitations.

We envision a variety of applications for distributed robotic macrosensors. Search and rescue operations can benefit from a mobile, rapidly deployable sensor capable of covering a wide area. Such sensors could discover persons in need of assistance, or map sources of danger such as wildfires. Macrosensors can also be deployed in order to quickly secure an area, or to detect and track intruders or other threats. Large numbers of tiny robots could be dropped from an aircraft or vessel in order to collect information about an area that is difficult to reach. For example, such deployments could be used to monitor a border. These macrosensors could be deployed rapidly into a new area because the robots' inherent mobility removes the need for precise manual placement.

Macrosensors also have significant scientific potential. For example, a macrosensor could perform continuous, high-resolution mapping of atmospheric or oceanic phenomena. Unlike a static sensor network, the nodes in a robotic macrosensor network need not be manually placed, so the macrosensor may be deployed rapidly in order to monitor an emergent event. Macrosensors can also be deployed in areas that are largely unreachable by humans, such as the surface of Mars.

## 2. Virtual Spring Mesh Based Control

Our approach to deployment and coordination of DRMs employs a virtual spring mesh as the underlying control mechanism. Virtual spring meshes are an extension of virtual physics-based control. Virtual physics-based robot control is inspired by natural phenomena and has been investigated primarily in the context of swarm robotics [10,11,14,25]. The general idea is that each robot is treated as a particle in a

simulated physical system, complete with virtual forces and rules of motion. While the forces only exist in simulation, the robots act in the real world as if the forces were real. The object is to define virtual forces and rules of motion in such a way that the local interactions between robots result in desirable global behavior.

Such systems exhibit several desirable characteristics. Each robot chooses its actions based only upon local information and a simple set of rules, but tends to act in a manner that contributes to a common goal. Thus, there is global cooperation without any global control, which makes the system both fault-tolerant and scalable. In our case, the macrosensor itself emerges as a result of the large-scale interactions of individual, nearly stateless components.

Previous attempts at virtual-physics based systems have generally focused on potential fields of some kind, using force fields analogous to gravity or the electromagnetic force. In contrast, the proposed virtual spring mesh only makes use of explicit connections between robots. More precisely, if robots are represented as vertices in a graph and force is transmitted through edges, the spring mesh is not a fully connected graph (even locally). Instead, virtual springs are created to transmit force only between selected adjacent pairs of robots. As we will describe further, besides providing efficiency gains, this approach allows for sophisticated control over the behavior of the group of robots that comprise the macrosensor. Since the spring mesh is created through a series of only local interactions between robots, robots are not required to have any explicit knowledge of the environment, or of other robots, beyond their immediate vicinity. All of the desired large-scale properties of the macrosensor emerge from these local interactions.

We chose the spring metaphor in part because the spring virtual force increases with error, which results in desirable control properties, and because it is beneficial for each connection to have natural length. If a pair of robots is too close together, the spring between them acts to push them apart; if they are too distant, it acts to pull them closer. This corrective force is directly proportional to error, which is defined as the difference between the spring's current length and its natural length. As with real springs, each virtual spring in the mesh has a natural length and a spring constant (that represents the "stiffness" of the spring). These parameters may be different for each robot.

## 2.1 Spring Formation

Since the proposed approach requires the explicit creation (and destruction) of virtual springs to transmit virtual forces between pairs of robots, the success of the macrosensor relies heavily upon the algorithm used for determining which pairs of robots to connect. We have developed and, using simulation, have evaluated several candidate algorithms for this purpose. An approach that we have found to be both effective and computationally efficient is based upon simple geometric relationships between robots. We describe this algorithm below.

A robot  $R$  will create a spring connection with its neighbor  $S$  if for every other neighbor  $T$ , the interior angle  $RTS$  is acute. This relation is simple to compute, is symmetric, and ensures a (locally) connected and planar graph. If all springs have the same natural length and stiffness, the mesh will have a total energy of zero when the robots form a hexagonal lattice, with the distance between each pair of robots equal to the springs' natural length. This is the only such formation where all angles are acute and all lengths are equal, so it is the only zero-energy configuration. Formal proofs of all these properties do exist, but are omitted here for the sake of brevity.

