NOTE TO USERS

This reproduction is the best copy available.

GREMLINS: AN ARCHITECTURAL FRAMEWORK FOR RECONFIGURABLE AUTONOMOUS ROBOTS

by

James Gaston, B. Eng
Ryerson University, 2003

A thesis
presented to Ryerson University
in partial fulfillment of the
requirements for the degree of
Master of Applied Science
in the Program of
Electrical & Computer Engineering

Toronto, Ontario, Canada, 2005

Author's Declaration

I hereby declare that I am the sole author of this thesis.

I authorize Ryerson University to lend this thesis to other institutions or individuals for the purpose of scholarly research.

I further authorize Ryerson University to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

Abstract

Gremlins: An Architectural Framework for Reconfigurable Autonomous Robots
James Gaston
Master of Applied Science
Electrical & Computer Engineering
Ryerson University 2005

.e work area of a team of small robots is limited by their inability to traverse a very nmon obstacle: stairs. We present a complete integrated control architecture and com- ınication strategy for a system of reconfigurable robots that can climb stairs. A modular)ot design is presented which allows the robots to dynamically reconfigure to traverse 'tain obstacles. This thesis investigates the implementation of a system of autonomous)ots which can cooperatively reconfigure themselves to collectively traverse obstacles such stairs. We present a complete behavior and communication system which facilitates this tonomous reconfiguration. The layered behavior-based control system is fault-tolerant and ;ends the capabilities of a control architecture known as ALLIANCE. Behavior classes are roduced as a mechanism for managing ordering dependencies and monitoring a robot's)gress through a particular task. The communication system compliments the behavioral ıtrol and implements inherent robot failure detection without the need for a base station external monitor. The behavior and communication systems are validated by implement- ; them on a mobile robot platform synthesized specifically for this research. Experimental als showed that the implementation of the behavior control systems was successful. The ıtrol system provided robust, fault-tolerant performance even when robots failed to per- m docking tasks while reconfiguring. Once the robots reconfigure to form a chain, a ferent control scheme based on gait control tables coordinates the individual movements the robots. Several successful stair climbing trials were accomplished. Improvements to ; mechanical design are proposed.

Acknowledgment

I would like to thank Dr. Kaamran Raahemifer and Professor Peter Hiscocks who served as my supervisors during this research. Dr. Raahemifar was always quick to offer encouragement and was very supportive. Professor Hiscocks, as always, was invaluable and provided practical suggestions and solutions to many challenges. I would also like to thank Dr. S. Krishnan and the Ryerson University Electrical Engineering Department.

To my loving wife, Karen, my thanks for her unending patience and loving support through this project, graduate studies, and my entire post-secondary career. I am also thankful to my parents for their support and once again allowing me to destroy their house in the pursuit of higher education.

James Gaston

Coming together is a beginning, staying together is progress, and working together is success

Henry Ford

Contents

List of Figures

List of Tables

;hapter 1

ntroduction

MULTI-AGENT robot systems facilitate the completion of tasks that are either too complex, too time consuming, or physically impossible for a single robot to accomsh alone. Cooperative robotic systems can be inherently more fault tolerant and adaptable ın their single complex and expensive robot counterparts.

otivation

.e of the drawbacks to a system of small, cooperative robots is their limited mobility. ıile such robots are typically highly maneuverable they are usually dwarfed by common ;tacles that humans overcome with ease. Currently there are no successful examples of autonomous multi-agent robotic system that implements stair climbing. The purpose of s work is to create a general purpose control and communication architecture that can be olied to tasks such as coordinating a system of robots that can collectively traverse stairs. e robots accomplish this feat by mechanically reconfiguring the entire robot system. The)ots autonomously couple together to combine their mechanical functionalities and create ıew entity with the capability to traverse stairs. The implementation of such a system :atly expands the capabilities and potential of multi-agent robotic systems, especially in loor environments. Now small groups of collaborative robots are no longer limited to an erational theater of a single floor.

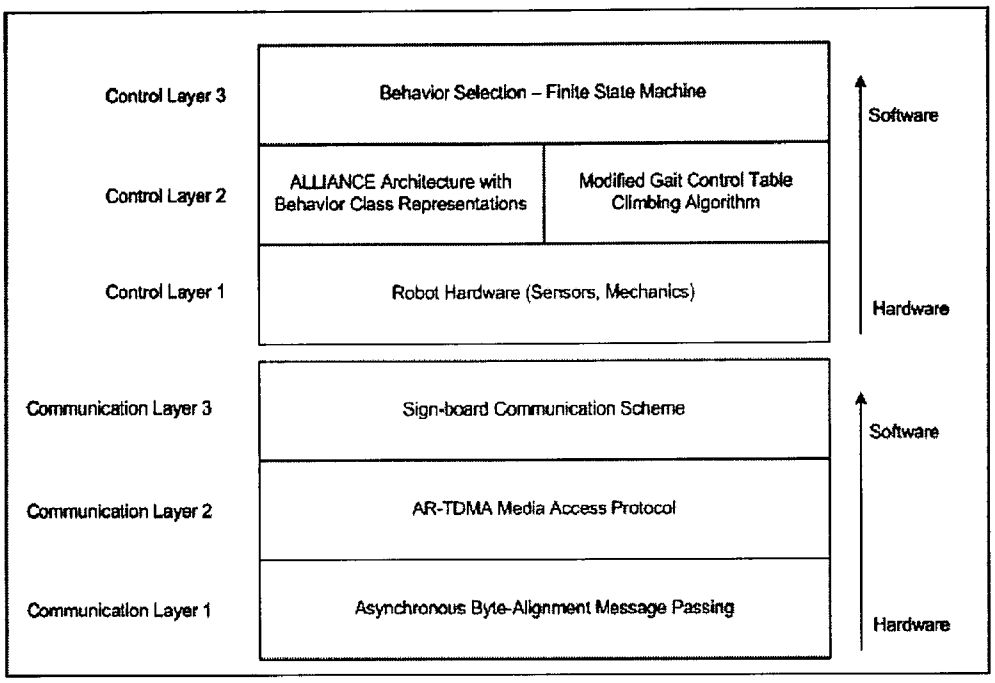


Figure 1.1: *Overview of the Proposed Control and Communication Architecture*

Contributions of this Work

We propose a complete integrated control architecture and communication strategy for a system of reconfigurable robots that perform tasks with ordering dependencies. The communication strategy to autonomously maintain a structured message passing network without the need for external supervision. To our knowledge, this is the first implementation of such an architecture that facilitates cooperative autonomous docking and reconfiguration of robots. Figure 1.1 shows the proposed control and communication architecture.

The Gremlin robots serve as a platform for the verification of the control and communication architecture. The contributions of this thesis to the current body of research are:

- A behavioral control architecture and communication system, integrated to facilitate autonomous mechanical reconfiguration of robots.

- Extension of a behavior-based control architecture called ALLIANCE which improve the efficiency of cooperative tasks with ordering dependencies.

- A new type of gait control table for motion planning is proposed which includes sensor feedback and implements a mechanical travelling wave on a set of linked stair climbing robots.

- A high level sign-board based communication scheme is integrated into the architecture which complements the behavioral system.

- A time domain multiple access (TDMA) communication scheme that allows the robot team to maintain a structured message passing system without the supervision of a base station or computer.

- A simple mechanical design augmented by a suitable control system.

This research represents the first successful implementation of cooperative autonomous stair climbing robots that we are aware of.

ıe Challenge of Stair Climbing

e difficulty of creating a system of reconfigurable, autonomous stair-climbing robots is ;hlighted in [1]. In this paper, the author outlined a system of autonomous robots that uld couple together using electromagnets and would use rotating armatures to lift one ɔther up the stairs. The author provided a comprehensive list of the challenges associated ;h this type of research including:

- Designing a power system substantial enough to sustain large motors and electromagnets for coupling and climbing

- Designing a control system to manage autonomous operation, reconfiguration and climbing

- Incorporating sensors to facilitate autonomous reconfiguration and climbing

- Designing a multi-agent communication mechanism

- Providing a mechanism for localization between robots

- Creating a physical design that is small and compact

The proposed research implements a system of autonomous, behavior-based robots that can dynamically and autonomously reconfigure to climb stairs. This implementation is achieved by building on previous research in the fields of behavior-based robot control, multi-agent robot communication, reconfigurable robots, and kinematic modelling. This work represents the first successful practical implementation of the fusion of these research fields.

The following outlines the goals of the proposed research:

- Develop a system of small autonomous mobile robots that can cooperatively climb stairs.
- The robots must function independently of each other when not climbing stairs to maintain the benefits of a multi-agent robotic system.
- The robots should be simple. Instead, the focus will be placed on the behavioral control of the robots.
- The robots should perform reconfiguration and climbing completely autonomously. No external control or communication supervision will be supplied.
- The robots should be as inexpensive as possible.

There were many electrical, mechanical, and conceptual challenges that are overcome to make the proposed implementation successful. Some of the most significant design challenges are:

- A mechanical lifting system that was simple, yet provided a mechanism that was strong and reliable for stair climbing
- A docking mechanism that was simple yet allowed a certain degree of misalignment
- A coupling mechanism to enable robot reconfiguration

- An electrical system that could implement the behavioral control system, control the mechanical systems, and fit within the constraints of the physical design of the robot
- An optical localization system to facilitate autonomous docking for reconfiguration
- A communication system using low cost hardware that can support multi-robot communication without a base station

ganization of the Thesis

is chapter has provided a brief introduction to the challenges of robot stair climbing. e following chapter expands on current research in the field of robotic stair climbing and o presents a brief history of mobile robotics and multi-agent robot research. Chapter ntroduces a behavior-based robot control structure that is robust, reliable, and, as was own in this thesis, well suited to autonomous robot reconfiguration. Chapter 4 presents nponents of a communication framework that is integrated with the control structure sented in Chapter 3, and facilitated inter-robot communication in this project. Chapter resents the implementation of stair climbing robots from mechanical and electrical design, behavioral control, and inter-robot communication. Chapter 7 outlines the experimental ification of this design and Chapter 8 presents conclusions based on observations from se experiments.

Chapter 2

Related Work

THIS chapter provides a background on multi-agent robotic systems and looks at the challenge of robot stair climbing by examining other stair climbing projects. Finally, the challenge of robot stair climbing in the realm of multi-agent robotic systems is defined.

2.1 Mobile Robotics Research

The field of robotics is vast and varied. Research ranges from industrial applications, to space exploration. Some researchers focus on mechanical design and functionality while others focus on robot control and behavior. The United Nations World Robotics survey [2] predicts a sevenfold increase in the use of robots for domestic use by the year 2007. The survey predicts that 4.1 million robots will be in use performing duties such as mowing lawns, vacuuming floors, and guarding homes. Towards the end of this decade, robots will be commonplace in such activities as assisting the elderly and handicapped, performing surgery, and dealing with hazardous situations such as fighting fires and disarming bombs.

One of the first mobile robots was the *Tortoise* developed by Grey Walter [1] in 1950. The robot consisted of vacuum tube amplifiers that drove a relay system used to control the steering and drive motors. The Tortoise exhibited simple tropism behaviors and would automatically recharge itself.

Ten years later a robot called the *Hopkins Beast* was created at the Johns Hopkins University [5] which navigated using sonar and was able to recharge itself by docking with

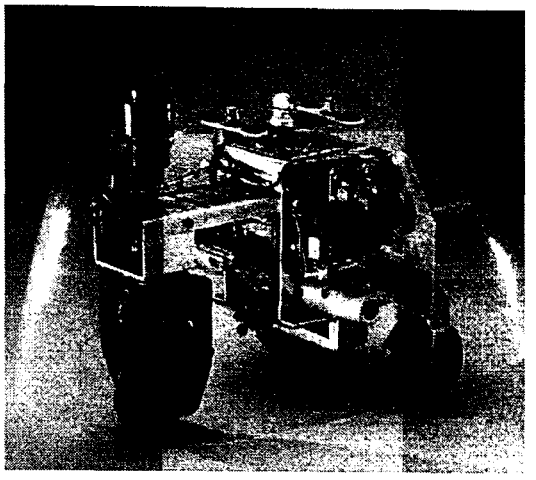


Figure 2.1: *The Tortoise [3]*

ll outlets.

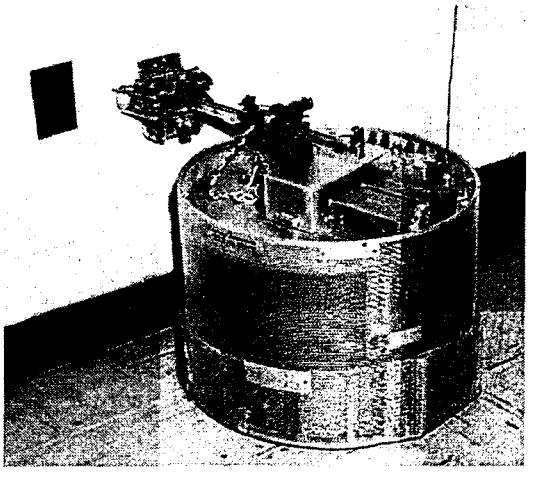


Figure 2.2: *The Hopkins Beast [6]*

A robot named *Shakey* was one of the first mobile robots to be controlled by a computer . Shakey, developed in 1970, used a television camera to navigate an office area and s able to recognize specially colored objects. The success of Shakey led to the *Stanford rt*, and the *CMU Rover* which implemented environment mapping using stereo vision and rasonic sensors, respectively.

Recently, large commercial companies have joined academic institutions in the devel-

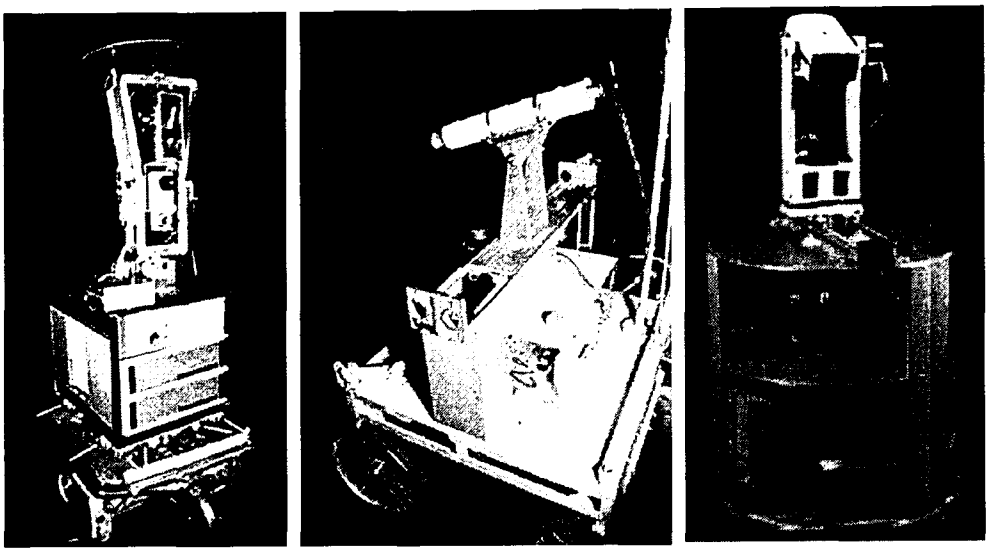


Figure 2.3: *(a) Shakey [8] (b) Stanford Cart (c) CMU Rover [9]*

opment of mobile robots. While historically limiting their robotics research to industrial applications, companies such as Sony and Honda have developed consumer robots such as the *AIBO* [10] robotic dog and the *Asimo* [11] humanoid robot.

2.2 Multi-Agent Robotic Systems Background

Multi-agent or distributed robotic systems are attractive because they allow for the completion of tasks that cannot be accomplished by a single robot, and the efficiency and productivity of a system can also benefit from using multiple robots [14]. Cooperative robotic systems can be inherently more fault tolerant and adaptable than a comparable single robot system.

The study of cooperative robotics is relatively new when compared to other fields of robotics research. Research into reconfigurable robot systems, motion planning, and cooperation architectures were popular during the birth of multi-robot systems research about 20 years ago [15]. Reconfigurable robot systems, a subset of cooperative robotics, grew out of cellular robotics which was actually an extension of distributed computing research [16]. Motion planning research has included studies of multi-robot movement in formations, as

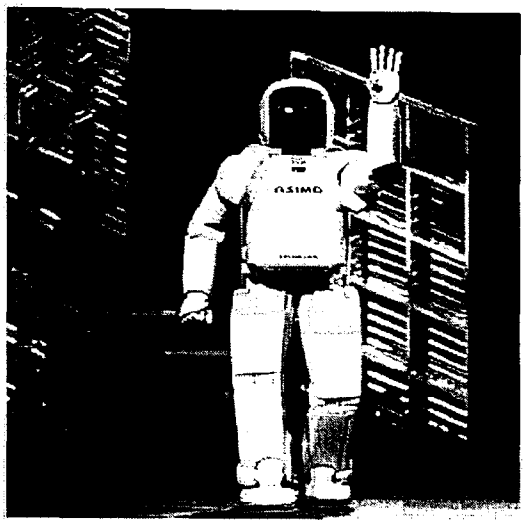


Figure 2.4: *The Honda Asimo Robot [12]*

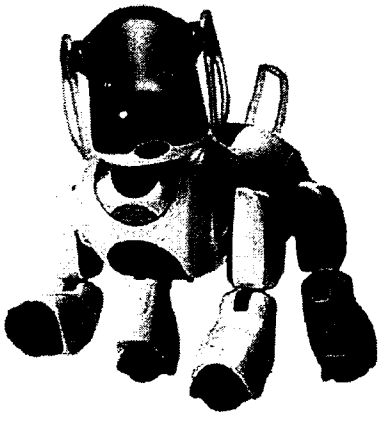


Figure 2.5: *The Sony AIBO Robot [13]*

well as path planning for situations where multiple robots must coexist in a finite amount of space without excessively interfering with the mobility of other robots [17–19]. Current research in multi-agent robotic systems has continued to expand to include research pertaining to communication, learning, biological parallels, localization, and mapping and exploration [20].

Distributed robotic systems have a plethora of applications including cooperative exploration, creating ad-hoc mobile communication networks to exchange sensor information, performing surveillance, or performing cooperative transportation of objects. One of the issues that can effect the usefulness and effectiveness of a robot, or a system of robots, working at a real world task such as these is mobility. Real world environments are dynamic and cluttered. Small robots, for example, can be thwarted by common indoor obstacles such as furniture or stairs. Larger robots can overcome these obstacles by virtue of their size and power output, but are poorly suited to navigating tight spaces, moving with stealth, and working in large groups.

2.3 Robot Stair Climbing Research

Robot stair climbing research can be divided into two broad categories: single robot systems, and multi-agent systems. All multi-agent systems use reconfigurable robots. The amount of research conducted concerning single robot stair climbing systems far outweighs that of multi-agent systems. This section describes the current state of research in both of these categories.

2.3.1 Single-Robot Systems

This section has provides an overview of single-robot stair climbing research by highlighting some of the notable projects. Robots such as URBIE [21], and QRIO [22] are successful implementations of fully autonomous robot stair climbing. Many other projects demonstrate viable mechanical platforms for stair climbing robots, but have yet to demonstrate autonomous operation.

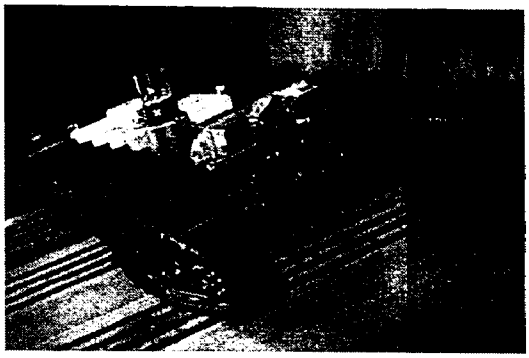


Figure 2.6: *Urbie Tactical Mobile Robot [23]*

·ban II Vehicle

mentioned earlier, there has been a substantial amount of research performed concerning gle-robot stair climbing systems. These robots are typically large and use wheels, tracks, legs for locomotion. The *Urban II Vehicle*, nicknamed *Urbie*, was developed in 2001 as rt of the Tactical Mobile Robot program funded by DARPA [24] and was a joint effort of e NASA Jet Propulsion Laboratory at the California Institute of Technology, the iRobot rporation, and the Robotics Institute of Carnegie Mellon University [21]. Physically, Ur- is a large tracked robot with a mass of approximately 20kg. It uses three gyroscopes, ctrolytic tilt sensors, and a LADAR laser scanner [25] to implement vision-guided au- nomous stair climbing. The Urbie robot is one of the most successful implementations autonomous stair climbing. In its "Autonomous Mode" the robot will autonomously ap- oach and climb stairs, traverse a landing and continue climbing the next set of stairs using sensors [26].

ELIOS Off-Road Robot

ıe development of the *HELIOS* series of off-road robots at the Tokyo Institute of Technol- y [27] in 1999 focused mainly on the mechanical design of the robot. The researchers in s project focussed on traction mechanics on the robot. The vehicle itself is equipped with ır low-pressure tires, two high-pressure tires, and two continuously variable transmissions. ıe most recent incarnation of the HELIOS series, the HELIOS V, was able to climb stairs

Figure 2.7: *The HELIOS Off-Road Robot [28]*

Figure 2.8: *Octopus Wheeled Climbing Robot [30]*

with a 16cm rise and 30cm tread by using the low-pressure tires to grip the edge of the stairs. The researchers note, however, that the vehicle would not be able to climb all types of stairs, and different wheels may be needed to traverse stairs with a different geometry [27]. The Helios robots are operated under manual control and offer no autonomous operation.

Octopus Legged and Wheeled Robot

Researchers at the Swiss Federal Institute of Technology in Switzerland have combined the concepts of legged and wheeled robots to create *Octopus* a robot with eight motorized wheels and 15 degrees of freedom [29]. Like the Helios robots, development of the Octopus was mainly focussed on mechanical design. In 2002, the researchers concentrated their efforts on integrating tactile wheels capable of detecting physical contact with the terrain. In [29], a stair-climbing algorithm using the Octopus hardware is formulated, but the researchers have yet to test the mechanical design and are planning to implement a controller for the robot.

:tal Wheeled Stair Climber

joint research project between the Japanese National Defense Academy and the Depart-nt of Mechanical Engineering at the Tokyo National College of Technology yielded another al wheeled stair climbing robot [31]. The project focussed mainly on the distributed con-l system necessary to manage the eight wheels of the robot, but was able to successfully nonstrate the traversal of a small staircase. Autonomous traversal of the stairs was ac-nplished using obstacle detectors mounted near the wheels on each of the arms.

g-Wheeled Robot

interesting hybrid between wheeled and bi-pedal robots was developed at the Gifu Labo-ory in Japan by Matsumoto et al [32]. This robot was bipedal, but had wheels for "feet" gure 2.9). The *Leg-Wheeled* robot was designed to overcome the shortcomings of its pre-:essor who was unable to traverse stairs with short treads. The researchers involved with s project proposed a trajectory planning method for control of the robot while ascending irs. The researchers modelled the mechanical system of the robot and conducted simu-ions before implementing the climbing algorithm on the physical robot. The researchers :cessfully used the control scheme to have the robot climb a set of stairs. The researchers l not, however, address the fact that their robot cannot traverse stairs over 7cm in height.

pedal Stair Climbing Robots

other researcher in Japan, [33], has undertaken robot stair climbing with a bipedal robot. :e his counterparts in Japan with the leg-wheeled robot, this researcher focussed on mod-ng the mechanical system of the bipedal robot and then used the model to implement imple stair climbing algorithm. One unique feature of this robot was its variable-length s which would be extended or retracted as needed (Figure 2.10). Due to the complexity the mathematical model of the robot, and the resulting complexity of the climbing algo-ım, sensor information from the robot was sent to a remote computer to be processed. tonomous operation was not achieved, and the robot could only traverse stairs with a

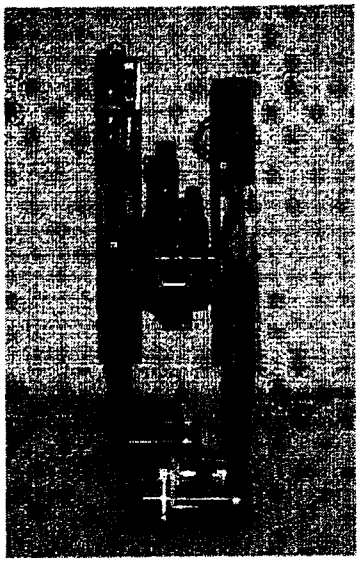


Figure 2.9: *The Leg-Wheeled Robot [32]*

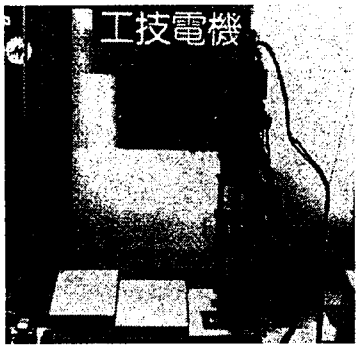


Figure 2.10: *Bipedal Robot with Variable-Length Legs [33]*

limited height.

Researchers in Italy at the Laboratory of Robotics and Mechatronics [34] have developed a bipedal mechanical robot system which incorporates pneumatics and suction cups (Figure 2.11). The robot has a programmable logic controller (PLC) which houses subroutines for walking and stair-climbing motions. A remote control system was used to trigger the subroutines in the PLD and have the robot ascend a set of stairs. Further research [35] showed that the robot could also descend stairs. The researchers plan to implement autonomous operation of the robot in the near future.

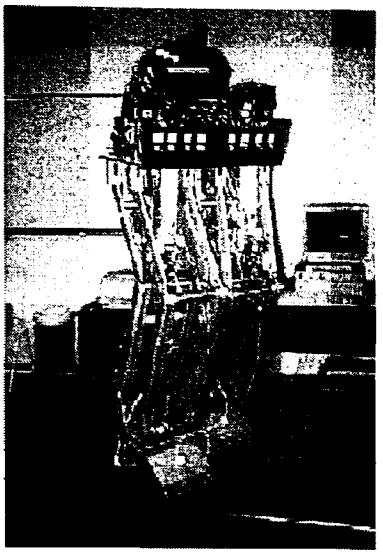


Figure 2.11: *EP-WAR2 Biped Robot [34]*

Two recently developed bipedal robots capable of autonomous staircase traversal are : *BARt-UH* and the *QRIO*, from the Intitut für Regelungstechnik in Germany, and the elligent Systems Research Laboratory at the Sony Corporation in Japan, respectively. ch of these robots, (Figure 2.12), used stereo machine vision to implement autonomous ir climbing [36] [22].

:OUT Four-Legged Robot

2004, researchers at McGill University in Montreal took a different approach to legged ir climbing with the quadruped *SCOUT* robot [37]. Their four-legged mechanical robot)totype uses servos as actuators. Despite its relative simplicity with one degree of freedom · leg, the researchers have shown through simulation that the SCOUT robot is capable of ir climbing. Physical implementation and autonomous operation are yet to be completed.

out Jumping Robot

unique entry in the realm of single-robot star climbers is the Scout robot [38] developed the Center for Distributed Robotics at the University of Minnesota. This robot is an :eption to the paradigm that stair climbing robots need to be large relative to the size

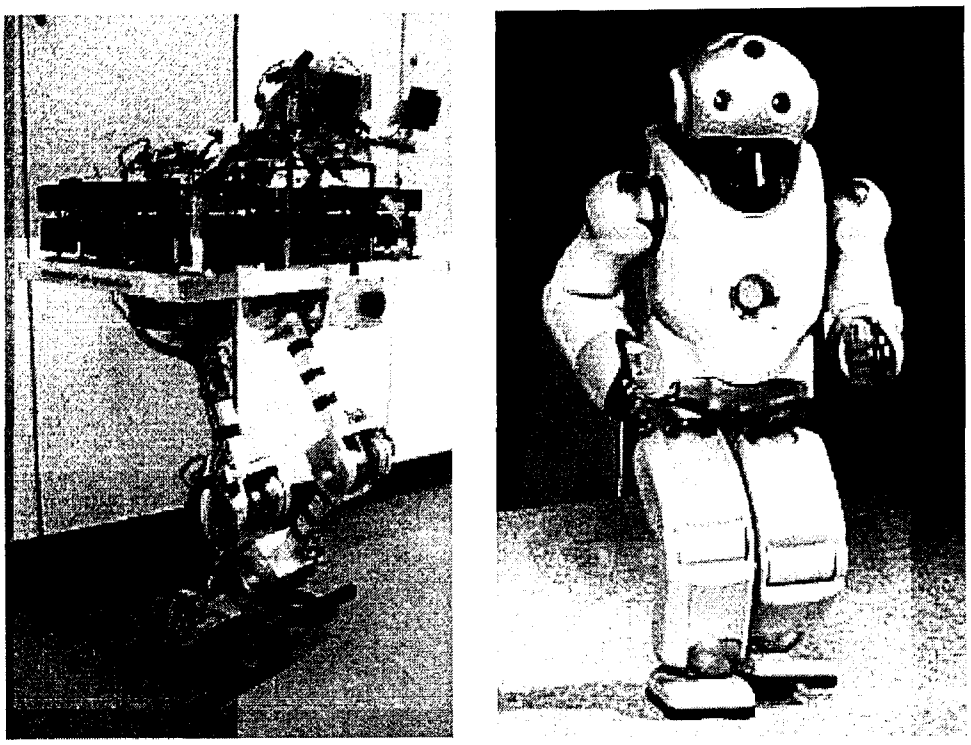


Figure 2.12: *(a) BARt-UH [36] (b) QRIO [22]*

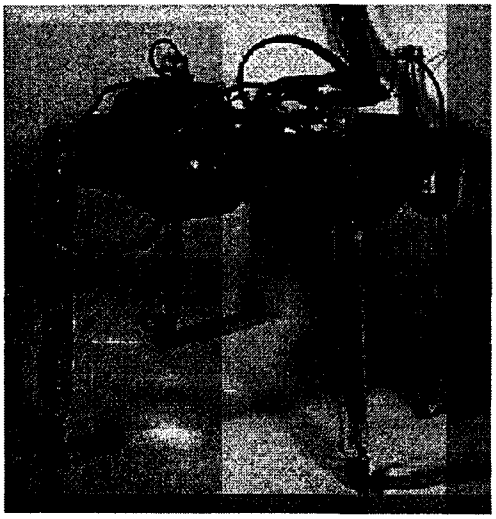


Figure 2.13: *SCOUT-1 Quadruped Robot [37]*

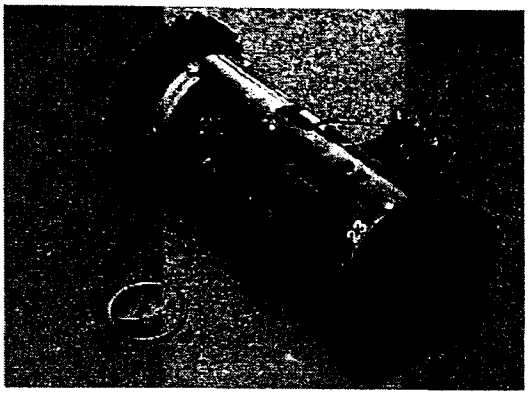


Figure 2.14: *The Scout Stair-Hopping Robot [38]*

ı stair. The Scout robot traverses stairs by jumping from one stair to another using a ing-loaded catapult foot. The only drawback to this type of staircase traversal is that the ıot can only perform this jumping maneuver on stairs with long treads.

ıir Climbing Assistance Robots

e of the practical applications for single-robot stair climbers is robotic assistance for the erly and disabled. In 1998, Takahashi [39] presented bipedal robots that acted as support ile ascending and descending stairs. Wiesspeiner, from the Graz University of Technology, ; developed a robotic stair climbing wheelchair to assist disabled persons [40].

3.2 Reconfigurable Robots

ıst of the current multi-agent robotic systems utilize small robots [41]. While these robots typically highly maneuverable in confined spaces their diminutive size makes them ill-ıipped to deal with all but flat, level terrain. In indoor environments, groups of small ıotics are confined to one level of the structure. Climbing stairs is impossible as the le of such robots is typically far below that of an average stair. Descending stairs is an ıally daunting task which usually consists of the robot driving off the edge of a stair with astrous results.

The Molecule

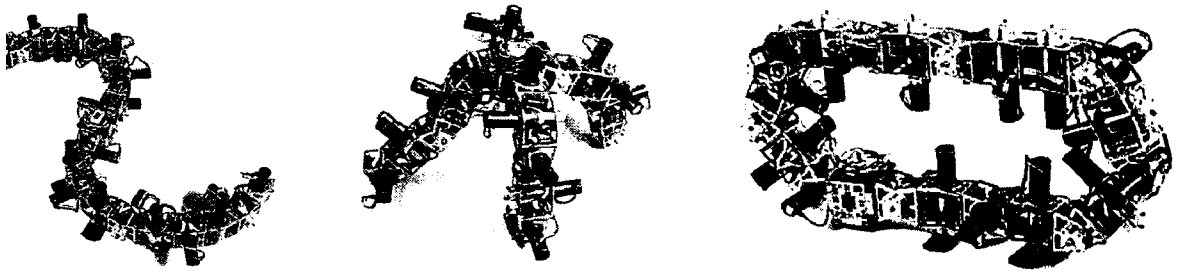
At the Dartmouth Robotics Laboratory, researchers have begun development on a modular robot called *The Molecule* that boasts self-reconfiguration abilities. The Molecule is currently in the early stages of development but when complete, the modular robot "can assume a snake shape to traverse a tunnel, reconfigure upon exiting as a six-legged robot to traverse rough terrain, and change shape and gait to climb stairs and enter a building" [42]. These abilities highlight just some of the qualities of a reconfigurable robot. One of the biggest advantages a reconfigurable robot has over a fixed-architecture robot is that a reconfigurable robot can support multiple modes of locomotion and manipulation. In terms of stair climbing, a reconfigurable robot could reconfigure to climb the stairs and then reconfigure again to perform some other function.

Polybot Modular Reconfigurable Robot

Some researchers have approached the problem of robot stair climbing using modular or reconfigurable robots. One of the most successful implementations of a reconfigurable, modular robot is the *Polybot* developed in 2001 at the Xerox Palo Alto Research Center in the United States [43]. The Polybot is a modular reconfigurable robot comprised of many simple robotic modules coupled to one other. The modules can be arranged in different configurations such as a snake, a spider, a biped, or other geometric configurations. The Polybot has undergone three generations of development [44], but version 4 of the first generation (Polybot G1v4) was the first incarnation of the Polybot to exhibit stair climbing capabilities. Research conducted with the Polybot has shown that gait control tables (GCT) are an effective mechanism for open loop movement control of a snake-like modular robot. A gait control table contains position entries for each degree of freedom used to implement a specific locomotion gait. Table 2.1 depicts a gait control table for a travelling arc using four robot modules coupled together in a simple chain, or snake configuration. The entries in the table represent the angle of each joint at a particular step in the motion. Gait control tables are discussed in more detail in relation to the Gremlin climbing algorithm in Section 5.3.4.

