

10

MARTIN MARIETTA ENERGY SYSTEMS LIBRARIES



3 4456 0422485 4

ORNL/TM-12920

ornl

**OAK RIDGE
NATIONAL
LABORATORY**

MARTIN MARIETTA

Alliance: An Architecture for Fault Tolerant Multi-Robot Cooperation

Lynne E. Parker

OAK RIDGE NATIONAL LABORATORY

CENTRAL RESEARCH LIBRARY

CIRCULATION SECTION

4500N ROOM 175

LIBRARY LOAN COPY

DO NOT TRANSFER TO ANOTHER PERSON

If you wish someone else to see this report, send in name with report and the library will arrange a loan.

FORM 2963 (3-4-75)

MANAGED BY
MARTIN MARIETTA ENERGY SYSTEMS, INC.
FOR THE UNITED STATES
DEPARTMENT OF ENERGY

This report has been reproduced directly from the best available copy.

Available to DOE and DOE contractors from the Office of Scientific and Technical Information, P.O. Box 62, Oak Ridge, TN 37831; prices available from (615) 576-8401, FTS 626-8401.

Available to the public from the National Technical Information Service, U.S. Department of Commerce, 5285 Port Royal Rd., Springfield, VA 22161.

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

ORNL/TM-12920

Computer Science and Mathematics Division

406

**ALLIANCE: AN ARCHITECTURE FOR FAULT
TOLERANT MULTI-ROBOT COOPERATION**

Lynne E. Parker

DATE PUBLISHED ---- February 1995

Research sponsored by the
Engineering Research Program
Office of Basic Energy Sciences
U.S. Department of Energy

Prepared by the
OAK RIDGE NATIONAL LABORATORY
Oak Ridge, Tennessee 37831
managed by
MARTIN MARIETTA ENERGY SYSTEMS, INC.
for the
U.S. DEPARTMENT OF ENERGY
under contract DE-AC05-84OR21400

MARTIN MARIETTA ENERGY SYSTEMS LIBRARIES



3 4456 0422485 4

Contents

1. Introduction	1
2. Related Work	3
3. ALLIANCE	5
3.1. Assumptions	5
3.2. Overview of ALLIANCE	6
3.3. Motivational Behaviors	8
3.4. Discussion of Formal Model of ALLIANCE	9
3.4.1. Threshold of activation	10
3.4.2. Sensory feedback	10
3.4.3. Inter-robot communication	10
3.4.4. Suppression from active behavior sets	11
3.4.5. Robot impatience	12
3.4.6. Robot acquiescence	13
3.4.7. Motivation calculation	13
3.5. Parameter Settings	14
3.6. Proofs of Termination	15
4. Results	21
4.1. The Robots	21
4.2. The Hazardous Waste Cleanup Mission	22
4.3. Experiments	26
4.4. Discussion	30
5. Conclusions	35
6. Acknowledgments	37

List of Figures

1	The ALLIANCE architecture.	7
2	The experimental mission: hazardous waste cleanup.	22
3	The ALLIANCE-based control of each robot in the hazardous waste cleanup mission.	23
4	The robot control organization within the <i>find-locations-methodical</i> behavior set.	24
5	The robot control organization within the <i>find-locations-wander</i> behavior set.	25
6	The robot control organization within the <i>move-spill(loc)</i> behavior set.	25
7	The robot control organization within the <i>report-progress</i> behavior set.	26
8	The robot team at the beginning of the hazardous waste cleanup mission.	27
9	Typical robot actions selected during experiment with no robot failures.	28
10	Interfering with <i>find-locations-methodical</i> task.	29
11	Typical robot actions selected during experiment when the initial robot action of finding the spill fails.	29
12	Two robots moving the two spills.	30
13	Robots delivering spill objects to the goal destination.	31
14	Typical robot actions selected during experiment when one of the robot team members which is moving a spill is removed.	31
15	Typical robot actions selected during experiment when one of the robot team members which is reporting the progress is removed.	32

Abstract

ALLIANCE is a software architecture that facilitates the fault tolerant cooperative control of teams of heterogeneous mobile robots performing missions composed of loosely coupled, largely independent subtasks. ALLIANCE allows teams of robots, each of which possesses a variety of high-level functions that it can perform during a mission, to individually select appropriate actions throughout the mission based on the requirements of the mission, the activities of other robots, the current environmental conditions, and the robot's own internal states. ALLIANCE is a fully distributed, behavior-based architecture that incorporates the use of mathematically-modeled motivations (such as impatience and acquiescence) within each robot to achieve adaptive action selection. Since cooperative robotic teams usually work in dynamic and unpredictable environments, this software architecture allows the robot team members to respond robustly, reliably, flexibly, and coherently to unexpected environmental changes and modifications in the robot team that may occur due to mechanical failure, the learning of new skills, or the addition or removal of robots from the team by human intervention. The feasibility of this architecture is demonstrated in an implementation on a team of mobile robots performing a laboratory version of hazardous waste cleanup.

1 Introduction

A key driving force in the development of mobile robotic systems is their potential for reducing the need for human presence in dangerous applications, such as the cleanup of toxic waste, nuclear power plant decommissioning, extra-planetary exploration, search and rescue missions, and security, surveillance, or reconnaissance tasks; or in repetitive types of tasks, such as automated manufacturing or industrial/household maintenance. The nature of many of these challenging work environments requires the robotic systems to work fully autonomously in achieving human-supplied goals. One approach to designing these autonomous systems is to develop a single robot that can accomplish particular goals in a given environment. However, the complexity of many environments or missions may require a mixture of robotic capabilities that is too extensive to design into a single robot. Additionally, time constraints may require the use of multiple robots working simultaneously on different aspects of the mission in order to successfully accomplish the objective. In some instances, it may actually be easier or cheaper to design cooperative teams of robots to perform some mission than it would be to use a single robot. Thus, we must build teams of possibly heterogeneous robots that can work together to accomplish a mission that no individual robot can accomplish alone.

The difficulties in designing a cooperative team are significant. In [4], Bond and Gasser describe the basic problems the field of Distributed Artificial Intelligence must address; those aspects directly related to situated multi-robot systems include the following:

- How do we formulate, describe, decompose, and allocate problems among a group of intelligent agents?
- How do we enable agents to communicate and interact?
- How do we ensure that agents act coherently in their actions?
- How do we allow agents to recognize and reconcile conflicts?

The ALLIANCE architecture described in this article offers one solution to the above questions for multi-robot cooperation. In addition to answering these questions, however, a primary design goal in the development of this multi-robot cooperative architecture was to allow the resulting robotic teams to be fault tolerant, reliable, and adaptable. Requiring fault tolerance in a cooperative architecture emphasizes the need to build teams that minimize their vulnerability to individual robot outages. Reliability refers to the dependability of a system, and whether it functions properly each time it is utilized. One measure of the reliability of the architecture is its ability to guarantee that the mission will be solved, within certain operating constraints, when applied to any given cooperative robot team. Adaptivity in a cooperative team allows that team to be responsive to changes in individual robot skills and

performance, to dynamic environmental changes, and to changes in the robot team composition as robots dynamically join or leave the cooperative team.

This article describes the control architecture, ALLIANCE, that we have developed which facilitates fault tolerant, reliable, and adaptive cooperation among heterogeneous mobile robots. We begin by describing the related work in this area, followed by a detailed discussion of the features of ALLIANCE, including a proof of mission termination for a wide variety of cooperative robotic applications. We then illustrate the viability of this architecture by describing the results of implementing ALLIANCE on a team of robots performing a laboratory version of hazardous waste cleanup, which requires the robots to find the initial locations of two spills, move the two spills to a goal destination, and periodically report the team's progress to a human monitoring the mission.

2 Related Work

A significant body of research in cooperative mobile robotics deals with the study of large numbers (often called swarms) of homogeneous robots. This approach to multi-robot cooperation is useful for non-time-critical applications involving numerous repetitions of the same activity over a relatively large area, such as cleaning a parking lot or collecting rock samples on Mars. The approach to cooperative control taken in these systems is derived from the fields of neurobiology, ethology, psychophysics, and sociology, and is typically characterized by teams of large numbers of homogeneous robots, each of which has fairly limited capabilities on its own. However, when many such simple robots are brought together, globally interesting behavior can emerge as a result of the local interactions of the robots. A key research issue in this scenario is determining the proper design of the local control laws that will allow the collection of robots to solve a given problem.

A number of researchers have studied the issues of swarm robotics. Deneubourg *et al.* [9] describe simulation results of a distributed sorting algorithm. Theraulaz *et al.* [24] extract cooperative control strategies, such as foraging, from a study of *Polistes* wasp colonies. Steels [22] presents simulation studies of the use of several dynamical systems to achieve emergent functionality as applied to the problem of collecting rock samples on a distant planet. Drogoul and Ferber [10] describe simulation studies of foraging and chain-making robots. In [15] Mataric describes the results of implementing group behaviors such as dispersion, aggregation, and flocking on a group of physical robots. Beni and Wang [3] describe methods of generating arbitrary patterns in cyclic cellular robotics. Kube and Zhang [13] present the results of implementing an emergent control strategy on a group of five physical robots performing the task of locating and pushing a brightly lit box. Stilwell and Bay [23] present a method for controlling a swarm of robots using local force sensors to solve the problem of the collective transport of a palletized load. Arkin *et al.* [1] present research concerned with sensing, communication, and social organization for tasks such as foraging. The CEBOT work, described in [12] and many related papers, has many similar goals to other swarm-type multi-robotic systems; however, the CEBOT robots can be one of a number of robot classes, rather than purely homogeneous.

Another primary area of research in cooperative control deals with achieving “intentional” cooperation among a limited number of typically heterogeneous robots performing several distinct tasks. In this type of cooperative system, the robots often have to deal with some sort of efficiency constraint that requires a more directed type of cooperation than is found in the swarm approach described above. Although individual robots in this approach are typically able to perform some useful task on their own, groups of such robots are often able to accomplish missions that no individual robot can accomplish on its own. The general research issues of adaptive action selection, communication, and conflict resolution are of particular importance in these types of systems.

Two bodies of previous research are particularly applicable to this second type

of cooperation. First, several researchers have directly addressed this cooperative robot problem by developing control algorithms and implementing them either on physical robots or on simulations of physical robots that make reasonable assumptions about robot capabilities. Examples of this research include the work of Noreils [16], who proposes a three-layered control architecture that includes a planner level, a control level, and a functional level; Caloud *et al.* [7], who describe an architecture that includes a task planner, a task allocator, a motion planner, and an execution monitor; Asama *et al.* [2] who describes an architecture called ACTRESS that utilizes a negotiation framework to allow robots to recruit help when needed; Cohen *et al.* [8], who use a hierarchical division of authority to address the problem of cooperative fire-fighting; and Wang [25], who proposes the use of several distributed mutual exclusion algorithms that use a “sign-board” for inter-robot communication.

The second, significantly larger, body of research related to intentional cooperation comes from the Distributed Artificial Intelligence (DAI) community, which has produced a great deal of work addressing this type of intentional cooperation among generic agents. These agents are typically software systems running as interacting processes to solve a common problem rather than embodied, sensor-based robots. In most of this work, the issue of task allocation has been the driving influence that dictates the design of the architecture for cooperation. Typically, the DAI approaches use a distributed, negotiation-based mechanism to determine the allocation of tasks to agents. See [4] for many of the seminal papers in this field.

3 ALLIANCE

3.1 Assumptions

In the design of any control scheme, it is important to make explicit those assumptions underlying the approach. Thus, before describing the ALLIANCE architecture in detail, we first discuss the assumptions that were made in the design of this architecture. These are as follows:

1. The robots on the team can detect the effect of their own actions, with some probability greater than 0.
2. Robot r_i can detect the actions of other team members for which r_i has redundant capabilities, with some probability greater than 0; these actions can be detected through any available means, including explicit broadcast communication.
3. The robots share a common language.
4. Robots on the team do not lie and are not intentionally adversarial.
5. The communications medium is not guaranteed to be available.
6. The robots do not possess perfect sensors and effectors.
7. If a robot fails, it cannot necessarily communicate its failure to its teammates.
8. A centralized store of complete world knowledge is not available.