A hexagonal lattice is in fact the shape that is observed when running the algorithm in simulation; results are typically within 2% of optimum, as measured by the mean spring length of the final state. A

hexagonal lattice is a practical arrangement with which to cover an area, because it provides uniform spacing between the robots. Figures 1 and 2 depict simulation output in which a random initial distribution of robots is drawn into a hexagonal lattice.

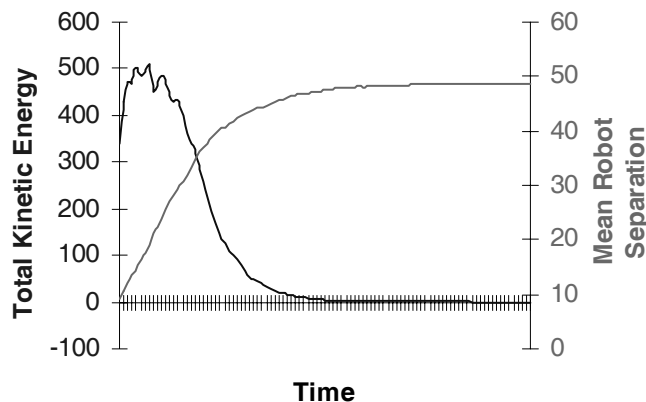


**Fig. 1.** Example of spring formation in a small, randomly distributed group of robots



**Fig. 2.** Final configuration of the same group of robots

We also employ a damping force that acts against the motion of each robot. The damping force serves to improve convergence time, i.e., the time it takes for the robot macrosensor to reach a stable state (at least temporarily). For a sufficiently large area, the basic spring and damping forces are sufficient to ensure convergent behavior--that is, the total amount of energy will decrease over time, and the system will reach a static state within a short period. More precisely, the total energy will decay exponentially, with the decay constant determined by the magnitude of the damping force (which is an adjustable parameter). Figure 3 depicts simulation output of an experiment in which 50 robots start in a tight cluster, and then deploy with a desired spring length of 50. Our simulator enforces a top speed for each robot, so the observed kinetic energy is being additionally reduced early in the simulation run, when the spring model would otherwise result in much higher energies. Also, the simulation includes a velocity “dead-band” such that any speed less than a certain limit is set to zero. This causes the robots to reach zero speed exactly, rather than just asymptotically approaching zero. A potential side effect is that the final configuration is slightly sub-optimal (from the standpoint of coverage), since very small adjustments are ignored. The final state in this example was a stable hexagonal lattice.



**Fig. 3.** Kinetic energy and mean robot separation as a function of time

It is possible to create formations other than a hexagonal lattice, either by adjusting spring parameters, altering the spring creation algorithm, or dividing the robots into different types of virtual particles, with rules of interaction that depend on the types of the participants. Various formations may have advantages for different applications; for example, square formations may be useful inside a building where the obstacles tend to have rectangular shapes.

## 2.2 Exploration

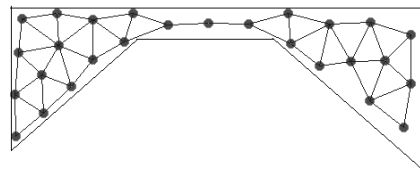
Real-world environments can be complex, containing walls and other impassable barriers. Exploring and mapping such environments has been the focus of a considerable amount of research [13,14,16,18,24,26]. Our approach to robot deployment builds upon this prior work, but differs from this work in that we seek to explore complex environments without the use of global information.

Environments that include concave spaces challenge most existing control algorithms. A narrow hallway may conceal a large unexplored room on the other side. In this case the locally optimal solution of simply spreading the robots out is not desirable, since this approach would not move many robots through the hallway.

Rather than building an explicit map and directing robots through a high-level protocol, we incorporate an additional local force, called the “exploration force.” The exploration force draws robots towards areas that are visible but not occupied by another robot. As a robot is drawn towards such open areas, it “pulls” other robots with it through its spring connections. This causes the mesh to expand into unexplored areas in a manner similar to a fluid flow. This approach effectively creates a high potential in areas that are well covered, and a low potential in areas that are unexplored. As the robots flow into a new area, the potential of this area increases until the flow stops. Robots will continue exploring until the potential is equal in all areas (that is, there is no visible area that is not covered), which will occur when the environment is covered uniformly.



