From Insect to Internet: Situated Control for Networked Robot Teams

Barry Brian Werger and Maja J Matarić

Interaction Laboratory, Robotics Research Labs
Department of Computer Science
University of Southern California
Los Angeles, CA 90089-0781
barry@usc.edu, mataric@usc.edu

Ant-like systems take advantage of agents' *situatedness* to reduce or eliminate the need for centralized control or global knowledge. This reduces the need for complexity of individuals and leads to robust, scalable systems. Such insect-inspired situated approaches have proven effective both for task performance and task allocation. The desire for general, principled techniques for situated interaction has led us to study the exploitation of *abstract situatedness* – situatedness in non-physical environments. The *port-arbitrated behavior-based control* approach provides a well-structured abstract *behavior space* in which agents can participate in situated interaction. We focus on the problem of *role assumption*, distributed task allocation in which each agent selects its own task-performing role. This paper details our general, principled Broadcast of Local Eligibility (BLE) technique for role-assumption in such behavior-space-situated systems, and provides experimental results from the CMOMMT target-tracking task.

Keywords: mobile robotics, multi-robot coordination, behavior-based control, group behavior

AMS Subject classification: 68T40, 68T35, 93C40, 93C85

1. Introduction

Ant-like systems take advantage of individual agents' *situatedness* to reduce or eliminate the need for centralized control or global knowledge. This reduces the need for complexity (of sensing, computation, and communication) of individuals and leads to robust, scalable systems. Such insect-inspired situated approaches have proven effective both for task performance [15,36,10,32] and for task allo-

cation [37,13,16]. While our previous work in ant-like robotics [37] has focused on physically situated interaction, the desire for general, principled techniques for situated interaction has led us to study the exploitation of *abstract situatedness*, that is, situatedness in non-physical environments. The *port-arbitrated behavior-based control* approach discussed below provides a well-structured abstract *behavior space* in which agents can participate in situated interaction. We focus on the problem of *role assumption*, distributed task allocation in which each agent selects its own task-performing role. This paper details our general, principled Broadcast of Local Eligibility (BLE) technique for role-assumption in such behavior-space-situated systems.

1.1. Situatedness

While the concept of intelligent behavior arising from environmental interaction goes at least as far back as Simon's wandering ant [28], Brooks' formulation of *situatedness* [9] and successful exploitation of such interaction is considered to be responsible for the behavior-based revolution in robot control [1,21]. The basic tenet of situatedness is "the world is its own best model" – given the dynamism and uncertainty of the world and robots' perception of it, it is better to rely on rapid sensor-based feedback than persisting internal models of world state. A number of robots implemented with extremely simple control systems demonstrated the effectiveness of a situated approach by robustly performing tasks that had proved problematic for previous deliberative robots, particularly in navigation and insect-like legged locomotion [8,1]. A variety of research projects began to study exploitation of situatedness for control of groups of robots. Many were inspired by natural phenomena such as flocking [20] and *stigmergic* [5,15,36] insect interaction (i.e., interaction through environmental effects), others examined applicability to tasks such as military formation control [24], and still others, focusing on manipulation tasks, characterized situated techniques as "externalization" of information into the environment, and began investigation of methods for rigorous analysis of what we call situated systems [12,11].

Whereas earlier work in situated control has focused on physically-situated aspects of multi-robot systems, below we extend the concept of situatedness to include environmental interaction between robots in an abstract task-space. Where a number of researchers have classified inter-robot communication as either state or goal transmission [29,1], we introduce a new form of "Ethereal" communication

that is analogous to environmental interaction.

1.2. Overview

Section 2 briefly presents some of our previous work in physically-situated role-assumption, to demonstrate both the benefits of such situated systems and the need for their generalization. Section 3 introduces the concept of *abstract situatedness*, focusing on the situatedness in the behavior space created by port-arbitrated behavior-based control techniques. Section 4 introduces our Broadcast of Local Eligibility (BLE) technique for multi-robot coordination, which structures behavior space for generalized types of situated interaction. This section uses the running example of a multi-robot, multiple-moving-target tracking task. This task is fully described in Section 5, which presents our experiments in this tracking domain. Section 6 provides some conclusions and directions for future research.

2. Physically-Situated Role Assumption

In this section we present some of our previous work involving the exploitation of physical situatedness for role assumption: a *robot chaining* system for foraging and a robot soccer team. Both of these systems demonstrate the simplicity, scalablity, and robustness that can result from the exploitation of situatedness, as well as the need for generality in situated techniques.

2.1. Robot Chaining

[36] and [37] describe in detail our *robot chaining* experiments which attempt to reproduce the stigmergic [15] techniques and benefits of pheromone-trail formation by natural ants [2,14,22,16].

The robot chaining system for foraging that we present replaces the chemical pheromones of the ant trails with the physical bodies of robots. We have demonstrated [37] that a group of robots equipped with only physical contact sensors is able to form itself into a physical pathway that members of the group can use for navigation (see Figure 1). Rather than depositing pheromones and having paths "emerge" through chemical processes, the chain links can collect some statistics of the activity of the chain-following robots, and use them to

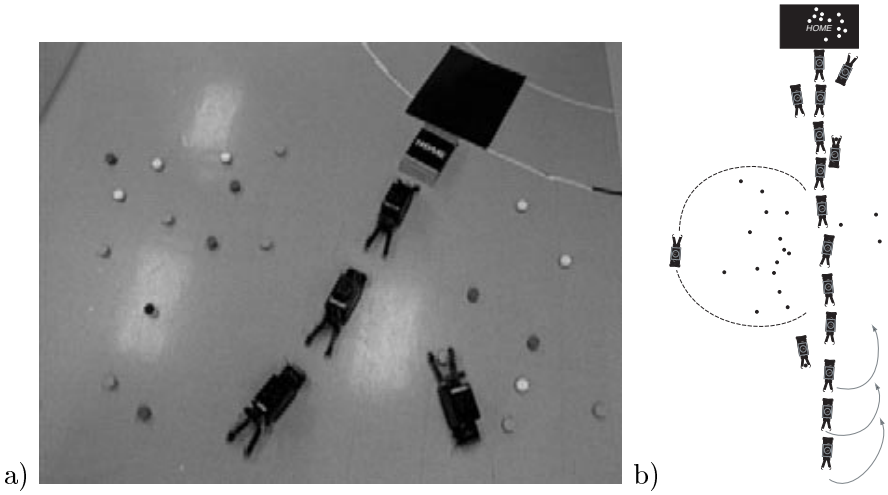

Figure 1. Foraging with a robot chain *a)* A robot returns to the chain carrying a puck after a circular excursion. *b)* The robot at the end of the chain leaves to become a forager if it notices that many successful foragers are coming along the chain (indicating that the chain has grown past a rich deposit).

adapt to the environment by physically modifying the chain, since the links of the chain are capable of computation and motion.

[13] describes findings about how ants change roles (e.g., from foragers to internal nest workers). This is found to happen in response to the number of encounters each ant has with ants fulfilling *other* roles – a nest worker that encounters a number of successful foragers in a given time period will decide to forage. The process we describe in [37] for adjusting the length of the chain functions in a similar manner: robots periodically make a decision to assume the role of forager or chain-link based on encounters with other robots.

Through a physically-situated approach, robots are able divide themselves efficiently into foragers and chain links and perform position-dependent tasks using only local sensing and interaction. [37] discusses further interesting properties of the chaining system regarding efficient role assumption given the inherent physical heterogeneity of the particular robots used.

2.2. Robot Soccer

As discussed, ant-like agents are able to determine, through their local interactions with other agents, what roles would be globally efficient for them to assume (e.g., forager vs. domestic nest worker), and our chaining robots are able

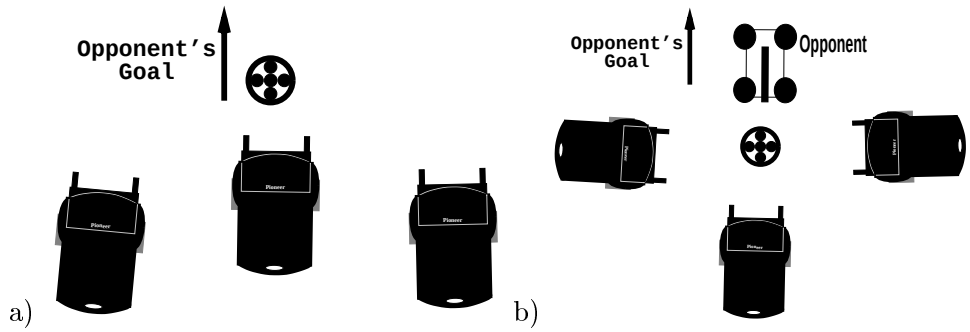

Figure 2. Formations in Robot Soccer. *a)* Offensive: Interaction of simple behaviors causes the robots to fall into a V-formation when the ball is in motion roughly towards the opponent's goal. Perceptual properties limit the formation to three robots. *b)* Defensive: When the ball is not moving roughly towards the opponent's goal, the robots cluster around it to form an effective barrier and be in good positions for recovery.

to use similar means of assuming efficient roles. Here we discuss a system we have implemented for a drastically different domain - robotic soccer - which is also able to use local interactions to determine globally efficient roles.

