REPORT DOCUMENTATION PAGE

AFRL-SR-AR-TR-04-

0197

Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE
	September 10, 2002	Annual Report: August 31, 2002

4. TITLE AND SUBTITLE

Hormone-Based Cooperative Communication

5. FUNDING NUMBERS

AFOSR F49620-01-1-0441

6. AUTHOR(S)

Wei-Min Shen

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

University of Southern California
Information Sciences Institute
4676 Admiralty Way, Marina del Rey, CA 90292

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

USAF, AFRL
Air Force Office of Scientific Research
801 N. Randolph Street
Arlington, VA22203-1977

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

20040423 031

13. ABSTRACT ***(Maximum 200 Words)***

The control and communication of massive robot swarms in a distributed manner is a difficult problem because global behaviors must be emerged as a collection of many local actions. This project uses a biologically inspired control method called Digital Hormone Model (DHM) to control the communication, tasking, and execution of massive robot swarms based on local communication, signal propagation, and stochastic reactions. This model is probabilistic, dynamic, fault-tolerant, efficient in computation, and can be easily tasked to deal with topology changes in the communication network and modify the global behaviors. Different from most existing distributed control and learning mechanisms, the DHM considers the topological structure of the organization, and supports dynamic re-configuration and self-organization. In the last year, we have formalized the concept of DHM and conducted experiments of simulating swarm behaviors in large scale for target attacking, network formation, self-repair, and avoid pitfalls in mission execution.

14. SUBJECT TERMS

Homrone-inspired communication, distributed cooperation.

15. NUMBER OF PAGES

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unlimitted	Unlimitted	Unlimitted	Unlimitted

Project Final Report

HORMCOMM:
Hormone-Inspired Cooperative Communication

USAF, Air Force Office of Scientific Research

Award Number: F49620-01-1-0441

Period of Performance: 06/15/2001 - 12/14/2003

Dr. Wei-Min Shen
Polymorphic Robotics Laboratory
Information Sciences Institute
University of Southern California
4676 Admiralty Way
Marina del Rey, CA 90292-6695
Phone: 310-448-8710
Fax: 310-822-0751
shen@isi.edu
http://www.isi.edu/robots

A Objectives

The goal of this research project is to design and implement a distributed architecture for communication among autonomous agents in a dynamic network that is capable of self-reconfiguration. In such a network, we assume that the autonomous agents are physically or logically interconnected and that can dynamically self-reconfigure their shape, formation, and configuration, in order to best function in uncertain and adversarial environments. Examples of such dynamic communication may occur in self-reconfigurable robots, sensor networks, mobile robot teams, and squads of unmanned combat or reconnaissance aircrafts.

The evaluation of this new communication architecture will be based on two test beds: one is for self-reconfigurable robots in which autonomous robot modules are to be reconfigured dynamically to form different shape and size and communication must be effective during and after all reconfigurations. The other is a set of simulated mobile robots that have wireless communication that can self-organize and self-repair for unexpected situations. The expected result of this project will be a set of communication protocols that can handle communication in dynamic network, tolerate noise and variable latency in communication channels, and robust to the topology reconfigurations of communication network.

B Status of Effort

The project started at 06/15/2001. We first started to formulate the dynamic communication problem in the context of self-reconfiguration systems, and developing a hormone-based approach for the problem. The key idea is to investigate and understand how biological cells communicate with each other using hormone-like signals and why such a communication method is capable of handling noises, reconfiguration changes, and do so without even to have identifiers or addresses for the biological cells.

During the first year of the project (06/15/2001 through 08/31/2002), we have achieved: (1) formalize the Digital Hormone Models using three mathematical components (dynamic network with connectors, stochastic selection of actions and diffusion-reactions of hormone messages); and (2) implement the DHM in a cellular automata environment and simulate the model to reproduce results that match the observations in feather formation in biological systems; (3) extend the simulation environment and demonstrate the DHM's diverse abilities for battlefield situations such as target attacking, network formation, self-repair, and avoid pitfalls in mission execution.

During the second year of the project (09/01/2002 through 12/14/2003), we have achieved: (1) extended the Digital Hormone Model proposed to become a theory of self-reconfiguration; and (2) designed a general framework for a hormone-inspired control language based on distributed functional languages.

C Accomplishments and New Findings

We recognize that biological systems offer many valuable lessons for achieving the desired features of future communication systems, and the key enabling technology seems to be the ability of self-organization or self-reconfiguration. High-performance can be achieved by restructuring structures and better distributing tasks to resources. Robustness can be obtained by adapting to environments and self-repairing component failures. Scalability can be accomplished by allowing autonomous components to self-organize for global performance

without pre-imposing any fixed superstructure. Furthermore, earlier results in cell pattern formation have shown that homogeneous embryos cells can communicate and self-organize into complex patterns such as feathers and scales and the morphological differences between different patterns appears to be controlled not by the presence or absence of particular molecules but by the level and configuration of their expression. To demonstrate the generality of similar approaches in robotics, we have also constructed a set of metamorphic robotic modules that can physically and autonomously reconfigure into robots that have different shapes and sizes yet the effective communication among themselves regardless of the change of topology.

Based on the above results, we have started investigate how biological cells communicate and control themselves, and begun to develop a *Digital Hormones Model* (DH-Model) for such purposes. Such a model of digital hormones is inspired by the fact that complex self-organization behaviors in biological systems appear to be triggered, controlled, and coordinated by configurations of cells through a set of hormone-like signaling molecules according to the physical and chemical principles of single cells. Preliminary experiments of this model on our existing self-reconfigurable robots have shown very encouraging results for building new communication systems that are robust to device damage, channel changes, or noises.

This project is evaluated in two domains: self-reconfigurable robots and massive robotic swarms. Among all computational studies of dynamic communication networks, modular self-reconfigurable robots are perhaps the best example to implement and demonstrate dynamic communication within a network that is capable of self-reconfiguration. These robots are constructed from a set of autonomous robot cellular modules that can self-reconfigure into structures of different shape, size, and configuration in order to accomplish complex task in dynamic and uncertain environments. These robots are highly desirable for tasks such as fire fighting, search/rescue after earthquake, and battlefield reconnaissance, where robots would encounter unexpected situations that are difficult for fixed-shape robots to deal with. For example, to maneuver through difficult terrain, a metamorphic robot may have to transform itself into a snake to pass through a narrow passage, grow legs to climb over obstacles, or become a ball to roll down a slope. Similarly, to enter a room through a closed door, a metamorphic robot may disassemble itself into a set of smaller units, crawl under the door, and then reassemble itself in the room. How can modules maintain their communication during all the reconfiguration process will be a challenge problem for testing and inspiring new protocols for hormone-based collaborative communication.

Massive robot swarms contain a great number of small and simple robots that are mobile, agile, affordable, locally communicated, and collaborative for common goals. Just like foraging Army ants in the rainforest, once triggered and driven by some given task signals, such robot swarms will pursue their goals relentlessly. They fear no sacrifice and surmount all difficulties, obstacles, destructions, and pitfalls in achieving their goals. Their courses may be non-deterministic but their overall behaviors are organized and targeted. They do not have fixed leaders but coordinate their actions via a totally distributed control mechanism. They can self-repair damage to their organization and self-adjust tactics and strategies.