We make the first assumption to ensure that robots have some measure of feedback control and do not perform their actions purely with open-loop control. However, we do not require that robots be able to measure their own effectiveness with certainty, because we realize this rarely happens on real robots.

The second assumption deals with the problem of *action recognition* — the ability of a robot to observe and interpret the behavior of another robot. However, we do not require that a robot be able to determine a teammate's actions through passive observation, which can be quite difficult to achieve. Instead, it is quite acceptable for robots to learn of the actions of their teammates through an explicit communication mechanism, whereby robots broadcast information on their current activities to the rest of the team.

This second assumption implies that the third assumption must be true — that is, that the robots must share an unambiguous common language, to the extent that their capabilities overlap, whether they interpret the actions of other robots passively or actively. If the actions are interpreted passively, the robots must in essence share a common *body language*, whereas the use of an explicit communication mechanism implies the presence of a more traditional language, including a vocabulary and usage

rules. Of course, the robots need not share a language concerning capabilities that are not shared by other robots.

Fourth, we assume that the robots are built to work on a team, and are neither in direct competition with each other, nor are attempting to subvert the actions of their teammates. Although at a low level conflicts may arise due to, for example, a shared workspace, we assume that at a high level the robots share compatible goals.

We further assume that subsystems of the team, such as communications, sensors, and effectors, are subject to failure, thus leading to assumptions five through seven.

Finally, we assume that robots do not have access to some centralized store of world knowledge, and that no centralized agent is available that can monitor the state of the entire robot environment and make controlling decisions based upon this information.

3.2 Overview of ALLIANCE

ALLIANCE is a software architecture that facilitates the fault tolerant cooperative control of teams of heterogeneous mobile robots performing missions composed of loosely coupled, largely independent subtasks. ALLIANCE allows teams of robots, each of which possesses a variety of high-level functions that it can perform during a mission, to individually select appropriate actions throughout the mission based on the requirements of the mission, the activities of other robots, the current environmental conditions, and the robot's own internal states. ALLIANCE is a fully distributed, behavior-based architecture that incorporates the use of mathematically-modeled motivations (such as impatience and acquiescence) within each robot to achieve adaptive action selection. Since cooperative robotic teams usually work in dynamic and unpredictable environments, this software architecture allows the robot team members to respond robustly, reliably, flexibly, and coherently to unexpected environmental changes and modifications in the robot team that may occur due to mechanical failure, the learning of new skills, or the addition or removal of robots from the team by human intervention.

In ALLIANCE, individual robots are designed using a behavior-based approach [5]. Under the behavior-based construction, a number of task-achieving behaviors are active simultaneously, each receiving sensory input and controlling some aspect of the actuator output. The lower-level behaviors, or competences, correspond to primitive survival behaviors such as obstacle avoidance, while the higher-level behaviors correspond to higher goals such as map building and exploring. The output of the lower-level behaviors can be *suppressed* or *inhibited* by the upper layers when the upper layers deem it necessary. Within each layer of competence may be a number of simple modules interacting via inhibition and suppression to produce the desired behavior. This approach has been used successfully in a number of robotic applications, several of which are described in [6].

Extensions to this approach are necessary, however, when a robot must select among a number of competing actions — actions which cannot be pursued in parallel. Unlike typical behavior-based approaches, ALLIANCE delineates several *behavior sets* that are either active as a group or hibernating. Figure 1 shows the general architecture

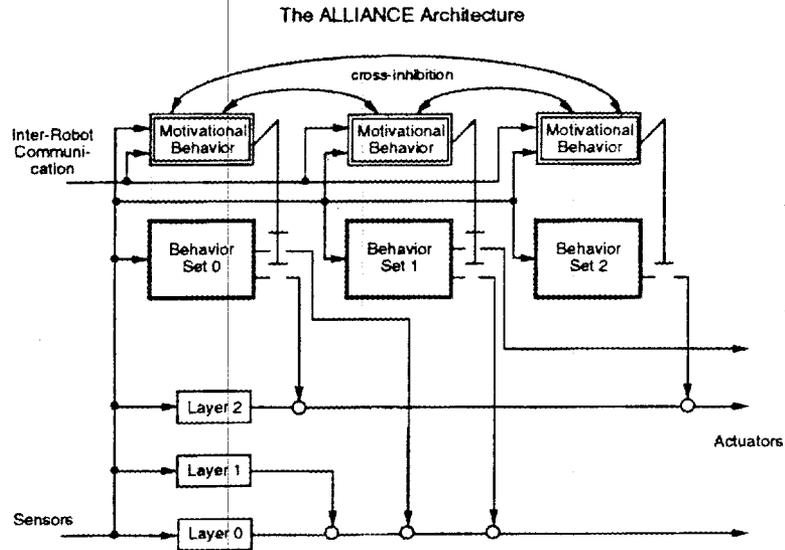


Figure 1: The ALLIANCE architecture, implemented on each robot in the cooperative team, delineates several behavior sets, each of which correspond to some high-level task-achieving function. The robot must have some mechanism allowing it to choose which high-level function to activate; the primary mechanism providing this ability is the *motivational behavior*. The symbols that connect the output of each motivational behavior with the output of its corresponding behavior set (vertical lines with short horizontal bars) indicate that a motivational behavior either allows all or none of the outputs of its behavior set to pass through to the robot's actuators. The non-bold, single-bordered rectangles correspond to individual layers of competence that are always active.

of ALLIANCE and illustrates three such behavior sets. Each behavior set a_{ij} of a robot r_i corresponds to those levels of competence required to perform some high-level task-achieving function. Because of the alternative goals that may be pursued by the robots, the robots must have some means of selecting the appropriate behavior set to activate. This action selection is controlled through the use of *motivational behaviors*, each of which controls the activation of one behavior set. Due to conflicting goals, only one behavior set should be active at any point in time. This restriction is implemented via cross-inhibition of behavior sets, represented by the arcs at the top of figure 1, in which the activation of one behavior set suppresses the activation of all other behavior sets. However, other lower-level competences such as collision avoidance may be continually active regardless of the high-level goal the robot is currently pursuing. Examples of this type of continually active competence are shown in figure 1 as layer 0, layer 1, and layer 2.

3.3 Motivational Behaviors

The primary mechanism for achieving adaptive action selection in this architecture is the *motivational behavior*. At all times during the mission, each motivational behavior receives input from a number of sources, including sensory feedback, inter-robot communication, inhibitory feedback from other active behaviors, and internal motivations called *robot impatience* and *robot acquiescence*. The output of a motivational behavior at a given point in time is the activation level of its corresponding behavior set, represented as a non-negative number. When this activation level exceeds a given threshold, the corresponding behavior set becomes active. Once a behavior set is activated, other behavior sets in that robot are suppressed so that only one behavior set is active in an individual robot at a time.

Intuitively, a motivational behavior works as follows. Robot r_i 's motivation to activate any given behavior set, a_{ij} , is initialized to 0. Then over time, robot r_i 's motivation $m_{ij}(t)$ to perform a behavior set a_{ij} increases at a fast rate of impatience as long as the task corresponding to that behavior set is not being accomplished, as determined from sensory feedback.

Additionally, the robots should be responsive to the actions of their teammates, adapting their task selection to the activities of other robot team members. Thus, if a robot r_i is aware that another robot r_k is working on a particular task, r_i should be satisfied for some period of time that that task is going to be accomplished even without its own participation in the task, and thus go on to some other applicable action. Robot r_i 's motivation to activate its corresponding behavior set continues to increase, but at a slower rate. This characteristic prevents robots from replicating each other's actions and thus wasting needless energy. Of course, detecting and interpreting the actions of other robots is not a trivial problem, and often requires perceptual abilities that are not yet possible with current sensing technology. As it stands today, the sensory capabilities of even the lower animals far exceed present robotic capabilities. Thus, to enhance the robots' perceptual abilities, ALLIANCE utilizes a simple form of broadcast communication to allow robots to inform other team members of their current activities, rather than relying totally on sensing through the world. At some pre-specified rate, each robot r_i broadcasts a statement of its current action, which other robots may listen to or ignore as they wish. No two-way conversations are employed in this architecture.

Each robot is designed to be somewhat impatient, however, in that a robot r_i is only willing for a certain period of time to allow the communicated messages of another robot to affect its own motivation to activate a given behavior set. Continued sensory feedback indicating that a task is not getting accomplished thus overrides the statements of another robot that it is performing that task. This characteristic allows robots to adapt to failures of other robots, causing them to ignore the activities of a robot that is not successfully completing its task.

A complementary characteristic in these robots is that of *acquiescence*. Just as the impatience characteristic reflects the fact that other robots may fail, the acquiescence characteristic indicates the recognition that a robot itself may fail. This feature

operates as follows. As a robot r_i performs a task, its willingness to give up that task increases over time as long as the sensory feedback indicates the task is not being accomplished. As soon as some other robot r_k indicates it has begun that same task and r_i feels it (i.e. r_i) has attempted the task for an adequate period of time, the unsuccessful robot r_i gives up its task in an attempt to find an action at which it is more productive. However, even if another robot r_k has not taken over the task, robot r_i may give up its task anyway if the task is not completed in an acceptable period of time. This allows r_i the possibility of working on another task that may prove to be more productive rather than becoming stuck performing the unproductive task forever. With this acquiescence characteristic, therefore, a robot is able to adapt its actions to its own failures.

The design of the motivational behaviors also allows the robots to adapt to unexpected environmental changes which alter the sensory feedback. The need for additional tasks can suddenly occur, requiring the robots to perform additional work, or existing environmental conditions can disappear and thus relieve the robots of certain tasks. In either case, the motivations fluidly adapt to these situations, causing robots to respond appropriately to the current environmental circumstances.

3.4 Discussion of Formal Model of ALLIANCE

Now that the basic philosophy behind the ALLIANCE architecture has been presented, let us look in detail at how this philosophy is incorporated into the motivational behavior mechanism.

First, let us formally define our problem as follows. Let the set $R = \{r_1, r_2, \dots, r_n\}$ represent the set of n heterogeneous robots composing the cooperative team, and the set $T = \{task_1, task_2, \dots, task_m\}$ represent m independent subtasks which compose the mission. We use the term *high-level task-achieving function* to correspond intuitively to the functions possessed by individual robots that allow the robots to achieve tasks required in the mission. These functions map very closely to the upper layers of the subsumption-based control architecture [5]. In the ALLIANCE architecture, each behavior set supplies its robot with a high-level task-achieving function. Thus, in the ALLIANCE architecture, the terms *high-level task-achieving function* and *behavior set* are synonymous. We refer to the high-level task-achieving functions, or behavior sets, possessed by robot r_i in ALLIANCE as the set $A_i = \{a_{i1}, a_{i2}, \dots\}$. Since different robots may have different ways of performing the same task, we need a way of referring to the task a robot is working on when it activates a behavior set. Thus, we define the set of n functions $\{h_1(a_{1k}), h_2(a_{2k}), \dots, h_n(a_{nk})\}$, where $h_i(a_{ik})$ returns the task in T that robot r_i is working on when it activates behavior set a_{ik} .

We now look in detail at the formal model of the motivational behavior. We first discuss the threshold of activation of a behavior set, and then describe the five primary inputs to the motivational behavior. We conclude this section by showing how these inputs are combined to determine the current level of motivation of a given behavior set in a given robot.