[34] describes at length our physically-situated approach to team cooperation for robot soccer. Though individual players can perceive only the ball, the goals, and obstacles (which are not distinguished but may be walls, opponents, or teammates), and have no communication equipment, the team displays fairly sophisticated cooperative behavior, falling into appropriate dynamic formations for offensive and defensive situations with the interesting property of formation size limitation.

The cooperative behaviors result from the interaction of simple individual behaviors such as attraction to the ball, repulsion from obstacles, and patrolling of an area when the ball is not visible. In an offensive situation, one robot pushes the ball forward, while the forces of attraction and repulsion cause the teammates to fall into a "V" formation (as shown in Figure 2a). This formation provides effective "fumble protection" that is essential in the robotic soccer domain. Robots often accidentally knock the ball off course while dribbling it forward; this formation provides backup and recovery. With this formation it is not uncommon for possession of the ball to transfer between the robots of an advancing group without loss of possession by the team. The formation also provides for a very quick defense if the ball is stolen. The size of the offensive formation is limited by the physical interaction between the robots. Once there are three robots in the

formation, other teammates are unable to pursue the ball without occasional visible occlusion by formation members, and will thus revert to defensive patrolling. In this way, necessary roles are filled (attacker, supporters, and defense) without negotiation, explicit definition or assignment of roles within the system, or even any explicit representation of teammates.

In a defensive situation the ball is not advancing toward the opponent's goal. The same forces described above cause the robots to fall into a semi-circular arrangement around the ball rather than the V-formation of the advance (see Figure 2b). This formation very effectively prevents the opponent from continuing to move the ball up the field, and places players in a good position to gain possession of the ball. An emergent "batting behavior" (described in [34]) makes it likely that the center robot will jostle the ball towards one of its teammates, which can smoothly begin an advance from the side; this can be seen as a rudimentary pass.

Transition between offensive and defensive formations is determined by motion of the ball, and is not even perceived by the robots; there is no concept of "offensive" or "defensive" (or even of "formation") anywhere in the behavior structure. Simple sensing of the local environment leads to flexible, dynamic team behavior that many researchers claim requires higher deliberation and explicit communication ([4,3,17,23,30,31]; see [18] for a detailed discussion of similar "physics-inspired" systems). Thus, in our soccer system, the situated approach allows robots to efficiently assume roles in offensive and defensive formations as determined purely by physics-inspired interaction and visual occlusion. Simple, stateless control allows sophisticated behavior including dynamically-determined limited-size formations, maintenance and recovery of ball possession, and simple passing. Assumption of roles takes place without any communication or explicit representation or coding of roles – the role behavior "emerges" from the interaction of a few simple behaviors.

2.3. Generality of Techniques Exploiting Physical Situatedness

Though we (and other researchers discussed above) have been able to build systems that effectively exploit their physically-situated nature, the techniques used have been fairly specific to their task domains and experimental environments. Our goal is to derive techniques that provide the benefits of situated approaches – simple agents, scalability, and robustness to environmental changes

and robot failures – in a generally applicable manner. For this, we have investigated situatedness in abstract environments.

3. Abstract Situatedness

The *Port-Arbitrated Behavior-Based Control* (PAB) paradigm [35] provides abstractions for constructing systems in *behavior space*, which shares certain key properties with the physical world: uncertainty, asynchronicity, and locality. As we will discuss below, entities in behavior space are able to influence or interfere with the operation of other entities, and thus we can speak of agents situated in behavior space, and of techniques that exploit this situatedness. In this section we describe the PAB paradigm and the concept of situatedness in PAB behavior space.

While the PAB approach to robotic control was introduced more than a decade ago, it is only with the recent introduction of our AYLLU language implementation [35,33] that PAB behavior interaction has been able to take place between networked robots – that is, that separate robots could inhabit the same behavior space. It is precisely this development that has motivated us to explore situatedness in behavior space and apply principles derived from our physically-interacting ant-like systems to ethereally-interacting robot teams.

3.1. Port-Arbitrated Behavior-Based Control

The behavior-based approach to robot control introduced by [6] revolutionized the field of mobile robotics in the mid-eighties [1]. One of Brooks' lesser-known (but importantly enabling) contributions is a set of well-defined abstractions and techniques for behavior interaction, implemented in a number of special-purpose languages which are more flexible successors to the well-known Subsumption Architecture [7]. We refer to these abstractions and techniques as the *Port-Arbitrated Behavior*, or PAB, paradigm [35].

3.2. Behavior-Producing Modules

In these PAB systems, controllers are written in terms of *behavior-producing modules*[1] (BPMs), each of which is an encapsulated piece of code that, when

[1] For clarity, we introduce this term to distinguish program code (*behavior-producing modules*) from observable behavior of a robot or system (*behaviors*). Previous referenced papers do

properly interfaced to sensors and actuators and run as a process, generates an observable behavior. BPMs run continuously and concurrently with other BPMs. Each BPM has a public interface for message-based interaction with other behaviors, the accessible elements of which are referred to as *ports*; they are registers that hold a single data item "message." (More detail on this interface, including the private elements, is available in [35]).

Ports are local to BPMs, and BPMs are local to robot hosts, so that an individual port is addressed by a triple $< robotname, BPMname, portname >$. BPMs can be multiply-instantiated under different names.

3.3. Connections Between Ports

Ports in different BPMs are linked together by *connections*, unidirectional data paths between a *source port* and a *destination port*. A port can have any number of incoming and outgoing connections. When a message (data item) arrives at a port, either written directly by code within the BPM or indirectly through a connection, it generally replaces the previous data item and is propagated along all of the port's outgoing connections. Such data flow can, however, be modified by connections which are specified to be *suppressive, inhibitory,* or *overriding*, as follows:

Given a connection $C_{s,d}$ from port s to port d with an associated period p, the following are the effects on d whenever a message m is propagated from s:

- If $C_{s,d}$ is *Normal*, then m is written to d, and is propagated along all of d's outgoing connections. p is not specified for Normal connections;
- If $C_{s,d}$ is *Suppressive*, then m is not written to d, and for period p, no incoming connections will be able to write to d (that is, for all ports x, any connection $C_{x,d}$ will be temporarily disabled);
- If $C_{s,d}$ is *Inhibitory*, then m is not written to d, and for period p, no messages will be propagated out from d (that is, for all ports x, any connection $C_{d,x}$ will be temporarily disabled);
- If $C_{s,d}$ is *Input Overriding*, then d is *suppressed* for period p, except that *only* messages arriving along $C_{s,d}$ (including m) are written to d and propagated as normal;

not make this distinction, though context generally suffices to distinguish between different meanings of *behavior*.

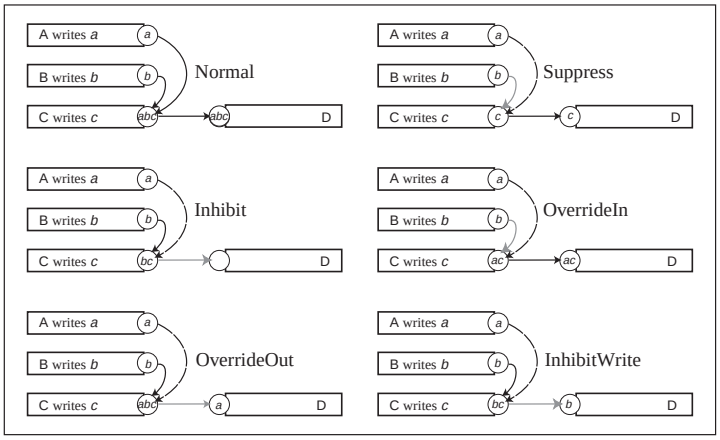

Figure 3. Effects of the Six Connection Types on Message Propagation. The rectangles are BPMs, and the circles are ports. All connections, indicated by arrows, are normal except for that between A and C, which is of the type indicated by the phrase next to the A-C arrow in each subdiagram. Letters in the circles indicate possible values of the port. In the *Normal* case, for example, D's port might hold any one of the values a, b, or c, depending on the timing of message arrival; while in the *Inhibit* case, A's message is not propagated (see Section 3.3) and no data is propagated from C's port, so C's port can hold values of b or c as determined by message arrival timing, but D's port will receive no messages.