D Digital Hormone Model

The Digital Hormone Model as a bio-inspired distributed control method for self-reconfigurable systems and networks. The model includes three basic components: a dynamic self-reconfigurable network with autonomous entities that have "connectors"; a set of probabilistic functions for entities to select actions based on local states, sensors, topology, and received hormone messages; and a set of equations for the diffusion-reaction of hormone messages. This model allows massive simple robots in large-scale systems to communicate and react to each other, and self-organize into global patterns that are suitable for the given task and environment. All communication and reactions are local and use signals that are similar to hormones that regulate cell activities in biological organisms. The proposed model combines advantages from Turing's reaction-diffusion model, stochastic reasoning and action, dynamic network reconfiguration, pheromone-based control, and individual learning techniques such as reinforcement learning. In the rest of the paper, we describe the key components of digital hormone models, and present the simulation results of hormone-controlled behaviors for massive robot swarms.

D.1 Biological Inspirations

Distributed massive systems that can self-organize into purposeful global behaviors are ubiquitous in nature. They appear in physics, chemistry, materials sciences, and others [2]. But perhaps the richest source for such phenomena are biological systems. Here, we will describe two of the most fascinating phenomena that inspired this proposal: morphallaxis and feather formation.

Morphallaxis is a process by which an organism can regenerate either a part or the whole from a fragment by self-reorganization of cells without cell proliferation. This is a process of tissue reorganization observed in many lower animals following severe injury, such as bisection of the animal, and involves the breakdown and reformation of cells, movement of organs, and re-differentiation of tissues. The result is usually a smaller but complete individual, derived entirely from the tissues of part of the original animal. It is believed that such a reorganization process is the most efficient way for simple organisms to self-heal and self-regenerate. One of the most remarkable examples of morphallaxis is a type of invertebrate freshwater animal called a hydra. If a hydra is cut in half, the head end reconstitutes a new foot, while the basal portion regenerates a new hydranth with mouth and tentacles. Even if a hydra is minced and the pieces scrambled, the fragments grow together and reorganize themselves into a complete whole. How this dramatic self-healing and self-adaptation process takes place is still a mystery. Some simple morphallaxis behaviors have been engineered into our CONRO self-reconfigurable robots supported by DARPA.

Another interesting biological example is the process of feather formation. Homogeneous skin cells first aggregate and form feather buds that have approximately the same size and space distribution. The feather buds then grow into different types of feathers depending on the region of the skin. Earlier theories believed that such a process was initiated and coordinated by some "key" or "leader" cells on the skin. However, recent findings in biological experiments [3, 4] have challenged these theories. Choung and his colleagues separated the related cells from the skin, randomly scrambled them and reset their initial positions, and observed their growth afterwards. The scrambled skin cells surprisingly grew into the patterns of feather buds as before.

These findings suggest that there are no predetermined molecular addresses, and that feather morphogenesis is likely a self-organizing process based on the physical-chemical properties and reactions between homogeneous cells. This example suggests that in order to build resolute robotic task forces for future combat systems, a totally distributed control mechanism similar to those in the biological systems may be a productive and feasible goal to achieve.

During the biological experiments, biologists observed some interesting relations among the reaction and diffusion characteristics of the hormones secreted from the cells, the size and space distribution of the final feather buds, and the initial density of cell population. In particular, they observed that, as shown in the left side of Figure 1, while the number of formed feather buds is proportional to the cell population density, the size of the feather buds remains approximately the same regardless of different population densities.

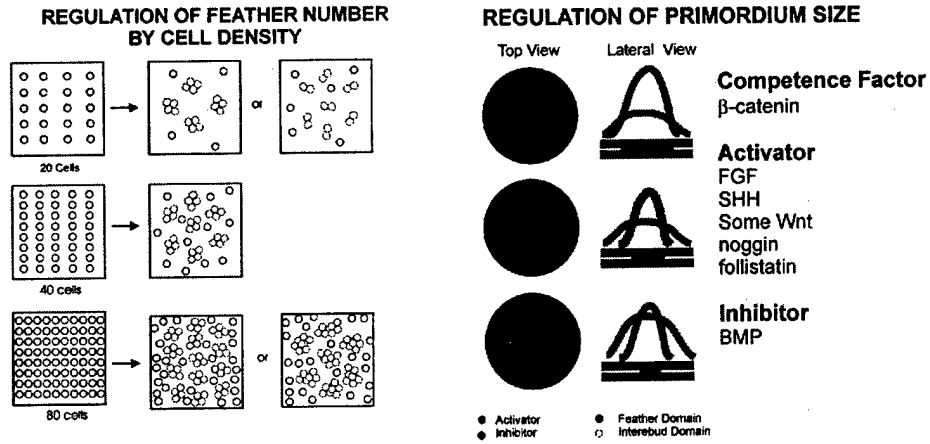


Figure 1: Coordination regulated by "activator" and "inhibitor" hormone profiles

The size of the feather buds, however, is related to the reaction and diffusion profiles of the activator and inhibitor hormones secreted from the cells. If the concentration ratio of the activator hormone (the inner red circle) to the inhibitor hormone (the outer green circle) is high (shown in the top right row in Figure 1), then the final size of the feather buds will be larger than usual. If the ratio is balanced (shown in the middle right row in Figure 1), then the size of the formed feather buds will be normal. If the ratio is low, then the size of the formed pattern will be smaller than usual (shown in the bottom right row in Figure 1). These observations are most interesting to us because they can be used as the basic criteria for evaluating computational models of hormone-based self-organization.

D.2 The Representation of Digital Hormone Model

The DHM is inspired by four factors: (1) biological discoveries about how cells self-organize into global patterns, (2) the existing self-organization models, such as Turing's reaction-diffusion model, (3) the stochastic cellular automata, and (4) the distributed control systems for self-reconfigurable robots.

The basic idea of Digital Hormone Models is that robots in a dynamically networked organization will use hormone-like messages to communicate and collaborate global behaviors

such as locomotion, tactic formation, self-reconfiguration, and self-repair. The hormone-like messages are similar, but not identical, to content-based messages. They do not have addresses but propagate through the network (note that hormone propagation is different from message broadcasting). There is no guarantee that every robot in the network will receive the same copy of the original message because a hormone may be modified during its propagation. Hormones are also similar, but not identical, to pheromones because hormones can propagate from cell to cell without leaving residues in the environment and their propagation is constrained by the current connections between robots. For example, two robots that are not directly reachable in a single hop are not considered connected, so they cannot directly send hormones to each other (while a pheromone is left in the environment to be sensed by any other robot). Although these differences are superficial, we feel hormones are more convenient to represent topological constraints between elements of an organization. All robots have the same decision-making protocol, but they will react to hormones according to their own local state and topological information so that a single hormone may propagate in the network and cause different robots to perform different actions.

Mathematically speaking, a Digital Hormone Model consists of three key components: a specification of a dynamic network of mobile robots, a probabilistic function for robot behavior, and a set of equations for hormone reaction, diffusion, and dissipation.