3.4.1 Threshold of activation

The threshold of activation of a behavior set is given by one parameter, θ . This parameter determines the level of motivation beyond which a given behavior set will become active. Although different thresholds of activation could be used for different behavior sets and for different robots, in ALLIANCE one threshold is sufficient since the rates of impatience and acquiescence can vary across behavior sets and across robots.

3.4.2 Sensory feedback

The sensory feedback provides the motivational behavior with the information necessary to determine whether its corresponding behavior set needs to be activated at a given point during the current mission. Although this sensory feedback usually comes from physical robot sensors, in realistic robot applications it is not always possible to have a robot sense the applicability of tasks through its sensors. Often, tasks are information-gathering types of activities whose need is indicated by the values of programmed state variables. The use of stored state in memory, therefore, can serve as a type of *virtual* sensor which serves some of the same purposes as a physical sensor.

At times, it is quite possible that the sensory feedback provides erroneous information to the robot. This erroneous information can lead the robot to assume that a task needs to be executed when, in fact, it does not (false positive), or that a task does *not* need to be performed when, in fact, it does (false negative). Although higher redundancy in individual robot sensors can help reduce this problem, at some point the levels of redundancy become exhausted, leading to robot failure. Thus, sensory failures as well as effector errors can lead to the team's failure to accomplish its mission.

We define a simple function to capture the notion of sensory feedback as follows:

$$\text{sensory_feedback}_{ij}(t) = \begin{cases} 1 & \text{if the sensory feedback in robot } r_i \text{ at time } t \\ & \text{indicates that behavior set } a_{ij} \text{ is applicable} \\ 0 & \text{otherwise} \end{cases}$$

Note that this use of sensory feedback serves the same purpose as “precondition lists” in traditional planning systems, such as STRIPS [11], or in situated agent planning systems, such as Maes' spreading activation networks [14]. In these planning systems, the precondition lists are collections of symbolic state descriptions that must hold true before a given action can be performed. One could impose a similar symbolic description on the required sensory feedback of each motivational behavior in ALLIANCE to make the environmental requirements of behavior set activation more explicit.

3.4.3 Inter-robot communication

The inter-robot broadcast communication mechanism utilized in ALLIANCE serves a key role in allowing robots to determine the current actions of their teammates.

As we noted previously, the broadcast messages in ALLIANCE substitute for more complex passive action interpretation, or action recognition, which is quite difficult to achieve.

Two parameters are utilized in ALLIANCE to control the broadcast communication among robots: ρ_i and τ_i . The first parameter, ρ_i , gives the rate at which robot r_i broadcasts its current activity. The second parameter, τ_i , provides an additional level of fault tolerance by giving the period of time robot r_i allows to pass without receiving a communication message from a specific teammate before deciding that that teammate has ceased to function. While monitoring the communication messages, each motivational behavior a_{ij} of robot r_i must also note when a team member is pursuing task $h_i(a_{ij})$. To refer to this type of monitoring in the formal model, the function *comm_received* is defined as follows:

$$comm_received(i, k, j, t_1, t_2) = \begin{cases} 1 & \text{if robot } r_i \text{ has received message from robot} \\ & r_k \text{ concerning task } h_i(a_{ij}) \text{ in the time} \\ & \text{span } (t_1, t_2), \text{ where } t_1 < t_2 \\ 0 & \text{otherwise} \end{cases}$$

3.4.4 Suppression from active behavior sets

When a motivational behavior activates its behavior set, it simultaneously begins inhibiting other motivational behaviors within the same robot from activating their respective behavior sets. At this point, a robot has effectively “selected an action”. The first motivational behavior then continues to monitor the sensory feedback, the communication from other robots, and the levels of impatience and acquiescence to determine the continued need for the activated behavior set. At some point in time, either the robot completes its task, thus causing the sensory feedback to no longer indicate the need for that behavior set, or the robot acquiesces the task either to another robot or because the robot is giving up on itself. In either case, the need for this behavior set eventually goes away, causing the corresponding motivational behavior to inactivate this behavior set. This, in turn, allows another motivational behavior within that robot the opportunity to activate its behavior set.

One additional detail has to be handled here to avoid problems when two or more motivational behaviors share exactly the same rate of impatience and which activate at the same instant. Although this situation is unlikely, if it ever occurs it can lead to the robot thrashing between the state in which multiple behavior sets are active and the idle state¹. To remedy this potential problem, a fixed priority among behavior sets is established, with the higher-priority behavior set “winning” in the case of simultaneous behavior set activations. In the formal model, however, we ignore this detail and simply refer to the cross-behavior set suppression with the following function:

¹The robot returns to the idle state after multiple simultaneous behavior set activations because all the active behavior sets send suppression messages, thus causing all the behavior sets to be deactivated.

$$activity_suppression_{ij}(t) = \begin{cases} 0 & \text{if another behavior set } a_{ik} \text{ is active, } k \neq j, \text{ on} \\ & \text{robot } r_i \text{ at time } t \\ 1 & \text{otherwise} \end{cases}$$

This function says that behavior set a_{ij} is being suppressed at time t on robot r_i if some other behavior set a_{ik} is currently active on robot r_i at time t .

3.4.5 Robot impatience

Three parameters are used to implement the robot impatience feature of ALLIANCE: $\phi_{ij}(k, t)$, $\delta_slow_{ij}(k, t)$, and $\delta_fast_{ij}(t)$. The first parameter, $\phi_{ij}(k, t)$, gives the time during which robot r_i is willing to allow robot r_k 's communication message to affect the motivation of behavior set a_{ij} . Note that robot r_i is allowed to have different ϕ parameters for each robot r_k on its team, and that these parameters can change during the mission, as indicated by the dependence on t . This allows r_i to be influenced more by some robots than others, perhaps due to reliability differences across robots.

The next two parameters, $\delta_slow_{ij}(k, t)$ and $\delta_fast_{ij}(t)$, give the rates of impatience of robot r_i concerning behavior set a_{ij} either while robot r_k is performing the task corresponding to behavior set a_{ij} (i.e. $h_i(a_{ij})$) or in the absence of other robots performing the task $h_i(a_{ij})$, respectively. We assume that the fast impatience parameter corresponds to a higher rate of impatience than the slow impatience parameter for a given behavior set in a given robot. The reasoning for this assumption should be clear — a robot r_i should allow another robot r_k the opportunity to accomplish its task before becoming impatient with r_k ; however, there is no reason for r_i to remain idle if a task remains undone and no other robot is attempting that task.

The question that now arises is the following: what slow rate of impatience does a motivational behavior controlling behavior set a_{ij} use when more than one other robot is performing task $h_i(a_{ij})$? The method used in ALLIANCE is to increase the motivation at a rate that allows the slowest robot r_k still under its allowable time $\phi_{ij}(k, t)$ to continue its attempt.

The specification of when the impatience rate for a behavior set a_{ij} should grow according to the slow impatience rate and when it should grow according to the fast impatience rate is given by the following function:

$$impatience_{ij}(t) = \begin{cases} \min_k(\delta_slow_{ij}(k, t)) & \text{if } (comm_received(i, k, j, t - \tau_i, t) = 1) \\ & \text{and} \\ & (comm_received(i, k, j, 0, t - \phi_{ij}(k, t)) = 0) \\ \delta_fast_{ij}(t) & \text{otherwise} \end{cases}$$

Thus, the impatience rate will be the minimum slow rate, $\delta_slow_{ij}(k, t)$, if robot r_i has received communication indicating that robot r_k is performing the task $h_i(a_{ij})$ in the last τ_i time units, but not for longer than $\phi_{ij}(k, t)$ time units. Otherwise, the impatience rate is set to $\delta_fast_{ij}(t)$.

The final detail to be addressed is to cause a robot's motivation to activate behavior set a_{ij} to go to 0 the first time it hears about another robot performing task $h_i(a_{ij})$.

This is accomplished through the following:

$$impatience_reset_{ij}(t) = \begin{cases} 0 & \text{if } \exists k. ((comm_received(i, k, j, t - \delta t, t) = 1) \\ & \text{and } (comm_received(i, k, j, 0, t - \delta t) = 0)), \\ & \text{where } \delta t = \text{time since last communication check} \\ 1 & \text{otherwise} \end{cases}$$

This reset function causes the motivation to be reset to 0 if robot r_i has just received its first message from robot r_k indicating that r_k is performing task $h_i(a_{ij})$. This function allows the motivation to be reset no more than once for every robot team member that attempts task $h_i(a_{ij})$. Allowing the motivation to be reset repeatedly by the same robot would allow a persistent, yet failing robot to jeopardize the completion of the mission.

3.4.6 Robot acquiescence

Two parameters are used to implement the robot acquiescence characteristic of ALLIANCE: $\psi_{ij}(t)$ and $\lambda_{ij}(t)$. The first parameter, $\psi_{ij}(t)$, gives the time that robot r_i wants to maintain behavior set a_{ij} activation before yielding to another robot. The second parameter, $\lambda_{ij}(t)$, gives the time robot r_i wants to maintain behavior set a_{ij} activation before giving up to possibly try another behavior set.

The following *acquiescence* function indicates when a robot has decided to acquiesce its task:

$$acquiescence_{ij}(t) = \begin{cases} 0 & \text{if } [(behavior\ set\ } a_{ij} \text{ of robot } r_i \text{ has been active for more} \\ & \text{than } \psi_{ij}(t) \text{ time units at time } t) \text{ and} \\ & (\exists x. comm_received(i, x, j, t - \tau_i, t) = 1)] \\ & \text{or} \\ & (behavior\ set\ } a_{ij} \text{ of robot } r_i \text{ has been active for more} \\ & \text{than } \lambda_{ij}(t) \text{ time units at time } t) \\ 1 & \text{otherwise} \end{cases}$$

This function says that a robot r_i will not acquiesce behavior set a_{ij} until one of the following conditions is met:

- r_i has worked on task $h_i(a_{ij})$ for a length of time $\psi_{ij}(t)$ and some other robot has taken over task $h_i(a_{ij})$
- r_i has worked on task $h_i(a_{ij})$ for a length of time $\lambda_{ij}(t)$

3.4.7 Motivation calculation

All of the inputs described above are combined into the calculation of the levels of motivation as follows:

$$\begin{aligned}
m_{ij}(0) &= 0 \\
m_{ij}(t) &= [m_{ij}(t-1) + \textit{impatience}_{ij}(t)] \\
&\quad \times \textit{sensory_feedback}_{ij}(t) \\
&\quad \times \textit{activity_suppression}_{ij}(t) \\
&\quad \times \textit{impatience_reset}_{ij}(t) \\
&\quad \times \textit{acquiescence}_{ij}(t)
\end{aligned} \tag{1}$$

Initially, the motivation to perform behavior set a_{ij} in robot r_i is set to 0. This motivation then increases at some positive rate $\textit{impatience}_{ij}(t)$ unless one of four situations occurs: (1) the sensory feedback indicates that the behavior set is no longer needed, (2) another behavior set in r_i activates, (3) some other robot has just taken over task $h_i(a_{ij})$ for the first time, or (4) the robot has decided to acquiesce the task. In any of these four situations, the motivation returns to 0. Otherwise, the motivation grows until it crosses the threshold θ , at which time the behavior set is activated and the robot can be said to have selected an action. Whenever some behavior set a_{ij} is active in robot r_i , r_i broadcasts its current activity to other robots at a rate of ρ_i .

3.5 Parameter Settings

Clearly, a robot’s selection of actions under ALLIANCE is dependent upon the parameter settings of the motivational behaviors — particularly, the settings of $\phi_{ij}(k, t)$ (time before impatient), $\psi_{ij}(t)$ (time before acquiescing to another robot), and $\lambda_{ij}(t)$ (time before giving up current task). If these parameters are set to very small values, the robots will tend to “thrash” back and forth between tasks, exhibiting very short attention spans. If the parameters are set to very large values, then the robots can be viewed either as showing remarkable perseverance, or as wasting incredible amounts of time.