- If $C_{s,d}$ is *Output Overriding*, then d is *inhibited* for period p, except that *only* messages arriving along $C_{s,d}$ (including m) are propagated outward along all $C_{d,x}$ as normal;
- If $C_{s,d}$ is *Write-Inhibitory*, then m is not written to d, and for period p, no messages written directly from *within the BPM* will be propagated out from d, though messages arriving *through connections* will be written and propagated.

Figure 3 illustrates the effect of the various connection types on message propagation. Any connection can be specified to be a *broadcast connection*, in which any message written to the source port results in message propagation to (or inhibition, suppression, or overriding of) the named destination port on each robot on the local network.

Further, although we have said that a new message arriving at a port generally overwrites the previous message in the port, a number of specialized port types facilitate scalability by filtering messages, which may arrive from different sources, for selective replacement. These port types are *MaxPorts, MinPorts, SumPorts*, and *PriorityPorts*. Thus, when a message $m_{incoming}$ is written to a

port p holding a data item $m_{current}$,

- if p is a normal port, $m_{incoming}$ replaces $m_{current}$
- if p is a *MaxPort*, $m_{current}$ becomes $max(m_{incoming}, m_{current})$
- if p is a *MinPort*, $m_{current}$ becomes $min(m_{incoming}, m_{current})$
- if p is a *SumPort*, $m_{current}$ becomes $m_{incoming} + m_{current}$
- if p is a *PriorityPort*, $m_{current}$ becomes $m_{incoming}$ iff $priority(m_{incoming}) > priority(m_{current})$

Our Broadcast of Local Eligibility technique described below demonstrates that a combination of only *MaxPort*s and *NormalPort*s is sufficient for implementation of some arbitrarily-scalable group coordination strategies.

It is through these mechanisms of suppression and inhibition that subsumption hierarchies, as well as other forms of arbitration, can be efficiently and intuitively implemented. Since connections are external to the BPMs, behavior code is easily re-usable, and interaction between BPMs can be modified dynamically. The port abstraction enforces a data-driven approach to programming that facilitates grounding of computation in sensor readings and effector actions. By placing coordination in the interaction between BPMs (connections) rather than in the behavior code, these systems allow complex controllers to be built "bottom-up" from simple, easily testable behaviors. The PAB approach allows a clean, uniform interface between encapsulated system components (BPMs) at all levels that abstracts away many issues of timing and communication; the "black boxes" of BPMs may contain reactive mappings or deliberative planners. While our research focuses on non-deliberative approaches, we believe that PAB interaction between system components can help reduce the complexity of the components themselves, whatever their type.

3.4. PAB Control as Situated Interaction

We make the claim that "ethereal" PAB interaction between robots, such as through wireless Ethernet, can be considered to be situated interaction, and this abstract situatedness can be exploited for many of the benefits of physical situatedness. The structure of a PAB behavior space provides something analogous to physical locality through addressable behaviors and ports, the port-BPM-robot hierarchy, and broadcast messages to related groups of ports such as "peer groups" (Section 4.2), in which certain broadcast messages are targeted only to

members of a restricted group of behaviors. These peer groups can be seen as spatial "neighborhoods." Robots are able to directly influence or interfere with each others' behavior through suppression and inhibition. Interactions, through unreliable messaging, are asynchronous and uncertain: messages sent may not arrive, as a result of either communications failure or arbitration processes. Due to the nature of connections and, especially, broadcast connections, behavior systems are scalable, and new BPMs or robots can begin interacting with a running system without modification or even notification.

We have developed a number of tools to exploit this "behavior-space situatedness," and dedicate the rest of this article to their presentation and analysis. Specifically, we show that when the PAB paradigm is extended across networks, the resulting systems are able to dynamically reconfigure themselves in order to efficiently allocate resources in response to changing environmental conditions, in a manner that is scalable and robust to robot failures.

4. Broadcast of Local Eligibility

We now introduce our Broadcast of Local Eligibility (BLE) approach to multi-robot coordination. The BLE mechanism involves a comparison of locally determined eligibility (i.e., eligibility determined through a robot's own sensory input) with the best eligibility calculated by a peer behavior on another robot. When a robot's local eligibility is best for some BPM B_n which performs task T_n, it inhibits the peer BPMs (that is, BPMs B_n) on all other robots, thereby "claiming" task T_n. Since this inhibition is an active process, failure of a robot which has claimed a task results in the task being immediately "freed" for potential takeover by another robot.

Since BLE is based on broadcast messages and receiving ports that filter their input for the "best" eligibility (see Section 4.1), BLE-based systems are inherently scalable. Up to the limit of communication bandwidth (see Section 4.8), any number of BLE-enabled robots added to a system will properly interact. As we will also see in Section 4.2, BLE allows heterogeneous robots to efficiently allocate themselves to appropriate tasks without the need for any explicit communication or global knowledge of particular abilities. The ability to dynamically instantiate and connect BLE-enabled BPMs allows systems to scale in capability as well as in number of robots.

Running Example In the following subsections and diagrams which explain BLE concepts, we will refer to an example target-tracking system, which is fully described in Section 5. Basically, a number of robots must attempt to keep a number of prioritized moving targets under constant observation. To do this, each robot has BPMs referred to as *Observers*, each of which is parameterized to cause the robot to attempt to stay within observation range of a specific Target (i.e., *Observer1* causes a robot to track *Target1*). A *Search* BPM on each robot causes the robot to wander randomly (intended to be used when no suitable Targets are within the visual field). BLE will be used to arbitrate between these BPMs, that is, to determine which task each robot in the system should attend to.

4.1. BLE-Enabled BPMs

BLE action selection requires that each BLE-arbitrated BPM include three ports: *Local*, *Best*, and *Inhibit* (see Figure 4a). Useful BPMs will usually have additional ports for task-related input and output. We generically refer to the BLE-arbitrated output of a BPM as *Output*, though the actual output may be through any number of arbitrarily-named ports. The *Best* port is a *MaxPort* (as described in Section 3.3), accepting only values that are larger than its current value.

4.2. Cross-Inhibition of BPMs

Cross-inhibition refers to the process of arbitration between *peer BPMs*, which are generally instances of the same BLE BPM on different robots. Given that there is some BPM instance B_n (which performs task T_n) on each robot, cross-inhibition results in the selection of at most a single robot to perform T_n. The selected robot is the one that is most eligible (according to local criteria) for the task. There may be multiple sets of cross-inhibiting BPMs active at the same time and, in general, cross-inhibition of BPMs B_i across robots is independent of cross-inhibition of BPMs B_j; Section 4.3 below discusses one manner in which local arbitration between different cross-inhibited BPMs can take place.

Cross-inhibition is performed in a continuous series of *decision cycles*, the maximum frequency of which is limited by network bandwidth as discussed in Section 4.8. In practice decision cycles are usually programmed to take place at a fixed rate between 10Hz and 1Hz. In each decision cycle, one robot from each peer group is selected to perform a task. As illustrated in Figure 4a, the *Local*

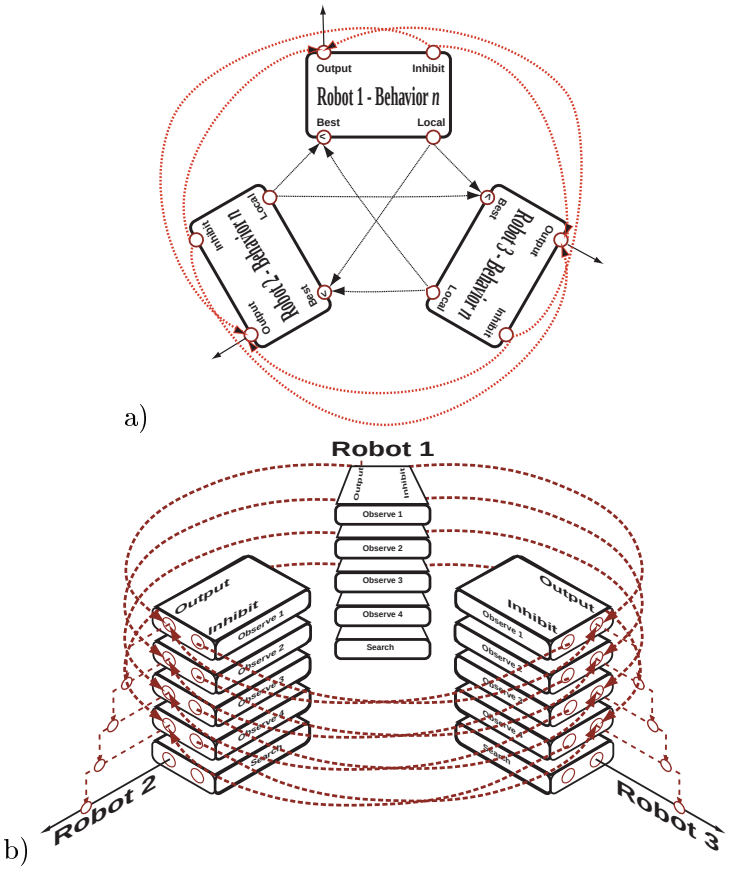

Figure 4. *a) Cross-Inhibition*: A cross-inhibited peer group. *b) Cross-Subsumption*: The structure of a cross-subsumptive system. Some lines are omitted for clarity; each "layer" is connected as in *a)*.