A *Dynamic Network of Mobile Robots* (DNMR) is specified as a network of N mobile and communicative robot nodes. Each node has a set of *connectors* through which the node can dynamically connect to other nodes to form *edges* for communication or physical (mechanical) coupling. The concept of a connector is theoretically new but exists in real systems. For example, in a wireless network, the connectors of a node are the channels it can use to communicate with others. A channel of a node must be "connected" to a channel of another node to form an edge of communication. In self-reconfigurable robots, the connectors are physical so that an edge is a physical coupling and a network of nodes can form physical structures with different shapes and sizes. The connectors are valuable but limited resources for nodes. Because connectors can be joined and disjoined, they make the edges in a network dynamic, and the analysis of network reconfiguration possible. Let N_t and E_t denote all the nodes and edges that exist in a dynamic network at time t, then a DNMR at time t can be defined as:

$$DNMR_t \equiv (N_t, E_t) \tag{1}$$

Note that both N_t and E_t can change dynamically because nodes can autonomously join, leave, or be damaged, and edges can be formed and disconnected by the connectors of the nodes. Different from classical models, nodes do not have unique IDs or addresses, the number of nodes and edges in the network is not known, and there is no global broadcast. A node can only communicate with its current neighbors through its current edges. This local communication assumption is realistic and necessary for large-scale robotic task forces, for two arbitrary nodes can be so far away that direct communication is not possible, especially when nodes only have limited resources. Through local communication, nodes can either generate hormones or propagate hormones. By default, a generated hormone will be sent to all the current edges of its generator, and a received hormone will be propagated to all the current edges except the one through which the hormone was received.

The second component of a Digital Hormone Model is a specification of robot node behavior. Nodes in the network can perform normal actions such as activating sensors or motors. A node

selects its actions, *B*, based on a probability function, *P*, that is conditioned on four local factors: the connector information, *C*; the sensor information, *S*; the values of local variables, *V*; and the received hormones, *H*:

$$P(B|C,S,V,H) \tag{2}$$

The actions, *B*, of a node include the commands to the local sensors and actuators, as well as actions that change connectors and generate or propagate hormones. Different from most existing probabilistic models, such as hidden Markov models, partially observable Markov decision processes, and reinforcement and Q-Learning models, the *P* function here considers not only sensor and state information *S* and *V*, but also topological information, *C*, and communication information, *H*. These allow Digital Hormone Models to support dynamic reconfiguration and self-organization in network structures. The function, *P*, although it is local and homogenous for all nodes, can greatly influence the global behaviors of the network and can be used to predict and analyze the global network performance "in the large." For example, in the simulation of feather formation to be discussed later, the characteristics of *P* can influence whether or not any global patterns can be formed. Biologically speaking, we believe that the function *P* partially simulates the hormone receptors and the control mechanisms found in the biological cells.

The third part of a Digital Hormone Model is the specification for hormone reaction, diffusion, and dissipation. Following Turing [5] and Witkin [6], we assume in the mathematical description that hormone reaction and diffusion occur through a two-dimensional medium, although analogous results can be derived for arbitrary dimensions and some higher dimensions are indeed used in applications of self-reconfigurable robots. The concentration of each hormone is a function of position and of time. We denote the concentration function for a particular hormone by *H(x, y)*. The reaction-diffusion-dissipation equation governing the hormone is given by:

$$\frac{\partial H}{\partial t} = \left(a_1 \frac{\partial^2 H}{\partial x^2} + a_2 \frac{\partial^2 H}{\partial y^2} \right) + R - bH \tag{3}$$

The first term on the right is for diffusion, and a_1 and a_2 are constants that represent the rate of diffusion in *x* and *y* dimension respectively. The function *R* is the reaction function governing *H*, which depends on all the other concentrations of hormones. The constant *b* is the rate for dissipation. The equation (3) is usually considered to be a part of an environmental function *G*, which is responsible for the implementation of the dynamics of communication or other effects of actions. For example, if two nodes sent out radio signals with the same frequency at the same time, then *G* would be responsible for simulating the interference between the two signals. Although the *G* function is not a part of any robot node, it sometimes can be simulated by the actions of the robots.

As we can see from the above definitions, a Digital Hormone Model is an integration of *dynamic networks* (Equation 1), *topological stochastic action selection* (Equation 2), and *distributed control by hormone reaction-diffusion* (Equation 3). As we will see in the rest of the paper, this integration provides a very powerful coordination mechanism for dynamic networks of mobile robots. The execution of DHM is very simple. All robot nodes in the network asynchronously execute the basic control loop as follows:

1. Select actions by $P(B|C,S,V,H)$;
2. Execute the selected actions in B;

3. Perform hormone generations and propagations;
4. (Optional) Simulate hormone reaction, diffusion, and dissipation;
5. Go to Step 1.

To illustrate the basic ideas of the DHM for self-organization, let us now define a simple DHM, call DHM_0, in Figure 2, where cells (shown as black dots) can move in a torus space of discrete grids. Each cell occupies one grid at a time and can secrete hormones (the red and the green) to neighboring grids to influence other cells' behaviors. For simplicity, we assume that all cells synchronize their actions and that the grids carry out the reaction and diffusion of hormones. A cell at a grid (a, b) can secrete two types of hormones, the activator A and the inhibitor I. The *diffusion* of A and I at a surrounding grid (x, y) are given by the standard distribution functions:

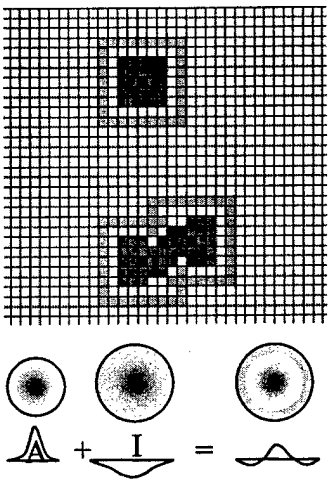


Figure 2: The Simple DHM_0

$$H_A(x,y) = \frac{a_A}{2\pi\sigma^2} e^{\frac{(x-a)^2+(y-b)^2}{2\sigma^2}} + R \quad (4)$$

$$H_I(x,y) = -\frac{a_I}{2\pi\rho^2} e^{\frac{(x-a)^2+(y-b)^2}{2\rho^2}} + R \quad (5)$$

where a_A, a_I, σ, and ρ are constants, and $\sigma<\rho$ in order to satisfy the Turing stability condition (see related work). Note that because $\sigma < \rho$, the hormone A has a sharper and narrower distribution than the hormone I, and these characteristics are similar to those observed in the biological experiments. We assume that the hormone A has the positive value and the hormone I has the negative value. For a single isolated cell, the hormone concentration in its neighboring grids looks like three "colored rings" (see the lower-right corner in Figure 2). The activator hormone dominates the inner ring (the red); the inhibitor hormone dominates the outer ring (the green); and the middle ring is neutral (the white grids) because it is where the hormones of A and I have canceled each other. The *reaction* between two hormones in a grid is computed by summing up all present "A"s and "I"s in the grid:

$$R = \sum_N (H_A + H_I) \quad (6)$$

When two or more cells are near each other, the hormones in the surrounding grids are summed up to compute hormone strength. In the upper part of Figure 2, we have illustrated combined hormones around a single cell and around two nearby cells. Since the grids are discrete, the rings around the cells are shown as squares instead of circles.

When all cells are moving in synchronization, there may be a chance that multiple cells will "collide" in the same grid. The collision of cells is solved in a simple manner. All cells first "virtually" move to the grids they selected. If there are multiple cells in the same grid, then the extra cells will be randomly distributed to those immediate neighboring grids that are empty. This is an environmental function, not a cellular action. But this action will ensure that no grid is hosting more than one cell at any time.