In practice, finding the proper parameter settings is not difficult. The ALLIANCE architecture has been implemented on a number of quite different robotic applications [21], one of which is reported later in this article, and parameter tuning did not prove to be a problem. Clearly, however, some attention should be paid to the parameters, as they do have a significant influence on the action selection of the robots. Ideally, the robots on the cooperative team should be able to adapt these values with experience to find the right parameter settings that moderate between the two extremes, rather than relying on human tuning. An extension to ALLIANCE, called L-ALLIANCE, provides mechanisms that allow the robots to dynamically update their parameter settings based upon knowledge learned from previous experiences. Since this dynamic parameter update mechanism is beyond the scope of this article, refer to [21] for details of L-ALLIANCE.

We also note that a number of issues regarding the efficiency of ALLIANCE are not addressed here. Among these issues include questions of how long robots remain idle before activating a task, how to ensure that robots failing at one task go on to attempt another task they might be able to accomplish, how robots deal with having more than one way to accomplish a task, and so forth. All of these issues are handled successfully using the dynamic parameter update mechanism L-ALLIANCE.

3.6 Proofs of Termination

When evaluating a control architecture for multi-robot cooperation, it is important to be able to predict the team's expected performance using that architecture in a wide variety of situations. One should be justifiably wary of using an architecture that can fail catastrophically in some situations, even though it performs fairly well on average. At the heart of the problem is the issue of reliability — how dependable the system is, and whether it functions properly each time it is utilized. To properly analyze a cooperative robot architecture we should separate the architecture itself from the robots on which the architecture is implemented. Even though individual robots on a team may be quite unreliable, a well-designed cooperative architecture could actually be implemented on that team to allow the robots to very reliably accomplish their mission, given a sufficient degree of overlap in robot capabilities. On the other hand, an architecture should not be penalized for a team's failure to accomplish its mission even though the architecture has been implemented on extremely reliable robots, if those robots do not provide the minimally acceptable mix of capabilities. A major difficulty, of course, is defining reasonable evaluation criteria and evaluation assumptions by which an architecture can be judged. Certain characteristics of an architecture that extend its application domain in some directions may actually reduce its effectiveness for other types of applications. Thus, the architecture must be judged according to its application niche, and how well it performs in that context.

ALLIANCE is designed for applications involving a significant amount of uncertainty in the capabilities of robot team members which themselves operate in dynamic, unpredictable environments. Within this context, a key point of interest is whether the architecture allows the team to complete its mission at all, even in the presence of robot difficulties and failure. This section examines this issue by evaluating the performance of ALLIANCE in certain dynamic environments.

Let us consider realistic applications involving teams of robots that are not always able to successfully accomplish their individual tasks; we use the term *limitedly-reliable* robot to refer to such robots. The uncertainty in the expected effect of robots' actions clearly makes the cooperative control problem quite challenging. Ideally, ALLIANCE's impatience and acquiescence factors will allow a robot team to successfully reallocate actions as robot failures or dynamic changes in the environment occur. With what confidence can we know that this will happen in general? As we shall see below, in many situations ALLIANCE is guaranteed to allow a limitedly-reliable robot team to successfully accomplish its mission.

It is interesting to note that with certain restrictions on parameter settings, the

ALLIANCE architecture is guaranteed to allow the robot team to complete its mission for a broad range of applications. We describe these circumstances here, along with the proof of mission termination.

We first define the notions of *goal-relevant capabilities* and *task coverage*.

Definition 1 *The goal-relevant capabilities of robot r_i , GRC_i , are given by the set:*

$$GRC_i = \{a_{ij} | h_i(a_{ij}) \in T\}$$

where T is the set of tasks required by the current mission.

In other words, the capabilities of robot r_i that are relevant to the current mission (i.e. *goal*) are simply those high-level task-achieving functions which lead to some task in the current mission being accomplished.

We use the term *task coverage* to give a measure of the number of capabilities on the team that may allow some team member to achieve a given task. However, we cannot always predict robot failures; thus, at any point during a mission, a robot may reach a state from which it cannot achieve a task for which it has been designed. This implies that the expected task coverage for a given task in a mission may not always equal the *true* task coverage once the mission is underway.

Definition 2 *Task coverage is given by:*

$$task_coverage(task_k) = \sum_{i=1}^n \sum_j \left\{ \begin{array}{ll} 1 & \text{if } (h_i(a_{ij}) = task_k) \\ 0 & \text{otherwise} \end{array} \right\}$$

The task coverage measure is useful for composing a team of robots to perform a mission from an available pool of heterogeneous robots. At a minimum, we need the team to be composed so that the task coverage of all tasks in the mission equals 1. This minimum requirement ensures that, for each task required in the mission, a robot is present that has some likelihood of accomplishing that task. Without this minimum requirement, the mission simply cannot be completed by the available robots. Ideally, however, the robot team is composed so that the task coverage for all tasks is greater than 1. This gives the team a greater degree of redundancy and overlap in capabilities, thus increasing the reliability and robustness of the team amidst individual robot failures.

Let us now define the notion of *sufficient task coverage* as follows:

Condition 1 (Sufficient task coverage):

$$\forall (task_k \in T).(task_coverage(task_k)) \geq 1$$

This condition ensures that, barring robot failures, all tasks required by the mission should be able to be accomplished by some robot on the team.

Now, we define the notion of an *active* robot team, since we consider our robots to be useful only if they can be motivated to perform some action:

Definition 3 *An active robot team is a group of robots, R , such that:*

$$\forall(r_i \in R). \forall(a_{ij} \in GRC_i). \forall(r_k \in R). \forall t. \\ [(\delta_{slow_{ij}}(k, t) > 0) \wedge (\delta_{fast_{ij}}(t) > 0) \wedge (\theta \text{ is finite})]$$

In other words, an *active* robot has a monotonically increasing motivation to perform any task of the mission which that robot has the ability to accomplish. Additionally, the threshold of activation of all behavior sets of an *active* robot is finite.

Finally, we define a condition that holds in many multi-robotic applications.

Condition 2 (Progress when Working):

Let z be the finite amount of work remaining to complete a task w . Then whenever robot r_i activates a behavior set corresponding to task w , either (1) r_i remains active for a sufficient, finite length of time ϵ such that z is reduced by a finite amount which is at least some constant δ greater than 0, or (2) r_i experiences a failure with respect to task w . Additionally, if z ever increases, the increase is due to an influence external to the robot team.

Condition 2 ensures that even if robots do not carry a task through to completion before acquiescing, they still make some progress toward completing that task whenever the corresponding behavior set is activated for some time period at least equal to ϵ . One exception, however, is if a robot failure has occurred that prevents robot r_i from accomplishing task w , even if r_i has been designed to achieve task w .

This condition also implies that if more than one robot is attempting to perform the same task at the same time, the robots do not interfere with each others' progress so badly that no progress towards completion of the task is made. The rate of progress may be slowed somewhat, or even considerably, but some progress is made nevertheless.

Finally, Condition 2 implies that the amount of work required to complete the mission never increases as a result of robot actions. Thus, even though robots may not be any help towards completing the mission, at least they are not making matters worse. Although this may not always hold true, in a wide variety of applications this is a valid assumption. As we shall see, this assumption is necessary to prove the effectiveness of ALLIANCE in certain situations. Of course, this does not preclude dynamic environmental changes from increasing the workload of the robot team, which ALLIANCE allows the robots to handle without problem.

What we now show is that whenever conditions 1 and 2 hold for a limitedly-reliable, active robot team, then either ALLIANCE allows the robot team to accomplish its mission, or some robot failure occurs. Furthermore, if a robot failure occurs, then we can know that any task that remains incomplete at the end of the mission is either

a task that the failed robot was designed to accomplish, or a task that is dependent upon the capabilities of that robot.

We can now show the following:

Theorem 1 *Let R be a limitedly-reliable, active robot team, and M be the mission to be solved by R , such that Conditions 1 and 2 hold. Then either (1) ALLIANCE enables R to accomplish M , or (2) a robot failure occurs. Further, if robot r_f fails, then the only tasks of M that are not completed are some subset of (a) the set of tasks r_f was designed to accomplish, unioned with (b) the set of tasks dependent upon the capabilities of r_f .*

Proof:

First, we show that the calculation of the motivational behavior guarantees that each robot eventually activates a behavior set whose sensory feedback indicates that the corresponding task is incomplete. From equation 1 in section 3.4.7, we see that at time t , robot r_i 's motivation $m_{ij}(t)$ to perform behavior set a_{ij} either (1) goes to 0, or (2) changes from $m_{ij}(t-1)$ by the amount $impatience_{ij}(t)$. The motivation goes to 0 in one of four cases: (1) if the sensory feedback indicates that the behavior set is no longer applicable, (2) if another behavior set becomes active, (3) if some other robot has taken over task $h_i(a_{ij})$, or (4) if the robot has acquiesced its task. If the sensory feedback indicates that the behavior set is no longer applicable, we know either that the task $h_i(a_{ij})$ must be successfully accomplished, or the robot's sensory system has failed. If another behavior set a_{ik} becomes active in r_i , then at some point task $h_i(a_{ij})$ will either become complete, thus allowing r_i to activate behavior set a_{ij} , or the robot has failed. If some other robot has taken over task $h_i(a_{ij})$, then either that other robot will eventually accomplish task $h_i(a_{ij})$, thus eliminating the need to activate task a_{ij} , or robot r_i will become impatient with that other robot. Since r_i is *active*, then we know that $impatience_{ij}(t)$ is greater than or equal to $\min_k(\delta_slow_{ij}(k, t))$, which is greater than 0. Therefore, we can conclude that an idle, yet *active* robot always has a strictly increasing motivation to perform some incomplete task. At some point, the finite threshold of activation, θ , will thus be surpassed for some behavior set, causing r_i to activate the behavior set corresponding to task $h_i(a_{ij})$.

We now build upon these observations to prove that either the mission becomes accomplished, or a robot failure occurs.

PART I (Either ALLIANCE succeeds or a robot fails):

Assume no robot fails. Then after a robot r_i has performed a task w for any period of time greater than ϵ , one of five events can occur:

1. Robot r_j takes over task w , leading robot r_i to acquiesce.
2. Robot r_i gives up on itself and acquiesces w .
3. Robot r_j takes over task w , but r_i does not acquiesce.
4. Robot r_i continues w .

5. Robot r_i completes w .

Since Condition 2 holds, we know that the first four cases reduce the amount of work left to complete task w by at least a positive, constant amount δ . Since the amount of work left to accomplish any task is finite, the task must eventually be completed in finite time. In the fifth case, since task w is completed, the sensory feedback of the robots no longer indicates the need to perform task w , and thus the robots will go on to some other task required by the mission.

Thus, for every task that remains to be accomplished, either (1) a robot able to accomplish that task eventually attempts the task enough times so that it becomes complete, or (2) all robots designed to accomplish that task have failed.

PART II (Incomplete tasks are dependent upon a failed robot's capabilities):

Let F be the set of robots that fail during a mission, and A_F be the union of (a) the tasks that the robots in F were designed to accomplish and (b) those tasks of the mission that are dependent upon a task that a robot in F was designed to accomplish.

First, we show that if a task is not in A_F , then it will be successfully completed. Let w be some task required by the mission that is not included in A_F . Since Condition 1 holds and this robot team is active, there must be some robot on the team that can successfully accomplish w . Thus, as long as w remains incomplete, one of these successful robots will eventually activate its behavior set corresponding to the task w ; since condition 2 holds, that task will eventually be completed in finite time. Thus, all tasks not dependent upon the capabilities of a failed robot are successfully completed in ALLIANCE.

Now, we show that if a task is not completed, it must be in A_F . Let w be a task that was not successfully completed at the end of the mission. Assume by way of contradiction that w is not in A_F . But we know from Part I that all tasks w not in A_F must be completed. Therefore, task w must be in A_F .