port of each robot's BPM B_n broadcasts a locally-computed eligibility estimate to the *Best* port of each other robot's BPM B_n. Each *Best* port maintains the maximum of the eligibility messages it has received in the current decision cycle. Whichever robot has a local eligibility better than or equal to the *Best* it receives writes to its *Inhibit* port, causing write-inhibition (described in Section 3.3) of BPM B_n's *Output* port(s) in the other robots for a period slightly longer than a decision cycle.

For cases where multiple robots are "most eligible" for some T_n, they all inhibit such that no robot performs the task for the period of inhibition; if the eligibility function is well-written and based at least in part on sensor data, then real-world uncertainty and dynamism should lead to a single robot quickly

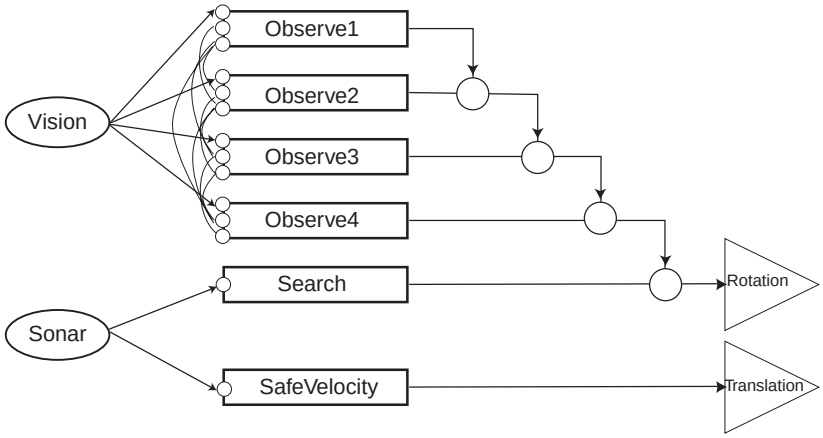

Figure 5. Local Control for CMOMMT. Arrows pointing into circles indicate *Output-Overriding* (see Section 3.3) connections; thus behaviors "higher" in the diagram have higher priority.

emerging as "most eligible." If this is not possible or acceptable in a given task domain, then various methods could be used to guarantee that ties do not occur, such as the use of a unique identifier for each robot as part of the eligibility function.

4.3. Cross-Subsumption

In BLE, Cross-inhibition arbitrates only between peer BPMs on different robots; some local mechanism must arbitrate between different BPMs on the same robot. While the PAB approach supports many types of arbitration, we believe that even simple Subsumption [7], when combined with BLE, is sufficient for flexible, scalable, and robust team cooperation in many tasks. We call the combination of cross-inhibition and local subsumption *cross-subsumption*.

In cross-subsumption, each robot has a local subsumption hierarchy (such as in Figure 5). Each layer of this hierarchy may be cross-inhibited (as in Figure 4a), resulting in a system similar to the one diagrammed in Figure 4b. As a result, each robot is controlled by its BPM B_n which has the highest priority of any BPM which is generating output. A BPM that is not generating output fails to do so either because its current input is unsuitable for the task (e.g., some necessary object is not in the field of view), or because its output is cross-inhibited. Thus, each robot will perform the highest-priority task that it is most suitable for.

4.4. Allocation of Multiple Robots to a Task

Subtasks that require more than one robot to assume the same role (e.g., 3 robots to patrol an area or surround a specific target) are implemented by multiply-instantiating the subtask-achieving BPM. To modify the control system in Figure 5 so that three robots attempt to observe Target 2, all that needs to be done is to modify the hierarchy so that instead of *Observe2* it includes *Observe2a, 2b,* and *2c*, connected as the other BPMs are, and all parameterized to track Target 2 (see Figure 6a). Given the resulting hierarchy, at most three robots will attempt to observe Target 2, and all three of these Target 2 observation roles will have lower priority than for Target 1 but higher than for Target 3. By leaving *Observe2a* in *Observe2*'s place, and placing *Observe2b* and *2c* between *Observe4* and *Search* (as in Figure 6b), the priorities will be changed such that only when all Targets are observed by at least one robot will multiple coverage of Target 2 take place.

4.5. Strict vs. Opportunistic Cross-Subsumption

It is possible to write BPMs so that they are either always active, or active only when certain prerequisites are met; for example, in a target-tracking system, the Target 1 Observer BPM might either always generate output, or only generate output when Target 1 is visible. We refer to the former as a *strict* BPM, and the latter as an *opportunistic* BPM. When constructing cross-subsumption hierarchies this difference is of utmost importance. Given the system of Figure 4b, if the BPMs are all strict, then no robot would attempt to track Targets 2, 3, or 4 until Target 1 was under observation; and if Target 1 was not present, regardless of the presence of other targets, no Targets would be tracked at all. If the BPMs were opportunistic, then the robots would distribute over whatever targets were available, and switch targets to cover any higher-priority, uncovered targets later encountered. A cross-subsumption system can blend opportunistic and strict BPMs; for example, there may be a case in which it is essential to observe Target 1 and merely desirable to observe the others; to achieve this, the Target 1 Observer BPM would be strict, and the other Observers opportunistic, guaranteeing that all robots try to observe Target 1 whenever it is not covered, but observe whatever other targets they can when 1 is covered.

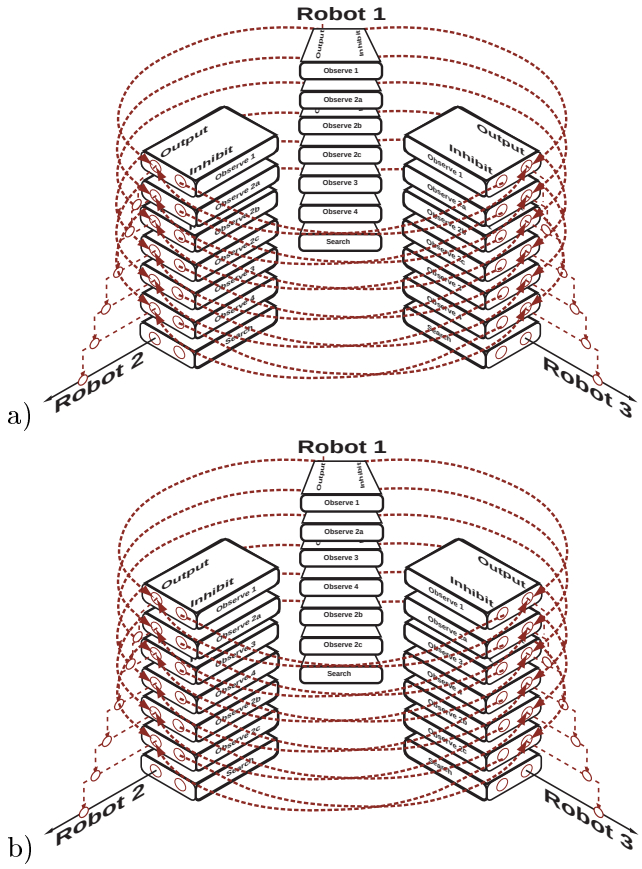

Figure 6. Assigning multiple robots to tasks. *a)* Multiple coverage of Target 2 has priority *b)* Target 2 is only multiply-covered if all Targets are covered at least once.

4.6. Heterogeneous Systems

Cross inhibition is particularly well-suited to heterogeneous systems, in which not all robots are able to perform all tasks. Robots in which some BPM B_n is not instantiated will naturally never inhibit B_n in other robots and thereby claim T_n; thus if robots locally instantiate only BPMs appropriate to their capabilities, nothing more needs to be done in order to assign heterogeneous robots to appropriate tasks. If the local arbitration (see Section 4.3) gives priority to the tasks each robot is specialized for, then this assignment of robots to tasks should be very efficient.