	Module Number			
	1	2	3	4
step 1	15	15	-15	15
step 2	15	15	15	-15
step 3	-15	15	15	15
step 4	15	-15	15	15

Table 2.1: *Simple Gait Control Table for Travelling Arc Locomotion*



(a) Polybot Chain (b) Polybot Spider (c) Polybot Loop

Figure 2.15: *Polybot Configurations*

The mechanical feasibility of Polybot stair climbing was demonstrated using a loop con-
ıration. The gait control table was supplied to the module via cables from a remote
top [43]. Due to the fact that there were no environmental sensors on the modules, the
ir climbing motion was mostly prescribed, with the loop of modules taking the shape of
stairs without sensing them (Figure 2.16). Researchers involved in the Polybot project
n to migrate the gait control tables into the individual modules to enable autonomous
ıration [45].

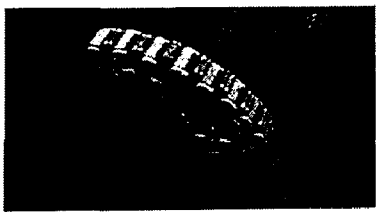


Figure 2.16: *Polybot Stair Climbing*

Figure 2.17: *The Inchworm Robot*

The Inchworm

In 1997, researchers at the Dartmouth College Department of Computer Science developed a reconfigurable robot called *The Inchworm* which is made up of a number of autonomous modules that can reconfigure according to a given task [46]. The design philosophy in this project differs from that of the Polybot in that the Polybot modules were not autonomous, and could not operate as individual entities outside of the Polybot chain. While the researchers at Dartmouth have not yet implemented dynamic autonomous reconfiguration with multiple Inchworm robots, the idea of autonomous modules in a reconfigurable robot remains attractive.

OmniTread Serpentine Robot

Two collaborative research groups at the University of Michigan in the United States and the Institute of Automatic Control in Poland, have developed the *OmniTread Serpentine Robot.* The OmniTread was comprised of five segments covered in tracks linked to one another by two degree of freedom joints. As of 2005, the OmniTread is purely a mechanical implementation with no autonomous or independent module operation [47]. However, their design did demonstrate the ability of a chain of robots to traverse tall vertical steps (Figure 2.18).

Millibots

The *Millibots* developed in 2001 at Carnegie Mellon University in the United States are a very successful implementation of a multi-agent robotic system. The Millibots use a modular architecture that allows different sensing and processing platforms for each robot depending

Figure 2.18: *OmniTread Robot Stair Climbing*

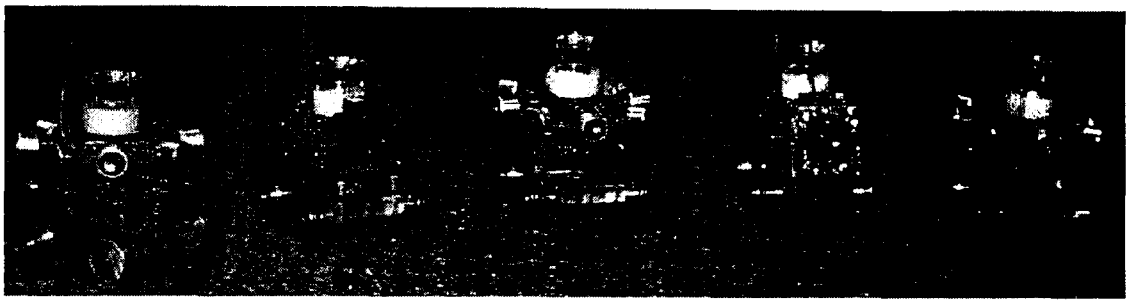


Figure 2.19: *Millibots*

the mission parameters [48]. The Millibots have been used to demonstrate applications multi-agent robot systems such as cooperative mapping of indoor environments, coopitive target tracking, redundant execution of a mission, localization for the purpose of laboration, and cooperative self-repair of the robot team [49]. Development of the Millis has also led to advances in sensor and communication capabilities of the robots as well high-level aspects of cooperative tasks [50, 51].

Despite the success of the Millibot project, the researchers realized that limited mobility s still a major drawback of using small robots. They proceeded to develop the *Millibot ain*. The goal of the Millibot Train project was to enhance the mobility of the robots to ow them to operate in environments with rough terrain and traverse larger obstacles [41]. The Millibot Train modular robot system was designed to incorporate individual Milot modules with powered tracks, and an electro-mechanical coupling/lifting mechanism gure 2.20). Work on the Millibot train did not include development of external sensors, onomous coupling, or high-level control. Instead, the researchers focussed on the electro-

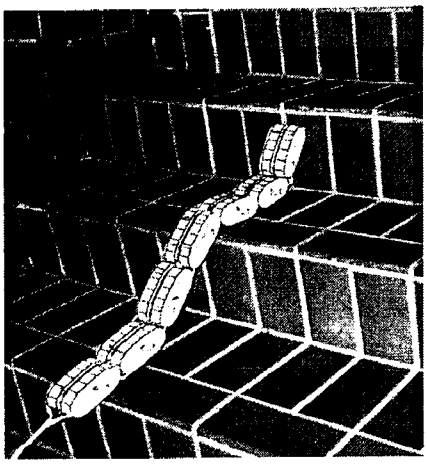


Figure 2.20: *Stair Climbing with the Millibot Train*

mechanical design of the robot. Fused-deposition-modelling for rapid prototyping was used to manufacture most of the parts for the robots. The robots also included several custom-built mechanical systems including a planetary-traction drive system, a coupling system based on shape-memory-alloy wire, and a harmonic-drive lifting mechanism with laser-cut gears. The researchers were able to demonstrate that the Millibot Train could traverse stairs up to 33cm in height under manual control. The authors cite autonomous operation, coupling, and stair-climbing as future work for their project.

The Millibot Train project demonstrates that it is possible to fabricate a system of small, modular, reconfigurable robots capable of climbing stairs. It also illustrates that there are research opportunities for creating robots that are less mechanically complex and can operate autonomously.

hapter 3

obot Behavior

HIS chapter presents behavior-based robot control systems and the ALLIANCE control architecture. Improvements necessary for adapting the ALLIANCE architecture facilitate the dynamic reconfiguration of robots is also discussed.

1 Behavior-Based Robotics

havior-based robotic control is a methodology for designing autonomous agents such as ots. It links artificial intelligence, engineering, and cognitive sciences [52]. Behavior-ed controllers are designed to realize particular high-level goals. They are comprised of ollection of behaviors which take inputs from a robot's sensors and control the robot's uators. Behaviors can also communicate and interact with each other. Behavior-based otic control places emphasis on creating agents that can *act* intelligently. It is important note the distinction between acting intelligently and producing cognition. Behavior based otics does not seek to produce cognition, but rather create robotic control systems where completion of a particular goal is paramount, and the coupling between perception of environment and action is as direct as possible. In fact, one of the guiding principles in ot design is the emphasis on understanding the environment in which a robot functions

The behavior-based design strategy has produced robotic systems used in military applications, mining, space exploration, disaster relief, and the modern home [53]. The Sojourner

Rover [54] which was part of the NASA Pathfinder mission to Mars (Figure 3.1), employed a behavior-based control layer developed at the Jet Propulsion Laboratory at the California Institute of Technology. This type of control system was able to acknowledge failure and adapt the robot controller accordingly.

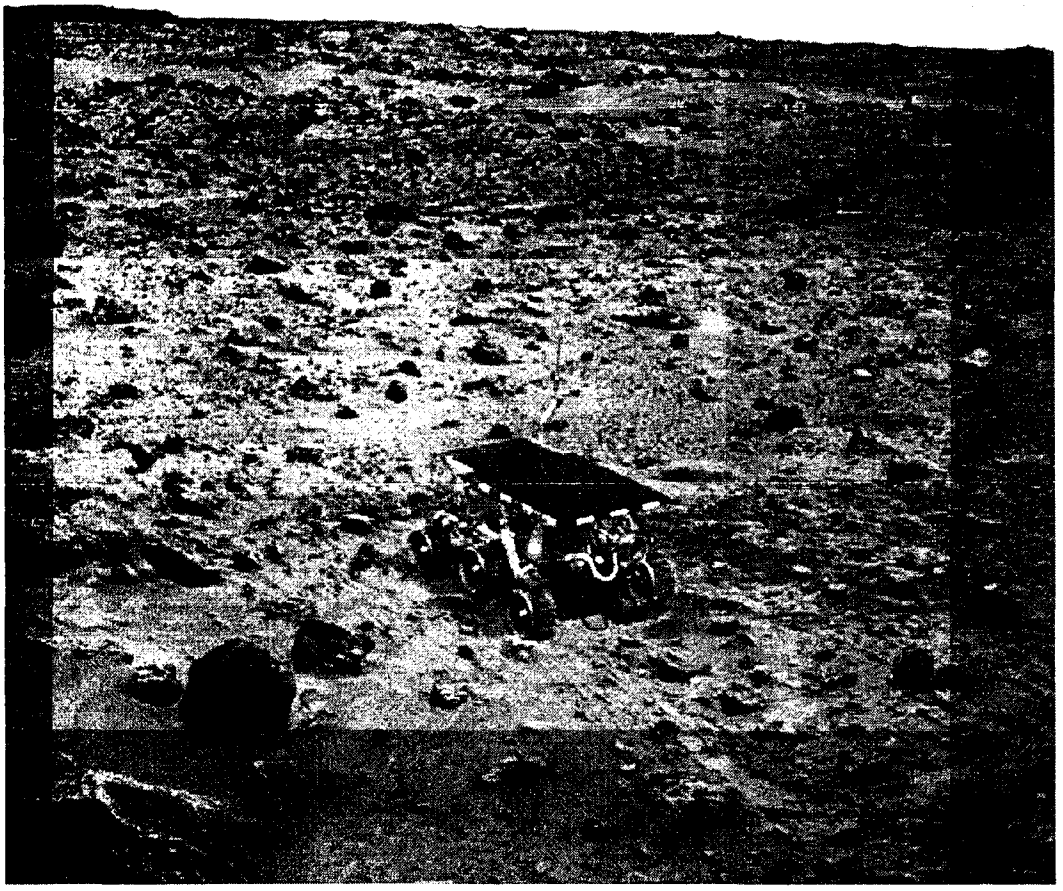


Figure 3.1: *The Mars Rover Sojourner*

3.1.1 A Brief History of Behavior-Based Robotics

Current behavior-based robot control is the fusion of the two basic types of robot control systems [55]. The first basic type of control system employs *planner-based strategies.* Under this control scheme a centralized world model is employed which is used to verify sensor feedback and generate actions as outputs [56–58]. Planner-based control strategies are also

erred to as deliberative and the two terms are used interchangeably. Information about world is provided by the sensors and acts as an input to the planner. The planner uses information in the world model in conjunction with the sensor inputs to select the best irse of action (Figure 3.2). Problems can occur when situations arise that are not included he world model. For this reason, a deliberative control model is best suited to situations ere the tasks is highly structured and the environment is predictable.

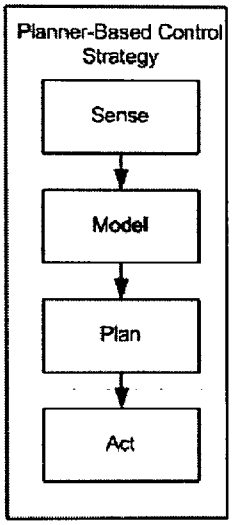


Figure 3.2: *Structure of a Deliberative Control System*

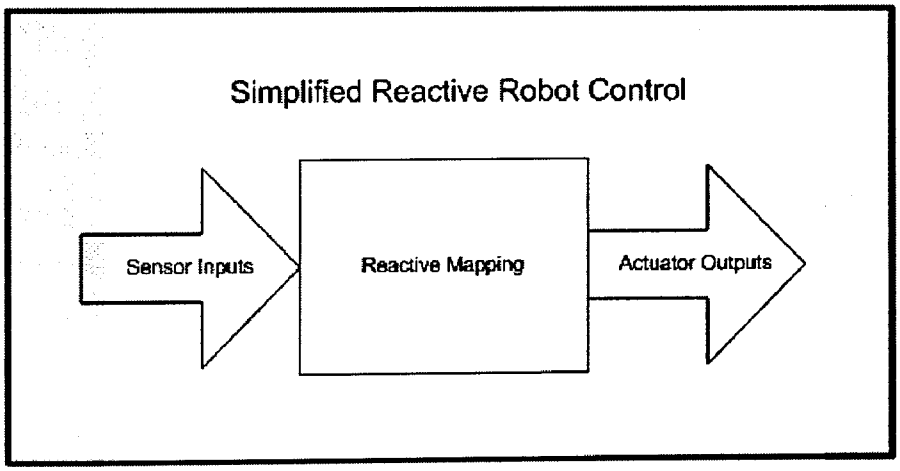


Figure 3.3: *Reactive Control Scheme*

The second basic control system exists at the opposite end of the spectrum. *Purely*

reactive control schemes contain no models of the world, and maintain minimal internal state. In software terms, reactive control schemes function like lookup tables and simply map sensor inputs to output actions, creating a direct connection between sensors and actions [59, 60]. Arkin [7] defines the following properties of reactive robot control systems:

- Behaviors serve as the basic building blocks for robotic actions.
- Abstract representation of sensor data is avoided in the generation of a response.
- Reactive systems are inherently modular from a software point of view.
- Animal models are often used as a basis for reactive systems.

The main advantage of reactive control systems is their fast response time. The responsiveness of a reactive control scheme is the result of directly coupling the sensor inputs to the actuator outputs. Figure 3.4 depicts a simple example of a reactive control scheme that implements obstacle avoidance by cross-coupling sensor inputs to actuator outputs. One of the major drawbacks of this approach is that it limits the robots' ability to perform high-level tasks which require planning and decision making.

Figure 3.5 shows a comparison between reactive and deliberative control systems. Reactive systems can provide real-time response and require no symbolic representation of their environment. However, reactive systems only implement low-level intelligence, and can only perform simple calculations. The performance of deliberative systems are dependent on the internal representation of the environment and offer high-level intelligence at the cost of slower response times.

3.1.2 A Hybrid Control Architecture

The subsumption architecture [61] provided a structured methodology for reactive systems. It is considered to be the foundation for modern behavior-based robotics [52]. This architecture used a bottom-up approach with a layered set of rules. Low-level layers such as

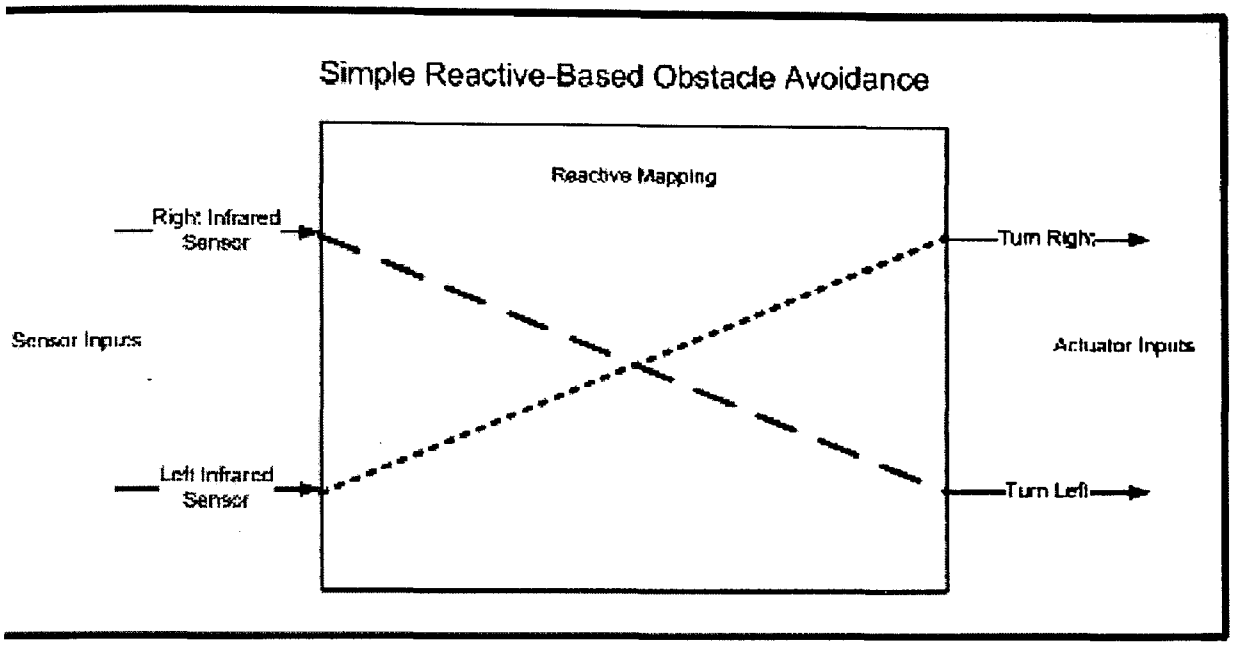


Figure 3.4: *Simple Reactive Control Example*

lision avoidance and obstacle detection were assigned the highest priority and were imple-nted using the reactive control system philosophy. High-level behaviors were rules which)lemented goal-achieving functions and utilized lower-level behaviors as outputs. An in-mediate module was employed which managed conflicts between the high-level behaviors l the purely reactive low-level behaviors.

One of the distinctive features of the subsumption architecture was the distributed nature the control system [55]. Figure 3.6 depicts a simple example of a subsumption-style ıtrol architecture designed to have a robot map obstacles in an environment. Note that ıaviors are implemented at different levels of abstraction. All of the behaviors are executed ıcurrently to exploit parallelism, making the system more powerful than a simple reactive tem.

Modern hybrid control systems seek to improve on the subsumption architecture. The 'elopment of the subsumption architecture yielded a control system that enabled real-ıe responses, and modular representation of behaviors but was inherently reflexive in ;ure. Being reactive in nature, the subsumption architecture lacked a mechanism for

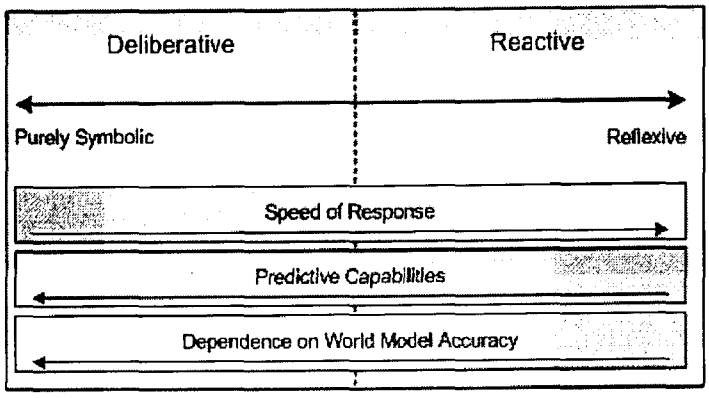


Figure 3.5: *The Robot Control System Spectrum [7]*

symbolic representation which could integrate knowledge of the environment into the system. Deliberative control systems provide a means of high-level planning for tasks that require purposeful action as opposed to reflexive action. The introduction of a deliberative control layer creates a control system where high-level planning is used to guide reactive control to achieve a particular task.

Advances in robot control architectures, [7], [62], led to *hybrid* architectures which employed aspects from both *planner-based* and *reactive* schemes. The hybrid deliberative/reactive approach used reactive control for low-level actions and planner-based strategies for high-level decision making [55]. In simple terms, the reactive portion of the robot control deals with the immediate safety of the robot while the deliberative control system maintains an internal representation of the world and makes high-level decisions according to this model.

Variations of the hybrid control architecture have been used in many multi-agent robot systems. Noreils [63], implemented a three-level control architecture to facilitate multi-robot cooperation which incorporated a planner, a control level, and a functional level. The ACTRESS architecture described in [64] introduced a negotiation framework which allowed robots to recruit help from other units. These implementations addressed the issue of behavior arbitration and task selection, the central design challenges of behavior-based robotics [52]. They did not, however, address some difficult real-world issues for physical

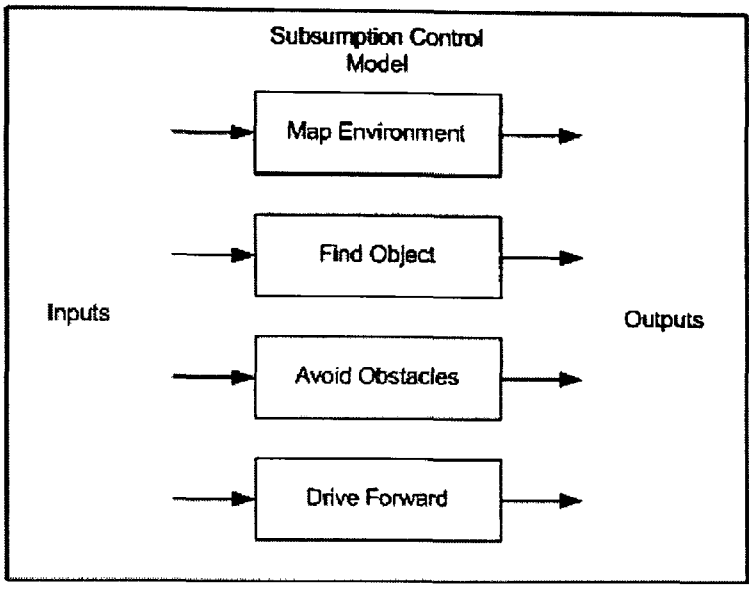


Figure 3.6: *A Simple Subsumption-Style Control Architecture*

ɔot teams such as dynamically changing environments, communication breakdowns, and ɔot failures. The next section presents a control architecture which emphasizes fault erance, and adaptive cooperative control, both of which are necessary for a system of onfigurable robots.

2 The ALLIANCE Architecture

e ALLIANCE behavior architecture, originally developed by Parker [65] in 1998, is a ıavior-based distributed architecture that achieves fault-tolerant robot cooperation using ıptive action selection. ALLIANCE is designed to exhibit robustness, fault tolerance, iability, flexibility, adaptivity, and coherence [66]. The main goal of the ALLIANCE hitecture is to implement a robot control scheme that will operate successfully amidst ariety of uncertainties such as sensor noise, robot failures, a dynamic environment, and ector noise. The ALLIANCE architecture seeks to improve on tradition behavior-based ıtrol by grouping behaviors into behavior sets. These behavior sets are activated or put o hibernation depending on the current high-level task. Each behavior set encapsulates ıavioral capabilities required to accomplish a high-level task.

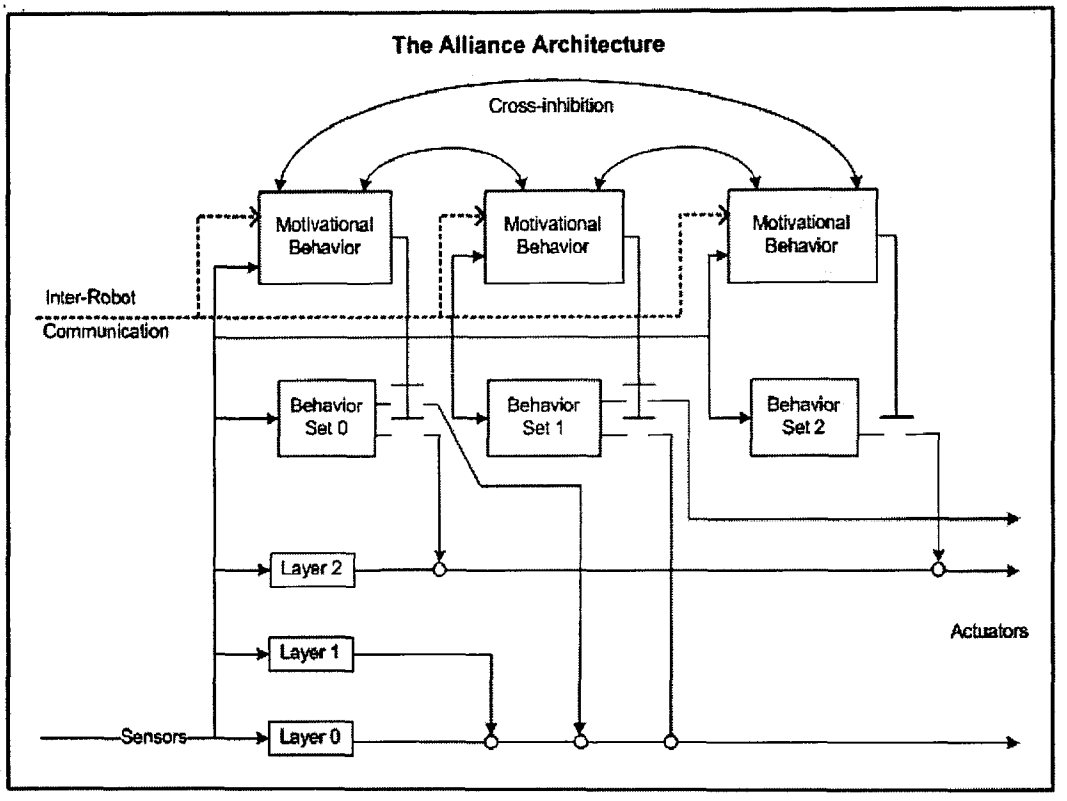


Figure 3.7: *The ALLIANCE Control Architecture*

Action selection in the ALLIANCE architecture is achieved through the use of motiva-
nal behaviors. These motivational behaviors represent the high-level goal-oriented be-
viors of the robot and control the activation of their associated behavior set. Only one
iavior set may be active at a time, but lower-level reactive behaviors may be activated
necessary. The motivational behavior mechanism for behavior selection is based on two
thematically modelled motivations: impatience and acquiescence. Each motivational be-
vior calculates a level of activation for its associated behavior set using sensory feedback,
nmunication with other team members, and the current rates of impatience and acquies-
ice. A behavior set is activated when its level of activation crosses a specified threshold
l the robot has selected an action.

The impatience and acquiescence mechanisms of this behavior architecture provide a
ans for fault-tolerant multi-robot cooperation. Robots will become impatient if a partic-
r goal is not being accomplished by another team member and will take over the task.
well, a robot will acquiesce a tasks if sensory input indicates that the task is not being
omplished in a timely and satisfactorily manner.

2.1 Applying the ALLIANCE Architecture to Reconfigurable Robots

e ALLIANCE architecture has been used to implement multi-agent robot cooperation
tasks such as hazardous spill removal, formation-keeping, cooperative tracking, and box
shing. Our proposed design adapts the ALLIANCE architecture to facilitate autonomous
iavior-based docking and reconfiguration. The ALLIANCE architecture is well suited
this task because it provides a robust, fault-tolerant control system. The motivational
iaviors impatience and acquiescence were crucial to implementing autonomous docking.
cking with mobile robots is a very difficult task [67–69] and can require several attempts
fore a successful dock is accomplished. Acquiescence and impatience in the ALLIANCE
hitecture provided mechanisms to ensure that unsuccessful docking attempts did not
pardize the overall success of the cooperative reconfiguration task. The implementation

of autonomous docking is discussed in Chapter 6.

One of the major shortcomings in the implementation of the ALLIANCE architecture presented in [65] was the inter-robot communication scheme. In previous experimental implementations of the ALLIANCE architecture, communication was not the focus. The researchers used a simple broadcast communication scheme managed by a central base station. While this scheme was adequate for early trials, the necessity of a central base station to manage communications does not allow for completely autonomous operation of the robot team. The next chapter presents a communication scheme that, when integrated with ALLIANCE, removes the need for a communication base station.

3.2.2 ALLIANCE Formal Model

The formal model of the ALLIANCE architecture is made up of the threshold of activation for behavior sets, and five primary inputs to the motivational behaviors. These five inputs are used in the computation of the current level of motivation for each behavior set.

Threshold of Activation

A single parameter determines whether a given behavior set is active. This parameter is the threshold of activation, θ. Different thresholds can be defined for each behavior set or a single threshold value can be supplied for all sets. Typically, one threshold is used for all behavior sets and the rates of impatience and acquiescence are different for each set.

Sensory Feedback

Sensory feedback in a robot can be either physical sensor information or virtual sensors. Virtual sensor information is supplied in the form of state information. Motivational behaviors use sensory feedback to determine whether the corresponding behavior set needs to be activated in order to proceed with the current mission. Sensory feedback is defined as:

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

ıer-robot Communication

e ALLIANCE architecture uses a broadcast communication system for inter-robot com-nication. Each robot monitors the broadcasts of other robots as a means of monitoring activities of other robots:

$$comm\_received(i,k,t_1,t_2) = \begin{cases} 1 & \text{if robot } r_i \text{ has received message from robot} \\ & r_k \text{ related to behavior set } a_{ij} \text{ in the time span} \\ & (t_1,t_2), \text{ where } t_1 < t_2 \\ 0 & \text{otherwise} \end{cases}$$

›adcast messages are used in the ALLIANCE architecture as a substitute for passive ion recognition. Passive action recognition is the process of detecting and interpreting actions of other robots. The detection and interpretation of actions by a robot requires y complicated sensors, substantial computing power, and sophisticated software. For se reasons, robots simply tell each other what they are doing using the communication chanism. Each motivational behavior monitors the communications of other robots and ermines if their activities are related to the associated behavior set.

Two other communication parameters are used by the team of robots to detect robot ures. The first of these parameters, ρ_i, defines the rate at which robot r_i broadcasts orts on its activities. The second parameter, τ_i, defines the period which robot r_i will ›w to elapse without receiving a communication from a teammate before determining that ticular teammate is no longer functioning.

ppression from Active Behavior Sets

ce a motivational behavior has activated its associated behavior set it has selected an ac-n and begins inhibiting other motivational behaviors from activating their behavior sets. e following function defines activity suppression:

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

Once a motivational behavior has activated its behavior set and is actively suppressing other motivational behaviors, it monitors feedback from the sensors, communications, and internal levels of impatience and acquiescence to determine whether the behavior set should remain active. Completion of the current task will be signalled by incoming sensor information and will result in the deactivation of the behavior set. Alternatively, the level of robot acquiescence will reach a point where the robot will give up on the current task and deactivate the behavior set. Once the behavior set is deactivated, other motivational behaviors can activate their behavior sets allowing the robot to select a new action.

Robot Impatience

The first parameter used to implement robot impatience in the ALLIANCE architecture is $\phi_{ij}(k,t)$. This parameter represents the amount of time robot r_i will allow robot r_k to affect the motivation of behavior set a_{ij} with communication traffic. This parameter can vary with time during the mission depending on capabilities of other robots. As well, a different $\phi_{ij}(k,t)$ value can be assigned to each robot by r_i enabling r_i to be influenced more strongly by certain robots.

Two other parameters, $\delta\_slow_{ij}(k,t)$, and $\delta\_fast_{ij}(t)$, are used to implement robot impatience. The $\delta\_slow_{ij}(k,t)$ parameter defines the rate of impatience of robot r_i concerning behavior set a_{ij} while another robot r_k is performing task $h_i(a_{ij})$. When no other robot is performing task $h_i(a_{ij})$ $\delta\_fast_{ij}(t)$ defines the rate of impatience. Clearly, $\delta\_fast_{ij}(t)$ defines a higher rate of impatience which motivates a robot to undertake a task that is not being pursued by another robot. The slower impatience rate, $\delta\_slow_{ij}(k,t)$, gives another robot r_k the opportunity to complete the task before robot r_i becomes impatient and takes over the task.

ALLIANCE increases the impatience rate for a motivation behavior controlling behavior set a_{ij} at a rate that allows the slowest robot to complete the task in its allowable time $\phi_{ij}(k,t)$. The impatience rate for a behavior set a_{ij} is given by:

$$impatience_{ij}(t) = \begin{cases} min_k(\delta\_slow_{ij}(k,t)) & \text{if } (comm\_received(i,k,j,t-\tau_i,t) = 1) \text{ and} \\ & (comm\_received(i,k,j,0,t-\phi_{ij}(k,t)) = 0) \\ \delta\_fast_{ij}(t) & \text{otherwise} \end{cases}$$

e following function is used to reset the motivation of behavior set a_{ij} when robot r_i t receives communication traffic indicating that another robot is performing task $h_i(a_{ij})$:

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

te that this reset function is only valid once for each robot r_k that indicates it is per- ming tasks $h_i(a_{ij})$. This prevents a robot from deadlocking the entire robot collective by sistently, and unsuccessfully, attempting a task.

bot Acquiescence

bot acquiescence is defined by two time parameters, $\psi_{ij}(t)$ and $\lambda_{ij}(t)$. The time that ot r_i will work on a specific task by maintaining an active behavior set a_{ij} before giving to try another behavior is given by $\lambda_{ij}(t)$. The parameter $\psi_{ij}(t)$ defines the minimum ount of time robot r_i will maintain the activation of a behavior set before it will yield to ther robot who has become impatient. The acquiescence function defines when a robot l abandon the current task and deactivate the currently active behavior set:

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

Motivation Calculation

The levels of motivation is calculated using all of the preceding parameters:

$$\begin{aligned} m_{ij}(0) &= 0 \\ m_{ij}(t) &= [m_{ij}(t-1) + impatience_{ij}(t)] \\ &\quad \times activity\_suppression_{ij}(t) \\ &\quad \times sensory\_feedback_{ij}(t) \\ &\quad \times acquiescence_{ij}(t) \\ &\quad \times impatience\_reset_{ij}(t) \end{aligned}$$

The motivation to activate behavior set a_{ij} increases from its initial value of 0 at a rate defined by $impatience_{ij}(t)$ unless one of the following conditions is met:

1. Another behavior set is activated in robot r_i.
2. Sensor information indicates that the task is complete and the behavior set is deactivated.
3. The robot has given up on the task.
4. Another robot has taken over the task.

The motivation returns to 0 if any of these conditions are met. If these conditions are not present, the motivation grows until it crosses the threshold of activation θ and the behavior set is activated. Once the behavior set is activated, the robot begins broadcasting its current activity to its teammates.

3.2.3 ALLIANCE Parameter Selection

The key parameters involved in action selection are:

- $\psi_{ij}(t)$: the amount of time r_i will attempt to execute behavior set a_{ij} before acquiescing

- $\delta\_slow_{ij}(t)$: the rate of impatience of robot r_i related to behavior set a_{ij} when robot r_k is attempting a task

- $\delta\_fast_{ij}(t)$: the rate of impatience of robot r_i related to behavior set a_{ij} when no other robot is performing that task

The parameter selections for the ALLIANCE architecture greatly influence the perfor- nce of the system [65]. Action selection depends on the parameters of the motivational iaviors. Parameter selection also has an impact on the efficiency of the system by affecting amount of time required to select between behavior sets, the robot idle time, and the ount of time required to reallocate a task. In practice, parameter selection is performed pirically and the parameters are tuned to provide optimum performance of a task.

2.4 ALLIANCE Performance Metrics

e of the challenges associated with implementing a behavior-based control architecture for /stem of cooperative mobile robots is the analytical evaluation of success. The ALLIANCE hitecture presented in Section 3.2.2 has been implemented in a wide range of cooperative otic applications, both physical and simulated [66]. Historically, quantitative indicators nission success have been used (Examples from [66] are given in Table 3.1).

plication Domain	Number of Robots	Metric Description	Metric Definition
Pushing	2-5 Physical	Distance pushed per unit time	$d(t)/t$ where $d(t)$ is the distance moved through time t
mation Keeping	4 Physical or Simulated	Cumulative formation error	$\sum_{t=0}^{t_{max}} \sum_{i \neq leader} d_i(t)$ where d_i is distance robot i is misaligned at t
lti-Robot Manipulation	2-4 Physical	Number of objects moved per unit time	$j(t)/t$, where $j(t)$ is number of objects at goal at time t
ardous Waste Cleanup	2-5 Physical	Time of Task Completion	t_{max}

Table 3.1: *Examples of ALLIANCE Performance Metrics*

In the above examples, improvements to the mission quality were evaluated on qualitative icators such as time or distance. In [66], it is proposed that these are natural indicators e the primary benefit of multiple robot teams is the advantage of parallelism which acts ncrease performance of the system. As well, system performance measures are defined to

be application dependent, and the most important issues in evaluating success are *whether* and *how well* the robots complete their mission.