We can thus conclude that if a task is not accomplished, then it must be a task for which all robots with that capability have failed, or which is dependent upon some task for which all robots with that capability have failed. \square

Note that it is not required here that robot team members be *aware* of the actions of their teammates in order to guarantee that ALLIANCE allows the team to complete its mission under the above conditions. However, awareness does have an effect on the *quality* of the team's performance, both in terms of the time and the energy required to complete the mission. These effects on team performance are discussed in [21].

4 Results

The ALLIANCE architecture has been successfully implemented in a variety of proof-of-concept applications on both physical and simulated mobile robots. The applications implemented on physical robots include two versions of a hazardous waste cleanup mission and a cooperative box pushing demonstration [18]. The applications using simulated mobile robots include a janitorial service mission [17] and a bounding overwatch mission (reminiscent of military surveillance) [21]. All of these missions using the ALLIANCE architecture have been well-tested. Over 60 logged (and many videotaped) physical robot runs of the hazardous waste cleanup mission and over 30 physical robot runs (many of which were videotaped) of the box pushing demonstration were completed to elucidate the important issues in heterogeneous robot cooperation. The missions implemented on simulated robots encompass dozens of runs each, most of which were logged in the study of the action selection mechanism.

The experimental mission we describe here to illustrate the fault tolerant action selection features of ALLIANCE is a laboratory version of hazardous waste cleanup. (Refer [19,21] for a somewhat different version of the hazardous waste cleanup mission, which involved the use of only one spill, rather than the two spills described below.) We first describe the robots used in these experimental studies, followed by a description of the mission the robots were given. We then describe the behavior set design of the robots for this mission, followed by the results of the implementation. The results described below are available on videotape [20].

4.1 The Robots

Our empirical studies were conducted on teams of three R-2 robots purchased commercially from IS Robotics. Each of these robots is a small, fully autonomous wheeled vehicle measuring approximately 25 centimeters wide, 31 centimeters deep, and 35 centimeters tall. The R-2 has two drive wheels arranged as a differential pair, two caster wheels in the rear for stability, and a two-degree-of-freedom parallel jaw gripper for grasping objects. The robot sensory suite includes eight infrared proximity sensors for use in collision avoidance, piezoelectric bump sensors distributed around the base of the robot for use in collision detection, and additional bump sensors inside the gripper for use in measuring gripping force.

We note here that although these robots are of the same type and thus have the potential of maximum redundancy in capabilities, mechanical drift and failure can cause them to have quite different actual abilities. For example, one of our robots had full use of its side infrared (IR) sensors which allowed it to perform wall-following, whereas the side IR sensors of two of the other robots had become dysfunctional. The L-ALLIANCE learning and parameter update system described in [21] gives these robots the ability to take advantage of these differences and thus determine from trial to trial which team member is best suited for which task.

A radio communication system allows robot team members to communicate with each other. This radio system is integrated with a positioning system, which consists of a transceiver unit attached to each robot plus two sonar base stations for use in

triangulating the robot positions. The positioning system is accurate to about 15 centimeters and is useful for providing robots with information on their own position with respect to their environment and with respect to other robot team members.

4.2 The Hazardous Waste Cleanup Mission

Illustrated in figure 2, the hazardous waste cleanup mission requires two artificially “hazardous” waste spills in an enclosed room to be cleaned up by a team of three robots. This mission requires robot team members to perform the following distinct tasks: the robot team must locate the two waste spills, move the two spills to a goal location, while also periodically reporting the team progress to humans monitoring the system. These tasks are referred to in the remainder of this article as *find-locations*, *move-spill(left)*, *move-spill(right)*, and *report-progress*, where *left* and *right* refer to the locations of the two spills relative to the room entrance.

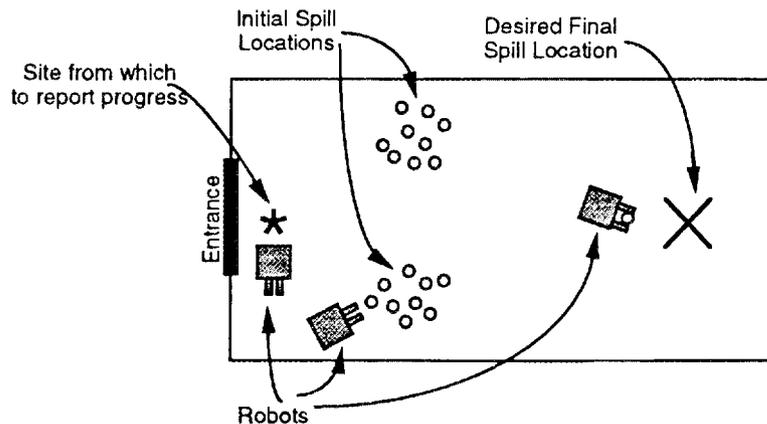


Figure 2: The experimental mission: hazardous waste cleanup.

A difficulty in this mission is that the human monitor does not know the exact location of the spills in robot coordinates, and can only give the robot team qualitative information on the initial location of the two spills and the final desired location to which the robots must move the spills. Thus, the robots are told qualitatively that one spill is located in the right half of the front third of the room, while the other spill is located in the left half of the front third of the room. Furthermore, the robots are also told that the desired final location of the spill is in the back, center of the room, relative to the position of the entrance. This information is used as described below to locate the initial and final spill locations. To prevent interference among robots, ideally only one robot at a time would attempt to find the spill, broadcasting the computed locations to the other team members once the task was complete.

Each robot was preprogrammed to have the following behavior sets, which correspond to high-level tasks that must be achieved on this mission: *find-locations-methodical*, *find-locations-wander*, *move-spill(loc)*, and *report-progress*. A low-level

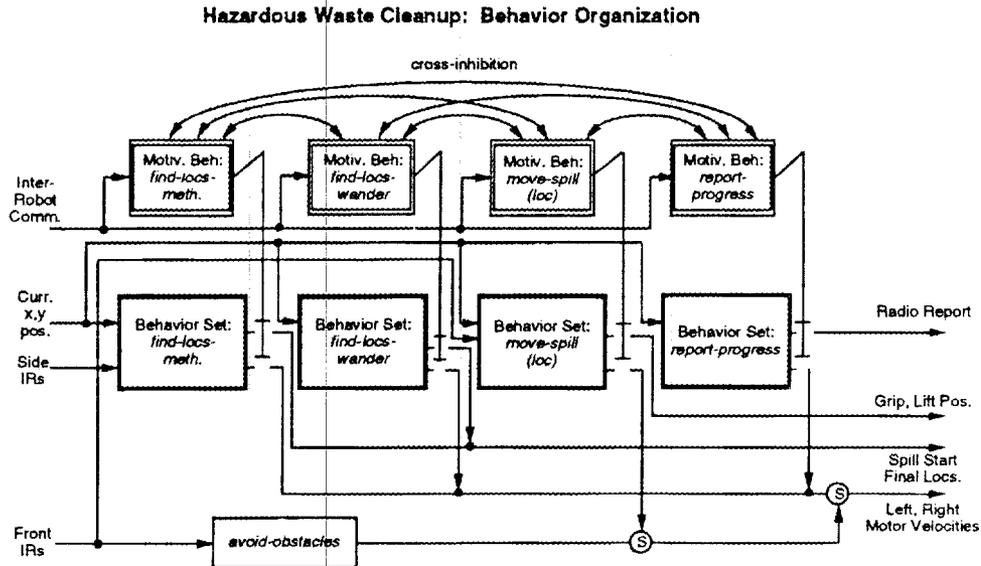


Figure 3: The ALLIANCE-based control of each robot in the hazardous waste cleanup mission. Not all sensory inputs to the behavior sets are shown here. In this figure, the high-level task achieving functions *find-locations-methodical* and *find-locations-wander* are abbreviated as *find-locs-meth* and *find-locs-wander*, respectively.

avoid-obstacles behavior was active at all times in these robots except during portions of the *move-spill* task, when it was suppressed to allow the robot to pick up the spill object. The organization of the behavior sets for this mission is shown in figure 3.

Two behavior sets are provided which both accomplish the task of finding the initial and final spill locations — *find-locations-methodical* and *find-locations-wander* — both of which depend upon the workspace being rectangular and on the sides of the room being parallel to the axes of the global coordinate system. Because of these assumptions, these behavior sets do not serve as generally applicable location-finders. However, we made no attempt to generalize these algorithms, since the point of this experiment is to demonstrate the adaptive action selection characteristics of ALLIANCE. Shown in more detail in figure 4, the methodical version of finding the spill location is much more reliable than the wander version, and involves the robot first noting its starting (or home) x, y position and then following the walls of the room using its side IRs until it has returned to its home location while tracking the minimum and maximum x and y positions it reaches. It then uses these x, y values to calculate the coordinates of the right and left halves of the front third of the room (for the two initial spill locations) and the back center of the room (for the final spill location). These locations are then made available to the *move-spill(loc)* behavior set, which requires this information to perform its task.

The wander version of finding the initial and desired final spill locations, shown in figure 5, avoids the need for side IR sensors by causing the robot to wander in each of the four directions (west, north, east, and south) for a fixed time period. While the

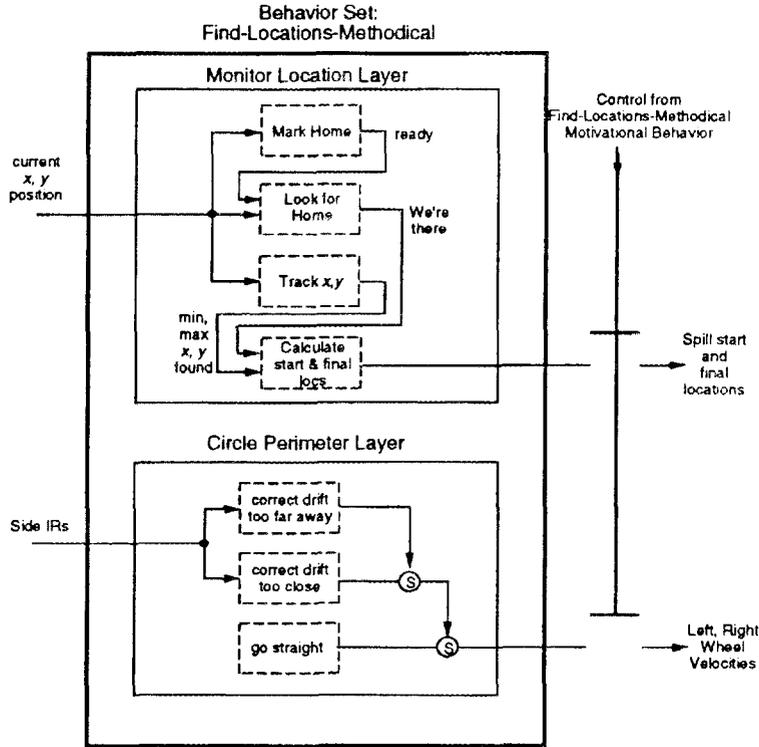


Figure 4: The robot control organization within the *find-locations-methodical* behavior set.

robot wanders, it tracks the minimum and maximum x and y positions it discovers. Upon the conclusion of the wandering phase, the robot calculates the desired initial and final locations from these minimum and maximum x, y values.

The *move-spill(loc)* behavior set, shown in more detail in figure 6, can be activated whenever there are spill objects needing to be picked up at *loc*, the locations of the initial and final spill positions are known, and the robot is not aware of any other robot currently working on the spill at *loc*. It involves having the robot (1) move to the vicinity of the initial spill location, (2) wander in a straight line through the area of the spill while using its front IR sensors to scan for spill objects, (3) “zero in” on a spill object once it is located to center it in the gripper, (4) grasp and lift the spill object, (5) move to the vicinity of the final spill location, and then (6) lower and release the spill object. To minimize interference among robots in a relatively small space, ideally only one robot at a time should work on a given spill.