**Fig. 4.** Robots in a cluster in a concave environment: initial configuration

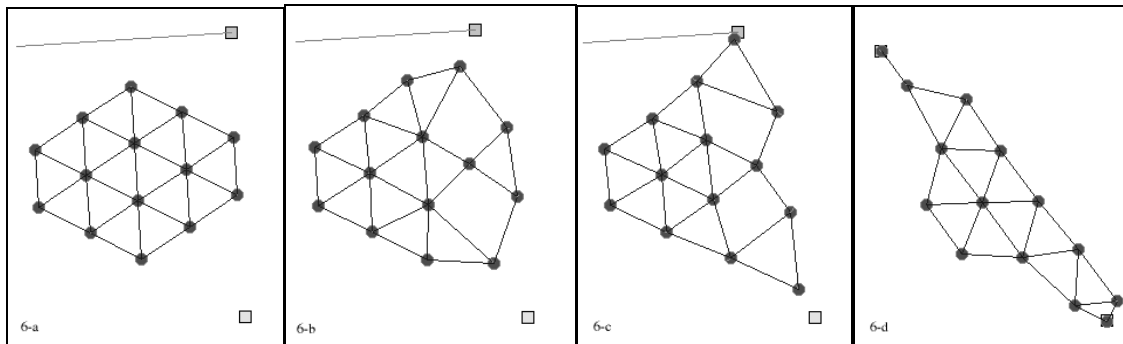


**Fig. 5.** Final configuration

The exploration force may take on several forms. Our current implementation simply adds a force in the direction of the longest apparent unobstructed path. Other choices include forces based on information-gain criteria, as well as adjustments to spring constants in order to draw “follower” robots. The exact nature of the exploration force influences global properties of the macrosensor, including the speed of deployment and the response to a situation in which there are insufficient robots to cover the entire area of interest. Characterizing various choices for the exploration force is a promising area for future research. Figures 4 and 5 depict simulation results of the simple exploration force in action, demonstrating that this simple concept handles concave areas.

## 2.3 Target Tracking

While the exact definition of a “target” is application-dependent, it is possible to classify targets into two basic categories: point targets and diffuse targets. Point tracking involves the detection and monitoring of a discrete target or targets; that is, targets with positions (which may change) that can be represented as points. In applications such as security, search and rescue and military reconnaissance, the people, vehicles or other objects are considered to be point targets. Point targets can be easily addressed within the spring mesh framework. We simply add an attractive force that draws robots toward these targets. Since intercepting the targets is likely to be appropriate in many applications, our current implementation of the point target force is designed to match the robot’s velocity with a vector that will intercept the target.

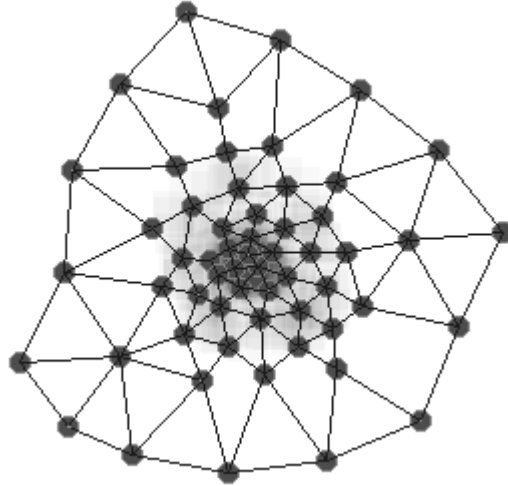


**Fig. 6.** Target tracking sequence.

Figure 6 shows simulation results of a small group of robots intercepting two point targets. The upper target is moving from right to left along the indicated path, at a speed that is 60% of the top speed of the robots. As seen in this example, targets are intercepted aggressively, and the spring mesh deforms in order to keep all of the robots connected with the desired spacing.

Diffuse target tracking is somewhat more complex, as these targets do not have a well-defined location. Heat plumes from a wildfire, or radioactive or chemical clouds are examples of diffuse targets. Here, in addition to identifying the location of the target, we also want to know both the extent of the target substance and its density gradient. We also may wish to increase the density of sensor coverage in the vicinity of a diffuse target. For example, when tracking a chemical plume, we may want to precisely map the plume extent in areas of significant concentration, while sacrificing detailed information about areas of lower concentration or about areas outside the plume. The spring mesh model is well suited to adaptive robot density, since each robot can control its own spring parameters. Robots that detect the desired diffuse target simply shorten their springs, thus drawing in their neighbors to areas of higher concentration. Figure 7 depicts simulation results showing this process in action.