The drawback to focussing on application-specific metrics is that more abstract qualities such as fault-tolerance, adaptivity, and robustness are not clearly defined. These abstract qualities become embedded in the qualitative performance metrics. For example, an behavioral implementation with poor fault tolerance may exhibit a higher failure rate or longer completion times for a given task. The development of metrics to evaluate high-level abstract qualities remains an area in need of further research. For the purposes of our design, qualitative performance metrics is used.

hapter 4

obot Communication

HIS chapter presents a communication framework that is complimentary to the ALLIANCE control architecture presented in the previous chapter. In earlier implemenions of the ALLIANCE architecture, [65], a radio communication system was used to nage the message traffic between robots which consisted of radio modems attached to h robot, and a base station responsible for time-slicing the communication channel. In apter 2 design goals for our design were presented which stated that fully autonomous eration would be achieved. This chapter presents an innovative communication scheme t is reliable and suitable to enable fully autonomous robot operation and removes the d for a communication base station.

1 High-Level Communication Organization

[70], the researchers have categorized three methods for high-level communication in lti-agent robotic systems: *direct routing*, *routing by signal propagation*, and *public notice ting*. Direct routing is the simplest mode of communication. A robot sends a message ectly through the communication channel to the receiving robot. Other robot agents are given the opportunity to receive the message.

In a system that uses routing by signal propagation, a robot broadcasts a signal into the ironment. The intensity of this signal decreases as the distance from the transmitting ot increases. For this reason, which robots receive the signal is somewhat random and

dependent on their distance from the transmitting robot. In more complicated systems, other robots may repeat the signal to increase the range.

The third method of communication is public notice routing. Public notice routing can take different forms, but in general the exchange of information is conducted in a common data area. When a robot wishes to communicate, it posts a message in a common message space which is visible to all of the robots in the system.

A public notice routing method is the most suitable to the ALLIANCE architecture. Direct routing is not suitable because communication occurs only between two robots, leaving other robots without information about the current actions of the transmitting robot. Signal propagation is very similar to broadcast communication and is inappropriate for this system since robots outside of the communication range of the transmitting robot are too far away to participate in docking maneuvers. A public notice routing method is attractive because it provides a mechanism where all of the robots in the system can access information about the actions of other robots at any time.

4.1.1 The Blackboard Architecture

The Blackboard Architecture [71] is a centralized public routing method originally developed in the field of artificial intelligence. It has been used as both a communication mechanism and a behavior control mechanism. The term Blackboard comes from the notion of a number of agents, or experts gathered around a blackboard exchanging ideas to solve a problem. The current problem state is contained on the blackboard and agents make changes as problem solving progresses (Figure 4.1). A blackboard system is comprised of several contributing agents, a control mechanism to perform arbitration in case of conflicts, and the blackboard itself. In an artificial intelligence system, the blackboard is typically a form of global shared memory.

The implementation of a blackboard system as a communication mechanism involves the use of shared memory [19]. The blackboard is a centralized mechanism that accepts communication messages from agents and saves them to the blackboard. Agents can also read

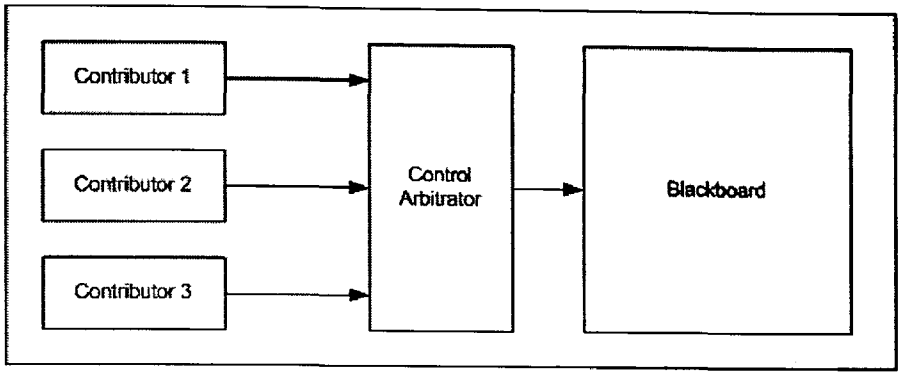


Figure 4.1: *A Simple Blackboard Architecture*

ssages posted by other agents. The blackboard model is general and does not prescribe any ticular message format for the blackboard. In the next section, a mode of communication resented which begins to formalize a blackboard-type communication system.

1.2 Publish-Subscribe Communication

blish-subscribe messaging [72] is a message delivery system that has been proven in multi-nt systems. In [73], the publish-subscribe model was shown to be popular for intercon-ting agents in a distributed environment, and was also shown to provide good scalability l reliability. The publish-subscribe paradigm has been used to build flexible, real-time, sely coupled distributed applications, and is used by companies such as Yahoo, Intuit, rade, and 300 of the world's financial institutions [74].

In a publish-subscribe communication system messages are addressed by subject or con-t instead of a particular recipient [75]. A data source, such as a robotic agent, addresses nessage with a subject which describes its content and publishes the message onto the work (Figure 4.2). Once the message has been published on to the network, other agents h an interest in that particular subject can subscribe and receive the message. In this way, nsmitting agents do not need any advance knowledge of receiving agents, and receiving nts can focus their attention on messages that are of a particular subject.

The publish-subscribe communication system is attractive for use with the ALLIANCE navior architecture for two reasons. First, messages transmitted in a publish-subscribe

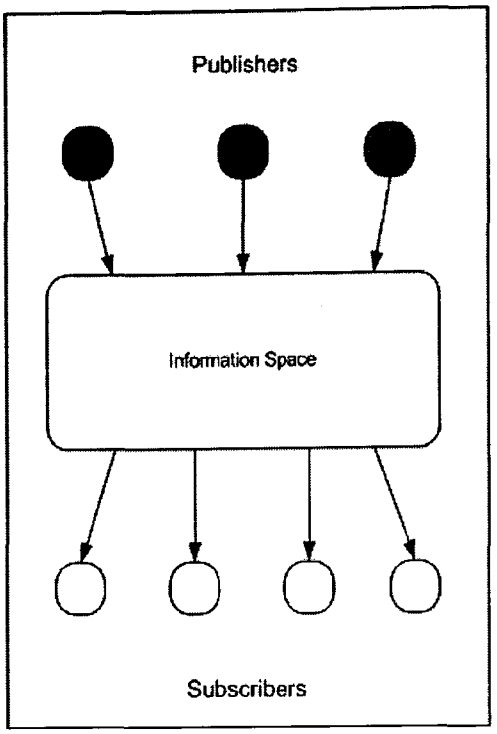


Figure 4.2: *The Publish-Subscribe Communication Model*

system are available to all agents of the system. In terms of ALLIANCE, if a robot sends a message regarding its current activities, that message is available to all other robots who are monitoring the transmitting robot's progress towards a given task. Secondly, the messages in the publish-subscribe paradigm are addressed according to subject, rather than addressee. This allows monitoring robots to focus on messages that pertain specifically to the task they are monitoring.

The only hurdle to overcome when adapting a publish-subscribe communication system for use with the ALLIANCE architecture is the necessity for a common information space. In practical implementations of a publish-subscribe communication mechanism such as [76] the publish-subscribe system functions as an instantaneous black board. Communication messages are addressed by subject, and transmitted in a broadcast fashion through the communication medium. The system maintains no history of the transmitted messages with the philosophy that information in a dynamic robot system quickly becomes irrelevant. In

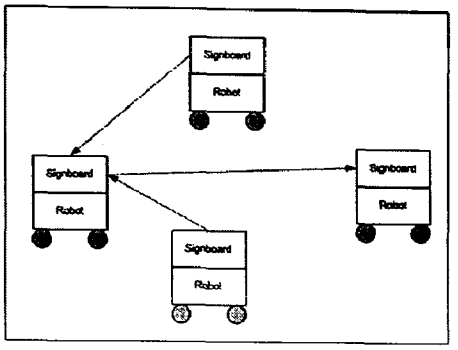


Figure 4.3: *Sign-Board Based Communication*

next section, a distributed blackboard communication is presented which maintains a rt history of transmitted messages without a centralized communication mechanism.

1.3 Sign-Board Communication

e *sign-board communication scheme* originally presented in [77] and [78] implements a ckboard communication model on physically distributed robots. Instead of a centralized ckboard which maintains a history of the transmitted messages, each robot has its own n-board to which it can post messages. Robots have the ability to read the sign-boards heir peers (Figure 4.3).

Like publish-subscribe messaging, messages in the sign-board communication scheme are addressed to a specific robot as they are available to all robots in the system after being ted to a sign-board. For this reason, the sign-board is "a massive parallel mechanism, all displayed messages are constantly available for other robots to *see.* " The sign-board nmunication scheme is complimentary to the publish-subscribe system presented in the vious section because it eliminates the need for a centralized communication monitor, enables subject-based message passing. Chapter 6 outlines the implementation of a blish-subscribe communication system which uses sign-board communication for message sing.

The only issue that remains in the implementation of a sign-board based publish-subscribe nmunication is that of media access control. The robots in this project shared a common

communication channel which placed limitations on the message traffic between agents. The next section describes a media access control protocol that enables the robots to read the sign-boards of their teammates in a manner that eliminates collisions on the communication channel.

4.2 Media Access Control

In order to meet the project goal of creating relatively inexpensive robots, a simple transceiver radio modem was used as the communication link between robots [79]. The transceiver used amplitude-shift keying and operated at a frequency of 315 MHz. A multiple access technique, or media access control, was required to allow the robots to share this single radio channel. There are three basic multiple access techniques: Time Division Multiple Access (TDMA), Frequency Division Multiple Access (FDMA), and Code Division Multiple Access (CDMA) [80]. In a TDMA system (Figure 4.4), several users share a single carrier frequency by making use of non-overlapping time slots. Each user is assigned one time slot to transmit and receive per frame. In a FDMA system, individual frequency channels are assigned to each user. CDMA systems implement multiple user access by having all users broadcast in the same frequency channel at the same time and multiply the narrow band signal from the user with a large bandwidth signal. This large bandwidth signal, or spreading signal, is based on a particular noise code sequence that is used at the receiver which performs a time correlation using the same code to extract the original signal.

The transceivers used in our design were simple devices that did not have the advanced features of their more expensive cousins. In a CDMA system, power control is necessary to prevent the signal from users transmitting closer to the receiver from interfering with users transmitting at a distance. This, coupled with the complexity of implementing a CDMA system, made code division multiple access a poor choice as media access control for the Gremlin robots. FDMA was also eliminated as a candidate for access control because it required more than one frequency channel. TDMA is chosen because it could be implemented on a single communication frequency, and allowed multiple user access.

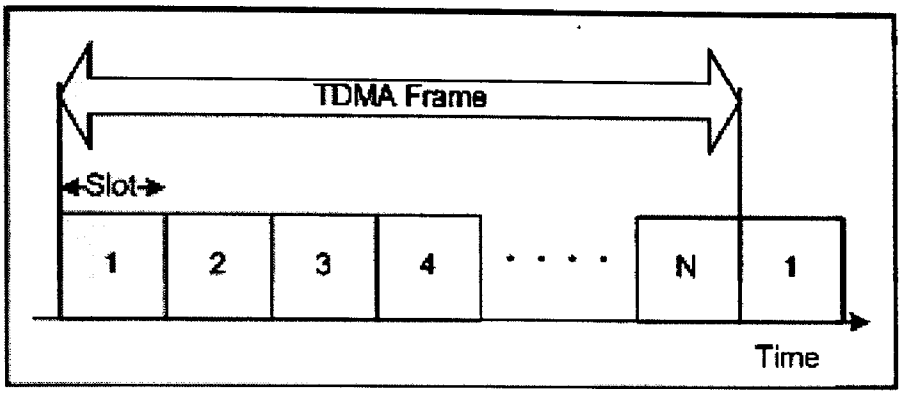


Figure 4.4: *TDMA Multiple Access*

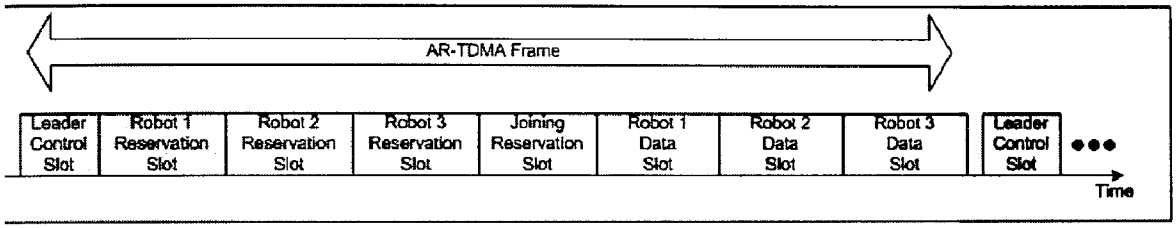


Figure 4.5: *AR-TDMA Slot Allocation*

In [81] a TDMA communication mechanism specifically designed to facilitate multi-agent otic systems was presented. Also the proposed Adaptive-Reservation Time Division Mul- e Access (AR-TDMA) is designed with a focus on high-traffic real-time communication robot cooperation. The AR-TDMA access control mechanism uses dynamic allocation h a reservation mechanism to manage time slots as robots joined the team.

Figure 4.5 shows the slot allocation of an AR-TDMA frame. The frame consists of a trol slot for the leader robot, reservation slots for follower robots, a reservation slots new robots joining the team, and data slots for each robot. The leader control slot is d to synchronize the frame, and provide routing information. The robot reservation slots ow robots to submit requests to join or leave the current cooperative task. The joining ervation slot is used to allow new robots to join the cooperative team. Data slots are used send and receive information between robots.

The AR-TDMA method was originally developed to enable cooperation within a team nobile robots and [81]. It is, however, the way in which the AR-TDMA communication

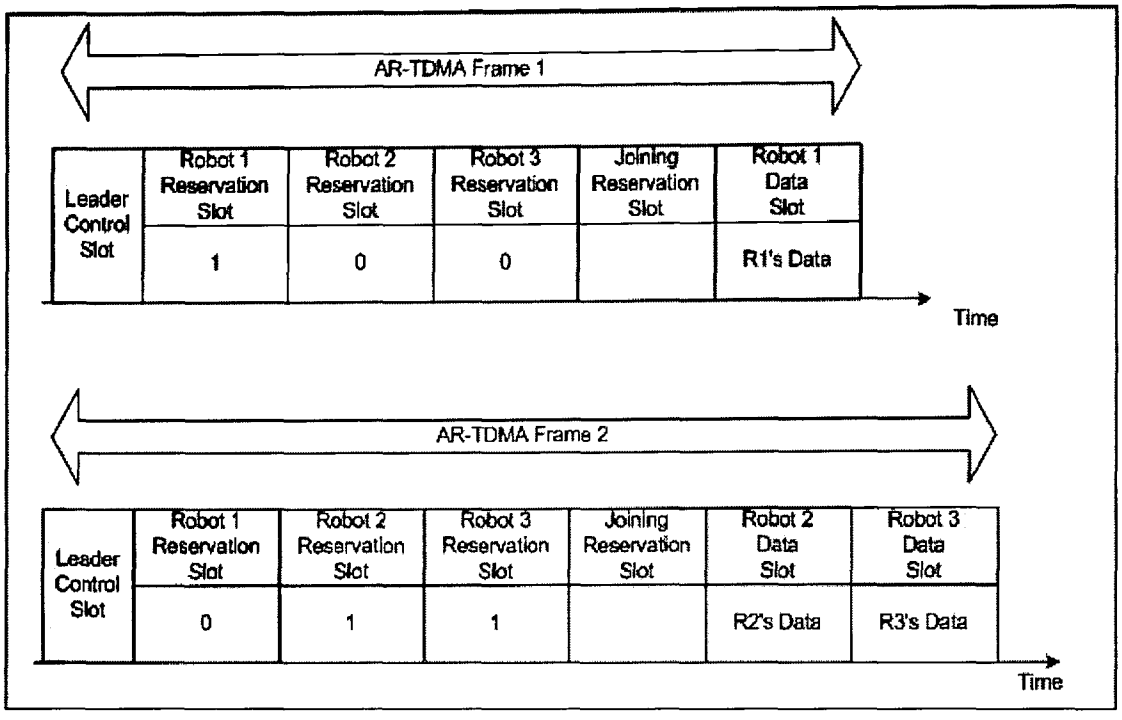


Figure 4.6: *Variable-Length Frames in AR-TDMA*

scheme allows for dynamic slot allocation that is of interest in relation to this project. The structure of an AR-TDMA frame allows the frame to be variable in length. The AR-TDMA frame has reservation slots which follow the leader's control slot at the start of each frame. If a robot has data to communicate and would like to use the channel it indicates this need to communicate during its reservation slot. Only robots which place a request in their reservation slot are allocated a data transmission slot in the AR-TDMA frame (Figure 4.6). In this manner, there are no time slots wasted on robots without any communication traffic.

hapter 5

roposed Architecture Synthesis and nplementation

WE present the behavioral control system and communication structure synthesized for the Gremlin robots. This research extends and integrates the ALLIANCE iavioral control architecture presented in Chapter 3, the sign-board communication tech-ue presented in Chapter 4, and the gait control table movement control scheme presented Chapter 2. The ALLIANCE architecture is extended significantly with specialized be-ior sets designed to facilitate autonomous reconfiguration via docking, a technique that ıld be applicable to many scenarios involving multiple autonomous agents.. The architec-e is also augmented significantly to integrate the successful completion of related behavior s into impatience calculations. The sign-board communication technique is integrated to ble fully autonomous operation of the ALLIANCE architecture without the need for a e communication station. The AR-TDMA protocol presented in Chapter 4 is modified enable automatic selection of a leader robot, automated TDMA frame construction, and erent detection of robot failure. Finally, the gait control table paradigm is improved by ling sensory feedback to state transitions in the table, allowing for closed loop operation.

1 Proposed Architectural Structure

e proposed control and communication architecture is shown in Figure 5.1. The behavioral ign of the Gremlin is separated into two components: the ALLIANCE-based control

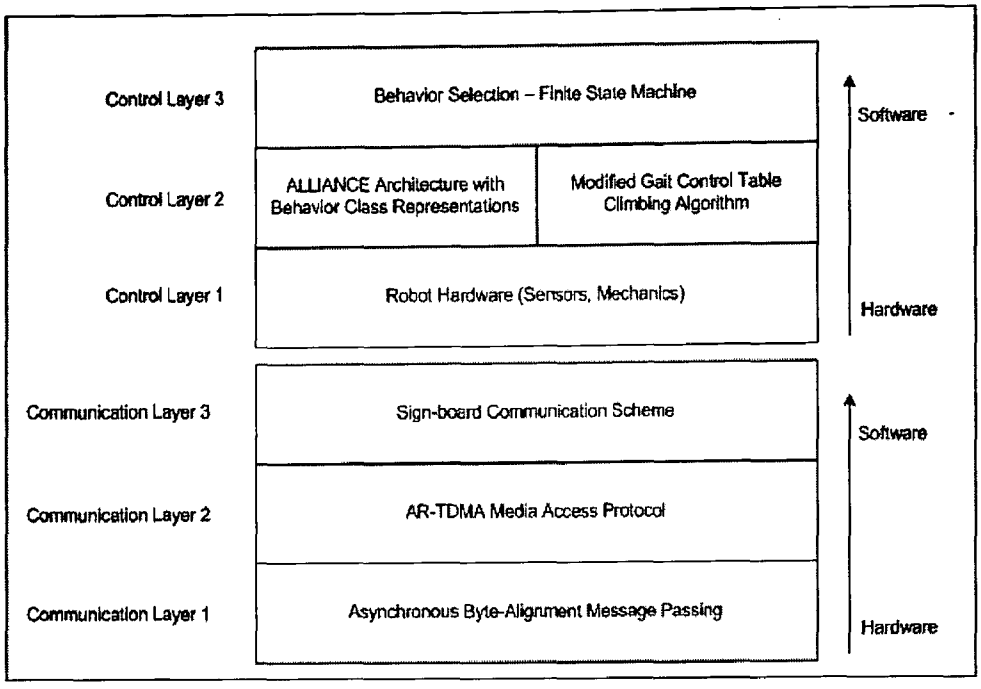


Figure 5.1: *Layered Representation of the Gremlin Control and Communication Architecture*

system for autonomous docking and reconfigurations, and a climbing control system based on gait control tables. At its highest level, the software for the Gremlin robots was organized as a finite state machine which manages the activation of the stair-climbing and reconfiguration behaviors. These behaviors interact with the robot platform and its sensors to accomplish the tasks of autonomous reconfiguration and stair climbing.

The communication architecture is divided into three layers. The highest level is based on the sign-board technique that allows the robots to constantly monitor the progress of other team members. The sign-board mechanism integrates well with the ALLIANCE behavior architecture. ALLIANCE requires that robot team members inform one another of their progress via communication messages. The sign-board mechanism allows all team members to view the current status of their team mates at all times. The middle layer of the communication architecture implements a media access protocol which facilitates automatic selection of a leader, dynamically adjustable frame sizes, and inherent robot failure detection. The lowest communication layer implements a simple, robust mechanism for transmitting data

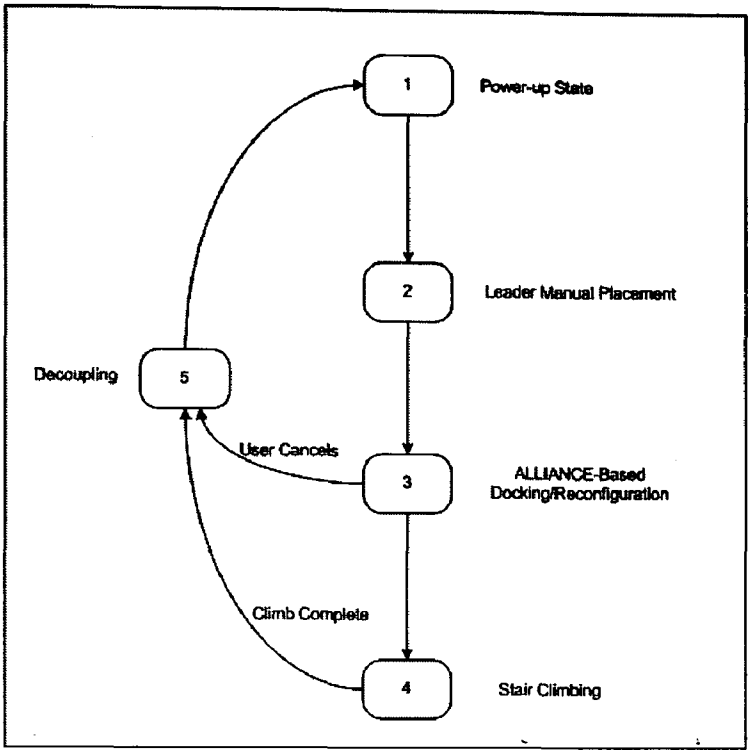


Figure 5.2: *High-Level Finite State Machine*

nchronously over a wireless link.

2 Proposed Behavioral Design: Finite State Machine

inite state machine was used as a high-level organizational structure for the different iaviors of the Gremlin robots. The state machine managed transitions between the dif- ent functions of the robots (Figure 5.2). When a robot first powers up it remains idle but nitors communications from other robots. The communication mechanism, discussed in tion 5.4, is designed to designate a leader robot. Once all of the robots have powered up; leader is manually positioned in front of the target staircase. Ideally, the leader would omatically detect the staircase and align itself, but that is an entirely different area of earch and beyond the scope of this thesis. See [26] for an implementation of staircase ection.

Once the leader has been positioned by the user, fully autonomous operation is initiated.

All operations from State 3 onward are conducted autonomously by the robots without any outside intervention from the user. In State 3, the robots performed autonomous reconfiguration by docking with one another using an ALLIANCE-based control architecture. Once all of the robots have docked to form a chain of robots, the control shifts to a climbing algorithm which directs the chain of robots while it climbs the stairs (State 4). At the top of the staircase the robots decouple and resume autonomous operation.

5.3 Proposed Behavioral Design: Modified ALLIANCE-Based Control

The ALLIANCE architecture was used as the basis for the autonomous docking and reconfiguration control layer. Development of an ALLIANCE-based control strategy includes the creation of behavior sets to implement autonomous docking and the creation of a new feature called behavior classes. The implementation of behavior classes integrates knowledge of task progression into impatience calculations and provides a mechanism for managing ordering dependencies within a task.

5.3.1 ALLIANCE Behavior Sets

In [65], Parker demonstrates that the ALLIANCE architecture enables cooperative robot control for tasks such as hazardous waste cleanup, cooperative box pushing, janitorial services, and bounding overwatch. In this thesis, we apply the ALLIANCE architecture to a completely new domain and create behavior sets designed for the tightly-coupled task of robot reconfiguration. The task of robot configuration via docking requires the development of specialized behavior sets with strict ordering dependencies. Parker notes that the primary weakness of ALLIANCE is its restriction to independent subtasks. We present an innovative mechanism that guarantees that behavior sets with ordering dependencies are executed in the proper order. As well, the ALLIANCE representation of impatience is modified significantly to take into account previous successes in a class of behavior sets to allocate extra time to docking tasks that are near completion.

A control structure based on the ALLIANCE architecture presented in Chapter 3 en-ed the robots to perform autonomous docking and reconfiguration. The proposed design resents the first implementation of the ALLIANCE architecture in a reconfigurable robot tem. The following behavior sets are defined to implement the autonomous docking con-. system:

- *Find-Leader-Methodical*: A methodical search procedure for locating the leader robot with the optical localization system.
- *Find-Leader-Wander*: A less systematic approach for finding the leader robot. The docking robot drives around the mission area while seeking the leader.
- *Determine-Relative-Position*: Used for determine the docking robot's position relative to the leader.
- *Navigate-to-Rear*: Enables the robot to navigate behind the leader robot as a precursor to docking.
- *Docking-Approach*: Approach the leader robot using the leader's fine docking LED as a guide.
- *Docking-Align*: Uses the front bumper switches to ensure proper alignment with the leader's rear docking plate.
- *Perform-Dock*: Raises and then lowers the front docking plate over the rear docking plate of the leader robot and drives the coupling mechanism into the rear docking plate of the leader to lock the two robots together.
- *Leader*: A special behavior set executed by the leader. This behavior set manages communications and the optical array while other robots that are attempting to dock.

The impatience and acquiescence mechanisms from the ALLIANCE architecture were ;inally developed to facilitate fault-tolerant cooperation in robot teams. In our design,

these mechanism are applied to the docking procedure. If a partner robot spends too much time attempting to dock with the leader, and is unsuccessful, it will either acquiesce the task of docking or another robot would become impatient and attempt to dock. The entire docking process is executed as follows:

1. The leader is chosen by the communication protocol and is navigated into position by a human observer.

2. The ALLIANCE behavioral control architecture takes over and the robots operate autonomously.

3. The leader selects the *leader* behavior set and activates all of its LEDs.

4. After receiving word from the leader that it has activated its LEDs all of the remaining robots on the team select either the *find-leader-methodical* or *find-leader-wander* behavior set to begin searching for the leader. The impatience rates for each of these behavior sets can be set differently in each robot. This will cause some robots to select the methodical set and some to select the wander set.

5. One of the robots detects the leader, selects the *determine-relative-position* behavior set, and immediately starts inhibiting the selection of this behavior set on other robots to prevent two robots from attempting to dock at the same time.

6. The docking robot uses its communication and optical systems to determine its position relative to the leader.

7. After its relative position is determine, the docking robot activates the *navigate-to-rear* behavior set to position itself behind the leader robot.

8. From its position directly behind the leader robot, the docking robot uses the *docking-approach* behavior set to track the light beam from the leader as it approaches from the rear.

). The docking robot collides with the back end of the leader robot and uses the *docking-align* behavior set to ensure proper alignment of the docking plates.

). The docking robot couples to the leader robot through the *perform-dock* behavior set designates itself as the new leader.

l. The process continues with the new leader and the remaining individual robots until all robots have docked and joined the chain.

If at any point during the procedure a docking robot fails to report progress by progressing ough the behavior sets another robot would become impatient and broadcast a message icated that it would like to attempt to dock with the leader. The robot currently docking h the leader backs away and monitors the progress of its counterpart.

der Behavior Set

e *leader* behavior is activated immediately after the leader robot has been navigated into ition in front of the stairs. The leader activates all four directions of the optical array ctively creating a beacon with 360° coverage of the mission area. Next the leader sends equest via the communication system for a partner robot who can dock with the leader. ce another robot visually detects the leader robot the leader is notified and then answers queries of the docking robot until it has docked with the leader. The queries from the king robot come in the form of requests to activate or deactivate different quadrants of optical array. After the docking robot has docked with the leader, the leader repeats the cess with other robots until all of the robots have joined the chain. As the leader, the ot also manages the TDMA protocol for inter-robot communication described in Section . Figure 5.3 shows the leader behavior set diagram.

Figure 5.3 is a standard behavior set organization diagram as defined in [65]. Inputs to behavior set are shown on the right side and outputs on the left. Behavior primitives h as "Activate Specific LEDs" are shown as small boxes interconnected by arrows. The ows represent data or information flow between the primitives. Behavior primitives are

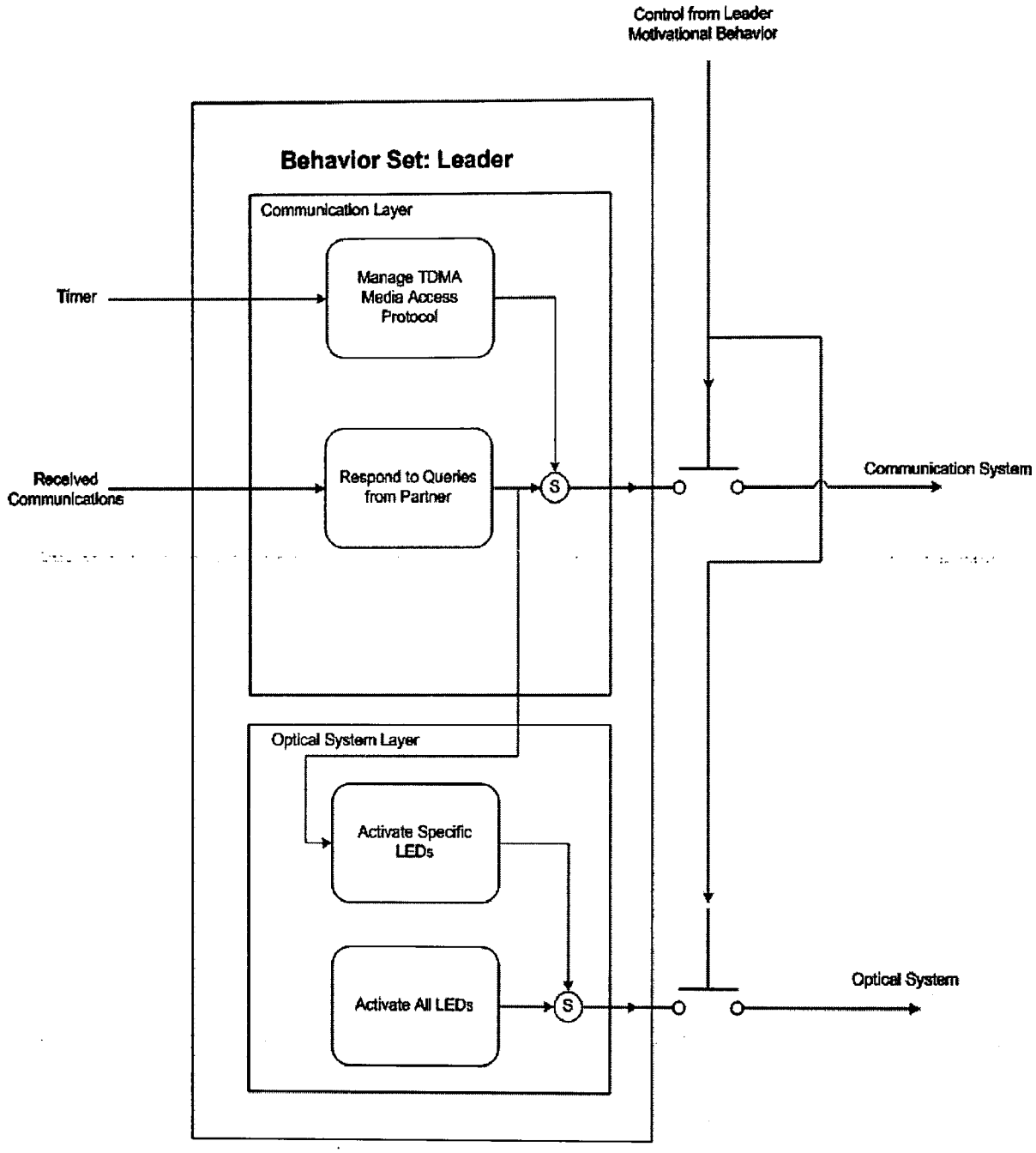


Figure 5.3: *The Leader Behavior Set*

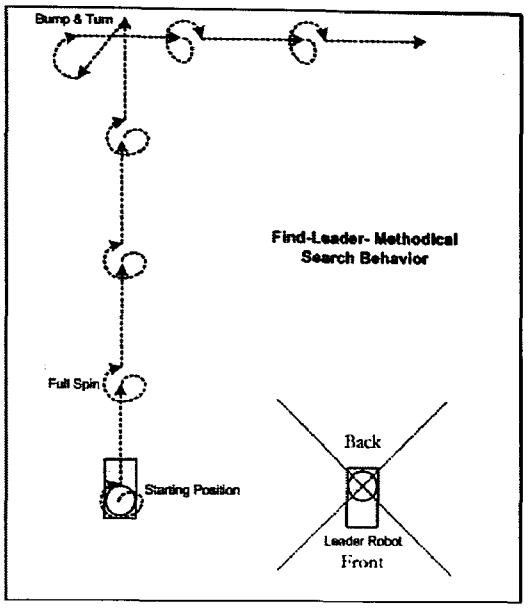


Figure 5.4: *Find-Leader-Methodical Behavior*

uped by function or robot system into layers. Behavior primitives can also suppress the puts of other primitives (denoted by an "s" enclosed in a circle). This allows for the ation of a hierarchy inside a behavior layer. For example, in the leader behavior set there two layers: a communication layer and an optical control layer. When the leader receives equest to activate a specific set of LEDs from its partner, the "Activate Specific LEDs" iavior primitive suppresses the "Activate All LEDs" behavior primitive and activates the uested LEDs.

d-leader-methodical Behavior Set

e *find-leader-methodical* behavior set is activated once a leader has begun to assemble the ots to form a chain for stair climbing. Once the leader has activated its optical array and t a communication message requesting a new docking partner, the find-leader-methodical iavior set can be activated to search for the leader and its LED beacon. The methodical rch procedure is depicted in Figure 5.4.