For cell behaviors, DHM_0 is governed by a function $P_0(B \mid C, S, V, H)$ defined as follows:

B: Each cell has ten actions. B_0 for secreting the A and I hormones, and B_1, ..., B_9 for moving into the nine neighboring grids: north, south, west, east, northeast, northwest, southeast, southwest, and self (the occupying grid);

C: Cells have no connectors in this simple model;

S: Each cell has nine hormone sensors, one for each of the neighboring grids;

V: Cells have no local variables in this model;

H: The nine hormone values sensed by the sensors;

$$P_0(B \mid C,S,V,H) = \begin{cases} P_0(B_0 \mid C,S,V,H) = 1.0; \\ P_0(B_i \mid C,S,V,H) = \textit{BestNeighborFunction}, \text{ where } i = 1,\ldots,9. \end{cases}$$

The function *BestNeighborFunction* is defined so that the probability of moving to a particular neighboring grid is proportional to hormone A and inversely proportional to hormone I in that grid, and the sum of these probabilities is 1. In the rest of paper, we present simulation results of four different scenarios for controlling massive robot swarms.

D.3 Simulating Cell Behaviors in Feather Formation

Given Equations 4, 5, 6 and $P_0(B \mid C, S, V, H)$, DHM_0 can be used to investigate how hormones affect self-organization and whether they can enable locally interacting mobile robots to form globally interesting patterns. We can also change the characteristics of these parameters, and observe and analyze the global effects at large. In particular, we would like to investigate:

- Will DHM_0 enable cells/robots to self-organize into patterns at all?
- Will the local hormone profiles or the initial population density affect the features (such as size or shape) of the final patterns?
- What types of hormone profiles can enable self-organization?

To answer these questions, we have conducted two sets of simulation experiments. In the first set of experiments, we use the same hormone diffusion profile and run a set of simulations on a space of 100x100 grids (all simulation experiments in this proposal are running in the same size environment) with different cell population densities ranging from 10% (~1000 cells) to 50% (~5000 cells). Starting with cells randomly distributed on the grids, each simulation runs up to 1,000 action steps, and records the configuration snapshots at steps of 0, 50, 500, and 1,000. As shown in Figure 3, cells in all simulations indeed form clusters with approximately the same size. These results demonstrate that HDM_0 does enable cells to form patterns. Furthermore, the results match the biological observations that the size of the final clusters does not change with cell population density, but the number of clusters does. Lower cell densities result in fewer clusters, while higher densities form more clusters.

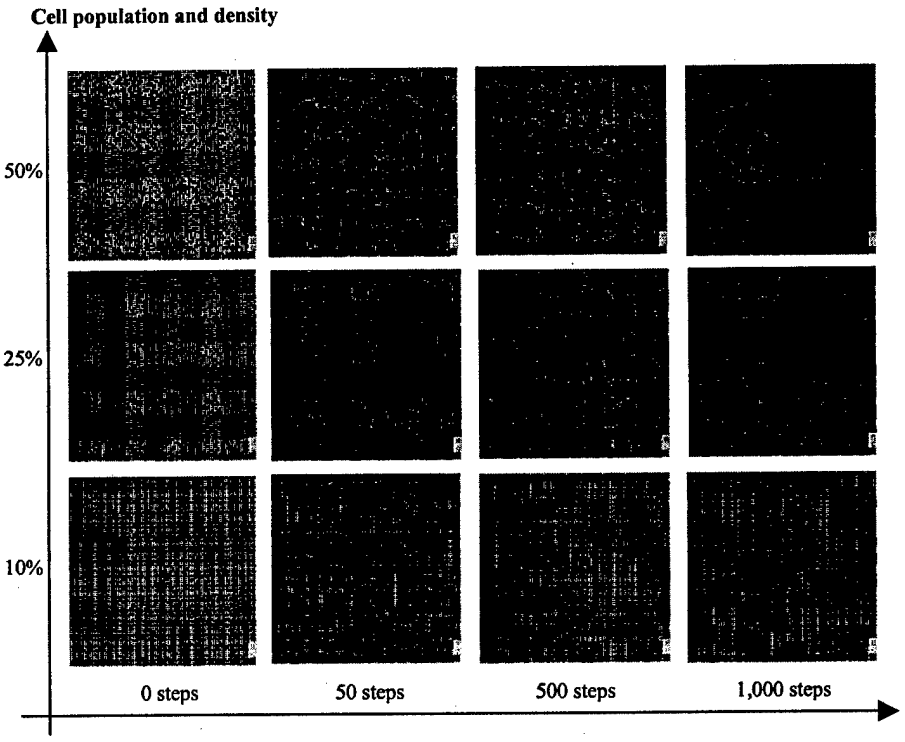


Figure 3: The same hormone profile but different cell population

To see how hormone profiles affect final patterns, we conducted the second set of experiments. We started with the same cell population density, but varied the local hormone profiles by changing the parameters for Equations 4 and 5. We wanted to observe the effects of different hormone profiles on the results of pattern formation. As shown in Figure 4, when a balanced profile of activator and inhibitor is given (see the second row), the cells will form final patterns as in the first set of experiments. As the ratio of activator over inhibitor (σ/ρ) increases, the size of final clusters will also increase (see the third row). These results are an exact match with the findings in the reported biological experiments [3].

When the ratio of A/I becomes so high that there are only activators and no inhibitors (increases a_A/a_I), then the cells will form larger and larger clusters, and eventually become a single connected cluster (see the fourth row). On the other hand, when the ratio is so low that there are only inhibitors and no activators, then the cells will never form any patterns (see the first row), regardless of how long the simulation runs. This shows that not all hormone profiles enable self-organization. These results are yet to be seen in biological experiments, but they are consistent with the principles of hormone-regulated self-organization and thus qualify as meaningful predictions of cell self-organization by hormones.