Figure 7 illustrates such a system of three heterogeneous robots: Robot 1 is specialized to "disable" targets and is also capable of tracking Targets 1 and

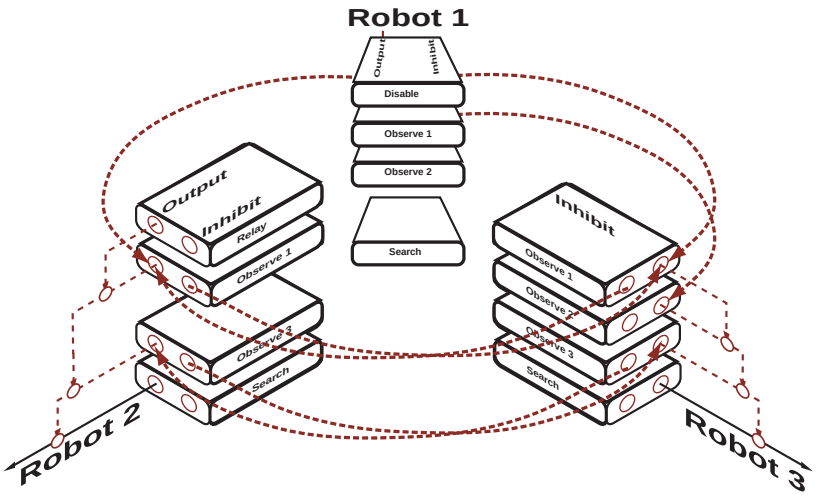

Figure 7. Cross-Subsumption of Heterogeneous Robots

2 and searching for targets; Robot 2 is specialized to act as a communication relay in case of certain types of failures, but will otherwise track Target 1 or 3, or search; and Robot 3 can track any Target or search. When these three robots are added to the network, the following peer groups will automatically form: all three robots will compete in the Observe Target 1 group; Robots 1 and 3 will compete in the Observe Target 2 peer group; and Robots 2 and 3 will compete in the Observe Target 3 peer group. Robots 1 and 2 prioritize their specialized roles.

4.7. Failure Recovery and Role Switching

In cases where a robot fails or drops out of communication, BLE provides automatic recovery, and when some robot becomes more eligible for some role than the robot that is currently filling the role, the more-eligible robot will automatically take over the role. As seen in Figure 8a, under normal circumstances when all robots are communicating, the most eligible robot (here, Robot 1) broadcasts inhibition to the rest of the peer group. If subsequently Robot 1 should suffer a communication failure, the peer group will effectively consist only of Robots 2 and 3, of which Robot 3 will take over the task (as in Figure 8b). It is possible that Robot 1 will continue performing the task, if the failure affects only its communication. In this manner, BLE is biased towards redundant coverage of high-priority tasks in cases of failure. In certain situations, it is conceivable that

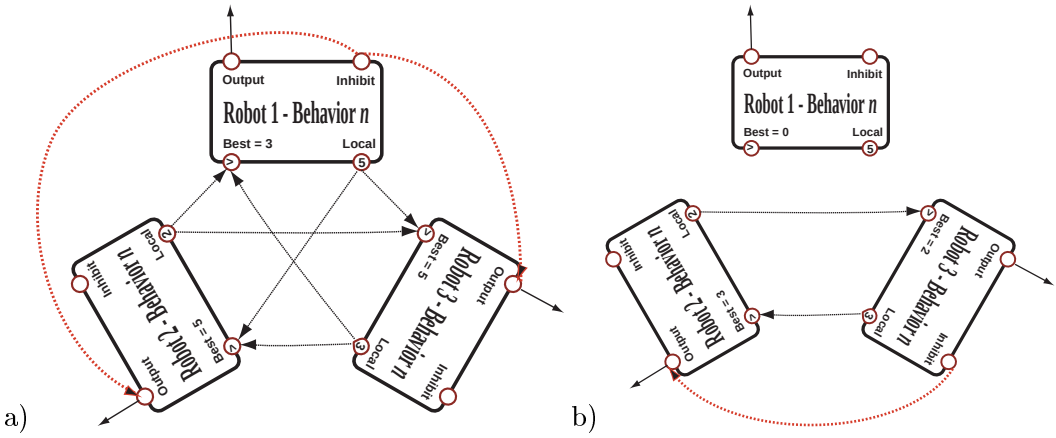

Figure 8. Robot or Communication Failure. *a)* When all robots are functioning and communicating, Robot 1 (with Local Eligibility 5) inhibits 2 and 3. *b)* If Robot 1 fails or leaves communication range, Robot 3 (Local Eligibility 3) takes over.

two or more subgroups might form, each subject to separate BLE competition for roles. In robotic systems that are situated in the physical world as well as behavior space, however, redundant coverage due to communication failure is rare (especially in opportunistic cross-subsumption systems), since physical proximity to environmental features necessary for task performance (e.g., visible Targets) lowers the likelihood that robots concurrently eligible for the same task will be out of communication range.

In cases where robots must switch roles because of changes in eligibility, the same type of reconfiguration of the peer group occurs. This case can be illustrated in Figure 8b) by imagining that there is no failure, but Robot 1's *Local* estimate of 5 is replaced by 1 due to, for example, an obstruction that does not let it stay near the target.

4.8. Scalability of BLE

As discussed above and illustrated specifically in Section 4.7, the nature of the broadcast-based cross-subsumption hierarchy allows BLE systems to adapt to changing numbers of robots. Subsumption-based arbitration in general is designed for scalability in the sense of being able to add new "layers" of behavior to existing systems [7]; Section 4.4, for example, demonstrates how peer groups can thus be added to existing systems. The practical question of scalability of BLE systems is how system requirements increase as robots and peer groups are

added to the system. The resources involved are communication bandwidth and processor time. Both of these scale linearly with number of robots $\times$ number of peer groups. We make the argument below for bandwidth, which, given current wireless Ethernet capabilities, is the limiting factor.

Since BLE is based on broadcast communication, the amount of bandwidth required scales linearly with the number of robots as follows: for each BLE peer group of N participating robots, each robot must send one broadcast message of its eligibility, and the most eligible robot must send one inhibiting message, resulting in $N + 1$ messages per peer group, per decision cycle.

In our current sub-optimal implementation, each message requires approximately 32 bytes[2]. Pessimistically assuming 64 bytes of network overhead (i.e., UDP, IP, and Ethernet headers) per message, we arrive at a figure of 768 bits per message[3] A 2Mb/s wireless network should therefore allow something on the order of 2,500 messages/second. This could allow for example, 2,500 robots participating in a single peer group at one decision cycle/second, 250 robots at 10 cycles/second; or 50 robots each participating in 50 peer groups at 1 cycle/second. In practice, wireless Ethernet has a lower effective bandwidth, and the limitations of BLE are proportionally reduced.

4.9. BLE Summary

The Broadcast of Local Eligibility technique structures behavior space so that a principled approach can be taken to building systems that exploit their abstract situatedness. Peer groups provide locality, local eligibility signals provide constant "environmental input," and inhibition allows robots to directly affect each other's actions. The broadcast basis of BLE leads to scalability limited only by communication bandwidth. BLE provides automatic recovery from robot failures and automatic retasking in response to changes in the environment or the robot team. BLE allows heterogeneous robots to assume task-achieving roles efficiently without shared knowledge of capabilities, and provides a number of means for prioritizing tasks.

[2] Assuming 8 bytes for the behavior name, 8 bytes for the port name, 8 bytes for the eligibility value, and 8 bytes which facilitate parsing.

[3] In practice, multiple messages (from different peer groups) are packed into each IP packet, so that network overhead is far less significant.

5. A BLE Example: Multiple-Target Observation

We have tested our BLE approach on a multi-target observation task known as CMOMMT (Cooperative Multi-robot Observation of Multiple Moving Targets) introduced by [25], and a prioritized variation that we call W-CMOMMT. CMOMMT is an NP-hard problem that requires strong cooperation [26] for good performance. It has the benefit of simple formulation and evaluation, and implemented systems for comparison. [27] gives a thorough overview of related work; [19] investigates efficient algorithms for the related multi-robot observation of entire areas, including trade-offs between communicative, non-communicative, and centralized methods.