The methodical search procedure starts with the robot performing a complete in-place n and then driving forward. After a short distance the robot spins again and then contin-

ues. If, at any point during these maneuvers, the robot detects the LED beacon of the leader it stops and the behavior is complete. If the robot reaches the boundary of the mission area it performs a right turn and continues. The robot periodically reports its progress to other robots while this behavior set is active.

find-leader-wander Behavior Set

The *find-leader-wander* behavior set is an alternative to the find-leader-methodical behavior set. This behavior (Figure 5.6) has the robot drive in a straight line until it reaches the mission area boundary where it changes direction, and then continues. The robot stops when the leader's LED beacon is detected.

determine-relative-position Behavior Set

The *determine-relative-position* behavior set is activated once a docking robot has detected the leader's optical signal and wishes to determine its position relative to the leader. The docking robot attempting to dock with the leader sends communication messages to the leader requesting that the leader enable one of its LED beacon quadrants. The port, starboard, front, and rear LED clusters are enabled in sequence on the leader at the request of the docking robot. When the docking robot optically detects the leader again it notes the last LED activation request and determines it position relative to the leader. If the leader is not detected after cycling through all of the LED clusters the docking robot acquiesces and the leader begins to look for another docking partner.

navigate-to-rear Behavior Set

The *navigate-to-rear* behavior set (Figure 5.10) was designed to position the docking partner robot directly behind the leader robot. If the partner robot is to the left or right of the leader robot it turns and uses its side-looking infrared receiver modules to drive parallel to the leader robot. For example, if the partner robot had discovered that it was on the starboard side of the leader robot from the determine-relative-position behavior set, it will turn and use its starboard infrared detector module to detect the light emitted from the starboard LED

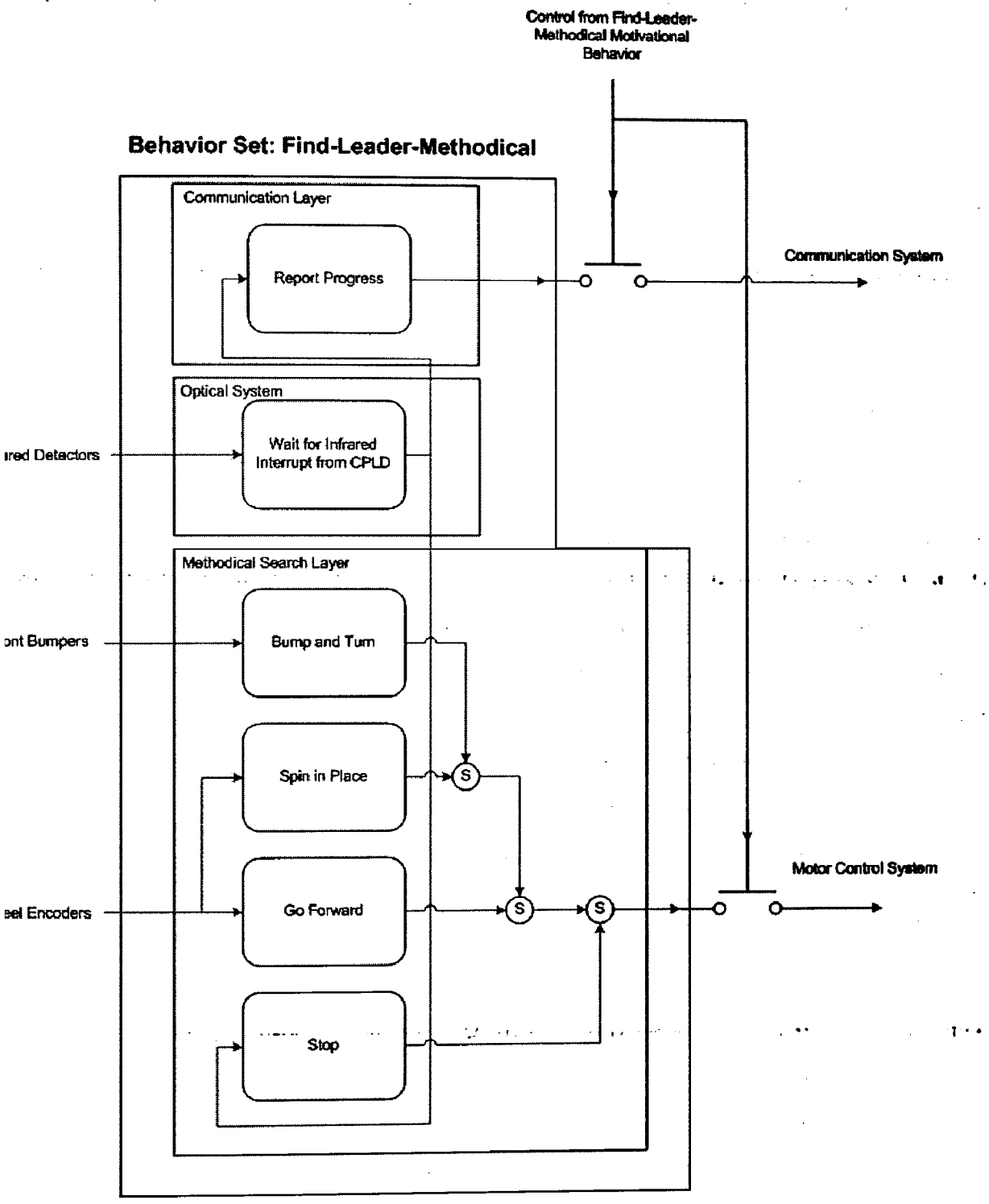


Figure 5.5: *The Find-Leader-Methodical Behavior Set*

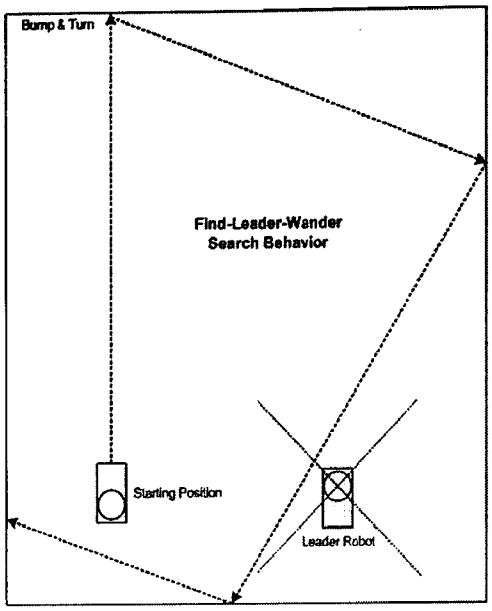


Figure 5.6: *Find-Leader-Wander Behavior*

cluster on the leader robot (Figure 5.9 (a)). The partner robot would then drive forward until it loses contact with the starboard LEDs on the leader. The partner robot would then signal the leader to disable its starboard LEDs and enable its rear LED cluster. Once the rear LEDs are enabled the partner robot makes a right turn and then begins counting wheel rotations as it drives through the sweep of the leader's rear LEDs (Figure 5.9 (b)). The partner robot reverses its course once it loses contact with the leader's rear LEDs. Back at the starting position, the partner robot tells the leader to enable its single fine docking LED and drives forward exactly half the number of wheel rotations recorded in the previous step (Figure 5.9 (c)). This places the robot approximately on the center line of the leader robot. The partner robot uses the signal from the leader's fine docking LED to perform a turn and ends up facing the rear docking plate of the leader (Figure 5.9 (d)).

docking-approach Behavior Set

The docking robot activates the *docking-approach* behavior set once it has navigated to a point directly behind the leader robot. At this point, the only LED actively emitting a signal on the leader robot is the fine docking LED. This provides a narrow beam of light that the

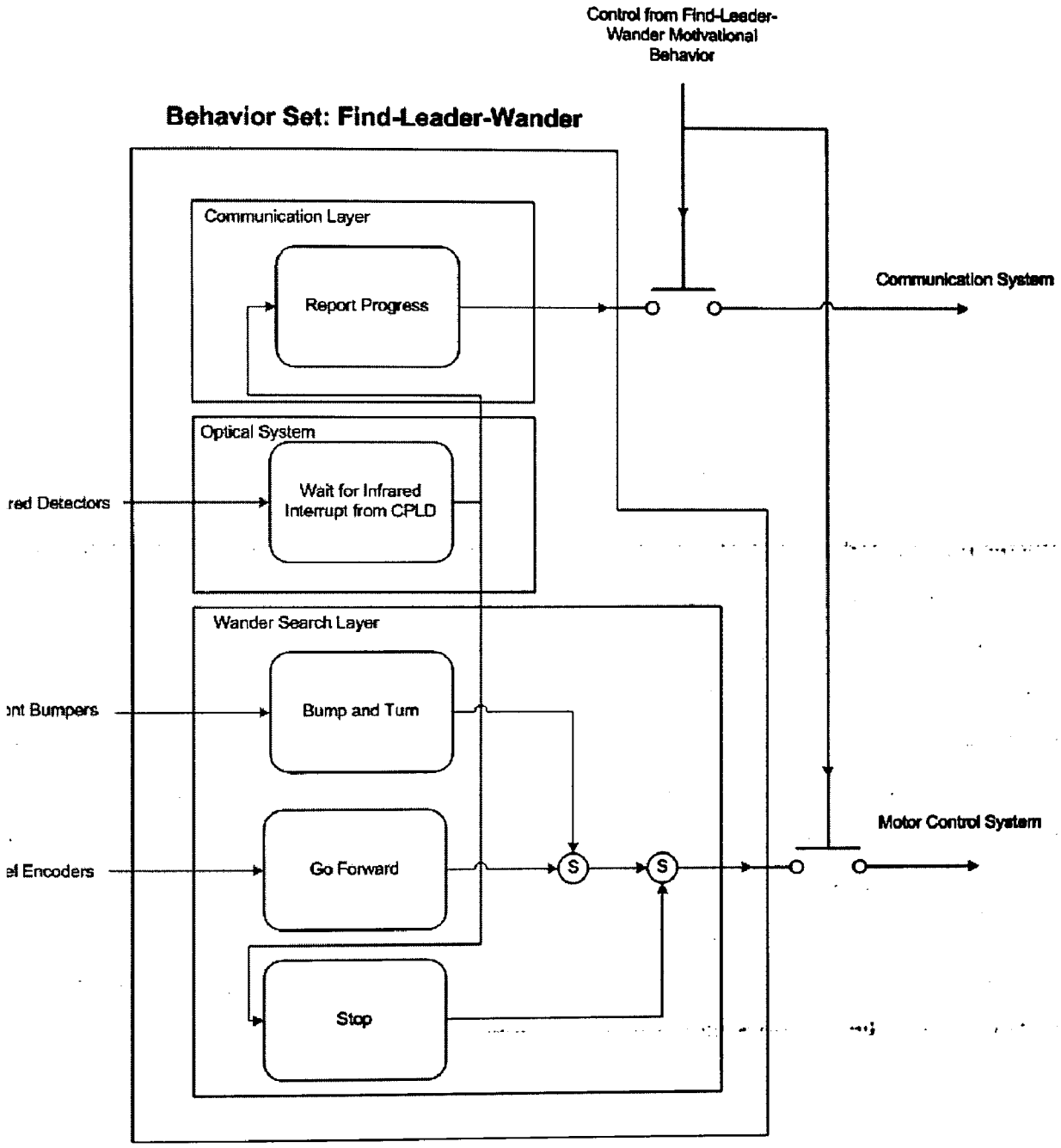


Figure 5.7: *The Find-Leader-Wander Behavior Set*

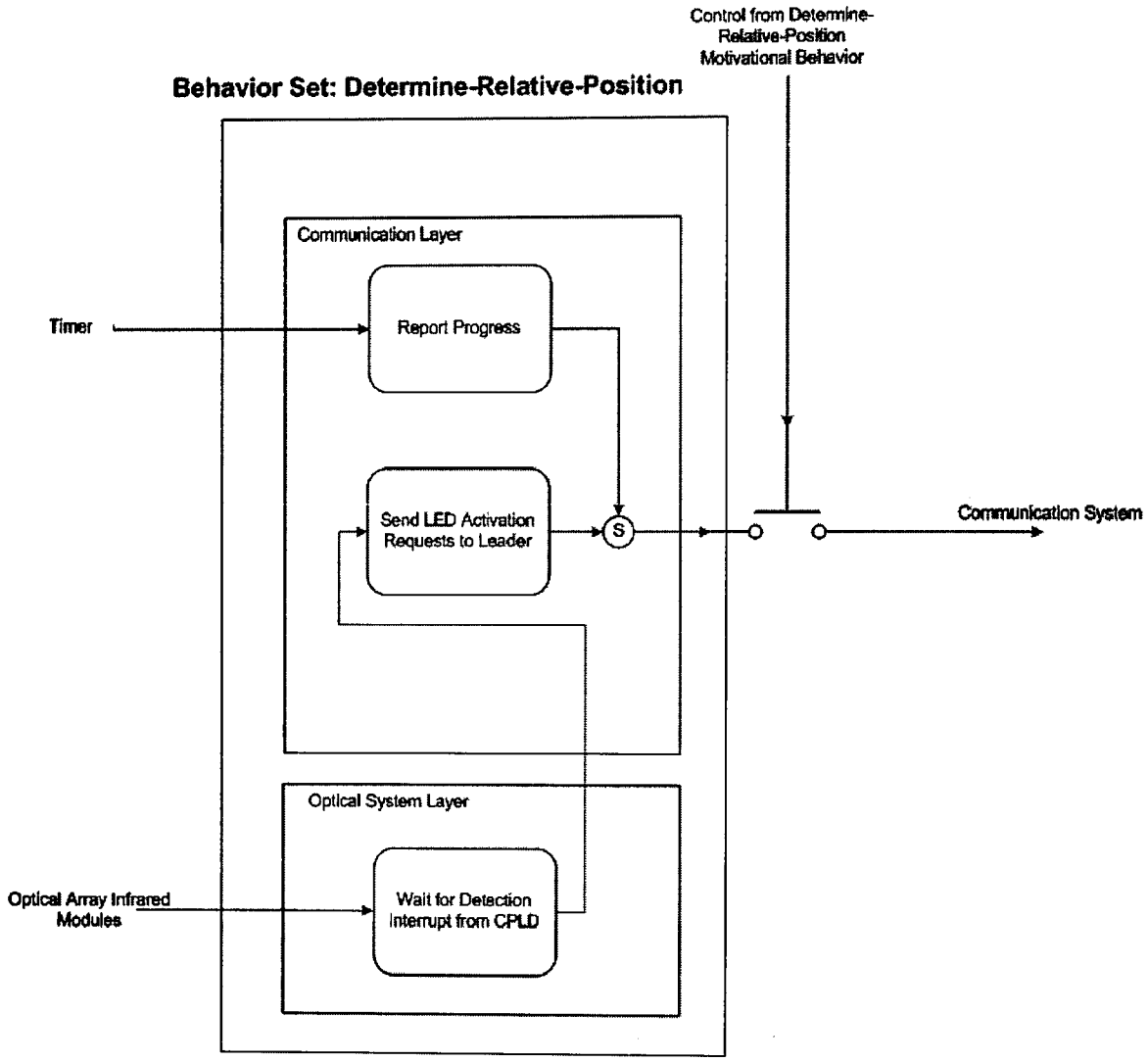


Figure 5.8: *The Determine-Relative-Position Behavior Set*

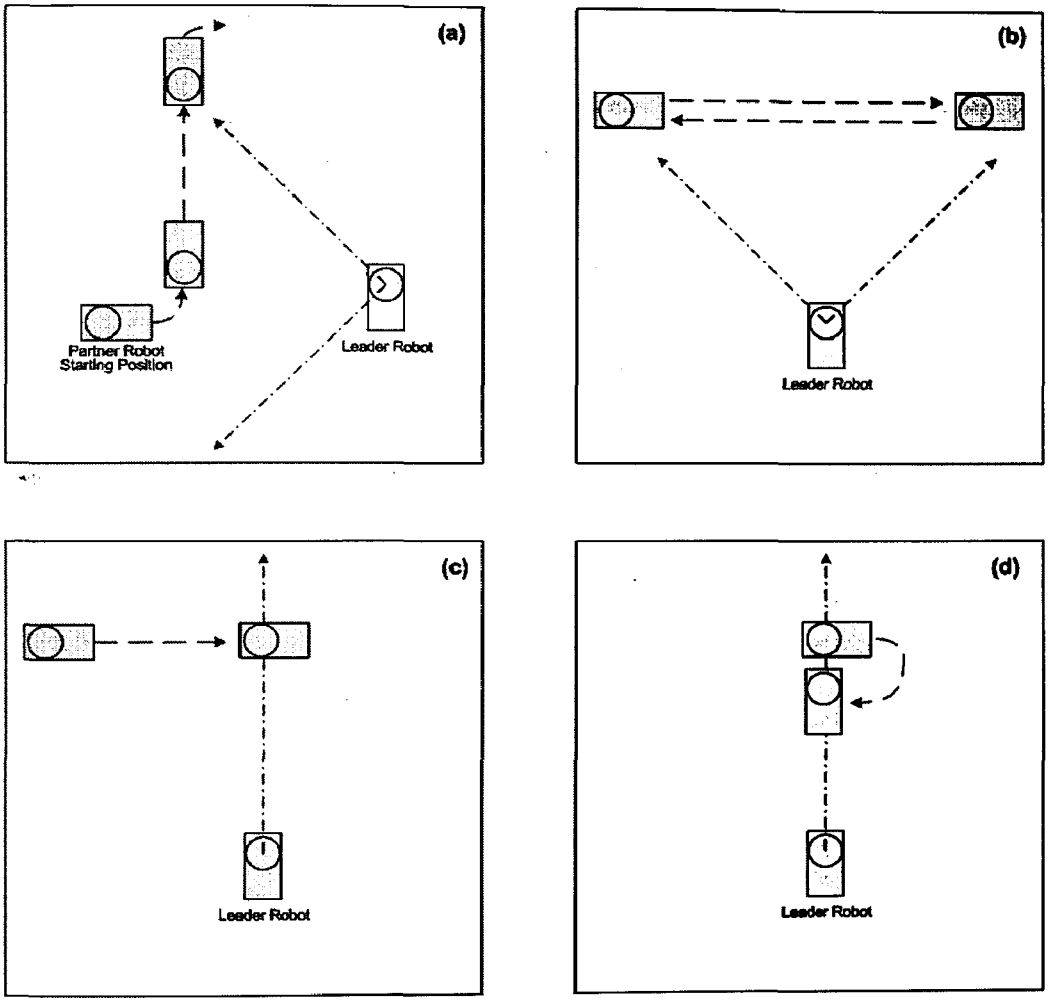


Figure 5.9: *Illustration of the Navigate-to-Rear Behavior Set*

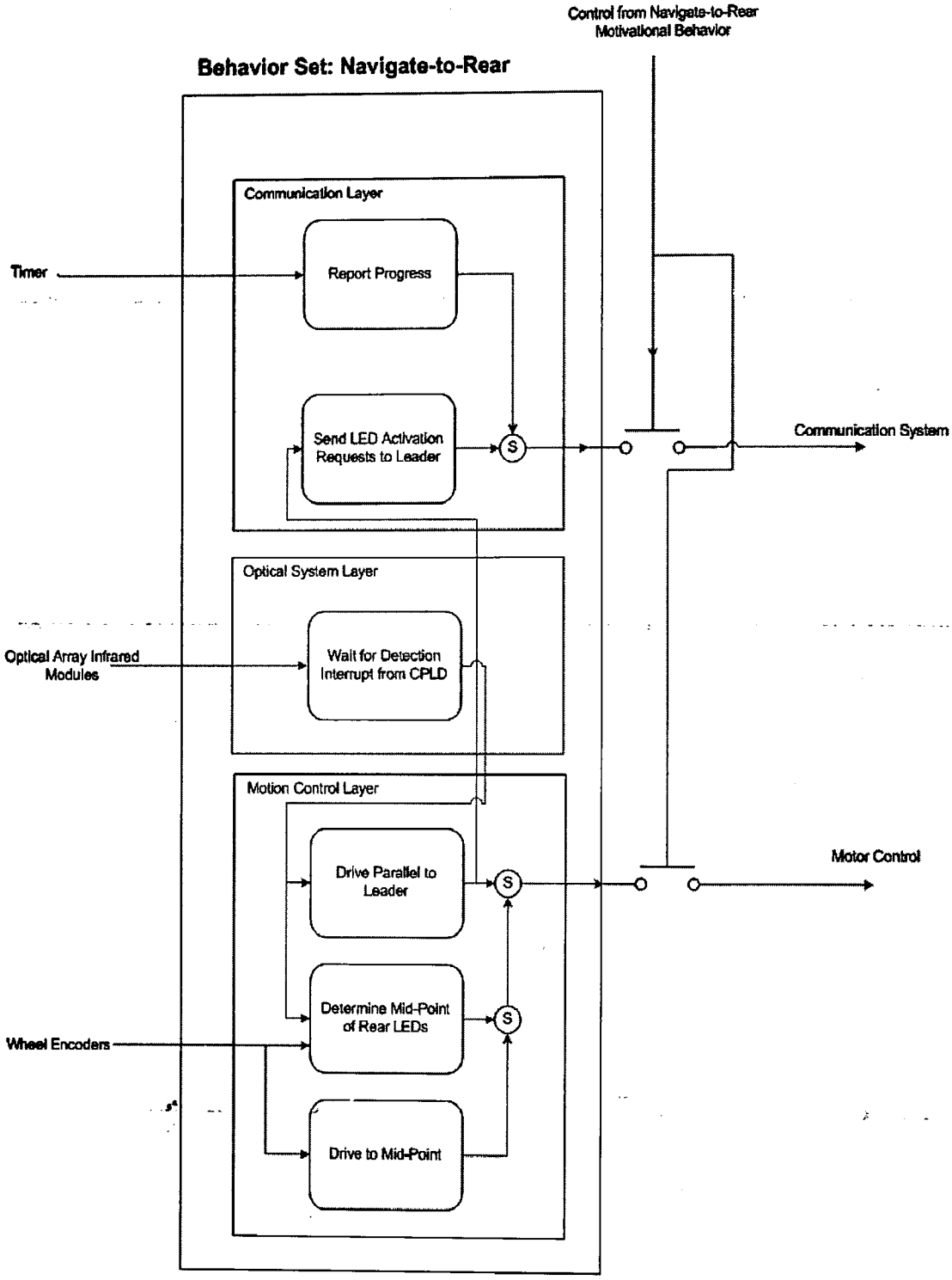


Figure 5.10: *The Navigate-To-Rear Behavior Set*

king partner can follow in to the rear of the leader. This behavior set terminates when docking robot collides with the back of the leader robot as detected by the front bumper tches.

:king-align Behavior Set

: *docking-align* behavior set is used to align the docking plates of the leader robot and docking robot (Figure 5.12). The docking robot uses its front bumper switches to place lf in line with the leader robot. This is accomplished when both bumper switches are uated against the rear docking plate of the leader robot. If only one switch is actuated, docking robot reverses a small distance and turns slightly towards the switch which was activated.

:form-dock Behavior Set

: *perform-pock* behavior set (Figure 5.13) is activated once the docking robot has aligned lf with the leader. The lifting mechanism is extended so that the front docking plate mates h the rear docking plate of the leader. Next, the lifting mechanism is retracted bringing front docking plate down over the rear docking plate. Once the lifting mechanism has y retracted the docking plates are aligned and sandwiched firmly together. Finally, the pling mechanism is driven into the rear docking plate of the leader locking the two robots ether.

3.2 Behavior Class Progression and the ALLIANCE Architecture

: docking procedure for the Gremlin robots is divided into behavior sets: *determine-itive-position*, *navigate-to-rear*, *docking-approach*, *docking-align*, and *perform-dock*. In this resentation of the ALLIANCE architecture we propose a behavior class called *docking-ss* which encompasses all of these behavior sets. The purpose of the behavior class is -fold. First, grouping all of the docking behavior sets into a single class allows robot m members to integrate the current progression through the docking maneuver into the

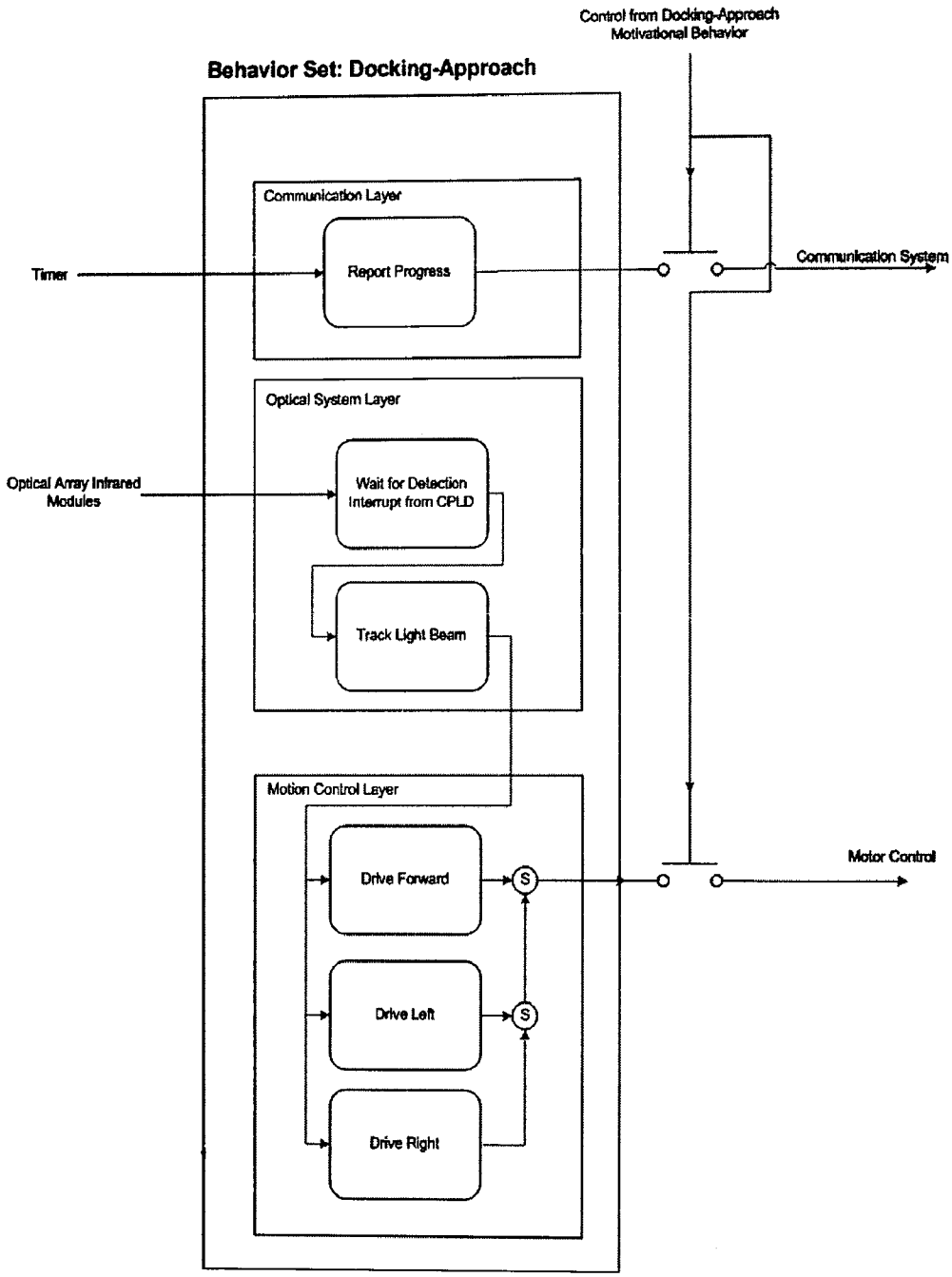


Figure 5.11: *The Docking-Approach Behavior Set*

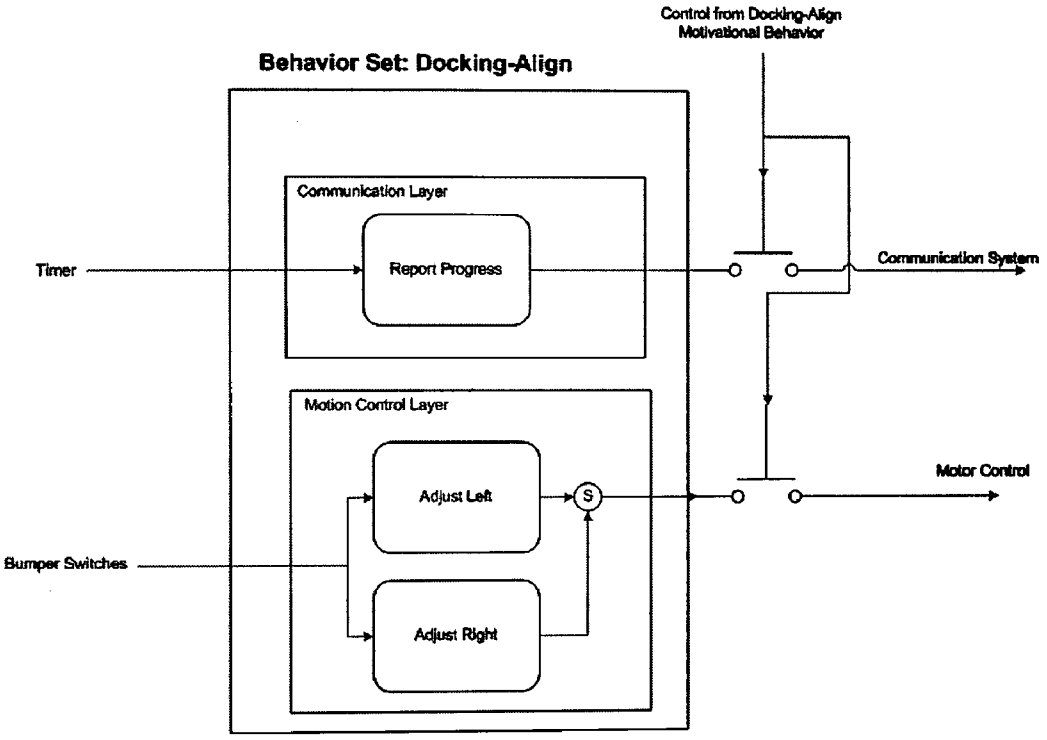


Figure 5.12: *The Docking-Align Behavior Set*

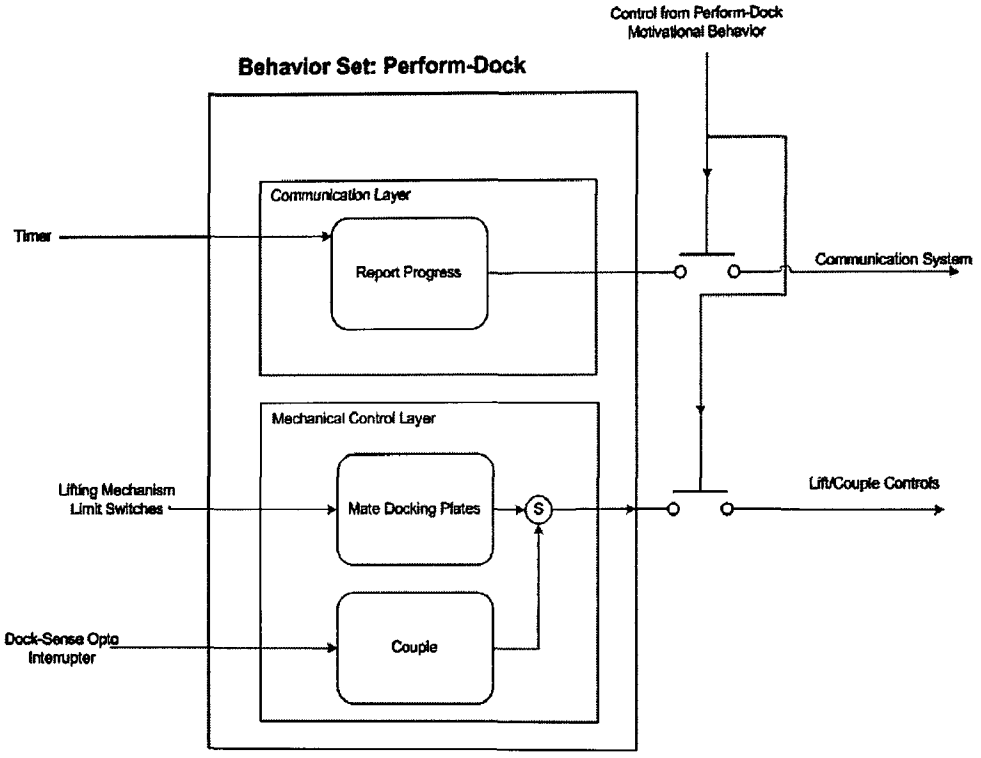


Figure 5.13: *The Perform-Dock Behavior Set*

:ulation of impatience. In other words, robots would be "more patient" with a robot that loser to completing the docking maneuver as opposed to a robot that has just begun. example, assume that Robot A is monitoring the progress of Robot B who is attempting lock with the leader. If Robot B is failing to report progress in the determine-relative-ition behavior set which is the first set in the docking class, Robot A will become impatient :he normal rate and take over the docking task from Robot B. However, if Robot B is gressing slowly through the docking-align behavior set, which is one of the last stages in docking procedure, Robot A will wait longer by taking into account that Robot B has :ady successfully completed three previous behavior set stages in the docking class.

We define a new function, $\gamma_{kc}(t)$, which represents robot r_k's progression through the avior class C. A behavior class in robot r_i is defined as the set $C_i = \{a_{in}, a_{im}, \ldots\}$ which ıade up of all the behavior sets required to perform a task with an ordering dependency. mally, behavior class progression is represented by redefining impatience in the behavioral del:

$$impatience_{ij}(t) = \begin{cases} \frac{min_k(\delta\_slow_{ij}(k,t))}{\gamma_{kc}(t)} & \text{if } (comm\_received(i,k,j,t-\tau_i,t) = 1) \text{ and} \\ & (comm\_received(i,k,j,0,t-\phi_{ij}(k,t)) = 0) \\ \delta\_fast_{ij}(t) & \text{otherwise} \end{cases}$$

: function $\gamma_{kc}(t)$ can be tailored to each behavior class. Its general form is:

$$\gamma_{kc}(t) = k_n \cdot h(a_{in}) + k_m \cdot h(a_{im}) + \cdots,$$

where $h(a_{in})$ represents information about r_i's progression through behavior set a_{in}. In-nation such as $h(a_{ij}) = 1$ means that robot r_i has successfully completed the behavior set while $h(a_{ij}) = 0$ means that the behavior set has not yet been completed. The constants and k_m are weighting coefficients assigned to each behavior set in the class. These coeffi-its can be set to values which reflect the difficulty or length of time required to complete isk with a particular behavior set.