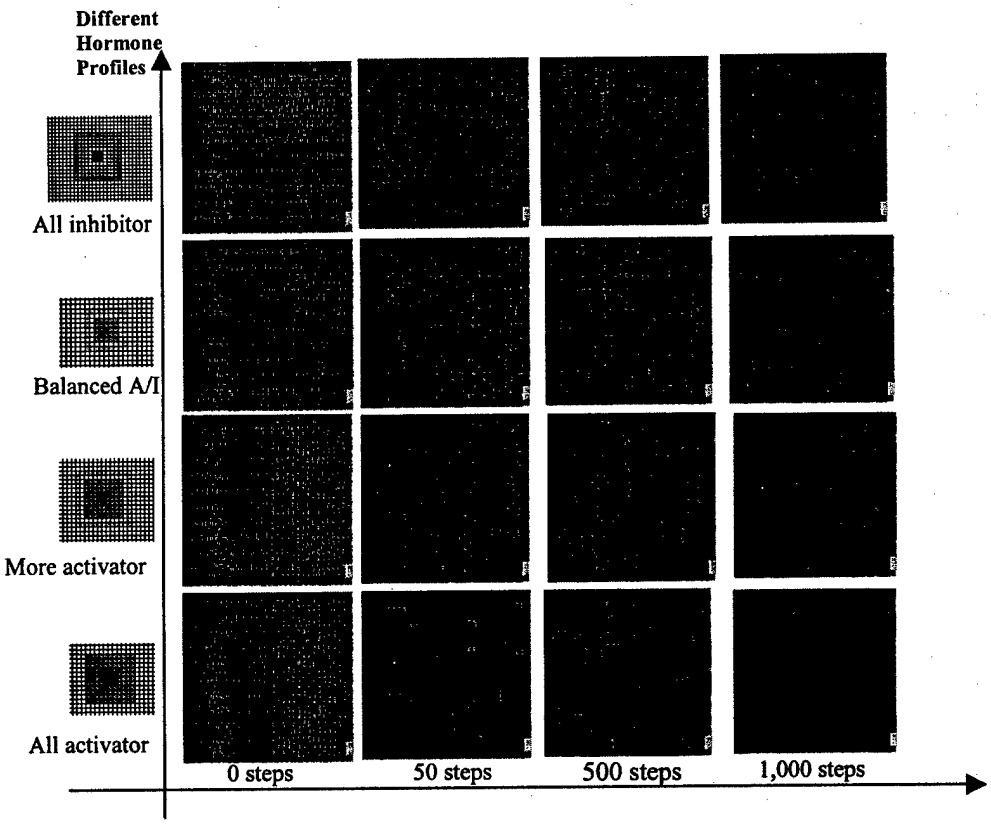


Figure 4: Simulation with different hormone profiles

In addition to changing the ratio of activator and inhibitor hormones, we also varied the shape of the hormone diffusion profiles and observed the effects on the features of the final patterns. For example, when we changed the profile to a narrow and long sandwich, the cells formed striped patterns as shown in Figure 5. This demonstrates that when local hormone profiles are anisotropic, we can predict that the global patterns will be anisotropic as well. This provides a hint that local action profiles can predict, control, and change global massive behaviors.

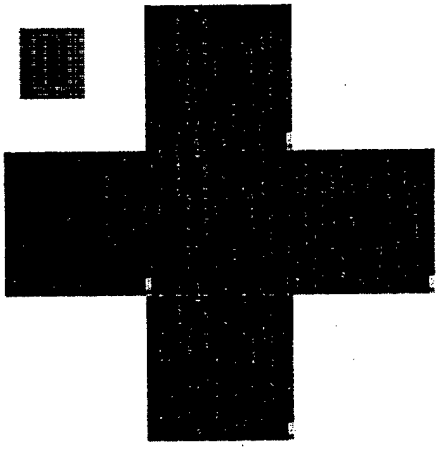


Figure 5: Diffusion profile for stripe patterns.

To test the effects of stochastic behavior function, we also replaced P_0 by a random function, or the Metropolis rule used in simulated annealing [7]. Experimental results showed that no global patterns were formed and neither function produced the same results as P_0 does. One possible explanation why Metropolis rule does not work here is that it first randomly selects a neighbor without considering the concentration of hormones, and then makes a go or no-go decision based on the energy difference and temperature (which we have experimented with using many different values).

In conclusion, the experimental results presented not only demonstrate that the proposed DHM is indeed an effective tool for simulating and analyzing self-organization phenomena, but that it is also capable of predicting and controlling global behaviors based on local hormone interaction profiles. The results suggest that hormones may play a critical role in networks of mobile robots because cells in the simulation can be viewed as intelligent and autonomous systems and they form global patterns based on stochastic local actions and information exchanges. This provides a departure point for new hypotheses, theories, and experiments. Since the model is mathematically adjustable, it is much more economic and efficient for scientists, especially researchers in MARS, to design new experiments and verify new hypotheses.

D.4 Searching and Seizing Targets

In this scenario, we assume that there are targets in the environment that can be sensed by the robots in short distance. The task for a robot battalion is to search and seize such a target. Shown in Figure 3, the robots first spread out to seek the target based on their sensors. As some robots find the target, they will generate hormone-like signals to their neighbors. Such signals will be propagated throughout the robot network and a gradient field for the target will be created. All robots will probabilistically follow the gradient and eventually surround the target. The location of the target may be static (such as a geographic location) or dynamic (such as an enemy vehicle). Since robot's actions are stochastic, they will wander in the field to ensure that all the targets will be found and tracked. To investigate this possibility, we introduced a source of target signal in the middle of the DHM_0 simulation environment. This signal source continuously generates activator hormone signals into its surrounding space and creates a hormone field to attract nearby robots. As shown in Figure 6, the robots are initially concentrated at the left bottom corner of the environment. But once they start running, they first wander around as before but soon are attracted by the target signal field. As time goes by, a sufficient number of robots are attracted by the signal and form an enclosure around the target. Notice that not all robots are devoted to the same target, and there are sufficient robots still searching for targets in the open space. This automatic task balancing is due to the non-deterministic robot behavior function in DHM.

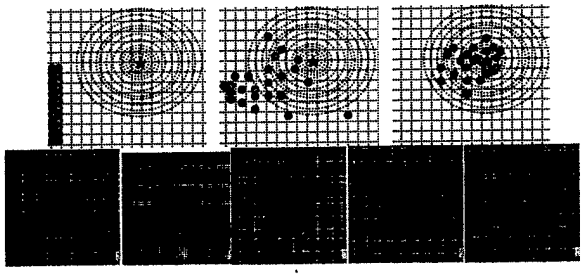


Figure 6: Simulation of searching and seizing a target.

In reality, the target signal field can be created in many different ways. One obvious implementation is to launch a signal source (invisible to the enemy) near the intended target. The other way is to use GPS (Global Position System) to specify a target location and simulate the hormone field in the robot's communication and sensing systems.

D.5 Spread and Scout in a Building

If the last scenario is about robots being controlled by attractive hormone signals, then the second scenario shows that robots can use repulsive hormone signals to spread out in a building to form a reconnaissance network. Shown in Figure 7, once robots enter the building, they are driven deeper into the building because of the repulsive hormones propagated in the network, and each robot has a higher probability of moving away from such hormone signals. The strength of the hormone can determine the density of the final network. To prevent robots from spreading too thin (or leaving the building), there is another attractive hormone active in the network. As we will see later, the balance between attractive and repulsive hormones ensures the formation of the desired global patterns.

As we see from the example of DHM_0, hormone controlled global behaviors are the result of balancing activator (attractive) and inhibitor (repulsive) hormones. The same mechanism can be used to allow a battalion of robots to automatically spread out in a building. To demonstrate this effect, we created some artificial walls in the DHM_0 simulation environment. As shown in Figure 4, there are three vertical zones separated by walls in the environment, and there are two "doors" in each wall that connect the zones.

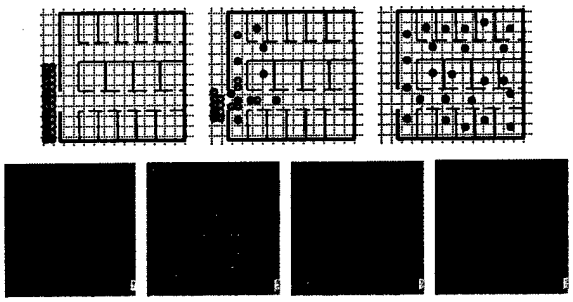


Figure 7: Spreading evenly in an indoor environment

Initially, all robots are in the left zone. Driven by their hormone signals, the robots "found" the doors in the walls and started to enter the middle zone. They then gradually spread out into the third zone. Notice that these robots automatically and evenly distributed themselves in these zones without explicitly being commanded to do so. The degree of spread can be controlled by the local hormone profiles of the robots. This is another demonstration that self-organizing behaviors can adapt to the tasks and environment at hand.