With a diffuse target it may be desirable to penetrate the target (as in the chemical plume example), or only to tightly surround the target and contain it (as with a fire). Either of these behaviors can be achieved by making the target attractive or repulsive, respectively. Attractive targets will draw a dense group of robots into the target area. Repulsive targets will cause a tight group of robots (with shortened springs) to stay just outside of the target area, since they attract each other but are repelled from the target itself.



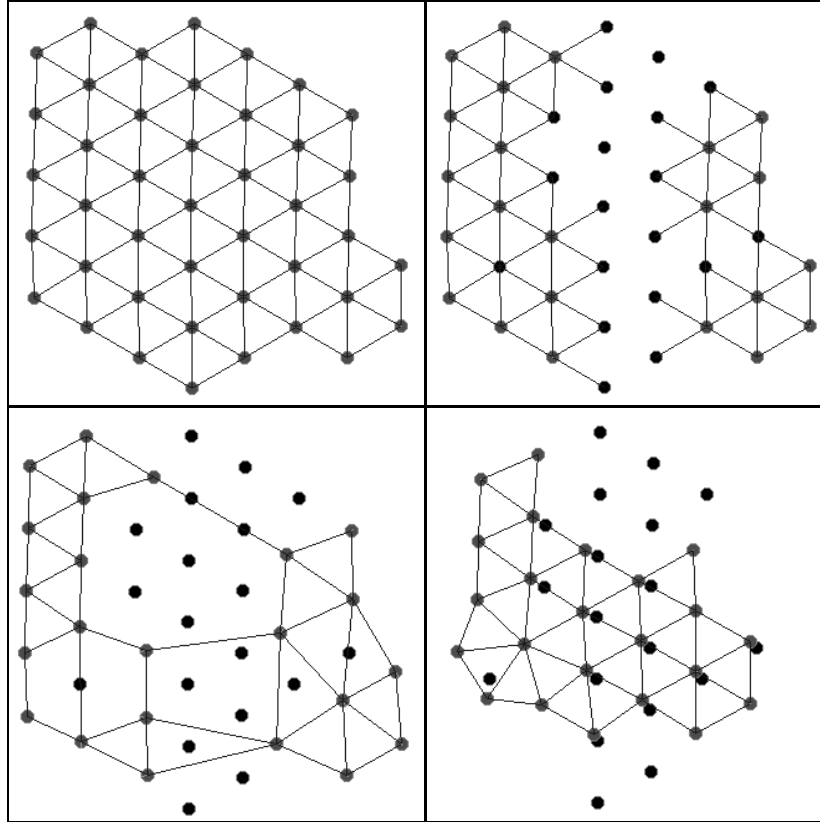
**Fig. 7.** Robots in the target area shorten their springs to increase density in the area.

### 3. Discussion

The virtual spring mesh approach is inherently scalable. The complexity of the algorithm running on each robot increases with the number of locally visible robots and targets (that is, the number of neighboring robots and targets that are currently in sight), but significantly, *complexity is independent of the total number of robots*. In fact, robots are not explicitly aware of the existence of any other robot that is not locally visible. For this reason, there is no particular limit on the number of robots that may be members of a single macrosensor. Additionally, the total area covered by the macrosensor increases linearly with the number of member robots, which satisfies our goal of extensibility.

Communication overhead of the virtual spring mesh is low. Since the spring-formation algorithm and force computations are symmetric, it is not necessary for robots to communicate with each other in order to form springs and execute the virtual physics model. Communication is only required for diffuse target mapping (since spring length adjustments must be announced), and possibly for fault detection. Even in these cases, it is only necessary to send messages to immediate neighbors; multi-hop routing is not required.

A significant advantage of the virtual spring mesh approach is that fault tolerance and attack resistance are inherent properties of the macrosensor. Since there is no hierarchy or centralized control, there are no single points of failure or obvious points of attack. Additionally, individual robots are nearly stateless, so recovery from robot failures is simple and rapid. Figure 8 shows a simulated example of a cluster of robots that experiences multiple simultaneous failures. As shown, the remaining robots automatically re-form a smaller spring mesh without the failed units. This recovery happens quickly and without any error handling beyond the detection of failed robots. The major cost associated with fault tolerance is the periodic communication required to assess the health of neighboring robots.