For example, if the navigate-to-rear behavior set requires a long time to complete its fficient would be set to a large value. This means that members of the robot team

observing a robot who has just completed the navigate-to-rear behavior set would wait longer for robot r_i to complete subsequent tasks because they know that r_i has just completed a task which is time consuming. This prevents the situation where a robot completes a very difficult task but has to acquiesce the following task because it takes slightly longer than normal to complete. As well, robots that are further along in the docking process would be allowed more time to complete the maneuver should they encounter small difficulties towards the end.

For the docking class, all of the coefficients in the progression function are set to unity:

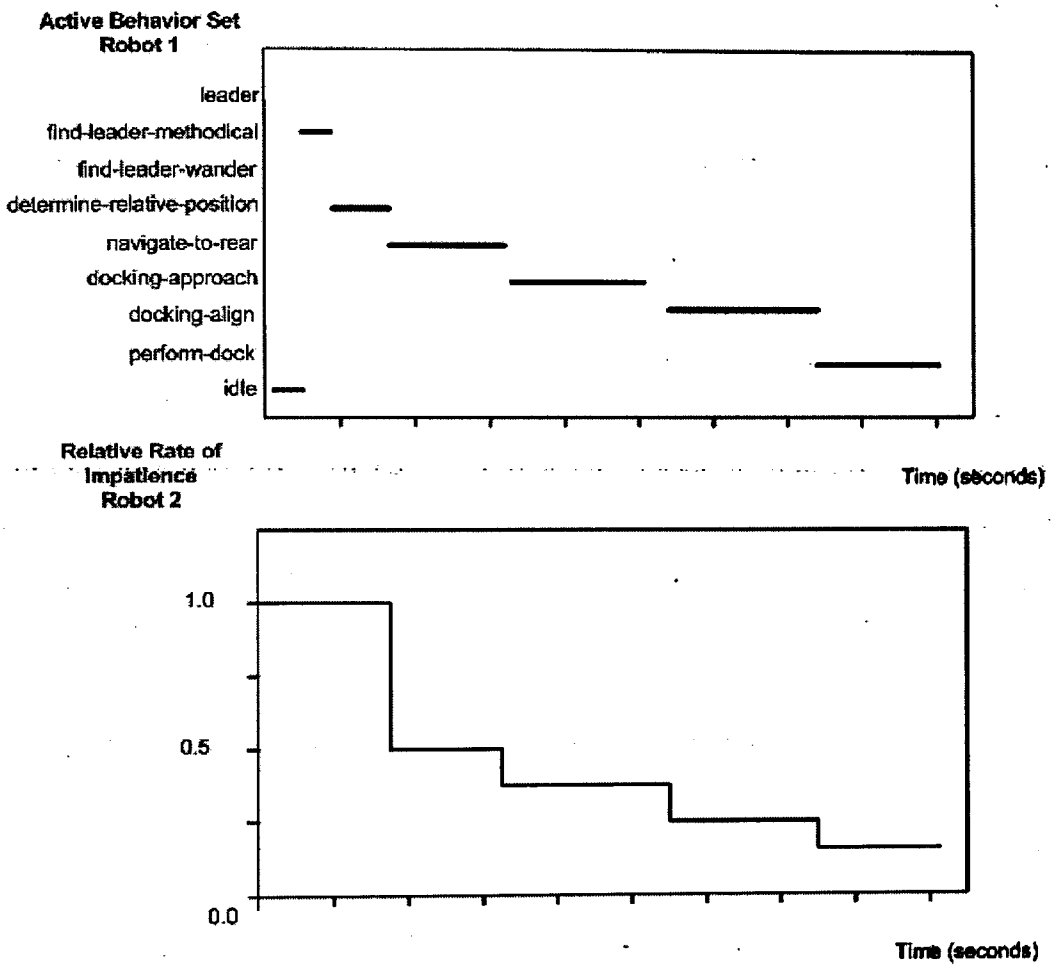
$$\gamma_{kc}(t) = h(a_{i1}) + h(a_{i2}) + h(a_{i3}) + \cdots$$

Clearly, in this case the rate of impatience decreases with each task in the behavior class that is completed. Figure 5.14 provides an example of the progression mechanism. In this example, Robot 1 is performing a docking maneuver be executing the behavior sets in the docking class in order. Each time that Robot 1 successfully completes a stage of the docking maneuver by progressing through a behavior set, the rate of impatience of Robot 2 decreases. In essence, Robot 2 is becoming "more patient" with Robot 1 because Robot 1 is making progress.

The second motivation behind creating behavior classes relates to the ordering dependencies of the behavior sets. The next section discusses how the behavior class organization facilitates ordering dependencies.

5.3.3 New ALLIANCE Ordering Dependency Mechanism

The behavior sets required to perform autonomous docking must be executed in a specific order. For example, there is little point for a robot to activate the docking-approach behavior set if it is not behind the leader robot. The behavior class provides an organizational mechanism to ensure that the behavior sets are executed in the proper order. The behavior sets belonging to a particular behavior class are numbered in ascending fashion in the order they must be executed. For example, in the behavior class $C_i = \{a_{i1}, a_{i2}, \ldots\}$ the behavior



ure 5.14: *Comparison of Behavior Class Progression to Rate of Impatience in an Observer* ot

set a_{i1} must be successfully executed before a_{ij} can commence. The first behavior set in a class, $a_(i1)$, is always assigned the highest $\delta_{fast}(t)$ so that it would automatically activate first. If behavior set a_in completes successfully, it modifies the current level of motivation for the next behavior set in the class so that it would be executed immediately. Formally:

$$m_{i(n+1)}(t) = m_{i(n+1)}(t-1) + (\theta - m_{i(n+1)}(t-1))$$

In other words, the successful completion of one behavior set in a class triggers the activation of the next set in the class. This mechanism ensures the correct sequence of execution of behavior sets with ordering dependencies within a given behavioral class.

5.3.4 Modified Climbing Control

The motion control system presented in [43] was implemented on the PolyBot G1v4 robot (Chapter 2). Recall that the PolyBot was made up of identical modules that could be assembled to create different robot configurations such as snakes, or long serial chains. A construct known as a gait control table (GCT) was used to control the climbing motions of the robots (Table 2.1). A gait control table imposes open loop control on the robot in the form of a prescribed motion. The elements of the table correspond to desired physical states of a robot mechanism such as a joint. Each row in the table corresponds to one step or state in the motion and each column corresponds to a robot in the chain.

A modified version of the gait control table is used as the basis for the Gremlin climbing control system (Table 5.1). The gait control table for the Gremlin robots implements a travelling square wave that meshes mechanically with the staircase. Entries in the table dictate whether the lifting mechanism of a robot is in the extended or retracted position. A "1" in the table indicates that the lifting mechanism is extended, and a "0" indicates that the lifting mechanism is retracted. The motion control column determines if the robot chain moves forward in a particular state (denoted by a "1" in the table). The state transition column indicates a condition that must be met before the robot chain would transition to the next state.

	Robot Number			Forward Motion	State Transition Condition
	1	2	3		
State 0	0	0	0	0	1F(1)→S1
State 1	1	0	0	0	1S(1)→S2
State 2	0	1	0	1	2S(1)→S3
State 3	1	1	0	0	1F(0)→S4
State 4	1	0	1	1	1S(1)→S5
State 5	0	1	1	1	3S(0)→S6
State 6	0	1	0	1	1S(1)→S2 else → S7
State 7	0	0	1	1	3S(1)→S8
State 8	0	0	0	1	Done

Table 5.1: *Gait Control Table for Gremlin Stair Climbing*

An entry in the state transition column begins with a number and a letter. The number otes the robot number in the chain to whom the condition is assigned with 1 being the ot at the front of the chain. The letter indicates either the bumper switches on the front king plate, "F", or the switches on the base tier which are used for detecting the face ı stair, "S". The number in brackets indicates the state of the switch which satisfies the nsition condition. A "1" indicates that the switch must be actuated and a "0" indicates t the switch must be open. The entry is concluded with an arrow which points to the t state in the table. State 6 is the only entry in the table which has two state transitions. e climbing procedure is defined by States 2 through 6. These states repeat until the robot in has reached the top of the stairs at which point State 6 transitions to State 7 which npletes the climbing procedure and brings all robots to the top of the stairs in retracted n.

Figure 5.15 is a visual representation of the stair climbing algorithm defined by the gait trol table.

4 Proposed Gremlin Communication System

e Gremlin communication system is organized into three levels. The highest level provided ymbolic representation for communications based on a sign-board model. The media ac-

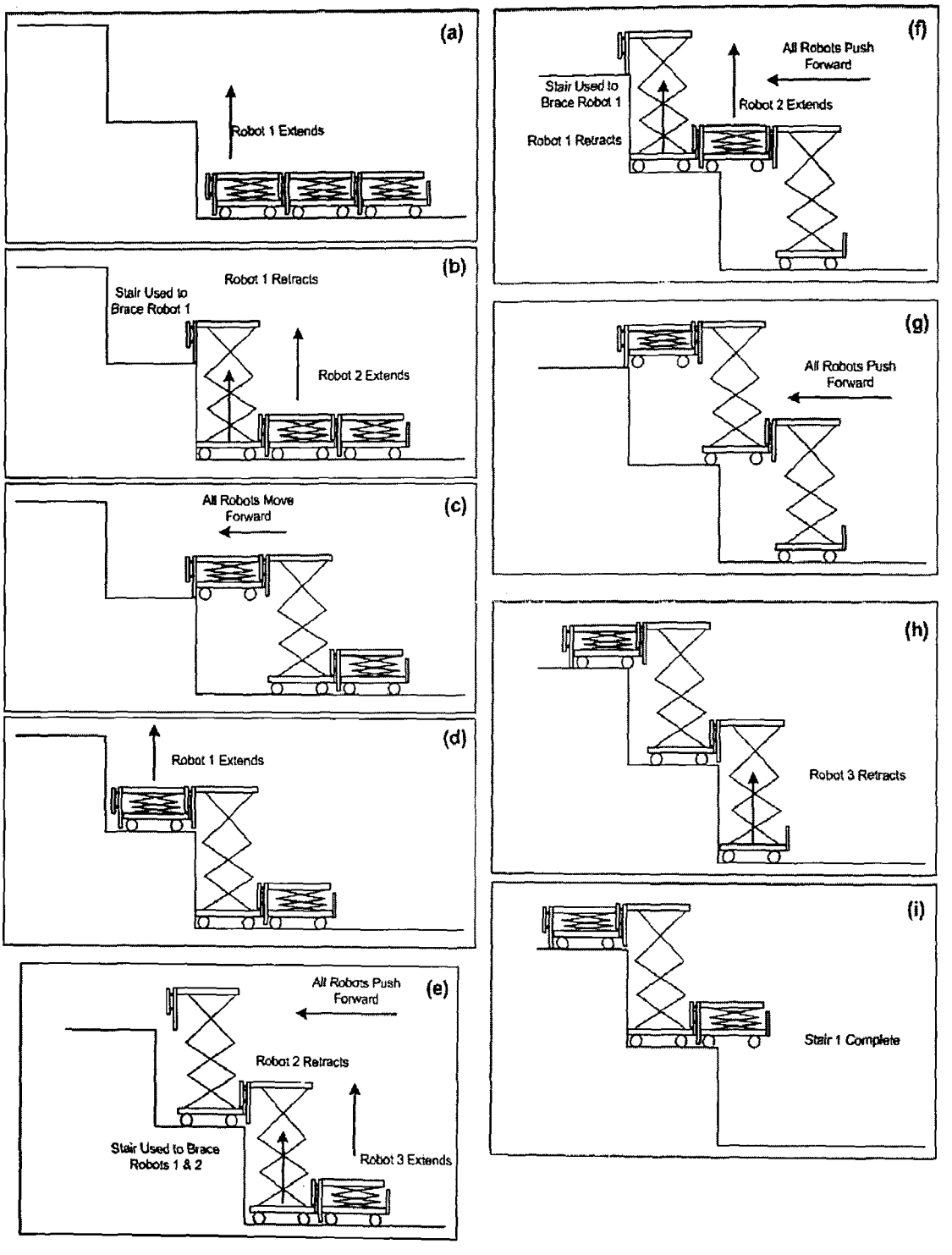


Figure 5.15: *Gremlin Stair Climbing Procedure*

control level is based on the AR-TDMA protocol and is used to structure the way in ch the robots accessed the communication channel. At the lowest level, a communica- protocol is implemented to facilitate the transmission of asynchronous data from the rocontroller through the wireless transceiver.

.1 High-Level Communication Organization

ts highest level of abstraction, the Gremlin communication architecture is based on the -board communication model presented in Chapter 4. Each robot maintained a sign- rd which contained information about its currently active behavior set, and the current ls of impatience and acquiescence in the robot (Figure 5.16). A robot with an active avior set inhibits the activation of similar behavior sets in other robots by inhibiting a icular class of behavior sets. Robots which are idle and are monitoring the activities of ther robot post their current levels of impatience on the sign-board.

The the example shown in Figure 5.16, the sign-board for the robot on the left indicates the robot is currently attempting to dock with the leader, is inhibiting other robots a activating their docking behaviors, and the current level of acquiescence is 65 with a shold of acquiescence of 128. The robot on the right is currently idle. It is currently nitoring the progress of the other robot and has two active motivational activities: find- ler-methodical and find-leader-wander. The impatience rate for the methodical search avior set is higher than that of the wander behavior. Assuming the robot on the left is uccessful in docking the robot on the right would execute its methodical search behavior when the level of impatience crosses the threshold. At this point, the robot on the left taken over the docking procedure.

During experimental testing, the sign-board information of each robot is recorded by a ote computer for analysis.

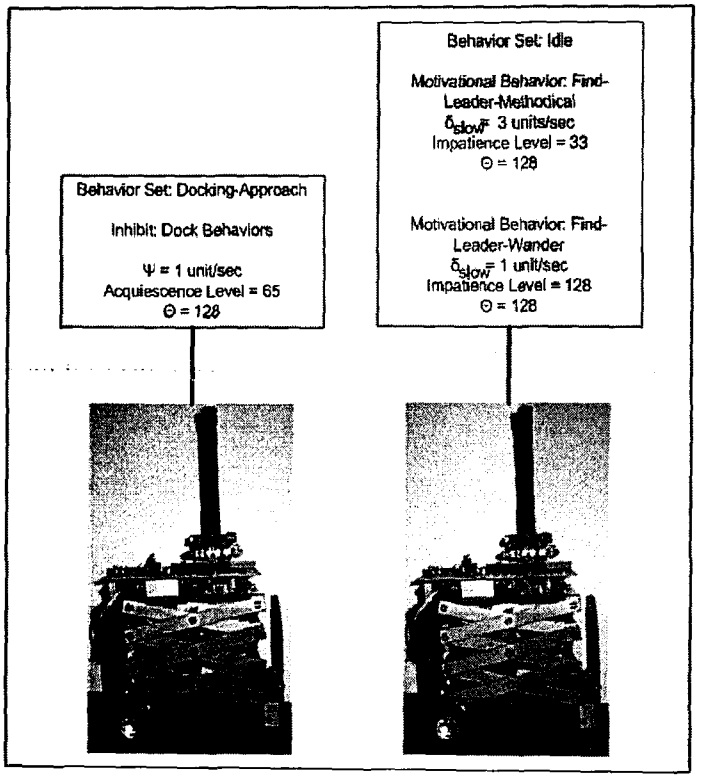


Figure 5.16: *Example of Gremlin Sign Boards*

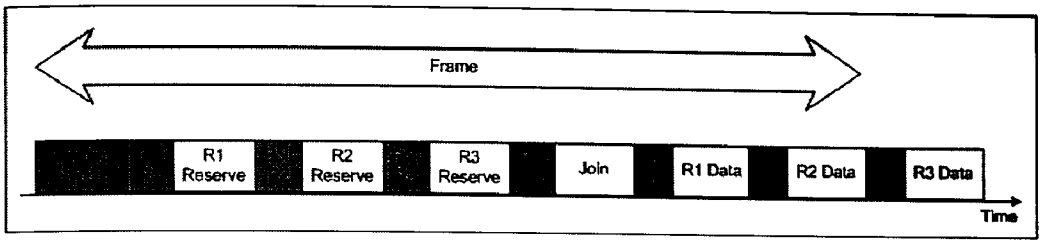


Figure 5.17: *Gremlin TDMA Frame*

.2 Proposed Media Access Control

propose a media access control scheme which synchronizes the communication messages ıe robot team, automatically builds the TDMA frame with each robot that is added to team, and inherently detects robot failures without the need for a communication base ion. Figure 5.17 shows the implementation of the AR-TDMA protocol used for media ːss control by the Gremlin robots. The dark grey slots indicate communication message ı the leader. In the first slot, the leader broadcasts a synchronization message to indicate ; a new TDMA frame is about to start. Next the leader polls each robot to determine if r have data to transmit. After polling each robot the leader sends a general "join" query. nother robot would like to join the team it responds with its identifier in the join slot is added to polling in the next frame. The leader then allocates data slots to each of robots who responded during polling earlier in the frame.

Figure 5.18 shows the progression as robots are added to the TDMA frame. The process ins with the first robot powering up. The robots monitors the communication channel ı short time and if no communications are detected the robot assumes that there are no ːr robots present and designates itself as the leader. The new leader begins sending AR-MA frames which consist of a synchronization slot and a "join" slot. When the second ɔt power up it detects the communication traffic from the leader and responds in the n" slot with its identifier. The leader recognizes that a robot has requested to join the n and creates a reservation slot for the second robot. The reservation slot signifies that leader recognizes the second robot as a member of the team and provides an opportunity the second robot to make a request to send data. The third robot powers up and follows

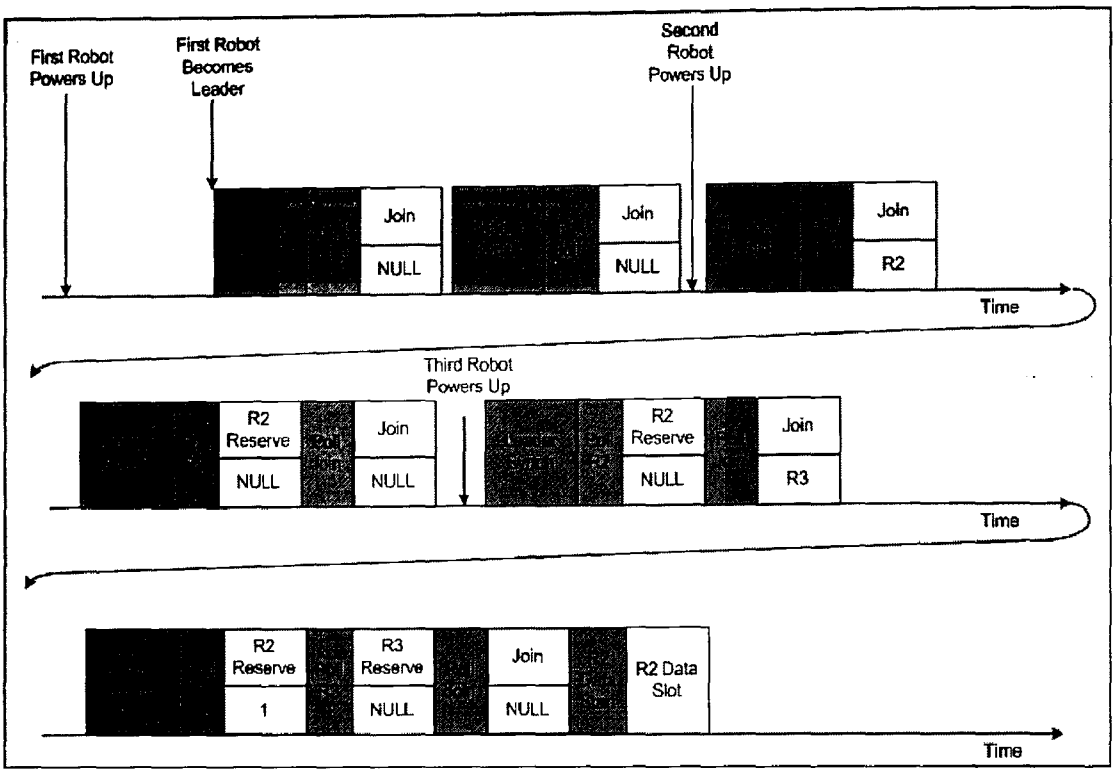


Figure 5.18: *Adding Robots to the TDMA Frame*

the same registration as the second robot. In the last TDMA frame, the third robot has been added to the team and the second robot makes a request to use the channel. The leader grants a data slot at the end of the frame to the second robot who then proceeds to send data over the channel.

The TDMA communication frame is constructed dynamically as robots join and leave the team. Each time a robot powers up to join the team, the TDMA frame is expanded with a polling frame for that robot. If a robot fails to answer a poll from the leader robot, it is automatically declared to be non-functional and is removed from subsequent frames.

5.4.3 Low-Level Communication Implementation

The lowest-level of the communication architecture defines the structure of the messages that are exchanged by the robots through the communication channel. Each message begins with a series of training bytes. These bytes are binary data of the form *101010101....* The purpose

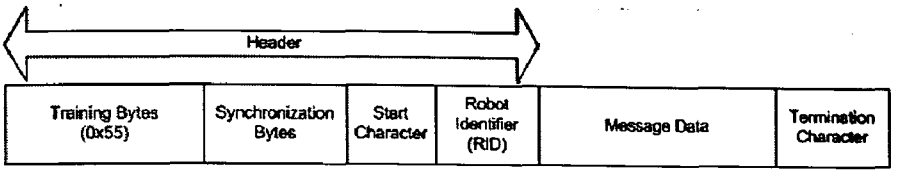


Figure 5.19: *Gremlin Message Structure*

he training bytes is to allow the data slicer in the receiving transceiver to stabilize. Next ɔecialized set of synchronization characters are sent. These characters ensure that the RT of the receiver is properly aligned to the bits being sent from the transmitter. A ıplete analysis of the synchronization procedure is presented in Appendix C. Once the ːiver has been aligned, a character signalling the start of the message is transmitted. This racter is followed by a number which identifies the transmitting robot. This identification racter also signals the end of the message header. After transmission of the header is ıplete the robot transmits its message data and then terminates the message with a ɔial character.

Chapter 6

Gremlin Robot Design

THE robots synthesized for this project are affectionately named *Gremlins* because of their small size and seeming affinity for mechanical problems. This chapter details the design of the Gremlin robots so that the reader is familiar with the robot platform including the high-level electrical and mechanical design. The robots that were designed are unique and were built by hand specifically for this project.

6.1 High-Level Robot Design

The purpose of this research is to design a system of autonomous mobile robots that could collectively reconfigure to climb stairs. The robots are required to operate individually and then reconfigure to create a larger entity capable of traversing stairs. The reconfiguration of the robot team entailed the robots docking and coupling with each other to form a single chain.

The following high-level design objectives outline the features necessary to implement the design goals:

- The robots must be equipped with a visual localization mechanism to facilitate docking.
- The robots, once coupled together, should be able to climb stairs of a standard height.
- The lifting mechanism employed for climbing should be compact in order to keep the robots as small as possible.

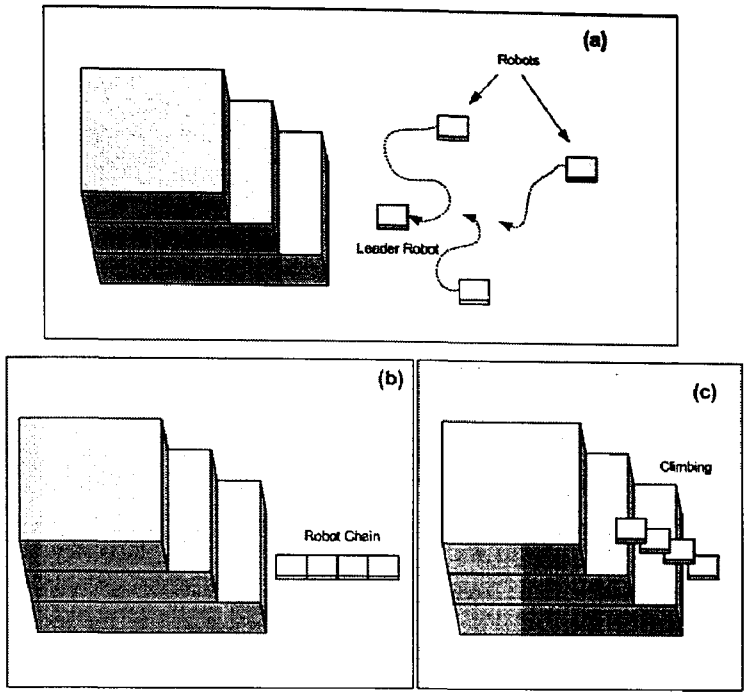


Figure 6.1: *Gremlin Stair Climbing Process*

- The coupling mechanism should provide a rigid, reliable mechanical link between the robots.
- The coupling mechanism should allow the robots to decouple.
- The docking mechanism should allow for misalignment of the robots and should facilitate automatic alignment.
- The robot should carry their own power supplies, function without external control and require no external tethers of any kind.

The modified ALLIANCE architecture presented earlier is the basis for the behavioral trol during docking and reconfiguration. A simple, yet effective, method for controlling robot chain during stair climbing is presented in Chapter 5. The entire reconfigura-/stair climbing procedure (Figure 6.1) is executed as follows:

. The first robot to be powered on becomes the leader, and is the first robot in the robot

chain.

2. The leader navigates into position in front of the stairs.

3. The leader begins to communicate with other robots to start the reconfiguration process.

4. The leader uses its optical localization system to present its position to the other robots.

5. The other robots attempt to find the leader robot visually.

6. Once another robot has detected the leader, it notifies the leader using the communication system.

7. Using the optical localization system, the leader guides the follower robot into docking position.

8. The follower robot docks with the leader robot.

9. The follower robot mechanically couples to the leader robot and becomes the new leader.

10. The docking/coupling process repeats until all robots have coupled.

11. Once the robot chain has been formed, the robots climb the stairs using their mechanical lifting systems.

12. At the top of the stairs, the robots decouple and resume independent operation.

The next section presents the electrical and mechanical features of the Gremlin robots that enable this stair climbing ability.

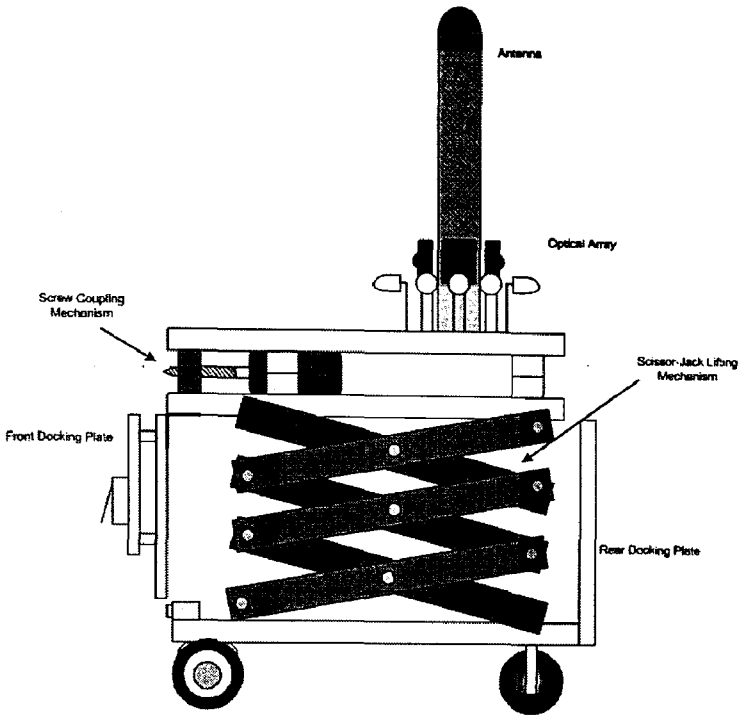


Figure 6.2: *Gremlin Robot Overview*

2 Mechanical Design

ıre 6.2 shows an overview of the Gremlin robot systems. There are three main compo-
ts to the mechanical design of the gremlin robots: the lifting mechanism, the docking
:hanism, and the coupling mechanism. The lifting mechanism is used to extend the over-
ıeight of the robot during climbing and docking. The docking mechanism, in conjunction
ı the lifting mechanism allowed the robots to attach to one another. The coupling mech-
ım allowed the robots to lock together once docked.

Small gear motors (Figure 6.3) are used in the mechanical design of the Gremlin robots
. These motors are compact, inexpensive, draw little current, and provid large amounts
orque relative to their size. They are instrumental in creating a small, compact robot
gn.

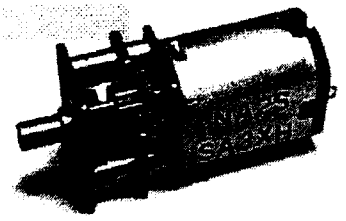


Figure 6.3: *Small Gear Motor*

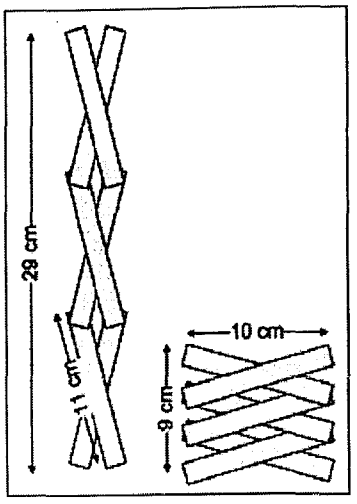


Figure 6.4: *Dimensions of the Scissor Jack Lifting Mechanism*

6.2.1 The Lifting Mechanism

The Gremlin lifting mechanism employs a scissor jack (Figure 6.4). A scissor lift mechanism offers large vertical travel and can be retracted to a fraction of its extended height. This feature is vital to making the robots small and compact. The mechanism is actuated by a gear motor driving a threaded rod (Figure 6.5). The threaded rod provides the additional mechanical advantage required to actuate the lifting mechanism.

Opto interruptors are employed to act as limit switches for the lifting mechanism. These optical switches are placed at the base of the lifting mechanism to allow the control electronics to detect when the mechanism was fully extended or retracted (Figure 6.6).

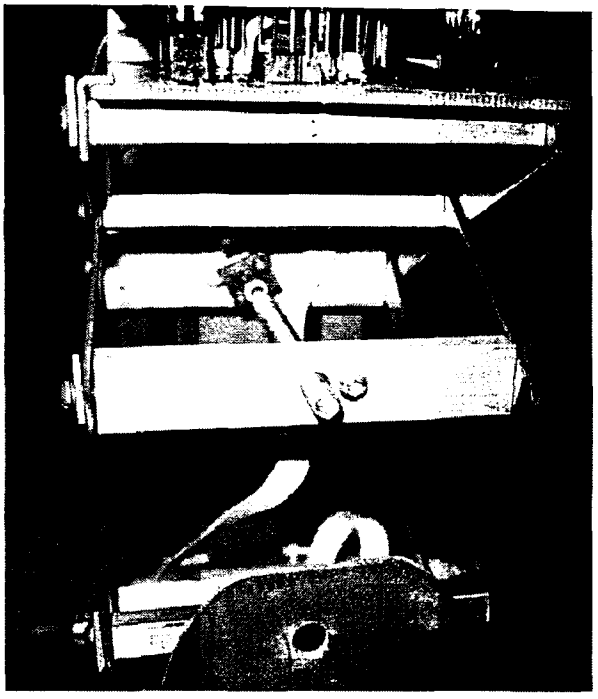


Figure 6.5: *The Threaded Rod Driving the Lifting Mechanism*

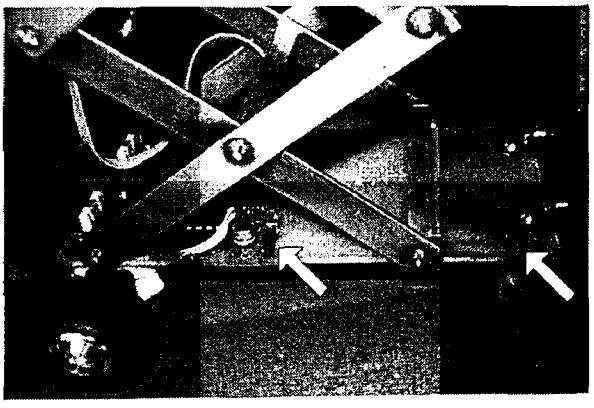


Figure 6.6: *Opto Interruptor Limit Switches*

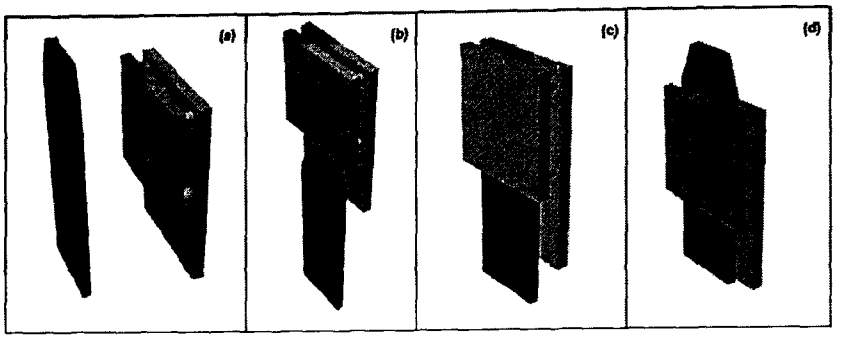


Figure 6.7: *Gremlin Docking Concept*

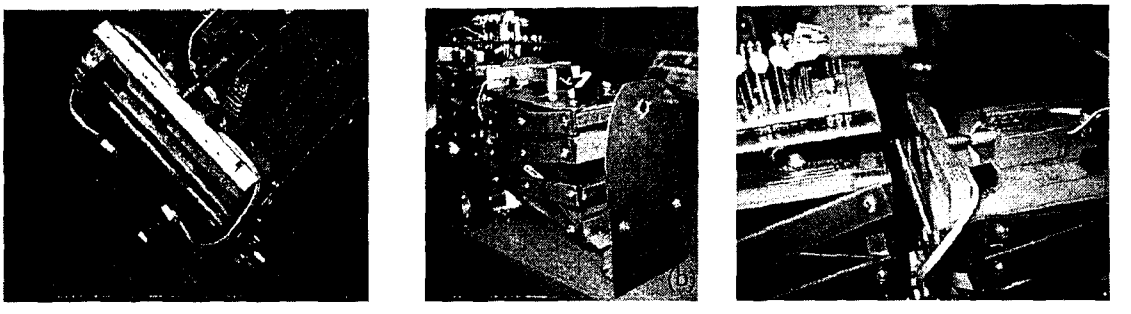


Figure 6.8: *(a) Front Docking Plate (b) Rear Docking Plate (c) Docked*

6.2.2 The Docking Mechanism

The docking concept for the Gremlin robots is shown in Figure 6.7. The docking mechanism employs a slide-and-lock strategy. On the front of each robot are two plates separated by spacers. These plates mate with a complimentary tapered plate on the back of each robot. Docking is performed by lifting the front plate of one robot over the rear plate of another robot and then lowering the front plate into place (Figure 6.8). The tapered design of the rear docking plate allows for a maximum of 2 cm of off-center misalignment in either direction.

The robots align the front and rear docking plates using bumper switches on the front docking plate. A successful docking alignment is detected when both bumper switches were in contact with the rear docking plate. Once the front docking plate of one robot slides into place over the rear docking plate of another robot they lock into place with the coupling mechanism.