Once the robots formed an even distribution in the building, they can start the scouting task. Every robot can constantly monitor connectivity with its neighbor robots (to be described later). If the enemy destroys some robots in the network, then the neighboring robots will be triggered to generate an "alert" hormone, and the hormone will propagate to the entire network. That way, a human operator can easily monitor the entire building without worrying about where and how the robots are distributed in the building. If it is desirable to also know the location of the damage, the receiver of the alert hormone can trace back to the origin of the alert hormone using the propagation path of that hormone. A propagation path is a list of connectors through which the hormone is propagated.

D.6 Self-Repair Unexpected Damages

The third scenario demonstrates that a massive robot network controlled by digital hormones can self-repair unexpected damages to their organization. Unlike classical network protocols that cannot adapt to dynamic network topology, hormone-controlled robots can use the presence/absence of hormone signals to self-adjust their topology connections (via changing locations in this example) to self-heal the damage. Shown in Figure 8, when a bomb damages the six robots in the center of the network, the remaining robots will adjust their arrangement so that a new (thinner) network will be formed. Such self-repair behavior is again enabled by the digital hormone control protocol.

To demonstrate the DHM's ability for self-repair, we have manually removed 15% of the cells from the middle of a stable global pattern produced by HDM_0 (see the square holes in the pictures in Figure 8). We then let the cells continue to run for 50, 100, 500, and 1000 steps (see the pictures in the second row of Figure 8), and observed that the cells self-repaired the hole completely.

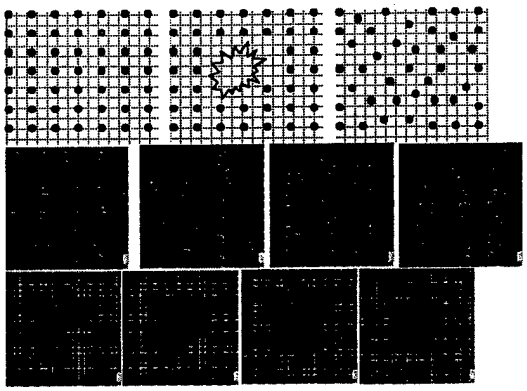


Figure 8: Self-Repairing Behavior of DHM

Furthermore, we have also manually created a hole in the robot enclosure, produced by the target seeking experiments, to simulate that the enemy is breaking the enclosure and attempt to escape (see the open space in the left bottom picture in Figure 8). After this severe damage, we let the robots continue to run for 50, 500, and 1,000 steps, and found that they repair the damage and complete the surrounding enclosure of the target again (see the pictures in the third row in Figure 8). From these two examples, we can see that as long as the local hormone profiles are in effect, the global patterns can repair themselves even after severe damage. The speed of repair is quite fast in simulation (1000 steps are about 30 seconds in real time). This provides the *resolute* ability for a massive robotic task force to accomplish its goals without any fear of sacrifice and destruction.

D.7 Surmount and Detour in Mission Execution

In the process of achieving their goals, massive robots will inevitably face barriers, obstacles, and pitfalls. As shown in the above pictures, some robots will run into barriers and traps. How do these robots get out of these difficulties? Using the hormone-inspired methods, we note that the robots closer to the target will generate stronger attractive hormone signals than those trapped at the dead end. Robots at the critical forking point between the correct path and the trapped path will be attracted more to the target direction. As more and more robots choose the correct path, the correct signals will become stronger and stronger, and eventually overcome the signals from the trapped robots. As a final result, the majority of robots will pursue the real target, and some of the trapped robots will be pulled out of the trap.

To demonstrate the ability of a massive robotic task force for surmounting difficulties and detouring barriers in operation, we have conducted the simulation in Figure 9, where the robots are to move from the top-left corner to the bottom-right corner in the environment. We placed an L-shaped barrier in the way to see if the robots would be trapped in the dead-end barrier. As shown in Figure 6, many robots first are trapped in the barrier on the way to the target position. Then, as some of the robots non-deterministically find a detour to bypass the barrier, their hormone signals get stronger and stronger and eventually attract many of the trapped robots, which cause them to move out of the trap. At the end, although there are still some robots trapped in the barrier, the majority robots have successfully passed the barrier and are on their way to the target position.

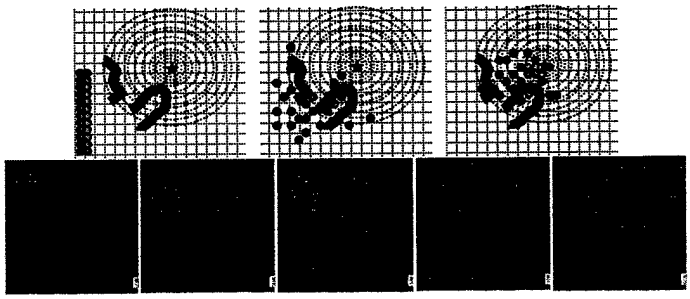


Figure 9: Bypassing a dead-end barrier in operation

In general, the situations of barriers and traps can be arbitrarily complex and impossible to predict. But with the ability to self-organize and self-repair, a massive robotic task force can find a way to bypass the barrier and traps. In the proposed work, we will conduct many more experiments and classify different situations based on the required tactics and strategies.

D.8 DHM by Wireless Communication

Although the above discussion assumes that the environment is responsible for computing the reaction and diffusion of hormones, a simple variation will allow it to be applied to a network of mobile robots that have short-range wireless communication. Different from pure biological experiments that require geographic proximity between cells, the DHM requires only topological proximity in which a neighbor robot is defined as one directly reachable in a single hop. To implement the hormone reaction and diffusion, each robot sends out "hormone signals" that carry the necessary information, such as hormone types, for the receivers to compute the reactions and diffusions of hormones. When a receiver receives such a signal, it uses the signal's strength (and possibly direction as well) to estimate the distance that this hormone signal has traveled, and convert that into a "hormone concentration" value based on the diffusion function of that hormone type. When the receiver receives multiple hormones that should react to each other, it uses the reaction function R to compute the reactions among related hormones. In this variation, space is continuous because a mobile robot can move in any direction it chooses. This is supported by DHM because Equations 2,3,5, and 6 are continuous. With this implementation, we believe that a hormone-controlled mobile robot network will be capable of self-organization and self-repair as we discussed above. The non-deterministic and hormone-guided actions will allow the robots to carry out such global behaviors without any centralized control.

E Theory and Language of Self-Reconfiguration

A theory of self-reconfiguration must answer the following challenging questions: (i) how to distribute complex global behaviors onto a large number of modules in a configuration, so that behaviors can be accomplished without any centralized controllers and avoiding any single-point failures; (ii) how to measure damages and detect the lack of capabilities in the current configuration for the task and environment in hand; (iii) how to select a configuration to restore the missing capabilities; (iv) how to morph into the selected configuration from a damaged configuration; and (v) how to select an appropriate behavior in the new configuration for the task. The solutions for these problems must be computationally feasible and enable implementation as a computational language.

Our current investigation for the theory of self-reconfiguration is primarily biologically inspired. We view a system configuration as a dynamic graph of reconfigurable nodes (cells). Each node has a set of reconfigurable *connectors* that can be dynamically controlled to change the connections with other nodes. The connectors can be physical, as in a self-reconfigurable modular robot, or logical, as in an agent/human organization. The self-reconfiguration process is triggered by the interaction between the system, the task, and the current environment, and it produces a new configuration along with a better solution for the task. This is a dynamic and learning process, for new configurations and solutions must be continuously proposed and executed for new tasks and new environments. Figure 10 shows such a model of self-reconfiguration in the graphical form.