The *report-progress* behavior set, shown in figure 7, corresponds to the high-level task that the robot team is required to perform approximately every 4 minutes during the mission. This task involves returning to the room entrance and informing the human monitoring the system of the activities of the robot team members and some information regarding the success of those activities. Note that this task only needs to be performed by the team as a whole every 4 minutes, not by all team members. In a real-life application of this sort, the progress report would most likely be delivered

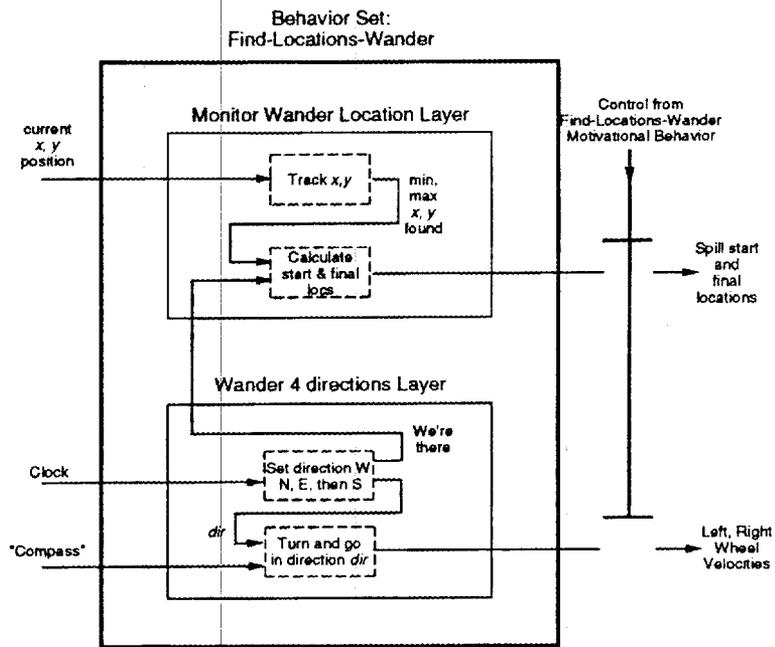


Figure 5: The robot control organization within the *find-locations-wander* behavior set.

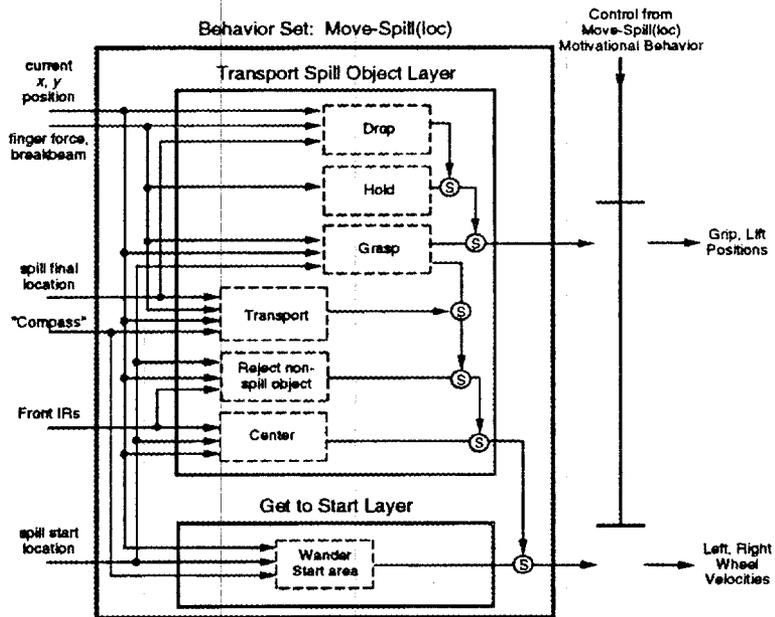


Figure 6: The robot control organization within the *move-spill(loc)* behavior set.

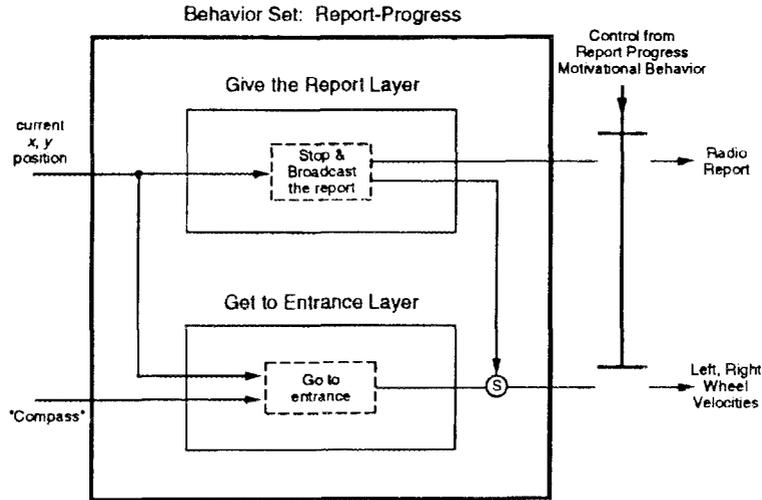


Figure 7: The robot control organization within the *report-progress* behavior set.

via a radio message to the human. However, in this experiment no actual progress information was maintained (although it could easily be accomplished by logging radioed the robot activities), and delivering the report consisted of playing an audible tune on the robot’s piezoelectric buzzer from the room entrance rather than relaying a radio message.

4.3 Experiments

We report here the experiments we conducted to test the ability of ALLIANCE to achieve fault-tolerant cooperative control of our team of mobile robots performing the hazardous waste cleanup mission. In all of the following experiments, teams of three R-2 robots were utilized in an environmental setup very similar to that depicted in figure 2; we will refer to these robots individually as GREEN, BLUE, and GOLD. All the robots began their missions at the room entrance, as shown in figure 8.

Figure 9 shows the action selection results of a typical experimental run when no robot failures occur. As reflected in this diagram, at the beginning of the mission, GREEN has the highest motivation to perform behavior set *find-locations-methodical*, causing it to initiate this action. This causes BLUE and GOLD to be satisfied for a while that the initial and final spill locations are going to be found; since no other task can currently be performed, they sit idle, waiting for the locations to be found. However, they do not idle forever waiting on the locations to be found. As they wait, they become more and more impatient over time, which can cause one of BLUE or GOLD to decide to find the spill and goal locations. Indeed, this does happen in a situation as shown in the photograph in figure 10, in which we intentionally interfere with GREEN’s ability to find the spill and goal locations. As shown in the action trace of figure 11, this leads to one of the remaining robots — namely, BLUE — to activate its *find-locations-wander* behavior set. (Note that BLUE does not activate its *find-*



Figure 8: The robot team at the beginning of the hazardous waste cleanup mission.

locations-methodical behavior set because its side infrared sensors prevent BLUE from successfully accomplishing that behavior set. As previously noted, automatic updates of the parameter settings to allow the proper prioritization of multiple methods of accomplishing the same task are possible through L-ALLIANCE, and are described in [21]. Alternatively, these settings can be accomplished readily by hand, which was done in the experiments reported here.) In this case, GREEN acquiesces its attempt to find the spill and goal locations to BLUE, since GREEN realized it was encountering difficulties of some sort. In either case, the robot finding the spill and goal locations reports these locations to the rest of the team.

At this point, the environmental feedback and knowledge of the spill and goal locations indicate to the robot team that the *move-spill(loc)* behavior set is applicable. As we see in figure 9, GREEN selects to move the *left* spill while BLUE selects to move the *right* spill. Since only one robot at a time should work on a given spill (as described in section 4.2), GOLD sits idle, satisfied that the left and right spills are going to be moved. Figure 12 shows a photograph of the robots at this stage in the mission.

In the meantime, the robots' impatience motivations to report the team's progress are increasing. Since GOLD is not performing any other tasks, it is the first to activate its *report-progress* behavior set. This reporting satisfies the remainder of the team, so they continue to move the two spills. This periodic reporting of the progress throughout the mission by GOLD is reflected in the diagrams in figures 9 and 11. In these particular examples, GOLD has effectively specialized as the progress reporting robot, whereas GREEN and BLUE have specialized as the move-spill robots. The mission continues in this way until both spills are moved from their starting location

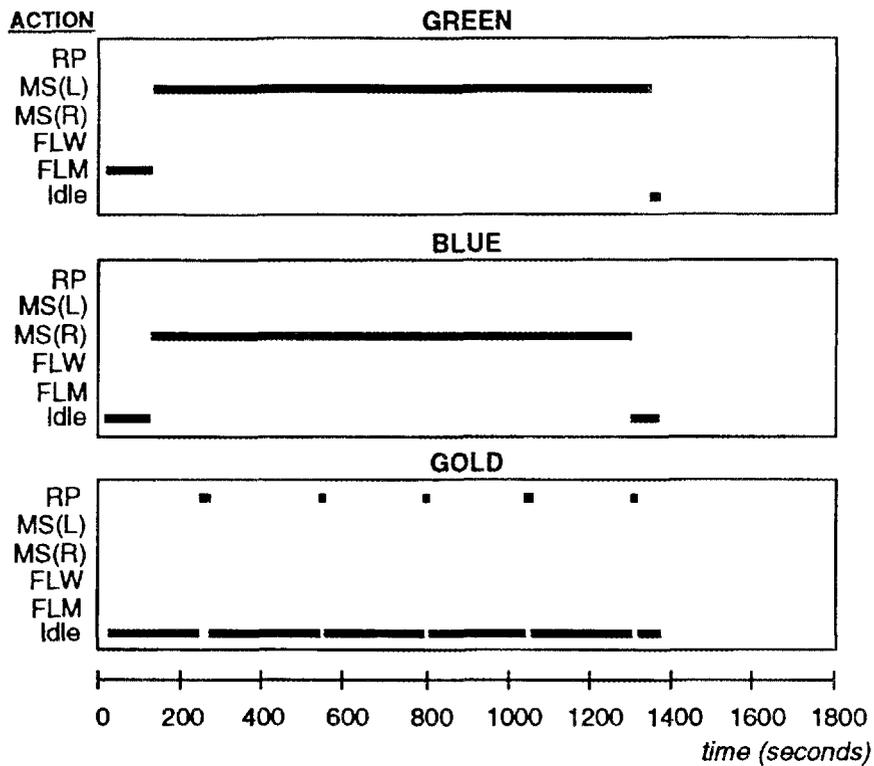


Figure 9: Typical robot actions selected during experiment with no robot failures. This is one instance of many runs of this mission. In this and the following figures showing traces of action selections, the meanings of the abbreviations are as follows: RP stands for *report-progress*; MS(L) and MS(R) stand for *move-spill(left)* and *move-spill(right)*, respectively; FLW stands for *find-locations-wander*; and FLM stands for *find-locations-methodical*.