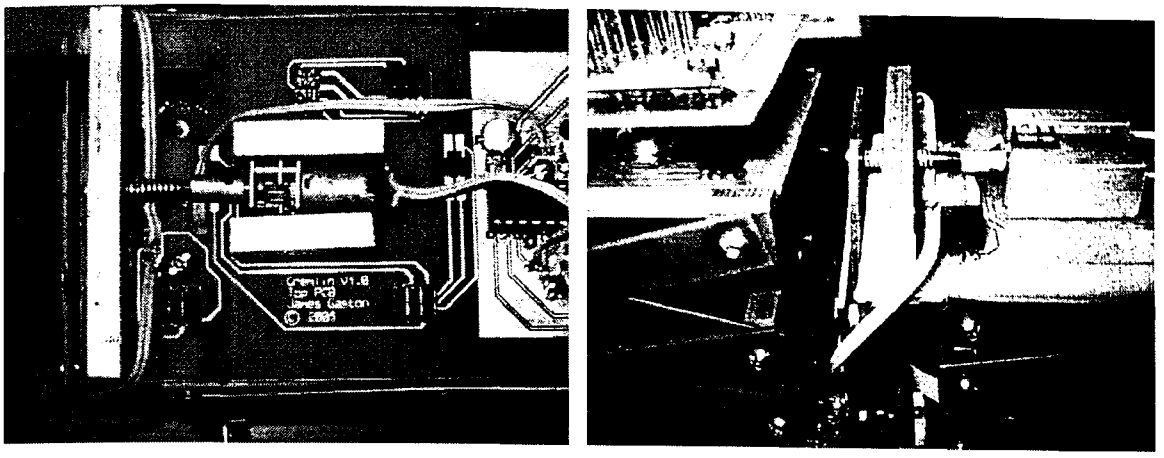


Figure 6.9: *(a) Top View of Coupling Mechanism (b) Actuated Coupling Mechanism*

2.3 The Coupling Mechanism

e coupling mechanism is designed to lock two robots together after they have docked. The chanism consists of a gear motor driving a threaded rod through an aluminum support ted to the chassis of the robot. The threaded rod is driven through a hole in the rear cking plate of the other robot by a gear motor to prevent the front and rear plates from arating.

3 Electrical Overview

e Gremlin electrical systems are divided into three main sections which mirrored the ysical construction of the robots: the base tier, the support tier, and the top tier. Each r represents a printed circuit board which doubles as the chassis of the robots. The base l support tiers make up the chassis of the robot which supports the lifting mechanism. e top tier houses the main command, control, communication, and optical systems of robot. As well as serving as part of the chassis and supporting the top of the lifting chanism, the support tier also houses motor control circuitry and sensors. The main drive tors are attached to the base tier which supported the lifting mechanism, battery pack, l houses control circuitry.

There are two methods for distributing power and control signals throughout the robot.

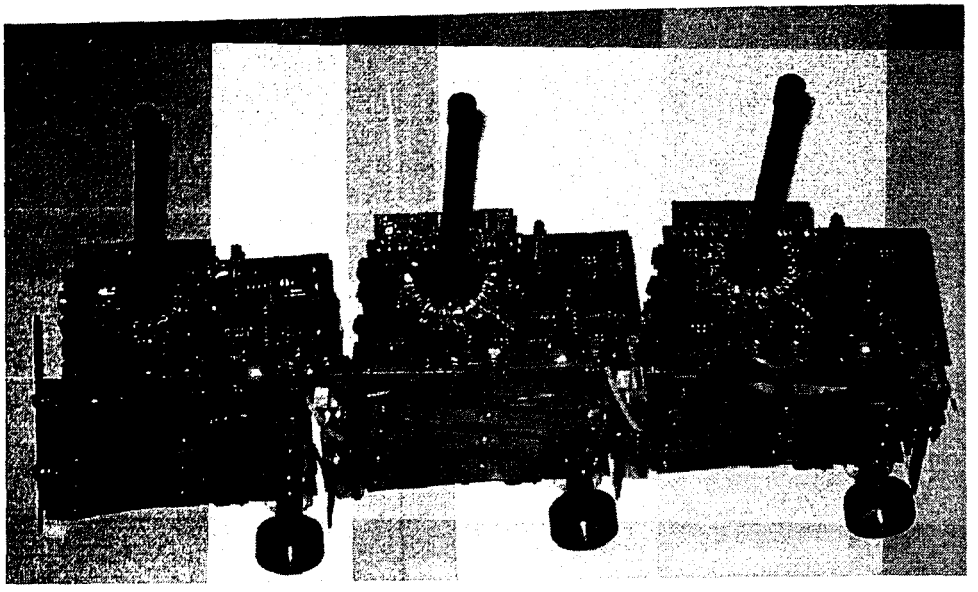


Figure 6.10: *Docked Gremlin Robots*

A long ribbon cable passes command and power signals between the base and support tiers. This ribbon cable allows the robots to extend and retract. A set of male connectors on the bottom of the top tier mate with matching female connectors on the support tier. These connectors firmly mount the top tier on top of the robot and pass power and control signals between the two tiers.

6.3.1 The Base Tier

The base tier of the Gremlin robots contains a rechargeable battery pack, the drive motors, and several sensor packages. The electrical design of the base tier is centered around an Atmel ATTiny2313 microcontroller [83]. This microcontroller monitors the status of the stair detection bumper switches (Figure 6.14), lifting mechanism opto interrupter limit switches, and wheel encoders. The microcontroller also maintains a serial peripheral interface (SPI) communication link with the microcontroller on the top tier. This communication link allows the ATTiny2313 to receive motor commands an send sensor information to the main microcontroller. The SPI protocol is chosen because it only requires three signal lines to implement two-way communication between the microcontroller on the base tier and the

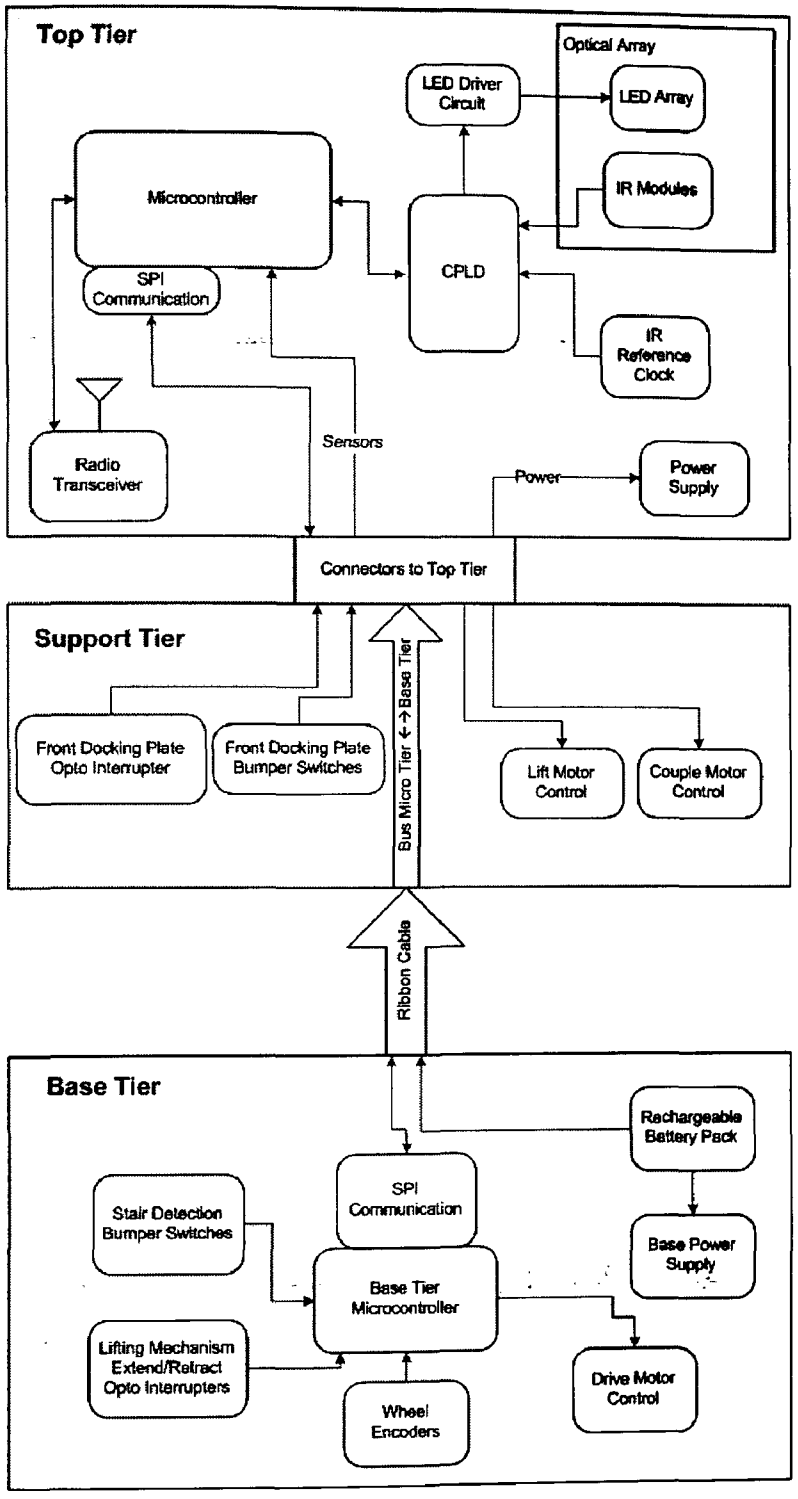


Figure 6.11: *Block Diagram of the Gremlin Electrical Systems*

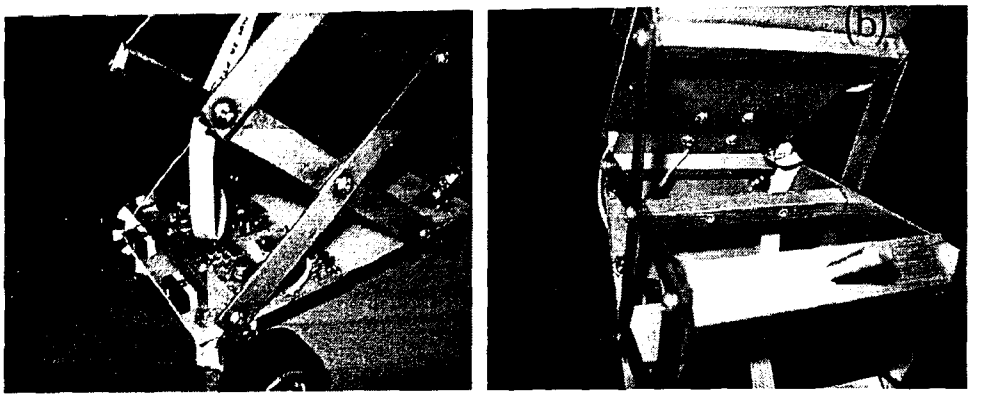


Figure 6.12: *Ribbon Cable Connector (a) Base Tier (b) Support Tier*

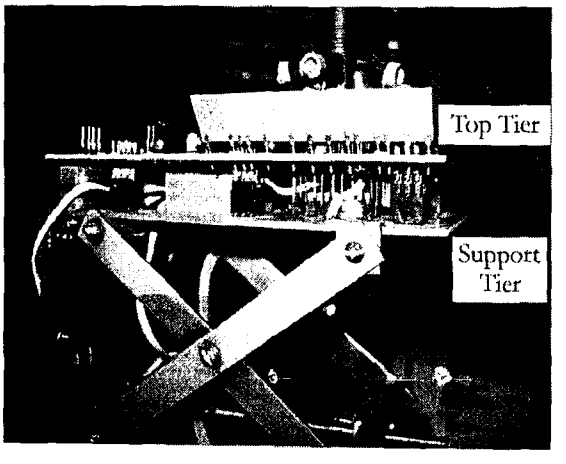
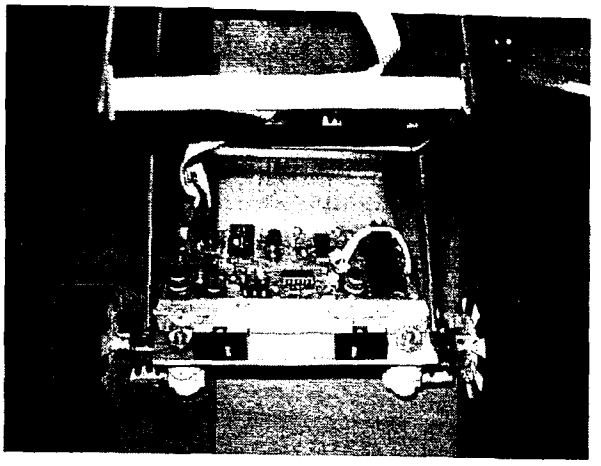


Figure 6.13: *Connectors Between the Top and Support Tiers*



ıre 6.14: *Gremlin Base Tier Printed Circuit Board Showing Stair Detection Bumper Switches*

Figure 6.15: *Wheel Encoders*

:ocontroller on the top tier.

The bumper switches on the front of the base tier allow the Gremlin robot to detect the of a stair when the lifting mechanism is extended. The opto interrupters on the base printed circuit board define the extended and retracted limits of the lifting mechanism. cial wheel encoders were designed (Figure 6.15) to act as odometers, and facilitate precise ements and turns.

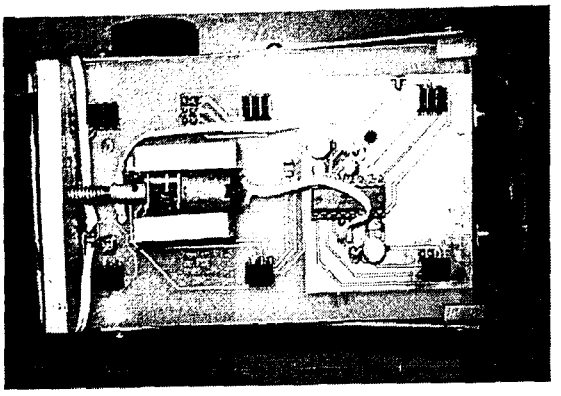


Figure 6.16: *Gremlin Support Tier Printed Circuit Board*

6.3.2 The Support Tier

This level of the robot chassis is bolted to the top of the lifting mechanism. The main electrical function of the top tier is to route control signals from the top tier to motor control circuitry on the support tier. The support tier also routes power and signals from the base tier to the top tier. Control circuitry for both the lifting mechanism motor and the coupling motor reside on the support tier. The support tier also housed the coupling mechanism.

6.3.3 The Top Tier

The top tier is regarded as the "brain" of the Gremlin robots. At the center of the top tier is an Atmel ATMEGA8515 microcontroller [83] and a Xilinx XC9532 complex programmable logic device (CPLD) [84]. The microcontroller is programmed with all of the command and behavior software developed in this project. It is interfaced to the Xilinx CPLD which acts as "glue" logic between the microcontroller and the optical array. The optical array, which acts as the visual localization system for the robots, is comprised of an array of superbright red light emitting diodes (LEDs), and four infrared receiver modules.

The LED array consists of 16 LEDs arranged in a circular fashion projecting outwards from the center of the robot. Electrically, the LEDs are divided into four groups, one for each cardinal direction facing outwards from the robot. Each direction, forward, rear, port, and

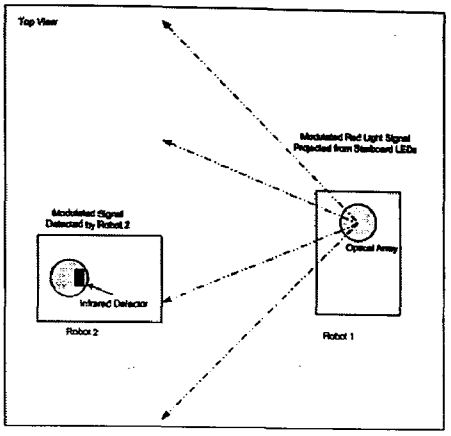


Figure 6.17: *Visual Localization Mechanism*

oard, can be turned on individually by the microcontroller. When enabled, the LEDs ect a modulated 38kHz signal in the red portion of the visible spectrum (approximately nm). This signal wavelength falls into the detection range of the infrared receiver modules h are designed to detect modulated infrared signals at 38kHz.

When an infrared receiver module detects the emissions from another nearby Gremlin t the CPLD stores a record of the detection and notifies the microcontroller. In this the microcontroller does not have to constantly monitor the outputs of the four in- d detection modules, but can query the CPLD regarding any recent detections. The ocontroller can also reset the detection history in the CPLD to begin monitoring for detections.

A single LED, called the *fine docking led*, positioned in the middle of the optical array pointed backwards serves as an alignment mechanism during docking. The narrow beam ted by this single LED ensures that robots approaching the rear docking plate will be to the center line of the stationary leader robot (Figure 6.19).

Wireless communication between the Gremlin robots is achieved using a radio transceiver ule from Abacom Technologies [79]. The transceiver is connected to the universal asyn- nous receiver transmitter (UART), also known as a serial port, on the microcontroller. lementation of the communication system is discussed in Section 5.4. Full electrical matics are presented in Appendix A.

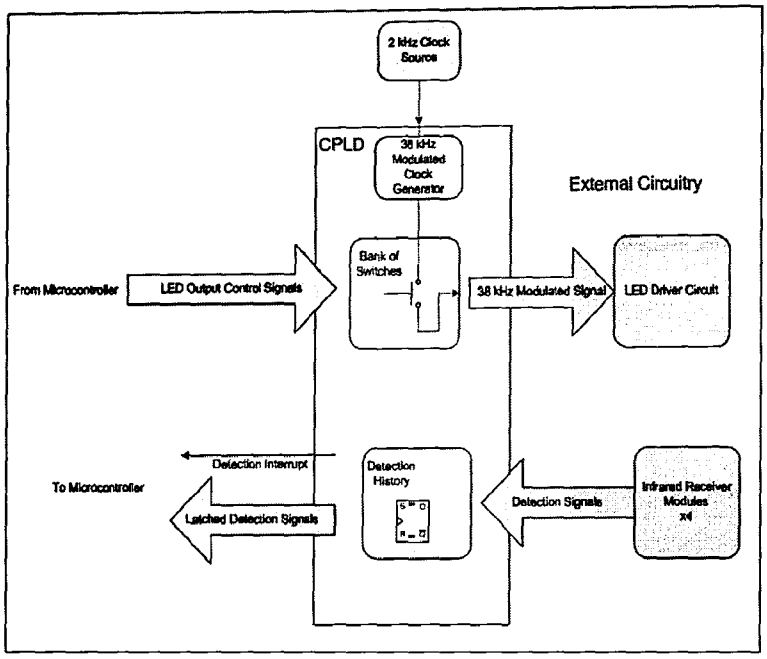


Figure 6.18: *Block Diagram of the Gremlin CPLD*

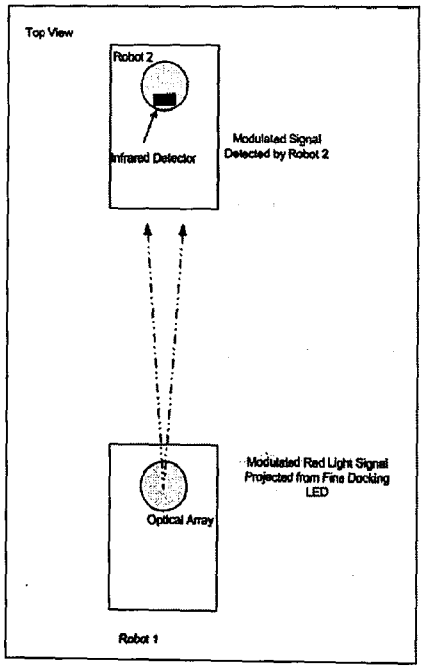


Figure 6.19: *Fine Docking Using a Single LED*

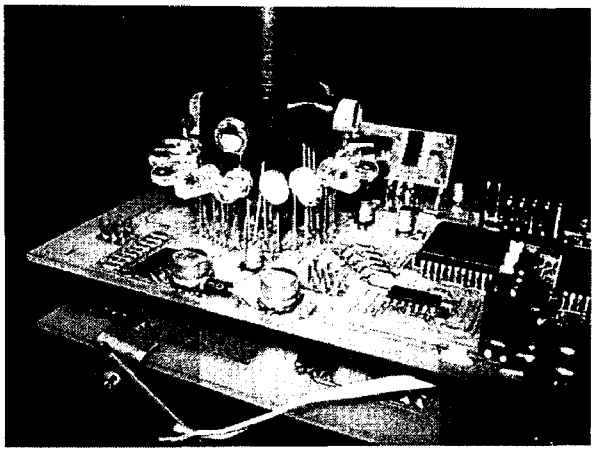


Figure 6.20: *Gremlin Optical Array*

Chapter 7

Experimental Verification

THIS chapter details the experimental verification of the Gremlin robot system. Experiments involving docking, reconfiguration, and stair climbing are described. The results from these experiments are presented and analyzed. The performance of the ALLIANCE behavioral control system for docking is examined by monitoring the parameters of the behavior model. Stair climbing performance is also investigated.

7.1 Experimental Setup

Two separate experiments were performed to examine the effectiveness of the Gremlin stair climbing robots. Docking experiments were conducted which were designed to investigate the effectiveness of the ALLIANCE control system for autonomous reconfiguration. Stair climbing experiments were also conducted to test the robots' ability to traverse a small staircase. A small three-stair staircase was fabricated for testing purposes. The *mission area* was 5'x5' area in front of the staircase.

7.1.1 Docking Experiments

Figure 7.1 shows the standard setup for a docking trial. The robots are powered up, one by one, and placed in the mission area defined by the white flooring. The white flooring is level, flat, and provided good traction for the robots. The first robot powered on becomes the leader according to the media access control strategy presented in Section 5.4. Once all

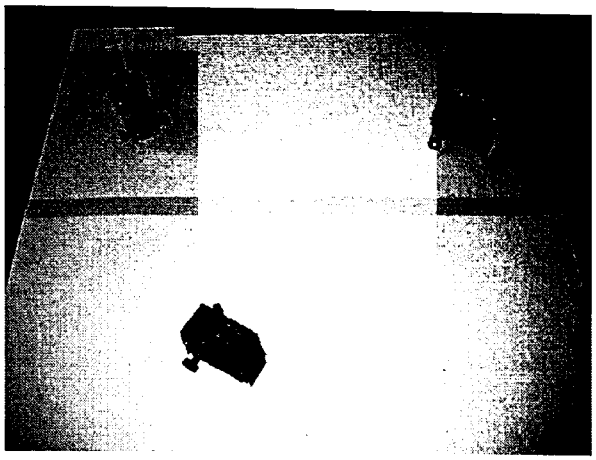


Figure 7.1: *Arrangement of Robots for a Docking Trial*

he robots are powered up, the leader is positioned in front of the staircase by remote trol. A special command is then issued to all of the robots to signal the start of the trial. m this point forward, the robots operate autonomously. A computer is used to monitor robot communications and monitor their behavior control systems.

.2 Climbing Experiments

ure 7.2 shows an isometric view of the experimental setup used to test the stair climbing ity of the Gremlin robots. In order to test the climbing ability of the Gremlin robots arately from their reconfigurations abilities, the robots the experimental stair climbing ls are initiated with the robots already in their chain configuration. The robots are ed 10 cm from the facing of the first stair and the climbing algorithm was activated. stairs are 20cm high and 20cm deep. A climbing trial is deemed successful if the robot in climbed all three stairs autonomously.

2 ALLIANCE-Based Docking Performance Results

ure 7.3 shows a photographic progress of the results of one of the successful docking trials. ure 7.4 shows the action selection for each robot during the experiment. Robot 1 was the robot to be powered up. This robot sat idle for several seconds while monitoring the

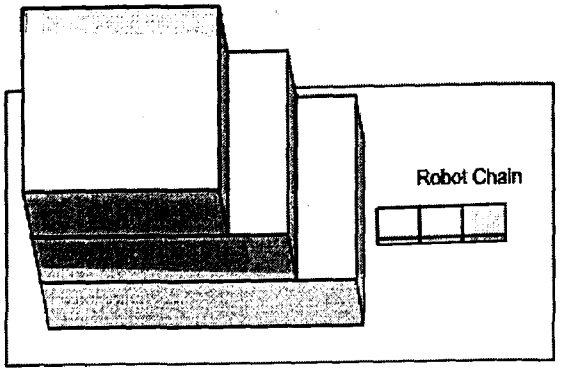


Figure 7.2: *Experimental Setup for Stair Climbing Exercises*

communications channel. During this idle time, the motivation level for the leader behavior set began to rise. Eventually, this motivation level crossed the threshold of activation and the leader behavior set was active. As the leader, Robot 1 took on the responsibility of managing the TDMA media access protocol and began broadcasting frames at approximately one frame per second. When each of the two remaining robots were turned on, they responded to the leader in the "join" slot of the communication frame(Figure 5.18 and were added to the polling routine of each frame.

With all of the robots now functioning, the leader enabled its LED array and the motivational levels for the find-leader behavior sets of Robots 2 and 3 began to rise. Robot 2 was programmed with the fastest motivational rate and was the first to select the find-leader-methodical behavior set. After selecting this behavior set, Robot 2 began inhibiting the activation of docking-related behavior sets in Robot 3 by posting its current status on its communication signboard. In practical implementation, sign-board data was transmitted at the beginning of each data slot so a robot with an active docking behavior set would update the other robots on its progress once per second.

Robot 2 progressed through the behavior sets and successfully docked with the leader, Robot 1. Once Robot 2 had docked, it stopped inhibiting the docking behavior selection of Robot 3. At this point the level of motivation for the find-leader behavior sets began to rise. The find-leader-wander behavior set was chosen because the impatience rate for this behavior was set to a higher rate. Robot 3 detected the optical signal from Robot 2

ich had now become the leader. Following the same progression was Robot 2, Robot 3 cessfully docked completing the chain. At this point, the robots were now ready to climb stairs.

Figure 7.4 shows the robot action selections during the successful docking trial that was described. Figure 7.5 shows the motivational levels for each of the behavior sets during same trial. The dotted lines in these graphs represent the threshold of activation for each avior set. When a motivational level reaches this threshold, the behavior set is activated. One of the main reasons for using the ALLIANCE architecture to implement the docking avior was its inherent fault tolerance. Figure 7.6 shows a photographic progression of ocking trial where one of the robots was unsuccessful at its first attempt to dock. As the trial described earlier in this section, Robot 2 was the first to detect the leader l attempt to dock. This robot followed the same behavior progression as in the successful king attempt progressing through the find-leader-methodical, determine-relative-position, igate-to-rear, docking-approach, and docking-align behavior sets. In this trial, however, robot had difficulty aligning itself for docking.

Robot 2 made several attempts to align itself with the leader while the docking-align avior set was active. As Robot 2 was attempting to align itself the impatience level in ot 3 started to rise. The motivation level for Robot 3 at this point was given by the ation:

$$
\begin{aligned}
m_{ij}(t) &= [m_{ij}(t-1) + impatience_{ij}(t)] \\
&\quad \times activity\_suppression_{ij}(t) \\
&\quad \times sensory\_feedback_{ij}(t) \\
&\quad \times acquiescence_{ij}(t) \\
&\quad \times impatience\_reset_{ij}(t) \\
&= [m(ij)(t-1) + impatience_{ij}(t)] \\
&= [m(ij)(t-1) + \delta\_slow_{ij}(k,t)]
\end{aligned}
$$

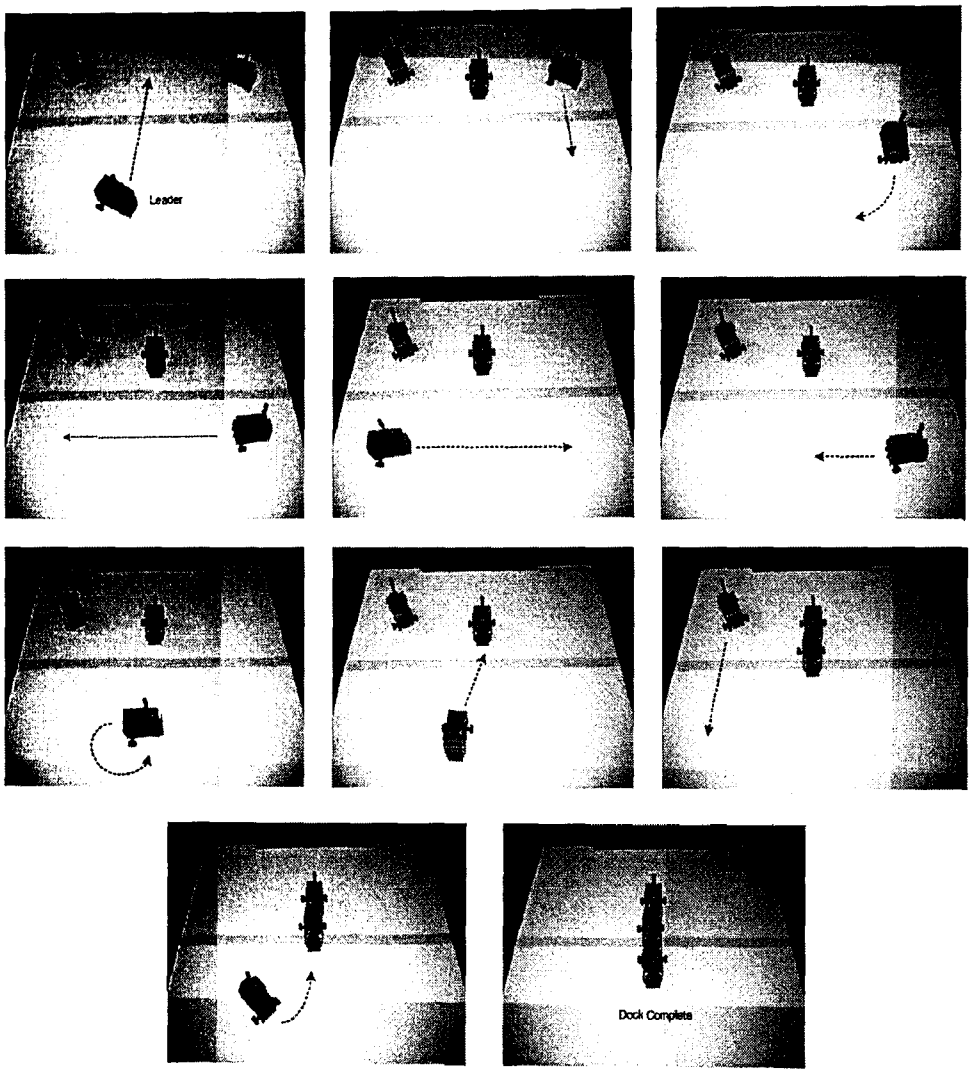


Figure 7.3: *Gremlin Docking Experiment*

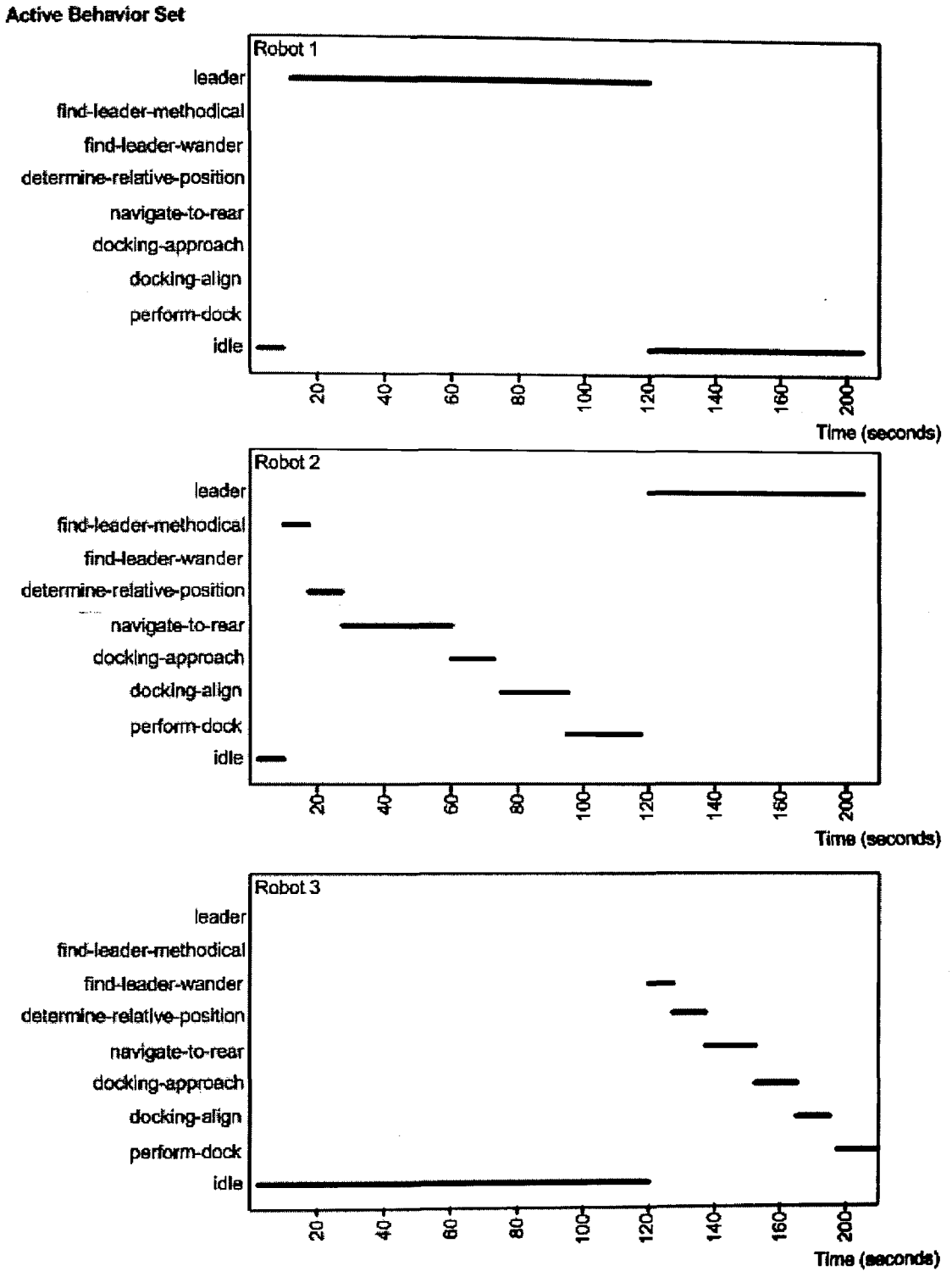


Figure 7.4: *Robot Action Selections During a Successful Docking Run*

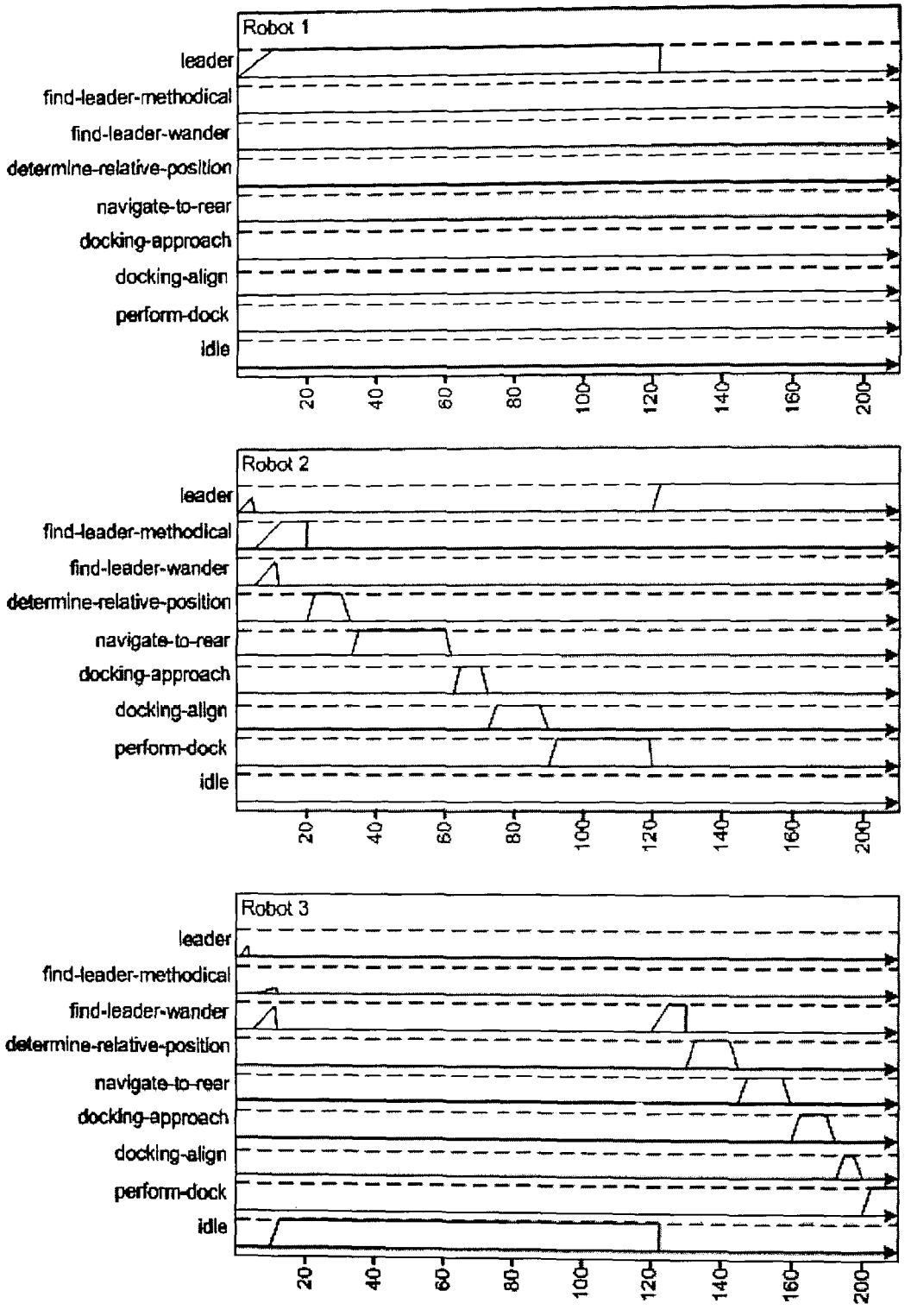


Figure 7.5: *Motivational Levels for Behavior Sets During Successful Trial*

In other words, the motivational level for Robot 3 to begin a docking attempt, (and take r for Robot 2 who was not completing the task) was increasing at a rate of $\delta\_slow_{ij}(k,t)$. ;e Robot 2 was unable to complete the docking maneuver before the motivational level lobot 3 reached the threshold of activation, Robot 3 had *become impatient* and took over task of docking. Also, since the docking-align behavior set in Robot 2 had been active more than $\psi_{ij}(t)$ time units, Robot 2 was ready to acquiesce the task when Robot 3 ounced it was going to make a docking attempt. Robot 2 gracefully backed away from leader robot to make room for Robot 3.