5.1. The CMOMMT Task

Our version of the CMOMMT problem is defined as follows. Given:

S : a bounded, enclosed region,

R : a team of m robots with noisy, limited range, limited field-of-view sensors, and

$O(t)$: a set of n targets $o_j(t)$ such that $In(o_j(t), S)$ is true, where $In(o_j(t), S)$ means that target $o_j(t)$ is within S at time t

Define an $m \times n$ matrix $A(t)$ where

$$a_{ij}(t) = \begin{cases} 1, \text{if robot } r_i \text{ is observing target } o_j \\ \quad \text{at time } t \\ 0, otherwise \end{cases} \tag{1}$$

A robot is said to be *observing* a target when the target is in the robot's field of view and within a certain distance d_{obs}.

We define a logical OR operator over a vector H:

$$\bigvee_{i=1}^{k} h_i = \begin{cases} 1, \text{if there exists an } i \text{ such that } h_i = 1 \\ 0, otherwise \end{cases} \tag{2}$$

The goal of the CMOMMT is then to maximize the following:

$$Observation = \frac{\sum_{t=1}^{T} \sum_{j=1}^{m} \bigvee_{i=1}^{r} a_{ij}(t)}{t \times m} \tag{3}$$

that is, to maximize the time during which each target in S is being observed by at least one robot. We assume that the area covered by the sensors of the robots is much smaller than the total area to be monitored and that targets

move slower than the robots. The original formulation of the problem [27] assumes that robots share a known global coordinate system; we replace this with the assumption that the robots can visually distinguish each target from the others[4]. Thus our formulation focuses on *task space* where Parker's formulation and implementation using predictive tracking and the *local force-vector* algorithm [27] tend to be more oriented towards physical space.

We also introduce a prioritized version of the problem which we call Weighted CMOMMT, or W-CMOMMT.

Given:

W : a vector of weights such that w_i reflects the priority of target o_i,

the goal of W-CMOMMT is to maximize

$$WObservation = \frac{\sum_{t=1}^{T} \sum_{j=1}^{m} w_j \bigvee_{i=1}^{r} a_{ij}(t)}{t \times m \times \sum_{v=1}^{m} w_v} \tag{4}$$

5.2. Experimental Design

We have implemented controllers for CMOMMT on a team of three ActivMedia Pioneer 2DX robots (see Figure 9b). These are differentially-steered wheeled bases with on-board sonar (for obstacle avoidance) and vision (for identifying and tracking targets). The video cameras have a 45-degree field of view. Each robot is connected to a wireless Ethernet LAN, and programmed using Aylu [35].

5.2.1. The Experimental Environment

Current experiments take place in an 18 by 22 foot enclosure. Targets are colored paper cylinders which an experimenter moves by hand in a fixed pattern at an average speed of about 2 feet/minute; sequences of positions for each target are marked on the floor, and a workstation provides verbal prompts for precise timing of target motion. The targets all begin at one end of the enclosure, and move in a criss-cross pattern that varies from a very dispersed to a very condensed formation (see Figure 9a). Trials are run with three robots and four targets.

[4] Parker [27] mentions that the ability to distinguish targets is important in her formulation as well, and might need to be performed through such techniques as analysis of target motion when targets are close together.

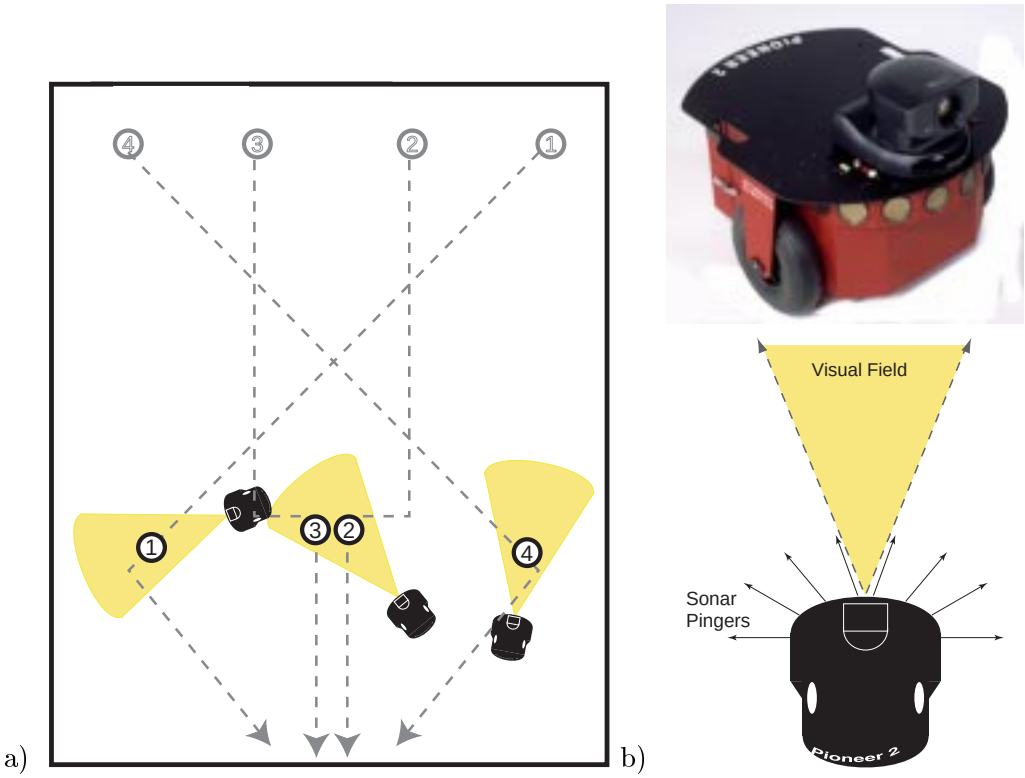

Figure 9. a) *Experimental Environment*: The 18 by 22 foot enclosure. Robots are shown with observation ranges; fields of view extend further. Targets are numbered circles. Light grey targets and dashed lines indicate initial positions and paths of targets. b) *Robotic Testbed*: Three Pioneer 2DX robots.

5.2.2. Robot Behaviors

Four controllers have been implemented for comparison in CMOMMT: a BLE controller, a local greedy controller, a local Subsumption controller, and a random controller. Each of these controllers was implemented using the same BPMs, with differences only in behavior arbitration.

Common Behaviors A single BPM on each robot controls translational motion to maintain a safe velocity based on the distance to sonar-detected obstacles. The task-oriented BPMs specified below only control rotational motion of the robots. Two classes of behavior are implemented:

- *Observer* behaviors: the target-observing BPMs rotate the robot towards a specific target in its field of view. This, combined with the common velocity

control behavior, causes the robot to approach a specific target and maintain a distance of approximately 1 foot. One observer BPM is instantiated for each target to be tracked. The observation range of the robots is approximately four feet, and the robots are able to perceive targets up to fifteen feet away, depending on lighting conditions in different parts of the enclosure.

- *Search* behavior: the search behavior is random wandering.

BLE Coordination: The BLE controller is a subsumption hierarchy of *Observer* BPMs, with the Target 1 observer having highest priority (diagrammed in Figure 5). Each is then joined into a cross-inhibiting peer group which consists of *Observer*s of the same target on each robot (Figure 4a), such that the controller becomes a cross-subsumption hierarchy (Figure 4b). The highest-priority BPM that is not cross-inhibited controls the robot - that is, each robot approaches and tracks the highest-priority target it sees that is not being observed by another robot.

The local evaluation function for each *Observer* BPM is proportional to the width of its associated target in the visual field, an approximation of distance. It favors observation of multiple targets by increasing for each additional target viewed in observation range. Given the definitions from Section 5.1, the local eligibility of the target j observer on robot i at time t is calculated as:

$$LE_{i,j}(t) = \begin{cases} \sum_{k=1}^{n} a_{i,k}(t) \times width_{i,k}(t), & \text{if } a_{i,j}(t) = 1 \\ -\infty, & otherwise \end{cases} \tag{5}$$

that is, if a robot is observing some target i, then its eligibility for the task of observing i is the sum of the widths of *all* targets that it observes, but if it is not observing target i, it has minimum eligibility for the task.

The wander BPM is active when all other BPMs are either cross-inhibited or unable to perceive any targets.

Local Subsumption Only The Local Subsumption (LS) controller is the same as the BLE controller, but connections are not made across the peer groups so that no cross-inhibition takes place. The robot approaches and tracks the highest-priority target it sees, or wanders if it sees no targets.