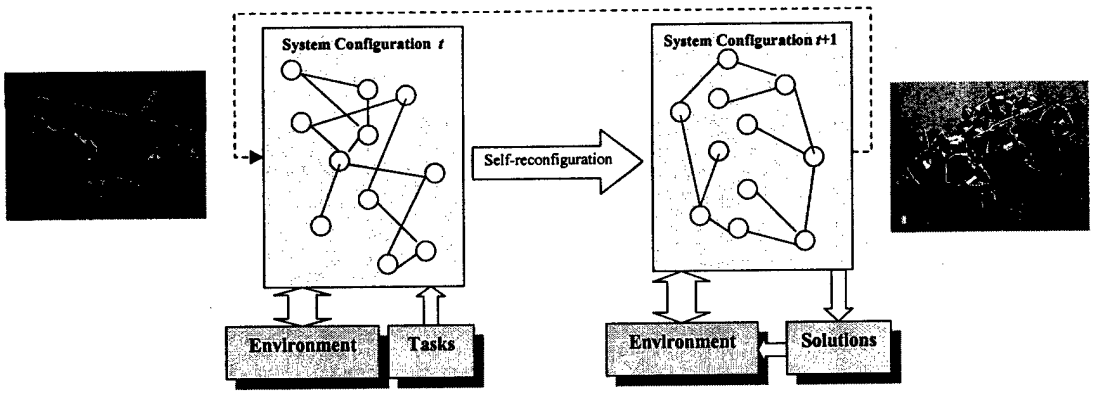


Figure 10: The proposed model for self-reconfiguration.

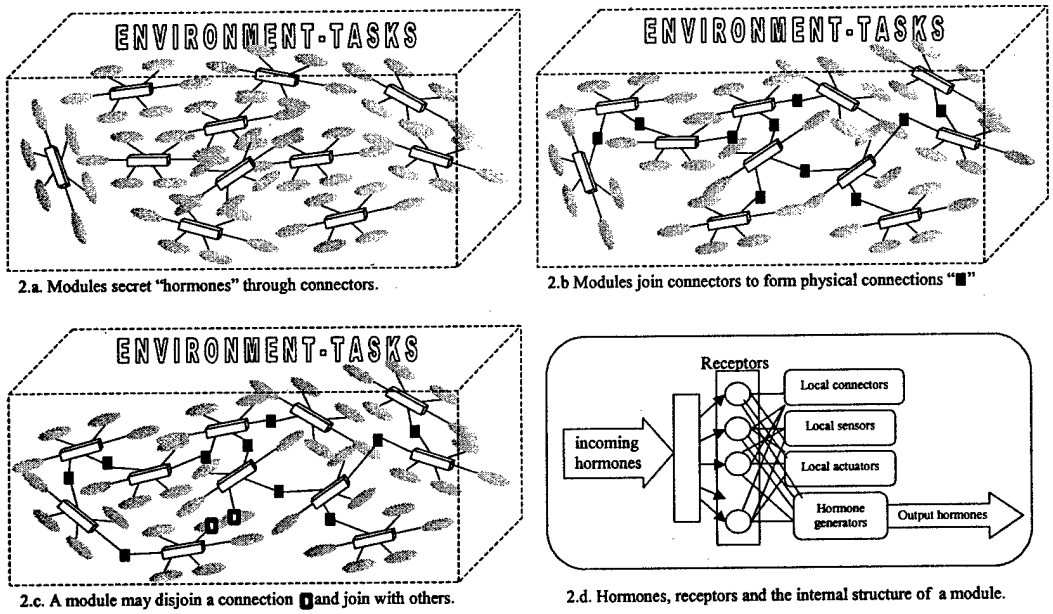


Figure 11: A hormone inspired approach to self-reconfiguration systems and robots

We mimic the hormonal and neurotransmitter mechanisms in biological systems for self-reconfiguration. We view a self-reconfiguration system as a collection of cell-like reconfigurable nodes (shown as the rectangle cylinders in Figure 11.a,b,c). Each node is an autonomous system with its own power, processors, actuators, sensors, and connectors (shown as the short lines from the modules in Figure 11.a,b,c) that can connect to the connectors of other nodes. A node in a configuration graph can secrete "hormone" signals (shown as the oval-shaped shadows) through its connectors. The secretion of hormone can be triggered by the existing hormones or by the environmental/task stimuli. Each node has a dynamic set of "receptor" processes (see Figure 11.d) for binding and processing hormones and for initiating local actions such as perception, locomotion, manipulation, navigation, making connections (small solid black rectangles in Figure 11.b and 11.c), disconnecting connections (small hollow rectangles in Figure 11.c), generating hormones, and propagating hormones. A computational, hormone-based language can be designed to dynamically change the receptors at a remote node, just as the distributed functional language (such as Erlang) used in the commercial cellular phone network. To *select*

an appropriate configuration for a given task and environment, one must have a deep understanding of the *interaction between configurations and environments*. Such interaction determines the impacts and requirements of different tactics and formations in different environments. We plan to study the characteristics of these interactions in many natural and artificial systems, and then abstract a common theme for our theory.

Our approach to achieving the challenging objectives is to (1) abstract a common model for self-reconfiguration; (2) combine knowledge of biological hormones/receptors with modern distributed functional programming languages to provide a foundation for building novel solutions for the challenging problems in self-reconfiguration; and (3) extend and improve our existing CONRO hardware reconfigurable robotic modules to demonstrate the new generation of self-reconfiguration robots.

E.1 A Theory of Self-Reconfiguration

To summarize the description above, a theory of self-reconfiguration theory must support at least the following requirements:

a. Execute a behavior of the current configuration in an environment;
b. Detect the impasses of a configuration for the current task and environment;
c. Select a better (or the best) configuration for the current task and environment;
d. Morph into the selected configuration from the current configuration;
e. Select an appropriate behavior in the new configuration for the task;
f. If the task has not achieved, then go to a, else accept new tasks.

Throughout the history, many theories and methodologies have been proposed and they are closely related to self-reconfiguration. For example, von Neumann's theory of self-reproducing automata [48], Turing's theory of learning machines [49], Genetic Algorithms [50-51], the PI's own theory for autonomous learning from the environment [52-54], and most recently IBM's Autonomic Computing (see http://www.research.ibm.com/autonomic). All this evidence suggests that there is probably an underlying theory for self-reconfiguration.