Robot 3 was able to perform the docking maneuver with the leader and then took over leader function itself. Once again, Robot 2 attempted to dock, this time with the new ler, Robot 3. Robot 2 was able to successfully dock with Robot 3 and complete the chain. s docking experiment highlights the utility of the ALLIANCE behavioral architecture in onomous mobile robot reconfiguration. Despite the fact that Robot 2 was unable to ;essfully dock with the leader on the first attempt, the inherent fault-tolerance of this trol architecture allowed Robot 3 to make an attempt and bring the system out of a ;ntial deadlock situation.

Figures 7.7 and 7.8 highlight the mechanism that define the fault tolerant features of ALLIANCE architecture. Figure 7.7 shows the action selection progression during the l. At approximately 110 seconds into the trial while Robot 2 was still attempting to n itself with the leader Robot 3 had become impatience and took over. This is shown on graph as a change in the active behavior set of Robot 2 from docking-approach to idle the change in Robot 3 from idle to find-leader-wander. Figure 7.8 shows the reasons for change in behaviors. From approximately 90 to 110 seconds, the motivational level to ate a docking behavior in Robot 3 was increasing at $\delta\_slow_{ij}(t)$. When Robot 2 failed lign itself and provide "reassurance" via the communication system to Robot 3 of its gress through the docking process, Robot 3 became impatient and "decided" to take over docking procedure itself as the motivation level passed the threshold of activation.

Figure 7.9 shows the average docking times for 30 docking trials. The graph displays the

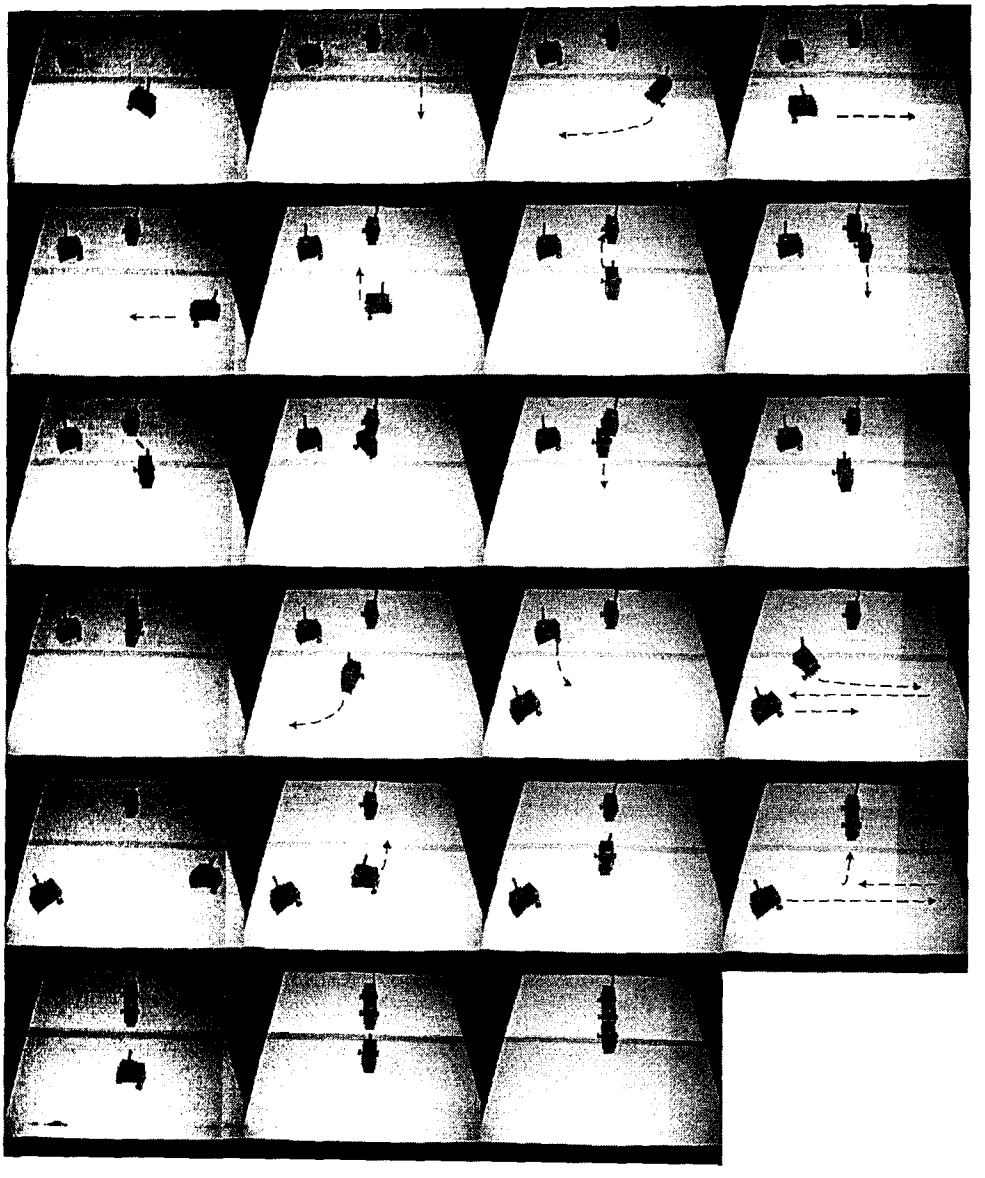


Figure 7.6: *Gremlin Docking Experiment with Docking Failure*

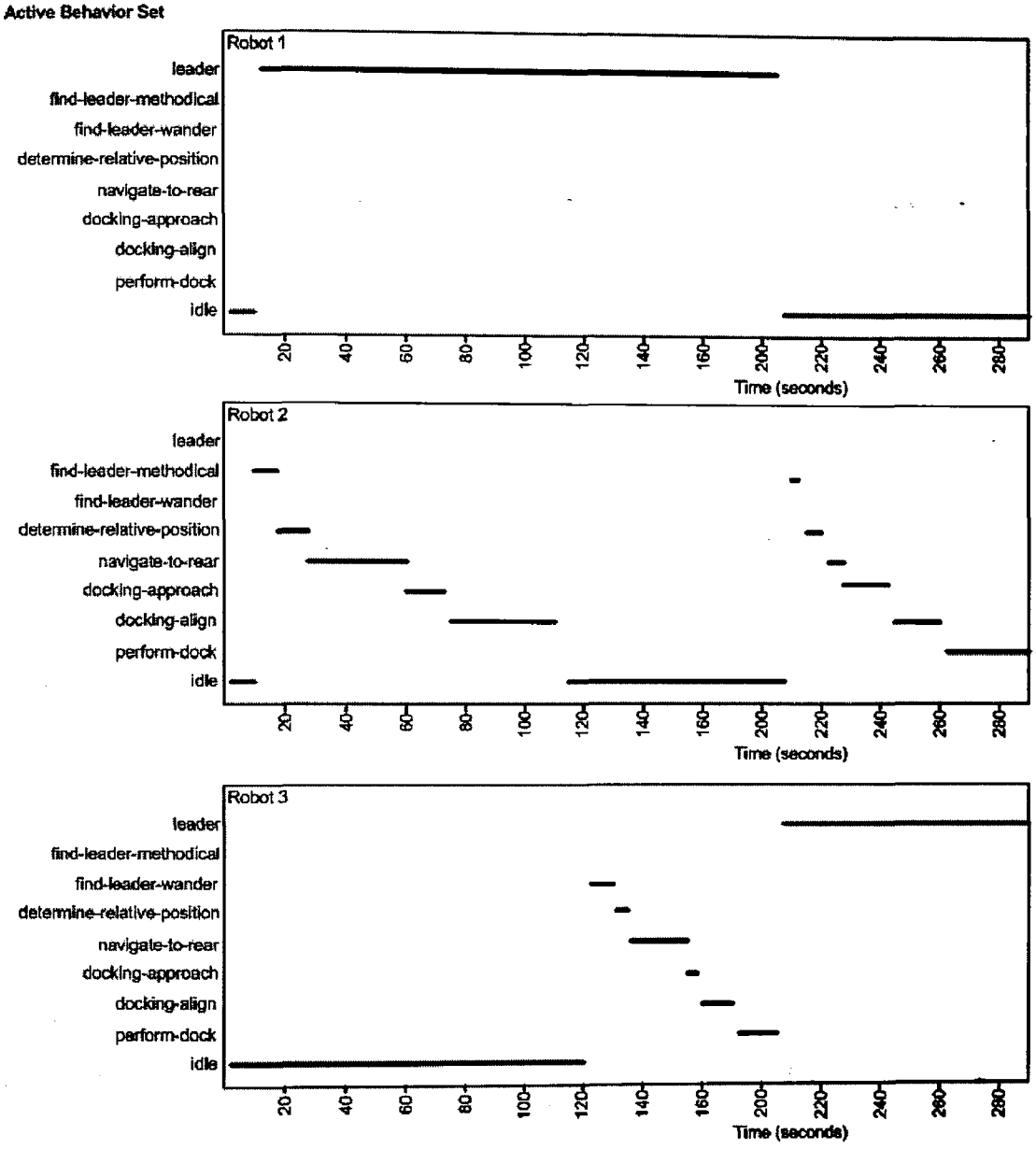


Figure 7.7: *Robot Action Selection During an Experiment with a Failed Dock*

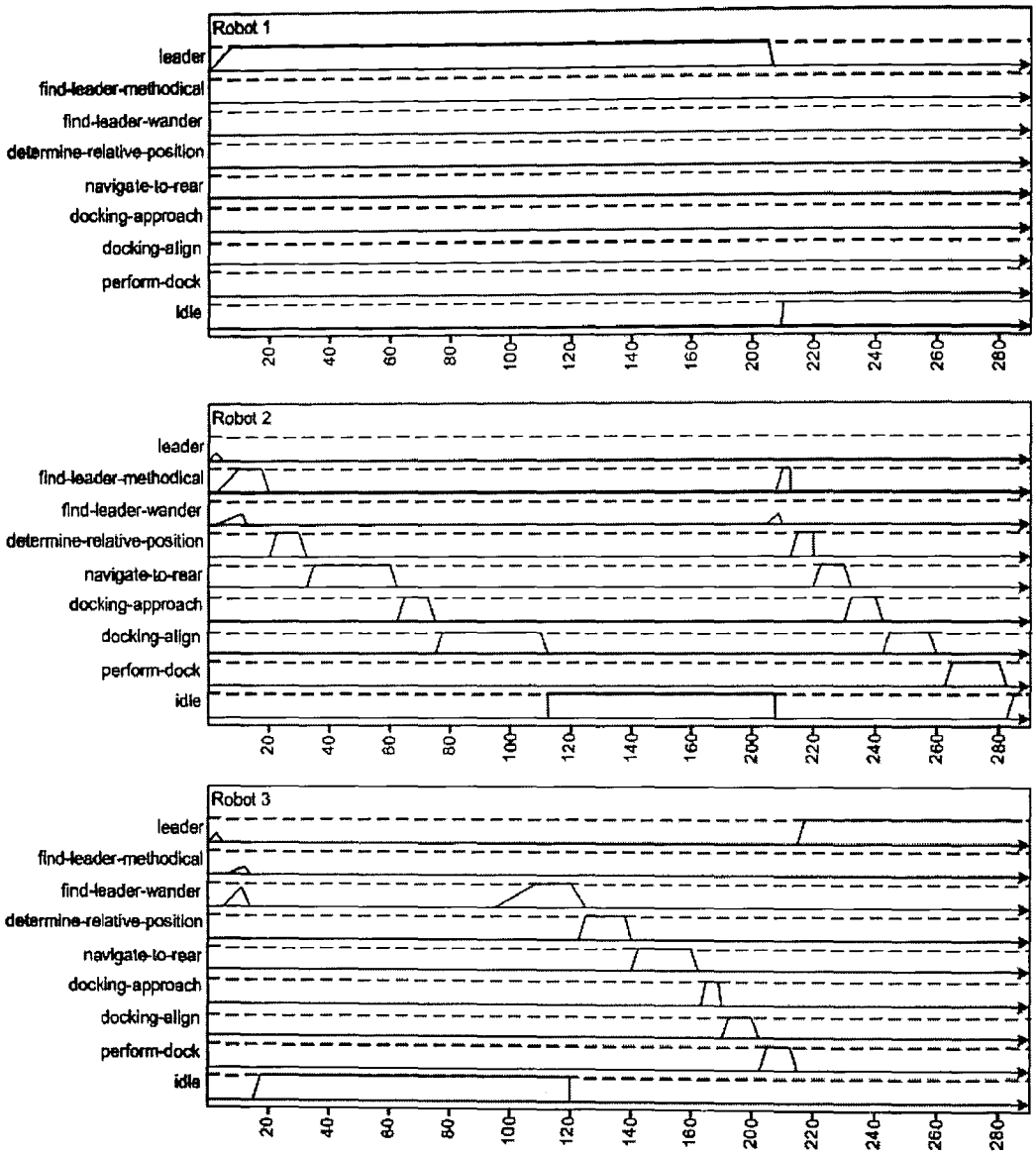


Figure 7.8: *Motivational Levels for Behavior Sets During A Trial with a Docking Failure*

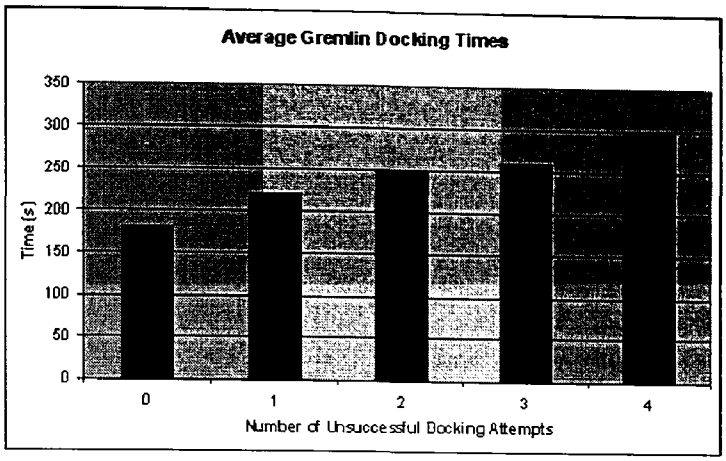


Figure 7.9: *Average Gremlin Docking Times*

rage docking time associated with the number of unsuccessful docking attempts during run. It is interesting to note that there is only about a 30 second increase in the time uired to dock for each unsuccessful attempt. This is due to the fact that a robot who is successful at docking remains behind the leader and is a good position to make another empt once the teammate who took over the task is finished.

3 Stair Climbing Performance Results

er 30 stair climbing trials were conducted. These trials showed that the Gremlin robots ld successfully and autonomously climb a set of stairs. Figure 7.10 shows a photographic gression of a successful stair climbing maneuver. The motions of the Gremlin chain tched the movements prescribed by the gait control presented in Section 5.3.4.

Figure 7.11 shows the average completion times for the successful climbing trials. The rage amount of time to climb three stairs was 4 minutes and 31 seconds, which translates o approximately 90 seconds per stair. Appendix C presents the results from all of the mbing trials. It was determined that the activity of stair climbing requires large amounts current because the motors driving the lifting mechanism frequently stall. These stall ditions were most frequent when one robot was extending while another was retracting ce the speed of retraction and extension were different in each robot.

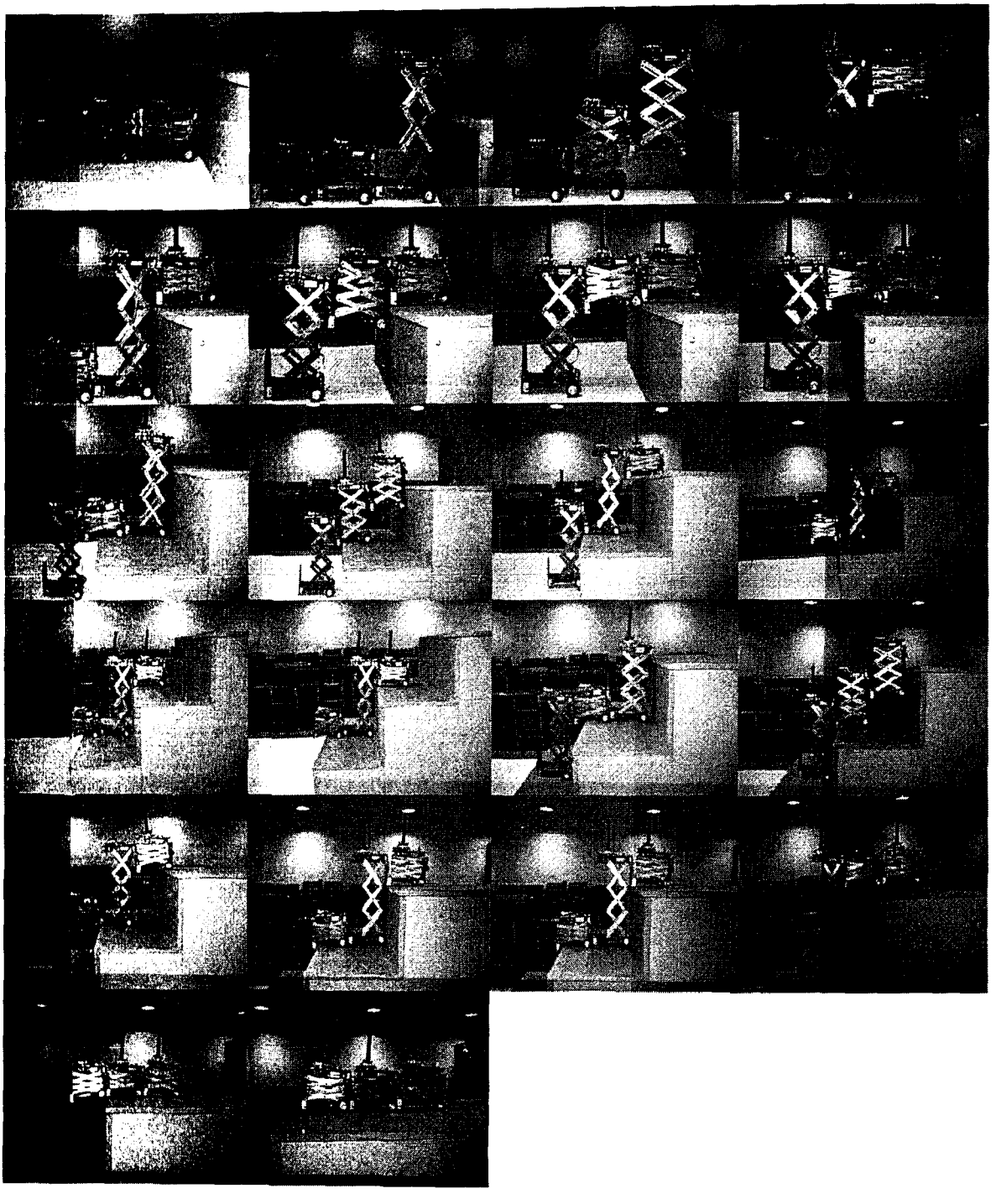


Figure 7.10: *Gremlin Stair Climbing Experimental Verification*

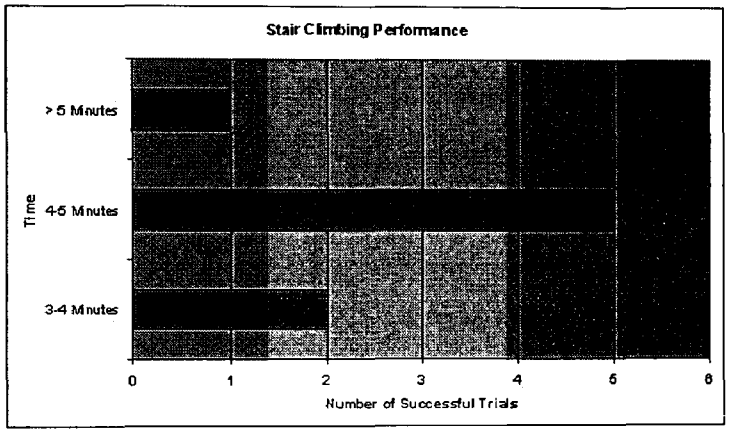


Figure 7.11: *Gremlin Stair Climbing Times*

Chapter 8

Conclusions and Future Work

WITH the proliferation of robots in our world, their ability mobility defines their usefulness. This project is a starting point for developing robot teams that have free reign over indoor environments. The opportunity exists to develop the notion of cooperative stair climbing robots. This development could lead to teams of vacuum cleaner robots that move from one floor to another in a dwelling. A team of mobile robots with the ability to move between floors of a building could also be used to augment an existing security system or provide reconnaissance for law enforcement. This chapter summarizes the contributions of this project, draws conclusions regarding the success of the implementation, and provides suggestions for future work on this topic.

8.1 Contributions

We propose a complete integrated control architecture and communication strategy for a system of reconfigurable robots that perform tasks with ordering dependencies. This architecture enables fully autonomous operation of a team of robots performing tasks with ordering dependencies. The communication strategy to autonomously maintain a structured message passing network without the need for external supervision. To our knowledge, this is the first implementation of such an architecture that facilitates cooperative autonomous docking and reconfiguration of robots.

This project has demonstrated the feasibility of developing a fully autonomous team of

onfigurable stair climbing robots. The following reiterates the significance of this demonstration and outlines the contributions of this project:

- A behavioral control architecture and communication system, integrated to facilitate autonomous mechanical reconfiguration of robots.
- Extension of a behavior-based control architecture called ALLIANCE which improve the efficiency of cooperative tasks with ordering dependencies.
- Behavior classes are introduced in the extended ALLIANCE architecture which group behavior sets according to a high level task.
- The behavior class is shown to be an effective mechanism for acknowledging the progress of a robot team member towards a task defined by several behavior sets with ordering dependencies.
- The behavior class is used to ensure that behavior sets with ordering dependencies are executed in the correct sequence.
- A new type of gait control table for motion planning is proposed which includes sensor feedback and implements a mechanical travelling wave on a set of linked stair climbing robots.
- A high level sign-board based communication scheme is integrated into the architecture which complements the behavioral system .
- A time domain multiple access (TDMA) communication scheme that allows the robot team to maintain a structured message passing system without the supervision of a base station or computer.
- The TDMA communication scheme facilitates dynamic frame size allocation and inherently detects robot failures.

- A simple mechanical design augmented by a suitable control system. This research represents the first successful implementation of cooperative autonomous stair climbing robots that we are aware of.

8.2 Conclusions

The proposed Gremlin Design has expanded on previous work which introduced the notion of cooperative robot stair climbing by presenting a system of autonomous stair climbing robots. In contrast to previous stair climbing implementations the proposed design shifts the focus from the mechanical design of the robots to the control and communication organization.

The behavioral control system of the Gremlin robots, which was based on the ALLIANCE architecture, is shown to be an effective and resilient scheme for autonomous reconfiguration. Experiments show that the fault-tolerant nature of the extended ALLIANCE architecture is well suited to the complex task of robot docking. In cases where a robot was having difficulty completing a docking maneuver, another robot would become impatient and take over the docking task. This feature of the behavior architecture is effective in preventing a single robot from jeopardizing the overall success of the docking task.

The gait control table paradigm is an effective method for controlling the Gremlin robots while stair climbing in their coupled state. Several successful trials were executed to demonstrate that this control scheme could autonomously guide coupled modular robots up a flight of stairs. The proposed design shifted focus from the mechanical design of the robots to the control and communication organization. This led to some minor drawbacks in the reliability of the mechanical system and its power consumption.

8.3 Future Work

This project uses the minimum number of robots (3) required to demonstrate stair climbing. Future research could investigate the prospect of scaling up the number of robots in the system. This research could investigate the effect of more robots on the efficiency and fault

rance of the system. With more robots, the effect of a total robot failure could also be stigated.

The Gremlin robots fabricated for this design are limited in their ability to detect the rcase. Future research could involve integrating image processing to enable the robots utonomously detect the staircase prior to reconfiguration. This ability could be coupled ı the selection of the leader robot such that the first robot to detect the staircase would ome the leader.

From a mechanical engineering perspective, the mechanics of the Gremlin robots could improved. This project has shown that the current mechanical design, even with its rtcomings, is capable of stair climbing. More precise construction techniques such as puter-aided manufacturing, and a complete analytical mechanical design could be pur- l that would undoubtedly improve the reliability and efficiency of Gremlin stair climbing.

Bibliography

[1] L. Graham, "Stair Climbing Co-Operative Mini-Robots A Novel Coupling Method, Control and Electronics," http://www.rdg.ac.uk/scarp/library/2005/papers/siu01llc.doc, 2004.

[2] J. Fowler, "U.N. Robot Use to Surge Sevenfold by 2007," http://biz.yahoo.com/ap/041024/, 2004.

[3] M. Gasperi, "Machina Speculatrix," http://www.plazaearth.com/usr/gasperi/walter.htm, 2005.

[4] H. Moravec, "Grey Walter Tortoise, Elsie," http://www.frc.ri.cmu.edu/~hpm/talks/revo.slides/1950.html, April 2005.

[5] H. Moravec, *ROBOT: Mere Machine to Tracendent Mind.* Oxford University Press, Inc, October 1998.

[6] H. Moravec, "The Hopkins Beast," http://www.frc.ri.cmu.edu/~hpm/talks/revo.slides/1960.html, 2005.

[7] R. C. Arkin, *Behavior-Based Robotics.* The Massachussettts Institute of Technology, 1998.

[8] S. Technology, "Shakey the Robot," http://www.sri.com/about/timeline/shakey.html, 2005.

[9] H. Moravec, "The Stanford Cart and CMU Rover," http://www.frc.ri.cmu.edu/~hpm/book97/ch2/, 2005.

S. Corporation, "AIBO Global Link," http://www.sony.net/Products/aibo/, July 2005.

H. Corporation, "Honda Worldwide: Asimo," http://world.honda.com/ASIMO/, July 2005.

H. Corporation, "The Honda Asimo Robot," http://world.honda.com/ASIMO/, 2005.

S. Corporation, "Sony AIBO," http://www.sony.net/Products/aibo/, 2005.

Y. U. Cao, A. Fukunaga, A. Kahng, and F. Meng, "Cooperative Mobile Robotics: Antecedents and Directions," *IEEE/RSJ International Conference on Human Robot Interaction and Cooperative Robots*, Vol. 1, pp. 5–9, 1995.

T. Arai, E. Pagello, and L. E. Parker, "Editorial: Advances in Multi-Robot Systems," *IEEE Transactions on Robotics and Automation*, Vol. 18, No. 5, pp. 655–661, October 2002.

G. Beni, "The Concept of Cellular Robotic System," *IEEE International Symposium on Intelligent Control*, pp. 57–62, 1998.

A. Yamashita, "Motion Planning for Cooperative Transportation of a Large Object by Multiple Mobile Robots in a 3D Environment," *Proceedings of IEEE International Conference on Robotics and Automation*, Vol. 4, pp. 3144–3151, 2000.

T. Suzuki, H. Ogata, and T. Arai, "Collision Avoidance Among Multiple Robots Using Virtual Impedance," *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 479–485, 1989.

P. Wang, "Navigation Strategies for Multiple Autonomous Mobile Robots," *Proceedings of the IEEE/RSJ International Conference on Intelligent Robotics and Automation*, pp. 486–492, 1989.

L. E. Parker, "Current Research in Multi-Robot Systems," *Journal of Artificial Life and Robotics*, Vol. 1, pp. 1–5, 2004.

[21] D. M. Helmick, S. I. Roumeliotis, M. C. McHenry, and L. Matthies, "Multi-Sensor, High Speed Autonomous Stair Climbing," *Proceedings of the 2002 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vol. 1, pp. 733–742, October 2002.

[22] J.-S. Gutmann, M. Fukuchi, and M. Fujita, "Stair Climbing for Humanoid Robots Using Stereo Vision," *Proceedings of 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1407–1413, October 2004.

[23] J. P. Laboratory, "URBIE Urban Robot," http://robotics.jpl.nasa.gov/tasks/tmr/homepage.html, 2005.

[24] "Defence Advanced Research Projects Agency, U.S. Department of Defence," http://www.darpa.mil/, 2005.

[25] S. Steplight, G. Egnal, S.-H. Jung, and D. B. Walker, "A Mode-Based Sensor Fusion Approach to Robotic Stair-Climbing," *Proceedings fo the 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vol. 2, pp. 1113–1118, 2000.

[26] Y. Xiong and L. Matthies, "Vision-Guided Autonomous Stair Climbing," *Proceedings of the 2000 IEEE International Conference on Robotics and Automation*, Vol. 2, pp. 1842–1847, April 2000.

[27] Y. Uchida, K. Furuichi, and S. Hirose, "Fundamental Performance of 6 Wheeled Off-Road Vehicle "HELIOS-V"," *Proceedings of the 1999 IEEE International Conference on Robotics and Automation*, pp. 2336–2341, May 1999.

[28] H. . Y. Lab, "HELIOS Off-Road Robot," http://www-robot.mes.titech.ac.jp/robot/wheeled/helios5/helios5_e.html, 2005.

[29] M. Lauria, Y. Piguet, and R. Seigwart, "Octopus: An Autonomous Wheeled Climbing Robot," *Proceedings of the Fifth International Conference on Climbing and Walking Robots*, 2002.

A. A. S. Lab, "Octopus," http://asl.epfl.ch/index.html?content=research/systems/Octopus/octopus.php, 2005.

Y. Takita, N. Shimoi, and H. Date, "Development of a Wheeled Mobile Robot "Octal Wheel" Realized Climbing Up and Down Stairs," *Proceedings of 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vol. 1, pp. 2440–2445, September 2004.

O. Matsumoto, S. Kajita, M. Saigo, and K. Tani, "Dynamic Trajectory Control of Passing Over Stairs By a Biped Type Leg-Wheeled Robot with Nominal Reference of Static Gait," *Proceedings of the 1998 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vol. 1, pp. 406–412, October 1998.

C.-L. Shih, "Ascending and Descending Stairs for a Biped Robot," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 29, No. 3, pp. 255–268, May 1999.

G. Figliolini and M. Ceccarelli, "Climbing Stairs with EP-WAR2 Biped Robot," *Proceedings of the 2001 IEEE International Conference on Robotics and Automation*, Vol. 4, pp. 4116–4121, May 2001.

G. Figliolini, M. Ceccarelli, and M. Gioia, "Descending Stairs with EP-WAR3 Biped Robot," *Proceedings of the 2003 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, Vol. 2, pp. 747–752, 2003.

A. Albert, M. Suppa, and W. Gerth, "Detection of Stair Dimensions for the Path Planning of a Bipedal Robot," *Proceedings of the 2001 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, Vol. 2, pp. 1291–1296, July 2001.

M. Buehler, R. Battaglia, A. Cocosco, G. Hawker, J. Sarkis, and K. Yamazaki, "SCOUT: A Simple Quadruped that Walks, Climbs, and Runs," *Proceedings of the 1998 IEEE International Conference on Robotics and Automation*, Vol. 2, pp. 1707–1712, May 1998.

[38] S. A. Stoeter, P. E. Rybski, M. Gini, and N. Papanikolopoulos, "Autnomous Stair-Hopping with Scout Robots," *Proceedings of the 2002 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vol. 1, pp. 721–726, October 2002.

[39] Y. Takahashi, H. Nakayama, and T. Nagasawa, "Biped Robot to Assist Walking and Moving Up-and-Down Stairs," *Proceedings of the 24th Annual Conference of the IEEE Industrial Electronics Society*, Vol. 2, pp. 1140–1145, August 1998.