Local Greedy The Local Greedy (Greedy) controller has neither cross-inhibition nor local subsumption; instead, the BPM with the highest evaluation function controls the robot. The robot approaches and tracks whatever target is most

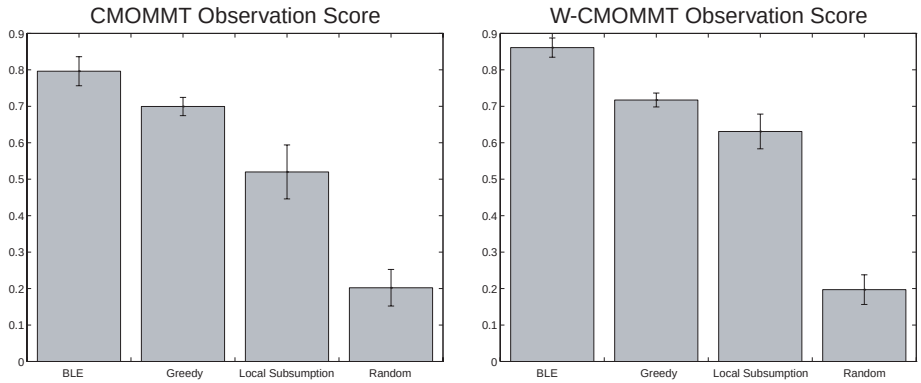

Figure 10. *Average Observation and W-Observation scores by algorithm.* Error bars span 2 standard deviations; metrics described in Section 5.1.

salient (as influenced by perceptual uncertainty) in the field of view, or wanders if no target is perceived.

Random In the Random controller, only the random wander BPM is active, at all times.

5.3. Results

Five trials of approximately 12 minutes each were run for each of the BLE, Greedy, and LS controllers; two trials of the Random controller were run for a baseline.

The most important measures are of course the *Observation* and *W-Observation* metrics of Section 5.1. As seen in Figure 10, on CMOMMT the BLE approach, with an average *Observation* of 0.7963, scored significantly higher ($p = 0.0017$ on a pairwise t-test) than the Greedy approach at 0.69940 and the LS approach at 0.51995 ($p < .0001$). Using the W-CMOMMT metric, the relative performance of BLE was even better, scoring 0.860984 to the Greedy score of 0.717251 ($p < 0.00001$) and the LS score of 0.630928.

The distribution of the robots across the targets can be clarified with information on simultaneous target coverage, illustrated in Figure 11. On average, the BLE approach observed all four targets 41.68 percent of the time, and observed at least three targets 82.17 percent of the time. The Greedy approach averaged four targets only 23.27 percent of the time, and at least three targets 66.12 percent of the time. LS observation of four targets averaging 9.71 percent,

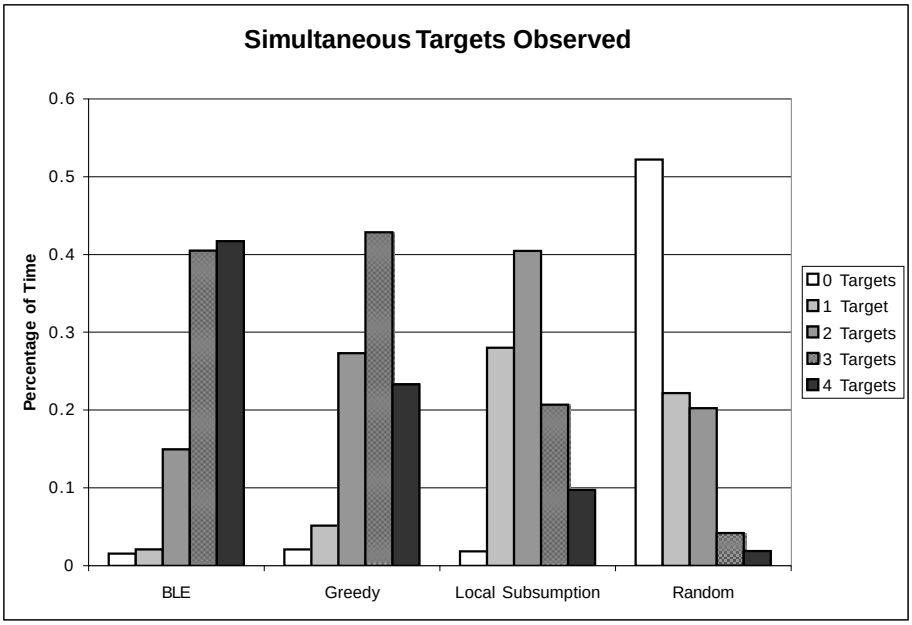

Figure 11. *Simultaneous Observation scores, by algorithm* The percentage of time during which 1, 2, 3, 4, or no targets were observed, averaged over trials.

and three or more, 30.38 percent. Thus, BLE achieved better distribution than either Greedy or LS. Surprisingly, BLE also achieved marginally higher observation of the highest-priority target than Local Subsumption (95.74 vs. 95.62 percent).

It can be seen from the target motion patterns of Figure 9a that during the last third of each trial, targets 2 and 3 were consistently close enough to be observed by a single robot. While the BLE approach resulted in a stable configuration of all four targets being observed for the majority of the final third of every trial, neither the Greedy nor the LS approaches maintained such a stable full observation in even a single trial. Figure 12 illustrates typical patterns of observation; for each algorithm we have chosen a trace of the trial which scored closest to the average. In the BLE approach, the three highest-priority targets are covered fairly constantly, although the observing robots switch off; overlap of observation is minimal. The stable four-target observation for the last third of the trial can be seen, with robot 1 covering both targets 2 and 3. In the Greedy trace, there are clearly both larger periods of overlap and larger periods in which some targets are not covered at all. In LS, as expected, the highest-priority target

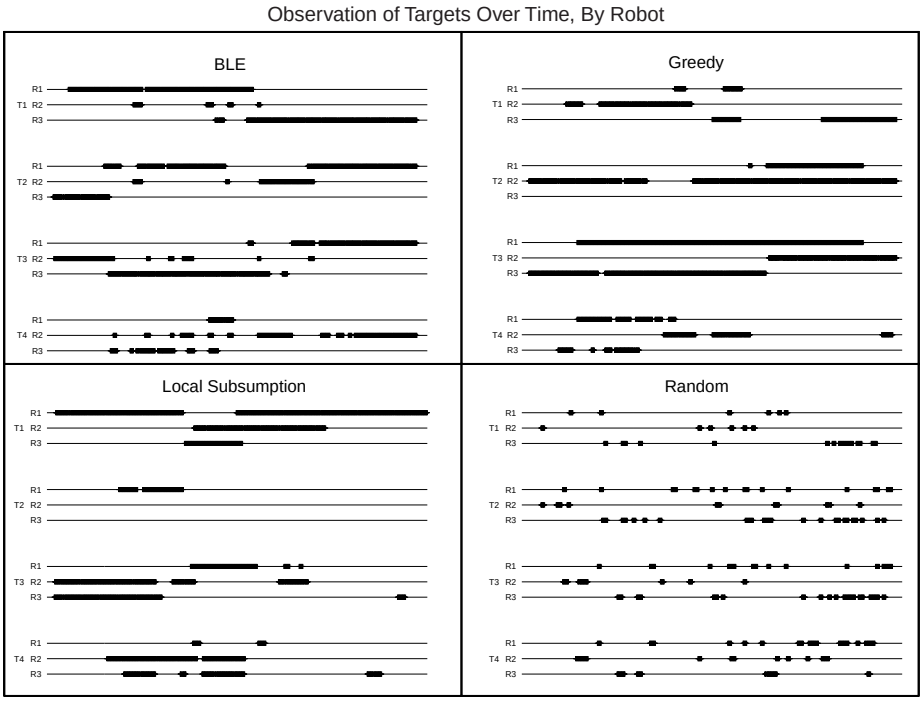

Figure 12. *Observation Over Time, by Algorithm* Each quadrant shows coverage of targets *T1–T4* by robots *R1–R3* during an average run of each algorithm. The horizontal axis is time. In the Local Subsumption quadrant, for example, we can see that Target 3 was covered by both Robots 1 and 2 for the first third of the trial. The BLE example demonstrates that BLE achieves high coverage of high-priority targets with low redundancy, even though robots switch targets.

was well and redundantly covered, while others were not.

In all trials, periods in which a particular robot seems not to be observing anything often reflect a blocked robot which is tracking a target, but not close enough to observe. This situation was common to Greedy and LS trials where robots often "queued up" behind other robots observing a salient target. Further, our collision avoidance, resulting only from the translational velocity control described in Section 5.2.2, did not deal effectively with robots approaching each other from the side, as when both were trying to get close to the same target; this resulted in occasional collisions during the LS and Greedy trials. The *task-space* separation of the BLE approach proved to be very effective in preventing both of these *physical-space* problems of interference.

Further, observation of the different approaches in action led to the realization that the BLE approach was effective in overcoming perceptual limitations of

the robots. While in the Greedy and LS trials robots tended to cluster around targets that were "better perceived" (due to details of the color-tracking implementation, and environment), exacerbating the physical-space problems described above, in the BLE trials, highly visible targets were quickly observed, driving other robots to pursue less salient targets.