**Fig. 8.** Massive failure situation. “Dead” robots are represented as dark points. The top right figure is immediately after the robots failed; the lower left is one simulation cycle later. Final configuration is shown on the lower right.

#### 4. Related Work

Explicitly coordinated exploration and mapping was examined in the “Cover Me!” [13] project, which defines a coverage metric and then uses an incremental greedy algorithm to deploy robots into locally optimal locations. This scheme is not designed to scale to large numbers of robots, as it makes use of global information and only deploys one robot at a time. Similar work by Simmons et al. [24] computes desired deployment locations by attempting to minimize overlap in information gain. Explicit loosely-coupled robot coordination for arbitrary goals (not just exploration) is implemented in the ALLIANCE system [20,21], which uses behavior-based task selection. RETSINA [9] operates a team of robots through a shared plan, which is communicated and refined over time. DINTA [3] takes a hybrid approach and uses a static sensor network to assign tasks to a set of mobile robots. This approach has many advantages, but requires a pre-deployed infrastructure.

Target tracking has been addressed within some of these systems. Targets are tracked in ALLIANCE [22] through a combination of local virtual forces and high-level behavior-based selection. Jung and Sukhatme [15] also use a multi-layered approach; their system computes a local solution for tracking groups of targets within the same field of view, operating within a framework that distributes robots into regions according to target density. A different approach has been proposed by Gage [8], who suggests randomized search strategies that use inexpensive systems and make detection very probable.

Fully distributed control based upon simple local behaviors has been used in several contexts. Brooks [4] has investigated behavior-based control extensively. Balch and Hybinette [1] suggest the use of “attachment sites” that mimic the geometry of crystals; this is used to create formations with large numbers of robots. A variety of projects have made use of “swarm robotics” e.g., [23,2] to carry out simple tasks such as light tracking. Gage [7] investigated the use of robot swarms to provide blanket, barrier, or sweep coverage of an area. Several researches have used models based on the interactions of ants within a colony [16,23].

Distributed control based on virtual physics (also called “artificial physics” or “physicomimetics”) has also been investigated, although not in the manner that we propose. Howard, Mataric and Sukhatme [14] model robots as like electric charges in order to cause uniform deployment into an unknown enclosed area. Spears and Gordon [10,11,25] use a more sophisticated model analogous to the gravitational force, but make the force repulsive at close range. Both of these models use fully connected graphs, although the latter model cuts off interactions beyond a maximum range.

## 5. Conclusion

We have developed a scalable approach toward control of distributed robotic macrosensors comprised of large numbers of potentially heterogeneous robotic sensor platforms. Such macrosensors can be tasked with automatically exploring complex and unknown environments, and can track targets of interest within that environment. In addition to their inherent scalability, these macrosensors are extensible and fault tolerant. Our analysis and simulation results indicate that the virtual spring mesh is an effective mechanism for controlling such macrosensors, and that macrosensors built using this approach exhibit the desired properties.

We have demonstrated through simulation that in principle, a virtual spring mesh can be used to explore and track targets in an unknown environment that includes nontrivial obstacles. We are currently attempting to build robotic sensors that can test the viability of virtual spring mesh based control in practice.

## 6. References

1. Balch, T. and Hybinette, M. “Behavior-based coordination of large-scale robot formations.” *Proceedings of the Fourth International Conference on Multiagent Systems (ICMAS '00)*. July 2000. pp. 363 - 364.
2. Baldassarre G. Nolfi S. Parisi D. “Evolving Mobile Robots Able to Display Collective Behaviors.” In *Proceedings of the International Workshop on Self-Organisation and Evolution of Social Behaviour*, pages 11-22, Monte Verità, Ascona, Switzerland, September 8-13, 2002.
3. Batalin, M. and Sukhatme, G.S. “Sensor Network-based Multi-Robot Task Allocation.” *2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)*, October, 2003, Las Vegas, Nevada.
4. Brooks, R.A., "Integrated Systems Based on Behaviors", *SIGART Bulletin* (2:4), August 1991, pp. 46–50.
5. Delin, K “The Sensor Web: A Macro Instrument for Coordinated Sensing.” *Sensors*. Vol 2, 2002. pp. 270-285.
6. Deng, J. Han, R. Mishra, S. "A Performance Evaluation of Intrusion-Tolerant Routing in Wireless Sensor Networks," *IEEE 2nd International Workshop on Information Processing in Sensor Networks (IPSN '03)*, 2003, Palo Alto, California.