[40] G. Wiesspeiner and E. Windischbacher, "Distributed Intelligence to Control A Stair-Climbing Wheelchair," *Engineering in Medicine and Biology Society IEEE 17th Annual Conference*, Vol. 2, pp. 1173–1174, September 1995.

[41] H. Brown, J. M. V. Weghe, C. A. Bereton, and P. K. Khosla, "Millibot Trains for Enhanced Mobility," *IEEE/ASME Transactions on Mechatronics*, Vol. 7, No. 4, pp. 452–461, December 2002.

[42] D. Rus, "Self-Reconfiguring Robots," *Intelligent Systems and Their Applications*, Vol. 13, No. 4, pp. 2–4, July 1998.

[43] M. Yim, S. Homans, and K. Roufas, "Climbing With Snake-Like Robots," *IFAC Workshop on Mobile Robot Technology*, May 2001.

[44] P. A. R. Center, "Polybot: A Chain of Reconfiguration Robot," http://www2.parc.com/spl/projects/modrobots/chain/polybot/, June 2004.

[45] M. Yim, D. G. Duff, and K. D. Roufas, "Walk on the Wild Side," *IEEE Robotics and Automation Magazine*, Vol. 9, No. 4, pp. 49–53, December 2002.

[46] K. D. Kotay and D. L. Rus, "Task-Reconfigurable Robots Navigators and Manipulators," *Proceedings of the 1997 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vol. 2, pp. 1081–1089, September 1997.

G. Granosik, M. G. Hansen, and J. Borenstein, "The OmniTread Serpentine Robot for Insdustrial Inspection and Surveillance," *International Journal on Industrial Robots*, Vol. IR32-2, pp. 139–148, March 2005.

L. E. Navarro-Serment, R. Grabowski, C. J. Paradis, and P. K. Khosla, "Millibots: The Development of a Framework and Algorithms for a Distributed Heterogeneous Robot Team," *IEEE Robotics and Automation Magazine*, pp. 31–34, December 2002.

R. Grabowski, L. E. Navarro-Serment, and P. K. Khosla, "An Army of Small Robots," *Scientific American*, pp. 63–67, November 2003.

R. Grabowski, L. Navarro-Serment, C. Paredis, and P. Khosla, "Heterogeneous Teams of Modular Robots for Mapping and Exploration," *Autonomous Robot*, Vol. 8, No. 3, pp. 293–308, 2000.

L. Navarro-Serment, R. Grabowski, C. Paredis, and P. Khosla, "Modularity in Small Distributed Robots," *Proceedings SPIE Conference on Sensor Fusion and Decentralized Control in Robotic Systems II*, Vol. 3839, pp. 297–306, September 1999.

M. J. Mataric, "Behavior Based Robotics," *MIT Encyclopedia of Cognitive Sciences*, pp. 74–77, April 1999.

I. N. Engineering and E. Laboratory, "Adaptive Robotics - Behavior-Based Robotics," http://www.inel.gov/adaptiverobotics/behaviorbasedrobotics/, 2004.

N. Aeronautics and S. Administration, "NASA's Mars Exploration Program," http://mars.jpl.nasa.gov/MPF/ames/ames-rovers.html, 2005.

M. J. Mataric, "Behavior-Based Control: Main Properties and Implications," *Proceedings, IEEE International Conference on Robotics and Automation, Workshop on Architectures for Intelligent Control Systems*, pp. 46–54, May 1992.

H. Moravec and D.Cho, "A Bayesian Method for Certainty Grids," *Proceedings, AAAI Spring Symposium on Robot Navigation*, pp. 57–60, March 1989.

[57] G. Giralt, R. Chatila, and M. Vaisset, "An Integrated Navigation and Motion Control System for Autonomous Multisensory Mobile Robots," *First International Symposium on Robotics Research*, pp. 191–214, 1984.

[58] R. Chatila and J. Laumond, "Position Referencing and Consistent World Modeling for Mobile Robots," *Proceedings of the 1985 IEEE International Conference on Robotics and Automation*, pp. 138–145, March 1985.

[59] J. H. Connell, "A Colony Architecture for an Artificial Creature," *MIT A.I. Lab Technical Report*, Vol. 1151, June 1990.

[60] R. C. Arkin, "Integrating Behavioral, Perceptual and World Knowledge in Reactive Navigation," *Designing Autonomous Agents*, pp. 105–122, 1990.

[61] R. A. Brooks, "A Robust Layered Control System for a Mobile Robot," *IEEE Journal of Robotics and Automation*, Vol. RA-2, No. 1, pp. 14–23, March 1986.

[62] J. H. Connell, "SSS: A Hybrid Architecture Applied to Robot Navigation," *IBM Reserch Report*, 1991.

[63] F. R. Noreils, "Toward a Robot Architecture Integrating Cooperation Between Mobile Robots," *The International Journal of Robotics Research*, Vol. 12, No. 1, pp. 79–98, February 1993.

[64] H. Asama, K. Ozaki, A. Matsumoto, and Y. Ishida, "Development of Task Assignment System Using Communication for Multiple Autonomous Robots," *Journal of Robotics and Mechatronics*, Vol. 4, No. 2, pp. 122–127, 1992.

[65] L. E. Parker, "ALLIANCE: An Architecture for Fault Tolerant, Cooperative Control of Heterogeneous Mobile Robots," *IEEE Transactions on Robotics and Automation*, Vol. 14, pp. 220–240, April 1998.

L. Parker, "Evaluating Success in Autonomous Multi-Robot Teams: Experiences from ALLIANCE Architecture Implementations," *Journal of Experimental and Theoretical Artificial Intelligence*, Vol. 13, pp. 95–98, 2001.

R. Kuc and B. Barshan, "Docking Mobile Robots Using a Bat-Like Sonar," *Proceedings fo the 1992 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vol. 2, pp. 1439–1444, July 1992.

K. Roufas, Y. Zhang, D. Duff, and M. Yim, "Six Degree of Freedom Sensing for Docking Using IR LED Emitters and Receviers," *Experimtal Robotics VII, Lecture Notes in Control and Information Sciences 271*, Vol. 1, 2001.

B. Minten, R. Murphy, J. Hyams, and M. Micire, "Low-Order-Complexity Vision-Based Docking," *IEEE Transactions on Robotics and Automation*, Vol. 17, December 2001.

Z. Gao, G. Yan, G. Ding, and H. Heng, "Research of Communication Mechanism of Multi-Agent Robot Systems," *Proceedings of the 2001 International Symposium on Micromechatronics and Human Science*, pp. 75–79, September 2001.

V. Jagannathan, R. Dodhiawala, and L. S. Baum, *Blackboard Architectures and Applications.* Academic Press, 1989.

S. McCanne, "Scalable Multimedia Communication with Internet Multicast, Light-Weight Sessions, and the MBone," *Technical Report CSD 981002, UC Berkeley*, March 1998.

G. Banavar, T. Chandra, B. Mukherjee, and J. Nagarajarao, "An Efficient Multicast Protocol for Content-Based Publish Subscribe Systems," *IEEE Transactions on Computers*, Vol. 47, No. 4, pp. 458–471, April 1998.

A. Chan, "Transactional Publish/Subscribe: The Proactive Multicast of Database Changes," *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data*, p. 521, 1998.

[75] B. P. Gerkey and M. J. Mataric, "Principled Communication for Dynamic Multi-Robot Task Allocation," *Experimental Robotics VII, LNCIS 271*, pp. 353–362, 2001.

[76] B. P. Gerkey and M. J. Mataric, "Sold!: Auction Methods for Multirobot Coordination," *IEEE Transactions on Robotics and Automation*, Vol. 18, October 2002.

[77] J. Wang, "Theory and Engineering of Cellular Robotic Systems," Ph.D. dissertation, Department of Computer Science, UCSB, 1990.

[78] J. Wang, "On Sign-Board Based Inter-Robot Communication in Distributed Robotic Systems," *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, Vol. 2, pp. 1045–1050, May 1994.

[79] A. Technologies, "AM-RTD-315 AM Data Transceiver," http://www.abacom-tech.com/transceivers.htm.

[80] T. S. Rappaport, *Wireless Communications: Principles and Practice.* Prentice Hall PTR, 2002.

[81] J. Arai, "A New MAC Protocol for Robot Communication and Its Performance Evaluation," *IEEE International Conference on Distributed Computing Systems Workshops*, pp. 743–748, June 2005.

[82] Solarbotics, "GM12 Gear Motor," http://www.solarbotics.com, 2005.

[83] A. Corporation, "Atmel Microcontrollers," http://www.atmel.com, 2005.

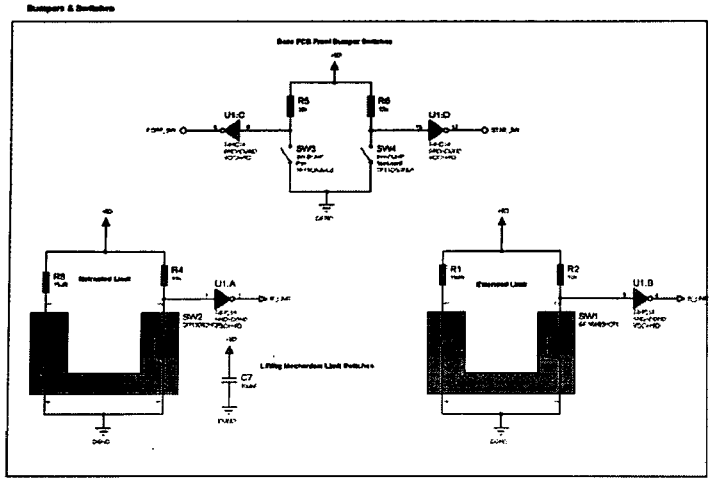
[84] X. Corporation, "Xilinx Programmable Logic," http://www.xilinx.com, 2005.

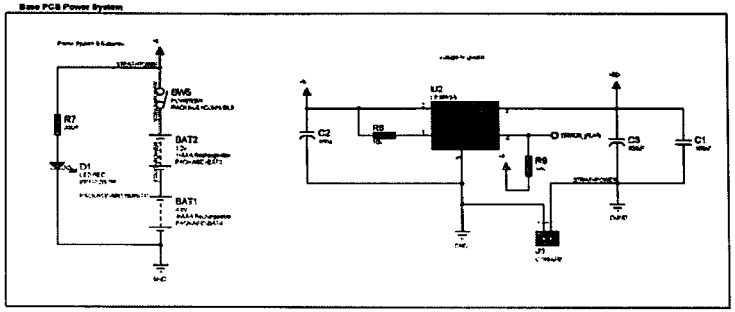
[85] R. V. Ammerman, "Ensuring Byte-Alignment for ASYNC over RF," http://www.piclist.com/techref/microchip/ammermansync.htm, May 2004.

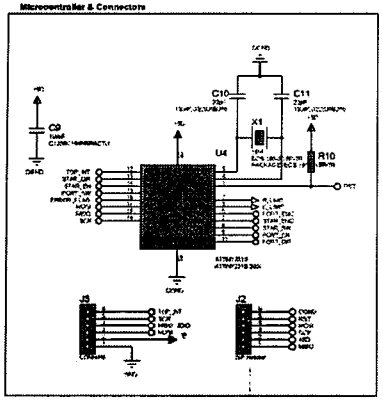
ppendix A

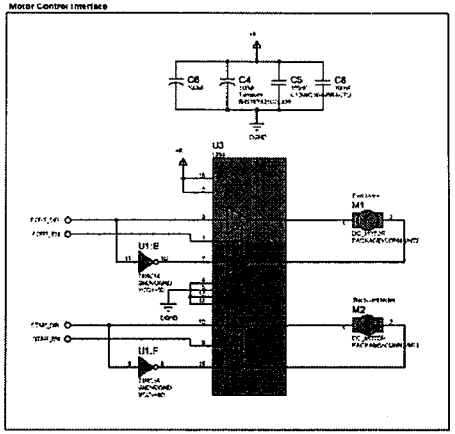
ectrical Schematics

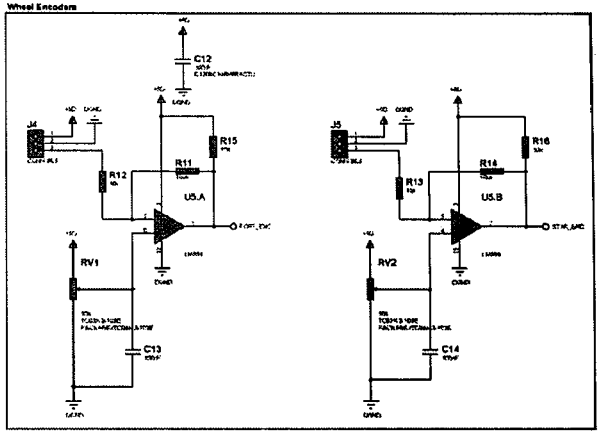
Base Tier



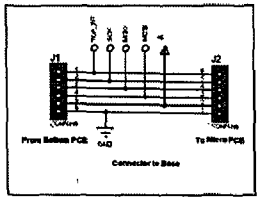


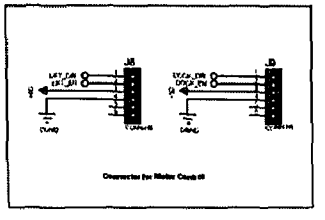


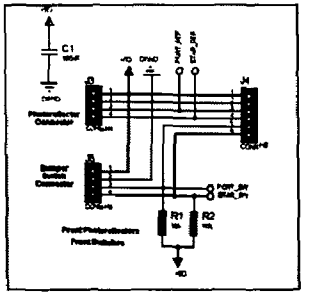




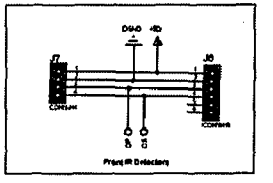
Connectors



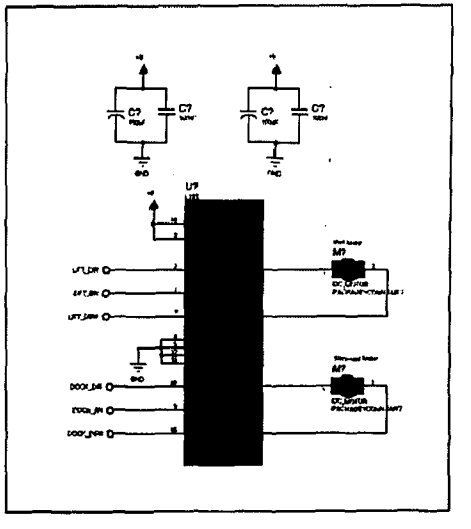


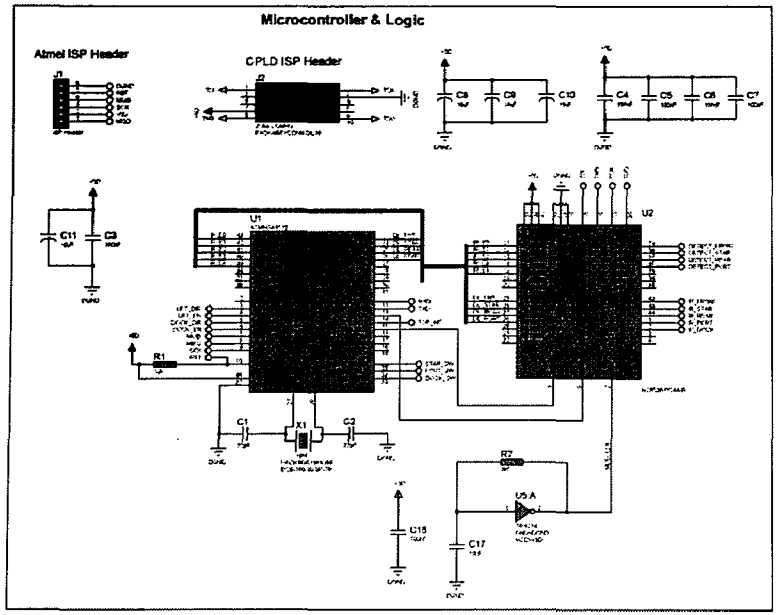


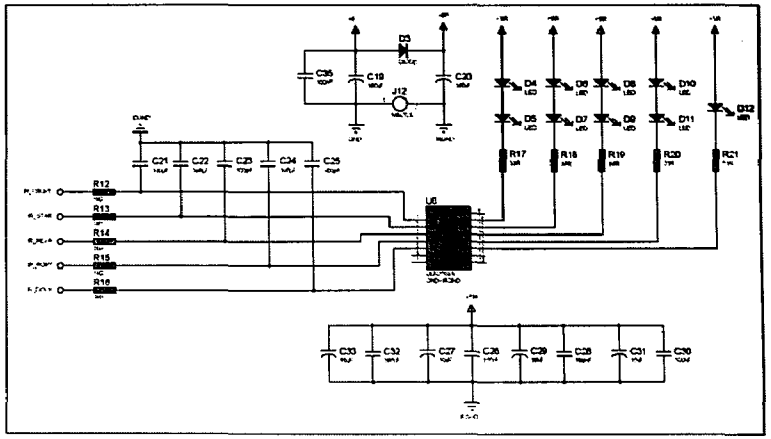
Sensors

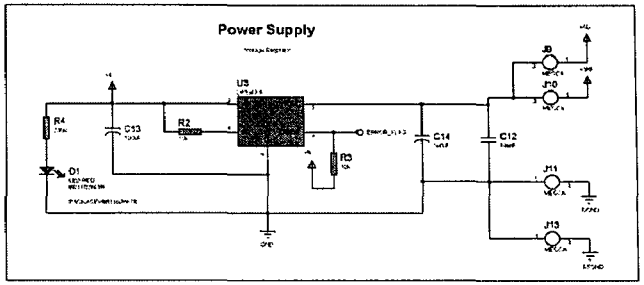


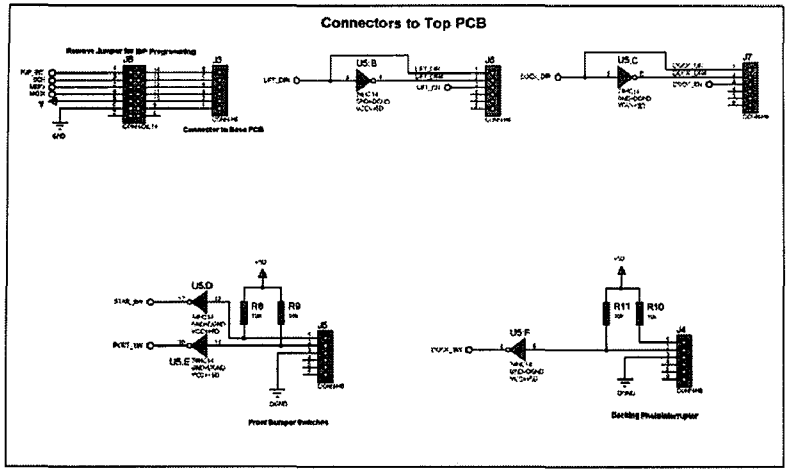
Motor Control

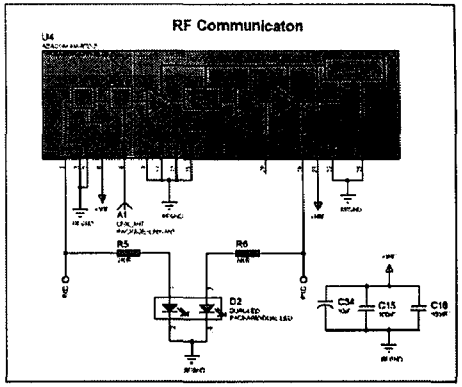



Microcontroller & Logic
Atmel ISP Header
CPLD ISP Header



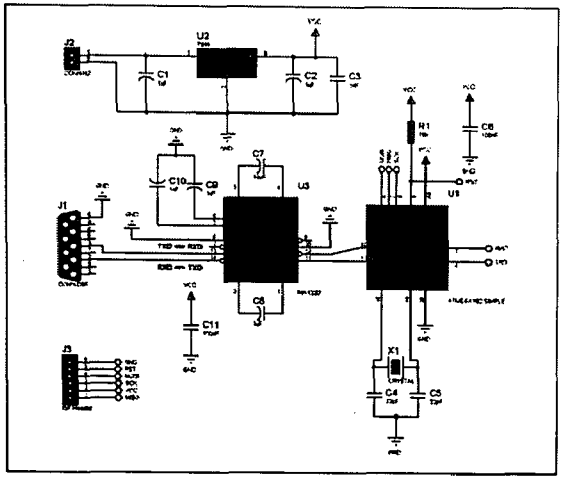

Power Supply


Connectors to Top PCB

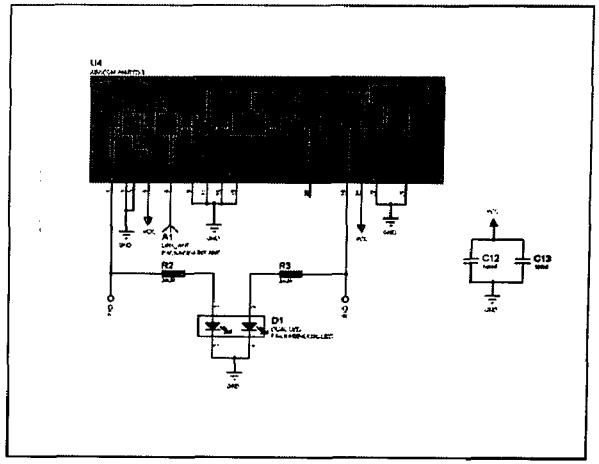

RF Communicaton

Top Tier

Microcontroller and RS232 Interface



RF Transceiver



Appendix B

UART Byte-Alignment Over a Wireless Communication Link

The wireless communications system is one of the most important systems of the robot. It allows the robots to communicate with one another, permits manual remote control of the robots and allows the user to monitor the status and activities of the robots. The wireless communications protocol was designed to use the hardware universal asynchronous receiver/transmitter (UART) feature of the microcontroller.

Due to the nature of the transceiver module it is not possible to simply send a character from one module to another. The modules use amplitude shift keying (ASK). These particular modules use a data slicer in the demodulation process. The data slicer uses a capacitor which charges as incoming data begins to appear at the receiver. It requires many high-low data transitions in order to properly stabilize. In other words, it takes time for the receiver to "turn on" and correctly recognize incoming bits. For this reason, the transmitted data must be *packetized* in order to ensure reliable communications.

A data packet consists of a preamble, synchronization bytes, a header, and the message. The preamble is a series of data bytes with many transitions. The binary sequence 10101010 (hex value 0x55) is transmitted five times. This gives the receiver a chance to stabilize. Next a sequence of four synchronization bytes is transmitted. The synchronization bytes avoid framing errors in the receiving UART and are based on the system presented by Ammerman [85]. Framing errors occur when the receiving UART is not properly synchronized to the

art and stop bits. This occurs because the receiving wireless transceiver requires time to abilize and may begin outputting data in the middle of one of the preamble bytes. In that se one of the "0" bits in a preamble byte may be mistaken for a start bit and a "1" as the op bit.

Once the receiver has stabilized the UART can be in any one of five possible states [85]:

State	Description
1	The receive UART is synchronized properly to the start and stop bits
2	The receive UART is shifted by 2 bit periods
3	The receive UART is shifted by 4 bit periods
4	The receive UART is shifted by 6 bit periods
5	The receive UART is shifted by 8 bit periods

Table B.1: *Byte Misalignment States*

Note that odd-numbered UART offsets are not considered since they would be discarded the framing error detection of the microcontroller. In every case except State 1 the "0" ta bits are being mistaken as the start bits and "1" data bits as the stop bit. States 2 ough 5 are corrected using the synchronization bytes. The synchronization bytes are de- ned to cause framing errors for certain UART states. The framing error will re-synchronize e receiving UART to the proper bit times. The synchronization bytes are sent in the fol- ing order:

1. 01000100 - Causes a framing error in State 3 and State 5. For State 3, the UART will use bit seven as the next start bit which translates the UART to State 2. For State 5, the UART will use bit 3 as the next start bit moving the UART into state 4.
2. 00010001 - Causes framing error in State 2 and State 4. For State 2, the UART will be translated to State A because it will use the start bit of the next byte. For State 4, the UART will translate to State 3 because it uses bit 5 as the start bit.
3. 01000101 - Causes a framing error only in State 3. The UART uses bit 7 as the next start bit translating to State 2.

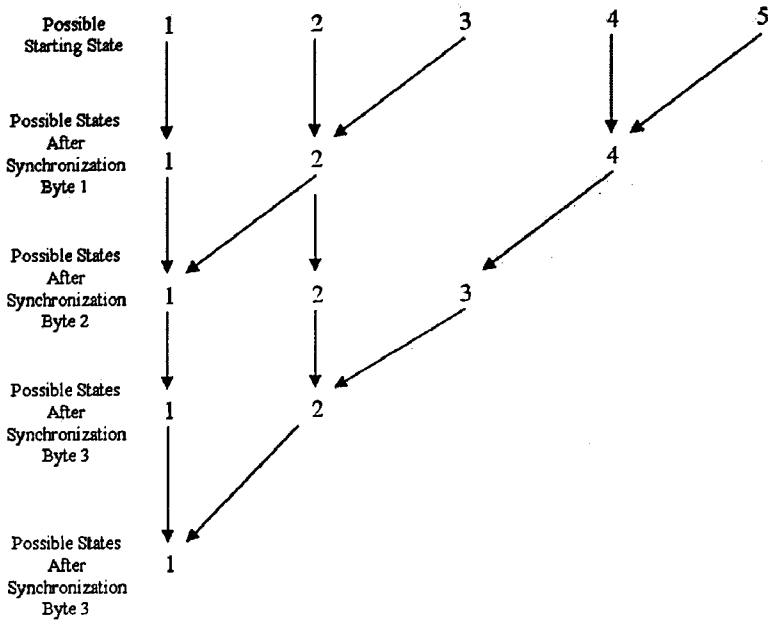


Figure B.1: *Synchronization of the Receiving UART*

4. 00010101 - Causes framing error only in State 2. The UART will use correctly use the next start bit, and the UART is properly synchronized.

Figure B.1 outlines the synchronization process graphically. An arrow indicates a framing error which causes a state transition. This transmission method ensures the reliable transmission of small packets of data over a wireless communications link by guaranteeing that the receiving UART is always synchronized.

Appendix C

Experimental Results

C.1 Stair Climbing Trials

Trial	Outcome	Stair Reached	Time (min:sec)	Notes
1	Power Failure	1	1:30	Robot 2 battery drained
2	Mechanical Failure	1	0:30	Robot 1 bumper switch caught on stair
3	Success	3	4:15	Successful ascension
4	Success	3	5:00	Successful ascension
5	Power Failure	2	2:00	Robot 1 battery drained
6	Sensor Failure	3	3:30	Robot 3 retracted opto did not trigger
7	Mechanical Failure	2	2:00	Robot 3 shaft coupling failed
8	Success	3	3:45	Successful ascension
9	Power Failure	2	1:30	Robot 3 battery drained
10	Sensor Failure	1	1:00	Couple disengaged
11	Success	3	3:45	Successful ascension
12	Mechanical Failure	2	2:15	Robot 2 lifting mechanism jammed
13	Power Failure	3	3:15	Robots 2 & 3 battery drained
14	Power Failure	2	1:45	Robot 1 insufficient power for lifting
15	Mechanical Failure	1	0:45	Robot 3 bumper jammed on stair
16	Power Failure	1	1:30	Robot 2 battery drained
17	Success	3	4:30	Successful ascension
18	Power Failure	2	2:30	Robot 3 battery drained
19	Power Failure	2	3:00	Robot 3 battery drained
20	Mechanical Failure	3	4:00	Robot 2 shaft couple failed
21	Success	3	6:00	Successful ascension
22	Power Failure	2	4:00	All robot batteries drained
23	Sensor Failure	2	2:30	Robot 1 stair sensor disengaged
24	Mechanical Failure	1	0:30	Robot 1 lifting mechanism failed
25	Mechanical Failure	2	2:15	Robot 1 bumper switch caught on stair
26	Success	3	4:45	Successful ascension
27	Sensor Failure	2	2:30	Robot 2 extended optp did not trigger
28	Success	3	4:15	Successful ascension
29	Power Failure	2	3:00	Robot 3 battery drained
30	Mechanical Failure	3	4:00	Robot 2 bumper torn off
31	Power Failure	2	2:45	Robot 2 battery drained
32	Power Failure	2	3:00	Robot 3 battery drained
33	Power Failure	1	1:30	Robot 3 battery drained
34	Mechanical Failure	2	3:00	Robot 1 lifting motor seized

Table C.1: *Results of Stair Climbing Trials*

:.2 Stair Climbing Trials Histogram

gure C.1 presents the results of all 34 stair climbing trials.

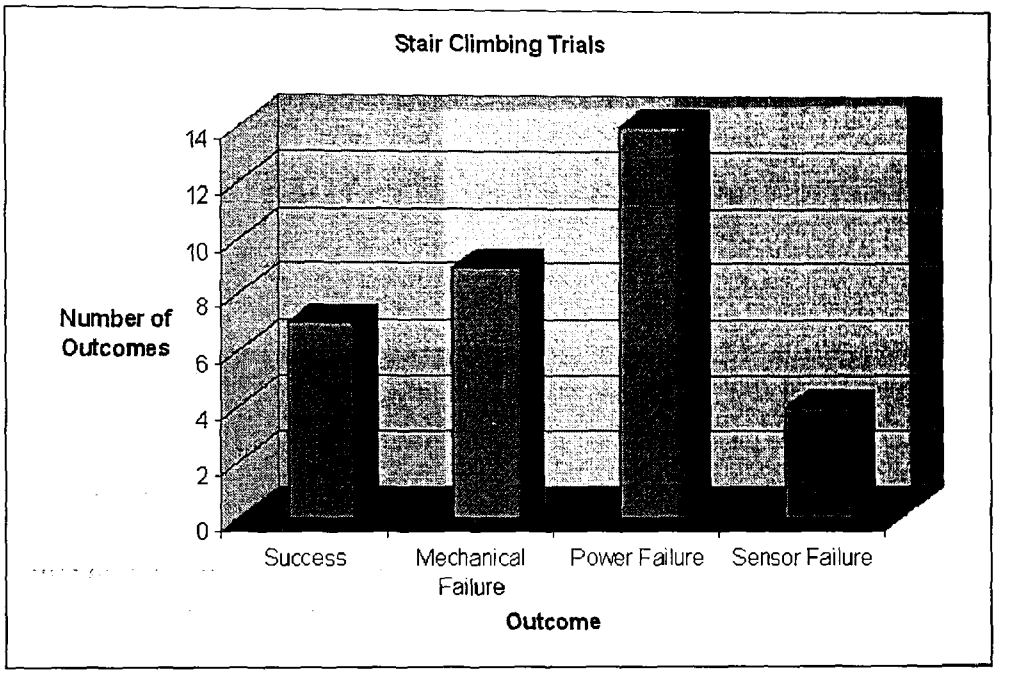
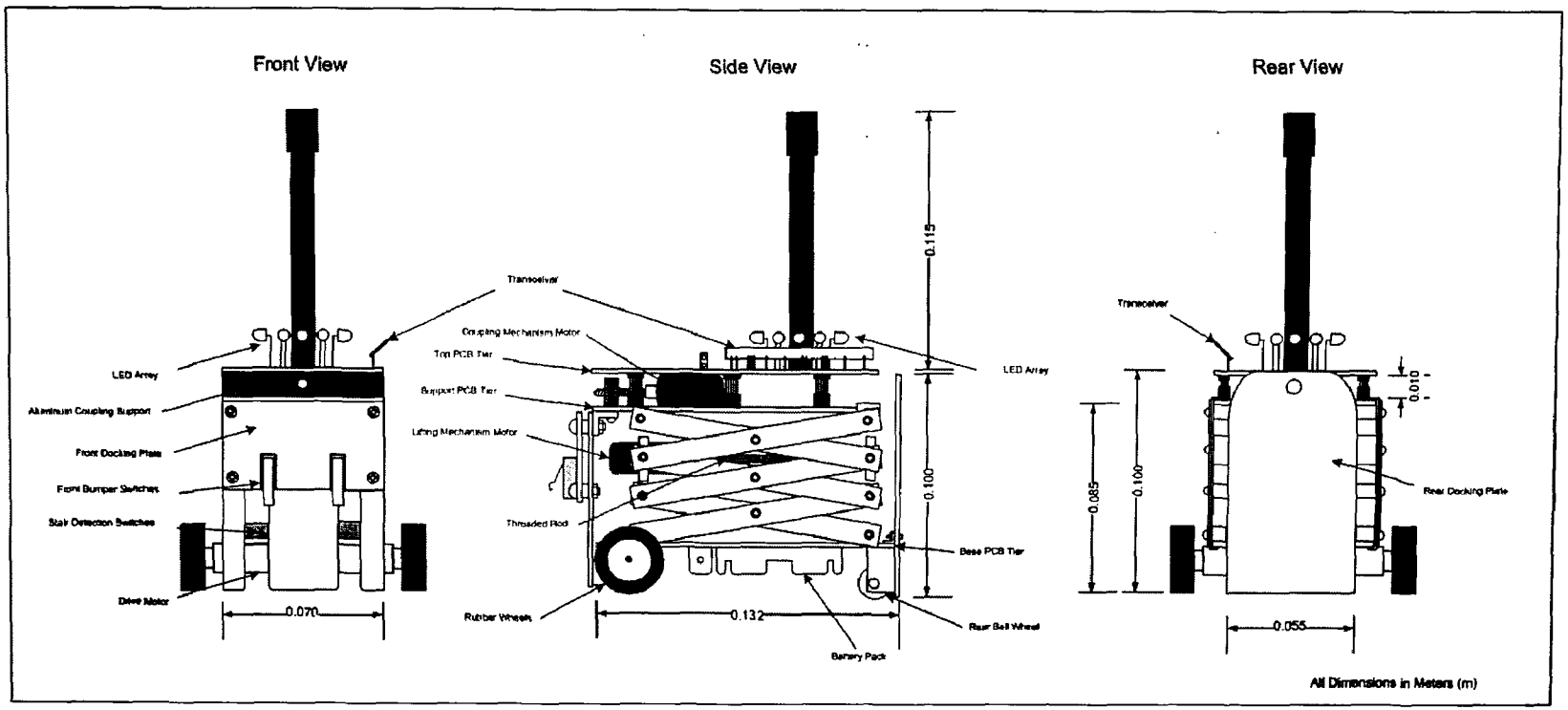


Figure C.1: *Stair Climbing Experimental Results*

Appendix D

Robot Mechanical Drawings


Front View
Side View
Rear View
LED Array
Aluminum Coupling Support
Front Docking Plate
Front Bumper Switches
Stair Detection Switches
Drive Motor
0.070
Transceiver
Coupling Mechanism Motor
Top PCB Tier
Support PCB Tier
Lifting Mechanism Motor
Threaded Rod
Rubber Wheels
0.132
Battery Pack
Rear Ball Wheel
Base PCB Tier
LED Array
0.115
0.100
Transceiver
0.085
0.100
0.010
Rear Docking Plate
0.055
All Dimensions in Meters (m)