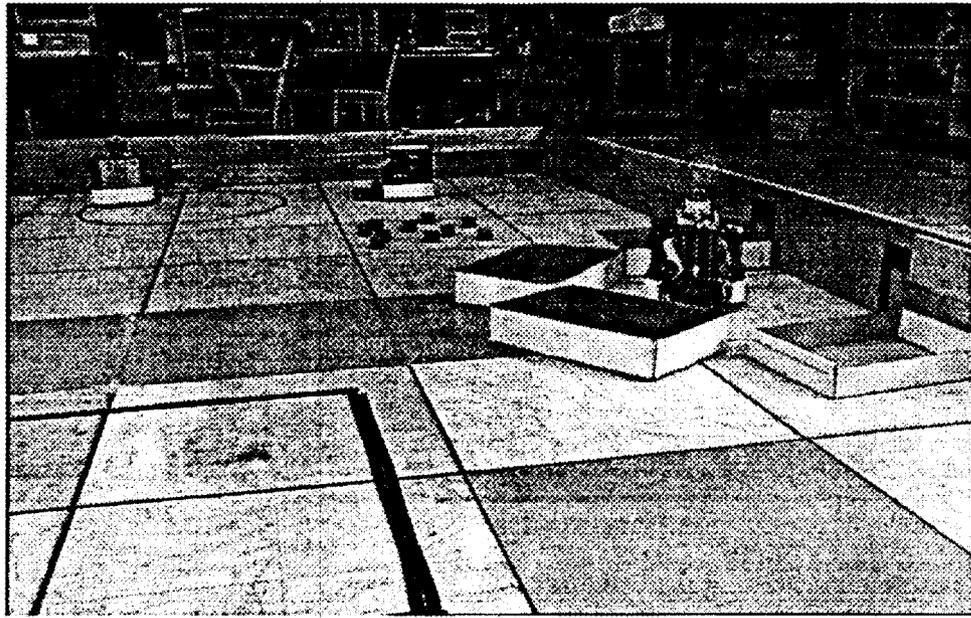


Figure 10: Here, we interfere with GREEN, which is attempting to locate the spill and goal locations, thus preventing it from completing this task.

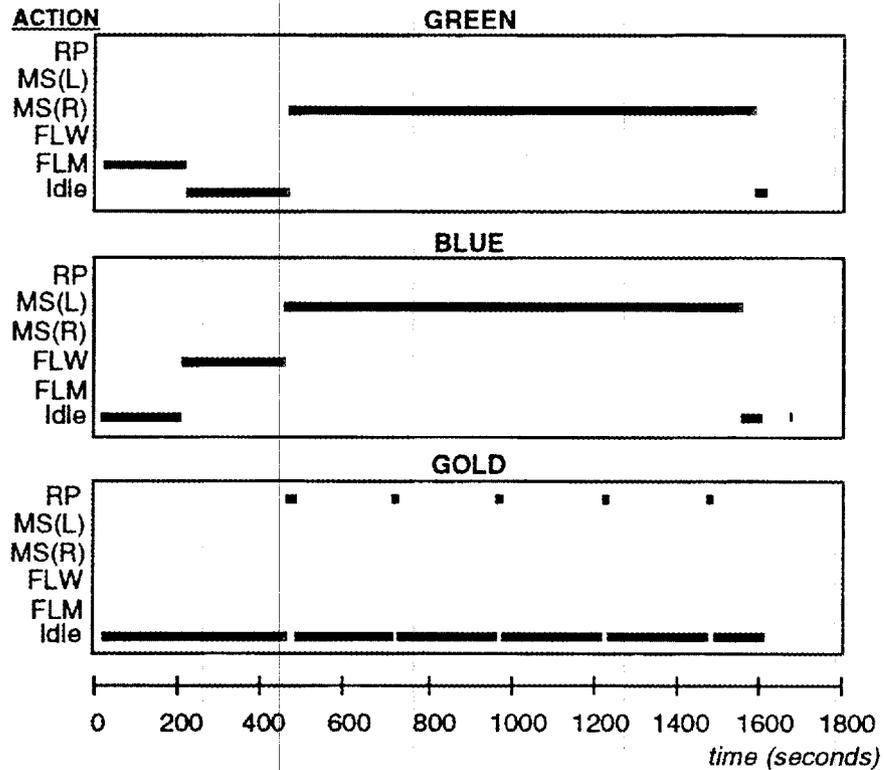


Figure 11: Typical robot actions selected during experiment when the initial robot action of finding the spill fails. This is one instance of many runs of this mission.

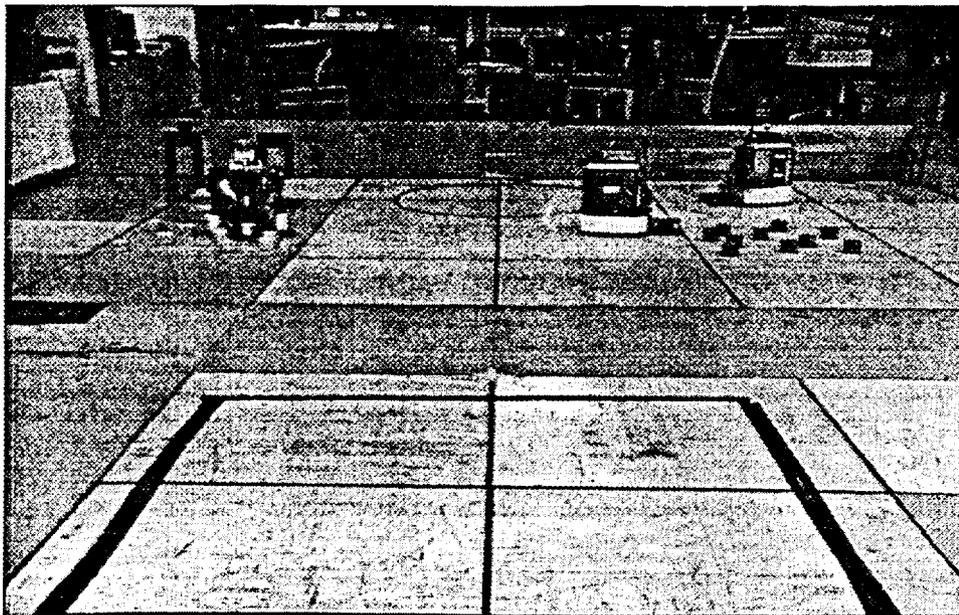


Figure 12: Now knowing the location of the two spills, two R-2 robots are in the process of moving their respective spills to the goal location.

to the goal destination. Figure 13 shows two of the robots delivering spill objects to the goal destination.

To illustrate the effect of unexpected events on the action selection of the team, we next experimented with dynamically altering the composition of the team during the mission. Figure 14 shows the effect on the mission when we removed BLUE from the team. This caused GOLD to become impatient that the *right* spill was not being moved, which in turn caused GOLD to activate its behavior set to move the *right* spill. However, this then effectively removes from the team the robot that is performing all of the progress reports, leading the remaining two robots — GREEN and GOLD — to have to interrupt their spill-moving activities to occasionally report the progress. A similar effect can be observed in figure 15, when we remove GOLD from the team.

4.4 Discussion

These experiments illustrate a number of primary characteristics we consider important in developing cooperative robotic teams. First of all, the cooperative team under ALLIANCE control is robust, in that robots are allowed to continue their actions only as long as they demonstrate their ability to have the desired effect on the world. This was illustrated in the experiments by BLUE and GOLD becoming gradually more impatient with GREEN's search for the spill. If GREEN did not locate the spill in a reasonable length of time then one of the remaining robots would take over that task, with GREEN acquiescing the task.

Secondly, the cooperative team is able to respond autonomously to many types

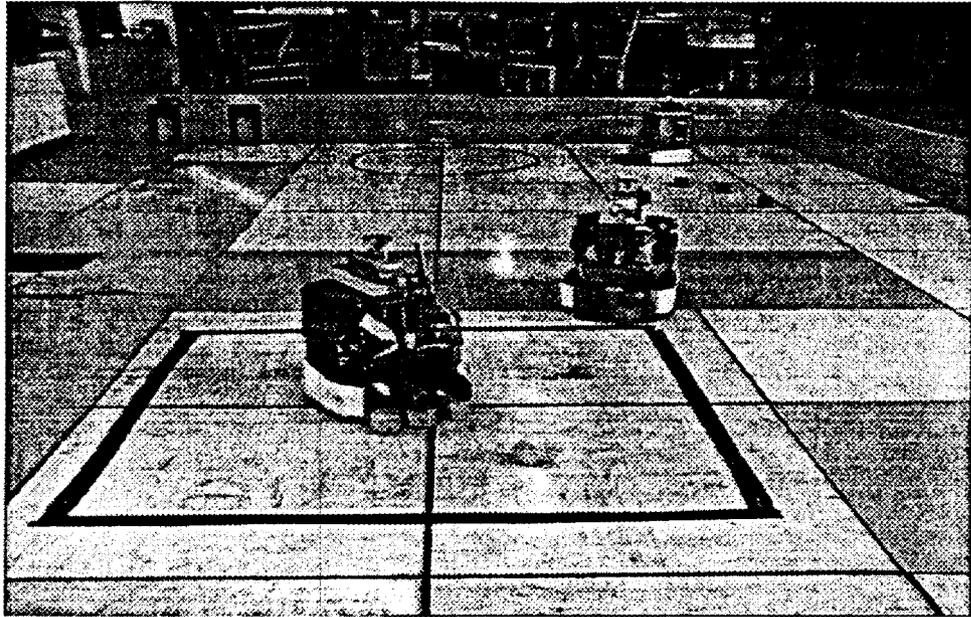


Figure 13: Robots delivering spill objects to the goal destination.

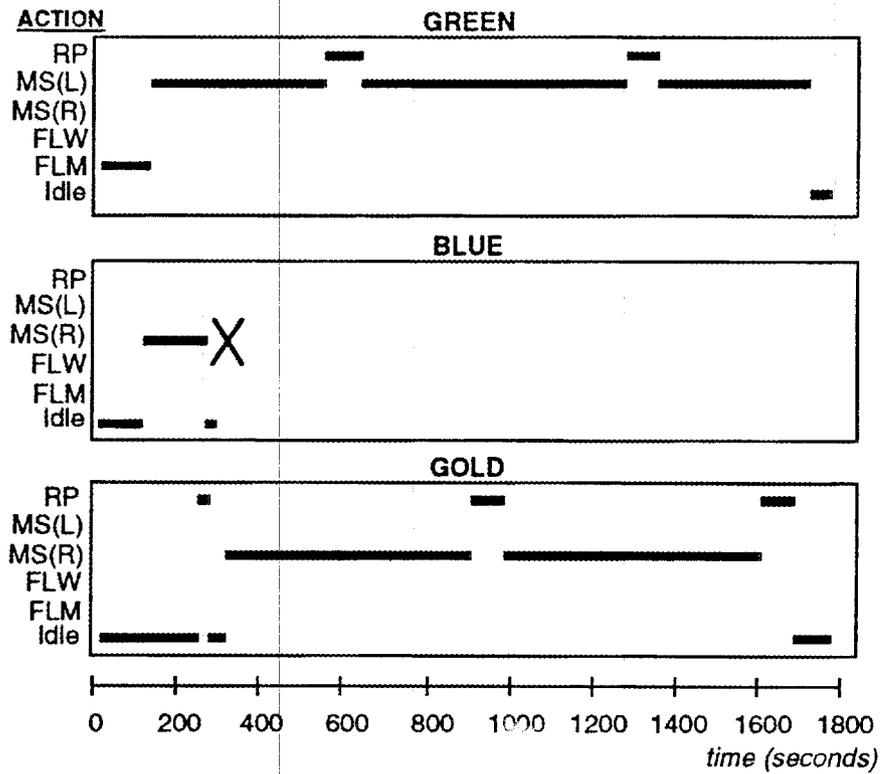


Figure 14: Typical robot actions selected during experiment when one of the robot team members which is moving a spill is removed. This is one instance of many runs of this mission. In this and the following figure, the "X" indicates the point in time when the corresponding robot was removed from the robot team.