7. Gage, D. "Command Control for Many-Robot Systems." *Proceedings of AUVS-92*. Huntsville, AL. June 1992.
8. Gage, D. "Randomized Search Strategies with Imperfect Sensors." *Proceedings of SPIE Mobile Robots VIII*. Boston. September 1993. vol 2058, pp 270-279.
9. Giampapa, J. and Sycara, K. "Team-Oriented Agent Coordination in the RETSINA Multi-Agent System." Tech report CMU-RI-TR-02-34. Robotics Institute, Carnegie Mellon University. December 2002.
10. Gordon, D. Spears, W. Sokolsky, O. Lee, I. "Distributed Spatial Control, Global Monitoring and Steering of Mobile Physical Agents." *IEEE International Conference on Information, Intelligence, and Systems*. November 1999.
11. Gordon-Spears, D. and Spears, W. "Analysis of a Phase Transition in a Physics-Based Multiagent System." In *Proceedings of the FAABS '02 Workshop*. 2002.
12. Hong, X. Gerla, M. Kwon, T. Estabrook, P. Pei, G. Bagrodia, R. "The Mars Sensor Network: Efficient, Energy Aware Communications" *Proceedings of IEEE MILCOM*. McLean, VA. 2001.
13. Howard, A. and Mataric, M. J. "Cover Me! A Self-Deployment Algorithm for Mobile Sensor Networks." *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA2002)*, 2002.
14. Howard, A., Mataric, M.J., and Sukhatme, G.S. "Network Deployment using Potential Fields: A Distributed, Scalable Solution to the Area Coverage Problem." *The 6<sup>th</sup> International Symposium on Distributed Autonomous Robotic Systems*, pp. 299-308.
15. Jung, B. and Sukhatme, G.S. "Tracking Targets using Multiple Robots: The Effect of Environment Occlusion", *Autonomous Robots*, **13**(3). 2002. pp. 191-205.
16. Koenig, S. and Liu, Y. "Terrain Coverage with Ant Robots: A Simulation Study." In *Proceedings of the International Conference on Autonomous Agents*, pages 600-607, 2001.
17. Konolige, K. "A gradient method for realtime robot control." In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2000.
18. López-Sánchez, M. Esteva, F. López de Mántaras, R. Sierra, C. Amat, J. "Map Generation by Cooperative Low-Cost Robots in Structured Unknown Environments." *Autonomous Robots*, 5. pp. 53-61.
19. Nissanka, B. Chakraborty, A. Balakrishnan, H. "The Cricket Location-Support System." *6<sup>th</sup> ACM International Conference on Mobile Computing and Networking (MOBICOM)*. Boston, 2000.
20. Parker, L. "ALLIANCE: An Architecture for Fault Tolerant, Cooperative Control of Heterogeneous Mobile Robots", *Proceedings of the 1994 IEEE/RSJ/GI International Conference on Intelligent Robots and Systems (IROS '94)*, September 1994. pp 776-783.
21. Parker, L. "On the Design of Behavior-Based Multi-Robot Teams." *Advanced Robotics*. 10 (6) 1996. pp 547-578.
22. Parker, L. and Emmons, B. "Cooperative Multi-Robot Observation of Multiple Moving Targets." *Proceedings of 1997 International Conference on Robotics and Automation*, Volume 3, pp. 2082-2089.
23. Sahin E. Franks N.R. "Measurement of Space: From Ants to Robots." In *Proceedings of WGW 2002: EPSRC/BBSRC International Workshop Biologically-Inspired Robotics: The Legacy of W. Grey Walter*, pages 241-247, Bristol, UK, August 14-16, 2002.
24. Simmons, R. Apfelbaum, D. Burgard, W. Fox, D. Moors, M. Thrun, S. Younes, H. "Coordination for Multi-Robot Exploration and Mapping." *Seventeenth National Conference on Artificial Intelligence (AAAI)*. 2000.
25. Spears, W., and Gordon, D. "Using Artificial Physics to Control Agents." In *IEEE International Conference on Information, Intelligence, and Systems*, 1999.
26. Thrun, S. Burgard, W. Fox, D. "A Real-Time Algorithm for Mobile Robot Mapping With Applications to Multi-Robot and 3D Mapping." *IEEE International Conference on Robotics and Automation*. San Francisco. April 2000.