6. Conclusions and Future Research

We have demonstrated that a situated approach to role assumption can be effective in physical domains to build flexible, scalable, systems; that this approach can be extended to abstract task spaces with similar effectiveness; and that the Broadcast of Local Eligibility provides a simple, general tool for building such abstract task spaces.

Experimentation has shown that the PAB paradigm, and BLE in particular, are able to support fully distributed, efficient coordination of teams of robots using simple and general low-level components. The resulting systems are scalable, robust, and flexible, adapting to changing environmental conditions and resource availability. Cross-subsumption can assign heterogeneous robots to tasks appropriately with no need for explicit negotiation or recognition. PAB is a principled approach, providing standard, well-defined abstractions for behavior coordination. Behaviors are fully encapsulated, facilitating "bottom up" system design and testing.

In the future, we plan to thoroughly analyze the class of tasks to which the PAB/BLE approach can scale. Papers in mobile robotics often make claims such as "[this architecture is] superior to subsumption for those applications which require higher-level reasoning to determine which behavior to activate" [27]. We plan to continue experimentation to increase the capabilities of behavior-based systems, and investigate and adapt analytic techniques in order to rigorously address such questions of relative capability. The development of simple, standardized coordination techniques such as PAB/BLE is an important step in the difficult problem of constructing analyzable behavior-based robotic systems.

We also show that situatedness can be extended to include situatedness in abstract behavior spaces, and that many if not all of the benefits of situatedness in the physical world are obtainable through situatedness in such abstract spaces. Further work and analysis will cast light on the nature of the non-symbolic "ethereal" interaction that we have claimed is analogous to physical-world interaction.

Acknowledgements

The research described here was supported by the Office of Naval Research Grants N00014-95-1-0759 and N0014-99-1-0162, and the Sandia National Labs Grant 53-4509-3060. Special thanks to the patient Warmitáy and the liberating Inti Qhocha Warak'a.

References

[1] R. Arkin. *Behavior-Based Robotics.* MIT Press, 1998.

[2] S. Aron, J. L. Deneubourg, S. Goss, and J. M. Pasteels. Functional self-organization illustrated by inter-nest traffic in ants: The case of the argentinian ant. In W. Alt and G. Hoffman, editors, *Biological Motion*, volume 89 of *Lecture Notes in BioMathematics*, pages 533–547. Springer-Verlag, Berlin, 1990.

[3] T. Balch and R. Arkin. Motor schema-based formation control for multi-agent robot teams. In *Proceedings of the International Conference on Multiagent Systems (ICMAS)*, pages 10–16, San Francisco, 1995.

[4] T. Balch and R. Arkin. Behavior-based formation control for multi-robot teams. *IEEE Transactions on Robotics and Automation*, 14(6):926–939, December 1998.

[5] R. Beckers, O. E. Holland, and J.L. Deneubourg. From local actions to global tasks: Stigmergy and collective robotics. In *Proceedings of the Fourth International Workshop on the Synthesis and Simulation of Living Systems*, pages 181–189. MIT Press, 1994.

[6] R. A. Brooks. *Cambrian Intelligence.* MIT Press, 1999.

[7] R. A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, 2(1):14–23, April 1986.

[8] R. A. Brooks. Elephants don't play chess. *Robotics and Autonomous Systems*, 6:3–16, 1990.

[9] R. A. Brooks. Intelligence without reason. In *Proceedings of IJCAI*, pages 569–595, Sydney, Australia, 1991.

[10] J. L. Deneubourg, S. Goss, N. Franks, A. Sendova-Franks, C. Detrain, and L. Crétien. The dynamics of collective sorting: Robot-like ants and ant-like robots. In *Proceedings of the First International Conference on Simulation of Adaptive Behavior: From Animals to Animats*, pages 356–363. MIT Press, 1991.

[11] B. Donald, J. Jennings, and D. Rus. Information invariants for distributed manipulation. In R. Wilson and J.-C. Latombe, editors, *The First Workshop on the Algorithmic Foundations of Robotics.* A.K. Peters, 1994.

[12] B. R. Donald. Information invariants in robotics. *Artificial Intelligence*, 72, 1995.

[13] D. M. Gordon. *Ants at Work.* The Free Press, New York, 1999.

[14] S. Goss, S. Aron, J. L. Deneubourg, and J. M. Pasteels. Self-organized shortcuts in the argentine ant. *Naturwissenschaften*, 76:579–581, 1989.

[15] O. Holland and C. Melhuish. Stigmergy, self-organisation, and sorting in collective robotics. *Artificial Life*, 5:2:173–202, 2000.

[16] B. Hölldobler and E. O. Wilson. *The Ants*. The Belknap Press of Harvard University Press, Cambridge, Massachusetts, 1990.

[17] N. Jennings. Controlling cooperative problem solving in industrial multi-agent systems using joint intentions. *Artificial Intelligence*, 75(2):195–240, 1995.

[18] S. Kraus. Negotiation and cooperation in multi-agent environments. *Artificial Intelligence*, 94:79–97, 1997.

[19] K. Kwok, B. Driessen, C. Phillips, and C. Tovey. Analyzing the multiple-target-multiple-agent scenario using optimal assignment algorithms. In *Sensor Fusion and Decentralized Control in Autonomous Robotic Systems, Proceedings of SPIE 3209*, pages 111–122, 1997.

[20] M. J. Matarić. Designing and understanding adaptive group behavior. *Adaptive Behavior*, 4:1:51–80, December 1995.

[21] M. J. Matarić. Behavior-based control: Examples from navigation, learning, and group behavior. *Journal of Experimental and Theoretical Artificial Intelligence*, 9(2–3):323–336, 1997.

[22] M. Muller and R. Wehner. Path integration in desert ants: Cataglyphis fortis. In *Proceedings of the National Academy of Sciences*, volume 85, pages 5287–5290, 1988.

[23] S. Onn and M. Tennenholtz. Determination of social laws for multi-agent mobilization. *Artificial Intelligence*, 95:155–167, 1997.

[24] L. E. Parker. Designing control laws for cooperative agent teams. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 582–587, 1993.

[25] L. E. Parker. Behavior-based cooperative robotics applied to multi-target observation. In R. Bolles, H. Bunke, and H. Noltemeier, editors, *Intelligent robots: Sensing, modeling, and planning*. World Scientific, 1997.

[26] L. E. Parker. Alliance: An architecture for fault tolerant multi-robot cooperation. *IEEE Transactions on Robotics and Automation*, 14, 1998.

[27] L. E. Parker. Cooperative robotics for multi-target observation. *Intelligent Automation and Soft Computing*, 5:5–19, 1999.

[28] H. A. Simon. *The Sciences of the Artificial*. MIT Press, Cambridge, Massachussetts, 1969.

[29] P. Stone and M. Veloso. Multi-agent systems: a survey from a machine learning perspective. *Autonomous Robotics*, 8(3), July 2000.

[30] P. Stone and M. M. Veloso. Task decomposition and dynamic role assignment for real time strategic teamwork. In *Intelligent Agents V – Proceedings of the Fifth International Workshop on Agent Theories, Architectures, and Languages (ATAL '98)*, Heidelberg, 1999. Springer Verlag.

[31] M. Tambe. Towards flexible teamwork. *Journal of Artificial Intelligence Research*, 7:83–124, 1997.

[32] G. Theraulaz, S. Goss, J. Gervet, and J.-L. Deneubourg. Task differentiation in polistes wasp colonies: a model for self-organizing groups of robots. In *Proceedings of the First International Conference on Simulation of Adaptive Behavior: From Animals to Animats*, pages 346–355. MIT Press, 1991.

[33] B. B. Werger. *Ayllu: Distributed Behavior-Based Control; User's Manual*. ActivMedia Robotics, 1999. available at http://robots.activmedia.com.

[34] B. B. Werger. Cooperation without deliberation: A minimal behavior-based approach to multi-robot teams. *Artificial Intelligence*, 110:293–320, 1999.

[35] B. B. Werger. Ayllu: Distributed port-arbitrated behavior-based control. In *Proceedings of DARS*, 2000.

[36] B. B. Werger and M. J. Matarić. Robotic food chains: Externalization of state and program for minimal-agent foraging. In *Proceedings of the Fourth International Conference on Simulation of Adaptive Behavior: From Animals to Animats 4*, pages 625–634. MIT Press, 1996.

[37] B. B. Werger and M. J. Matarić. Exploiting embodiment in multi-robot teams. Technical Report 99-378, USC Institute for Robotics and Intelligent Systems, 1999.