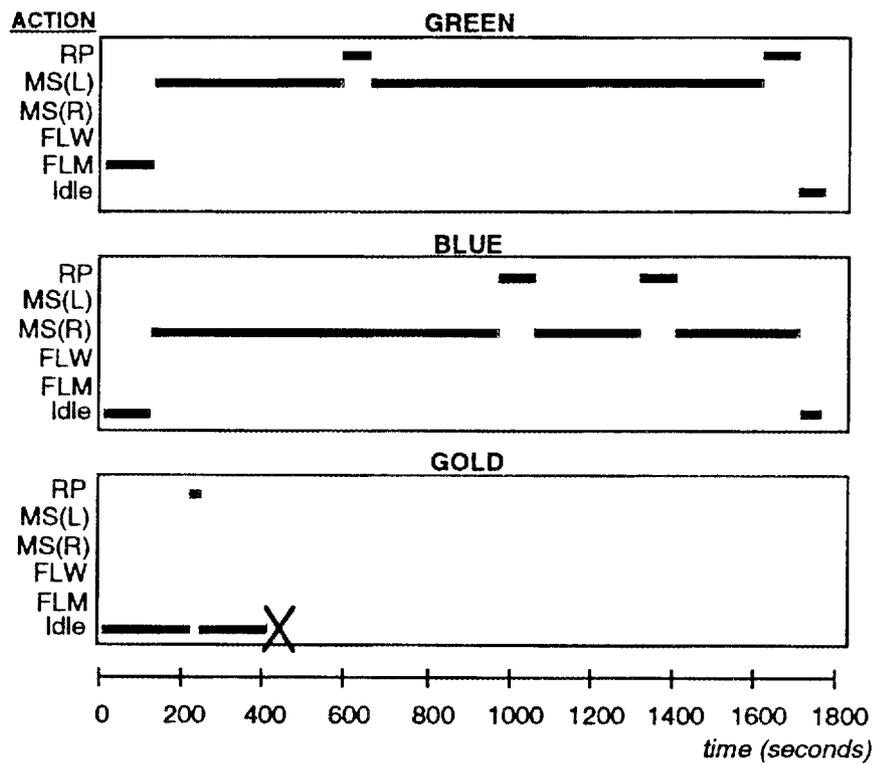


Figure 15: Typical robot actions selected during experiment when one of the robot team members which is reporting the progress is removed. This is one instance of many runs of this mission.

of unexpected events either in the environment or in the robot team without the need for external intervention. As we illustrated, at any time during the mission, we could disable or remove robot team members, causing the remaining team members to perform those tasks that the disabled robot would have performed. Clearly, we could also have easily increased or decreased the size of the spill during the mission and the robots would not be adversely affected.

Third, the cooperative team need have no *a priori* knowledge of the abilities of the other team members to effectively accomplish the task. As previously noted, the learning/ parameter update system, L-ALLIANCE, does allow the team to improve its efficiency on subsequent trials whenever familiar robots are present [21]; however, a description of this mechanism is beyond the scope of this article.

Other characteristics of ALLIANCE have also been studied which show that this architecture allows robot teams to accomplish their missions even when the communication system providing it with the awareness of team member actions breaks down. Although the team's performance in terms of time and energy may deteriorate, at least the team is still able to accomplish its mission. Refer to [21] for a deeper discussion of these and related issues.

5 Conclusions

We have presented a fully distributed, behavior based architecture called ALLIANCE, which facilitates fault tolerant mobile robot cooperation. A number of key characteristics of ALLIANCE provide these fault tolerant cooperative features. ALLIANCE enhances team robustness through the use of the motivational behavior mechanism which constantly monitors the sensory feedback of the tasks that can be performed by an individual robot, adapting the actions selected by that robot to the current environmental feedback and the actions of its teammates. Whether the environment changes to require the robots to perform additional tasks or to eliminate the need for certain tasks, ALLIANCE allows the robots to handle the changes fluidly and flexibly. This same mechanism allows robot team members to respond to their own failures or to failures of teammates, leading to adaptive action selection to ensure mission completion. ALLIANCE further enhances team robustness by making it easy for robot team members to deal with the presence of overlapping capabilities on the team. The ease with which redundant robots can be incorporated on the team provides the human team designer the ability to utilize physical redundancy to enhance the team's fault tolerance. This is the first cooperative control architecture for multi-robot cooperation that has achieved this level of fault tolerance, and which has been demonstrated on actual mobile robot teams.

6 Acknowledgements

The author wishes to thank Prof. Rodney A. Brooks of the Massachusetts Institute of Technology's Artificial Intelligence Laboratory, who supervised this research, and the IS Robotics Corporation for building the robots.

Support for this research was provided in part by the University Research Initiative under Office of Naval Research contract N00014-86-K-0685, in part by the Advanced Research Projects Agency under Office of Naval Research contract N00014-85-K-0124, and in part by the Mazda Corporation. Additional support has been provided by the Office of Engineering Research Program, Basic Energy Sciences, of the U.S. Department of Energy, under contract No. DE-AC05-84OR21400 with Martin Marietta Energy Systems, Inc.

References

- [1] Ronald C. Arkin, Tucker Balch, and Elizabeth Nitz. Communication of behavioral state in multi-agent retrieval tasks. In *Proceedings of the 1993 International Conference on Robotics and Automation*, pages 588–594, 1993.
- [2] H. Asama, K. Ozaki, A. Matsumoto, Y. Ishida, and I. Endo. Development of task assignment system using communication for multiple autonomous robots. *Journal of Robotics and Mechatronics*, 4(2):122–127, 1992.
- [3] Gerardo Beni and Jing Wang. On cyclic cellular robotic systems. In *Japan – USA Symposium on Flexible Automation*, pages 1077–1083, Kyoto, Japan, 1990.
- [4] Alan Bond and Less Gasser. *Readings in Distributed Artificial Intelligence*. Morgan Kaufmann, 1988.
- [5] Rodney A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, RA-2(1):14–23, March 1986.
- [6] Rodney A. Brooks. Elephants don't play chess. *Robotics and Autonomous Systems*, 6:3–15, 1990.
- [7] Philippe Caloud, Wonyun Choi, Jean-Claude Latombe, Claude Le Pape, and Mark Yim. Indoor automation with many mobile robots. In *Proceedings of the IEEE International Workshop on Intelligent Robots and Systems*, pages 67–72, Tsuchiura, Japan, 1990.
- [8] Paul Cohen, Michael Greenberg, David Hart, and Adele Howe. Real-time problem solving in the Phoenix environment. COINS Technical Report 90-28, University of Massachusetts at Amherst, 1990.
- [9] J. Deneubourg, S. Goss, G. Sandini, F. Ferrari, and P. Dario. Self-organizing collection and transport of objects in unpredictable environments. In *Japan-U.S.A. Symposium on Flexible Automation*, pages 1093–1098, 1990.
- [10] Alexis Drogoul and Jacques Ferber. From Tom Thumb to the Dockers: Some experiments with foraging robots. In *Proceedings of the Second International Conference on Simulation of Adaptive Behavior*, pages 451–459, 1992.
- [11] R. E. Fikes and N. J. Nilsson. STRIPS: a new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2(3/4):189–208, 1971.
- [12] T. Fukuda, S. Nakagawa, Y. Kawauchi, and M. Buss. Self organizing robots based on cell structures — CEBOT. In *Proceedings of 1988 IEEE International Workshop on Intelligent Robots and Systems (IROS '88)*, pages 145–150, 1988.

- [13] C. Ronald Kube and Hong Zhang. Collective robotic intelligence. In *Proceedings of the Second International Workshop on Simulation of Adaptive Behavior*, pages 460–468, 1992.
- [14] Pattie Maes. How to do the right thing. *Connection Science*, 1(3):291–323, 1989.
- [15] Maja Mataric. Designing emergent behaviors: From local interactions to collective intelligence. In J. Meyer, H. Roitblat, and S. Wilson, editors, *Proceedings of the Second International Conference on Simulation of Adaptive Behavior*, pages 432–441. MIT Press, 1992.
- [16] Fabrice R. Noreils. Toward a robot architecture integrating cooperation between mobile robots: Application to indoor environment. *The International Journal of Robotics Research*, 12(1):79–98, February 1993.
- [17] Lynne E. Parker. Adaptive action selection for cooperative agent teams. In Jean-Arcady Meyer, Herbert Roitblat, and Stewart Wilson, editors, *Proceedings of the Second International Conference on Simulation of Adaptive Behavior*, pages 442–450. MIT Press, 1992.
- [18] Lynne E. Parker. ALLIANCE: An architecture for fault tolerant, cooperative control of heterogeneous mobile robots. In *Proceedings of the 1994 IEEE/RSJ/GI International Conference on Intelligent Robots and Systems (IROS '94)*, pages 776–783, Munich, Germany, September 1994.
- [19] Lynne E. Parker. An experiment in mobile robotic cooperation. In *Proceedings of the ASCE Specialty Conference on Robotics for Challenging Environments*, Albuquerque, NM, February 1994.
- [20] Lynne E. Parker. Fault tolerant multi-robot cooperation. MIT Artificial Intelligence Lab Videotape AIV-9, December 1994.
- [21] Lynne E. Parker. *Heterogeneous Multi-Robot Cooperation*. PhD thesis, Massachusetts Institute of Technology, Artificial Intelligence Laboratory, Cambridge, MA, February 1994. MIT-AI-TR 1465 (1994).
- [22] Luc Steels. Cooperation between distributed agents through self-organization. In Yves Demazeau and Jean-Pierre Muller, editors, *Decentralized A.I.* Elsevier Science, 1990.
- [23] Daniel Stilwell and John Bay. Toward the development of a material transport system using swarms of ant-like robots. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 766–771, 1993.
- [24] Guy Theraulaz, Simon Goss, Jacques Gervet, and Jean-Louis 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*, pages 346–355, 1990.

- [25] Jing Wang. DRS operating primitives based on distributed mutual exclusion. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1085–1090, Yokohama, Japan, 1993.

INTERNAL DISTRIBUTION

- | | |
|---------------------|------------------------------|
| 1. J. Barhen | 19. S. Shekhar |
| 2. J. E. Baker | 20. R. F. Sincovec |
| 3. M. Beckerman | 21. M. S. Smith |
| 4. C. W. Glover | 22. E. C. Uberbacher |
| 5. J. P. Jones | 23. M. A. Unseren |
| 6. H. E. Knee | 24-25. Laboratory Records |
| 7. R. C. Mann | Department |
| 8. E. M. Oblow | 26. Laboratory Records, |
| 9. C. E. Oliver | ORNL-RC |
| 10-14. L. E. Parker | 27. Document Reference |
| 15. V. Protopopescu | Section |
| 16. S. A. Raby | 28. Central Research Library |
| 17. N. S. V. Rao | 29. ORNL Patent Section |
| 18. D. B. Reister | |

EXTERNAL DISTRIBUTION

30. Dr. Peter Allen, Department of Computer Science, 450 Computer Science, Columbia University, New York, NY 10027
31. Dr. Wayne Book, Department of Mechanical Engineering, J. S. Coon Building, Room 306, Georgia Institute of Technology, Atlanta, GA 30332
32. Dr. Steven Dubowsky, Department of Mechanical Engineering, Massachusetts Institute of Technology, 77 Massachusetts Ave., Building 3, Room 469A, Cambridge, MA 02139
33. Mr. Steve Holland, Robotics, B/MD-63, General Motors Corporation, NAO Manufacturing Center, 30300 Mound Rd., Warren, MI 48090-9040
34. Dr. Avi Kak, Robot Vision Lab, Department of Electrical Engineering, Purdue University, Northwestern Ave., Engineering Mall, West Lafayette, IN 47907
35. Dr. Oscar P. Manley, Division of Engineering, Mathematical, and Geosciences, Office of Basic Energy Sciences, ER-15, U.S. Department of Energy-Germantown, Washington, DC 20545
36. Dr. Wes Snyder, Department of Radiology, Bowman Gray School of Medicine, North Carolina Baptist Hospital School of Medicine, 300 S. Hawthorne Dr., Winston-Salem, NC 27103
37. Office of Assistant Manager, Energy Research and Development, Department of Energy, Oak Ridge Operations, Oak Ridge, TN 37831
- 38-39. Office of Scientific and Technical Information, P.O. Box 62, Oak Ridge, TN 37831