E.2 The Basic Model

The basic model for the proposed theory of self-reconfiguration is a dynamic network of self-reconfigurable modules. The modules can be software or physical, but they must be autonomous, self-sufficient, and have *connectors* that can dynamically join or disjoin with other modules in the network. For example, software modules in an integrated information system may be active databases and their connectors may be a virtual link on the network. Modules on an Ethernet network may be PCs, printers, or other physical or virtual devices and the connectors are their links to the Ethernet cable. Modules in a self-reconfigurable robot are physical robot-units that have independent power, processors, actuators, sensors, and connectors that can dynamically join or disjoin with connectors of other modules. In general, we denote a module m_i 's connectors as $(c_{i1}, \ldots, c_{in})$ and assume that every module has sensors to monitor the status of its connectors. Two modules m_i and m_j can form a link $l(c_{ix}, c_{jy})$ by joining their connectors c_{ix} and c_{jy} if the connectors are free. A module can disconnect an existing link by disjoining its connector from the link. When this happens, the module at the other end of the link will sense the change immediately. Modules can autonomously join or leave (e.g., when damaged) the network, and links can be dynamically formed or disconnected by the modules. Let M_t and L_t denote

respectively the modules and links existing in the network at time t, then a *configuration* at time t can be defined as:

$$C_t \equiv (M_t, L_t) \quad (1)$$

To ensure the distributed nature and self-reconfiguration, no single module is assumed to know the total number of modules and links in the network, and modules do not have unique global identities. This is very different from any classical computational models. After all, biological cells do not have unique global identities, and a true self-reconfigurable system must not have any single failure point such as a global identifier server. What each module can and should know is its local topological information, or its *topological type*, which reflects how a module is linked with neighbors in the current configuration. For example, a link in a CONRO robot is a pair of the female connector *b* (back) and one of the three male connectors, *f* (front), *l* (left), and *r* (right), so modules have 32 different types listed in Table 1 depending their links. For example, Type T0 has no links; types T1 through T6 have one link, types T7 through T18 have two links, types T19 through T28 have three links, and types T29 through T31 have four links. Some example types are illustrated in Figure 12. Type T1 has a link *l*(*b*,*f*), which means its *b* connector is connected to the *f* connector of another module. Type T16 has two links *l*(*b*,*f*) and *l*(*f*,*b*). Type 21 has three links *l*(*b*,*f*), *l*(*r*,*b*), and *l*(*l*,*b*). Semantically, a module's type reflects the module's position in the current configuration. For example, in a snake, T1 is the head, T16s are the body, and T2 is the tail. In a hexapod, T5s and T6s are the right and left legs respectively, T21 is the head, T29s are the spinal cord, and T19 is the tail.

TABLE 1: LOCAL TOPOLOGICAL TYPES OF MODULES

Connected to other modules: This Module					
b	*f*	*r*	*l*	Type
				T0
f				T1
	b			T2
		b		T3
			b	T4
l				T5
r				T6
	b	*b*		T7
		b	*b*	T8
	b		*b*	T9
l	*b*			T10
l		*b*		T11
l			*b*	T12
r	*b*			T13
r		*b*		T14
r			*b*	T15

This Module				
b	*f*	*r*	*l*	Type
f	*b*			T16
f		*b*		T17
f			*b*	T18
	b	*b*	*b*	T19
f	*b*	*b*		T20
f		*b*	*b*	T21
f	*b*		*b*	T22
l	*b*	*b*		T23
l		*b*	*b*	T24
l	*b*		*b*	T25
r	*b*	*b*		T26
r		*b*	*b*	T27
r	*b*		*b*	T28
f	*b*	*b*	*b*	T29
l	*b*	*b*	*b*	T30
r	*b*	*b*	*b*	T31

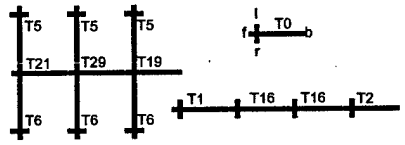


Figure 12: Topological types of CONRO modules in different configurations.

Given a configuration C_t, the current actions by all actuators in all modules constitute an "action snapshot" vector $A^{Ct}{}_i = <a1, ..., a_i, ..., a_c>$, where a_i is the current action by module m_i. We define a *behavior* of a configuration as a sequence of action snapshot vectors of the configuration:

$$B^{C_t}{}_i = < A^{C_t}{}_{i1}, A^{C_t}{}_{i2}, ..., A^{C_t}{}_{ij} > \quad (2)$$

The notion of time and synchronization is implied by the sequence of the action snapshots. One snapshot can trigger the next, or snapshots can have durations. A single configuration can have many different behaviors. For example, a snake configuration (where modules are linked into a chain) can behave in lateral undulation, concertina, sidewinder, or rectilinear (similar to a caterpillar).

An *environment* is defined as the current surrounding of the robot. It is reflected in the robot's perceived consequences of its actions. An *observation* is a vector containing the current values of all sensors in the modules. It is a snapshot of all sensors. A *task* can be defined as a targeted observation. For example, a climbing task can be defined as a goal observation in which the elevation sensor value is above a threshold.

The concepts of configuration, module type, behavior, tasks, and environment are all important for self-reconfiguration. The environment and the task are given, while the configuration and behavior can be changed. In a given environment, the same configuration can be good for one task but bad for the other. For example, in a slope environment, a ball configuration is good for going down, but bad for going up. For a given task, a self-reconfigurable robot can change to a new behavior in the current configuration, or change to a new configuration completely. The tradeoff is a very interesting problem and depends on the cost of changing and the urgency of the task. For example, to reach an object at the far end of a table, one can stand up and walk around the table (changing behaviors from sitting/reaching to walking/grasping), or simply grow a longer arm to reach the object (changing the configuration but using the same behavior). Although this choice is unnatural for humans, changing configuration is sometimes a much better, easier, and only choice for accomplishing certain tasks in certain environment. It is almost like treating your own body as a generator of new tools!

E.3 Hormones, Receptors, and Distributed Collaborations

Given the basic model, we propose a distributed functional language called DH2 to program and control all activities of a self-reconfigurable system. DH2 is an unique integration of the biological hormonal mechanism with distributed functional languages. The language has explicit constructs to create lightweight "receptor" processes on any module in a configuration and use hormone messages to propagate through the links to trigger modules to perform actions according to their positions or roles in the network. Many syntax notations of DH2 are similar to Ericsson's distributed functional language Erlang [1,8,9], but DH2 uses no global identifiers. Advanced from our earlier work in hormone-inspired control [2-5], DH2 can dynamically create receptors and simulate the bonding process between hormones and receptors, thus modules can differentiate and evolve themselves from homogeneous to heterogeneous, just as biological cells.

The basic objects (constants, compound terms, and variables) and functions in DH2 are similar to that of Erlang. But the most important notion in DH2 is a *receptor*, which is a process explicitly created with a built-in function

$$\text{receptor(Mod, [}P_H\text{(H), }P_T\text{(T), }P_S\text{(S), }P_V\text{(V)], [action1(Args1), ..., actionJ(ArgsJ)])} \quad (3)$$

which will execute at the module Mod, evaluating the expression as follows. The second element in this expression specifies a pattern matching process with matching functions as P_H, P_T, P_S, and P_V, and parameters as H (the expected hormone), T (the expected local topology), S (the expected local sensor values), and V (the expected local variable values). When the matching is successful, this receptor will bond to the hormone and execute the actions specified in the third element of the expression: action1(Args1), ..., actionJ(ArgsJ). The actions can be commands to local sensors, actuators, computations, and connectors, or can generate, propagate, and terminate

hormones and receptors. When bonded to a hormone, a receptor is consumed or terminated but its action may generate new receptors, e.g., a copy of itself.

A *hormone* is a message propagating in the links and to be bonded by receptors. It has the format

hormone(Content, Path) (4)

where Content can be any term in DH2, and Path is a list of connector names through which the hormone has been propagated. Hormones can be created by actions of receptors when triggered by other hormones or certain local events at the module. A newly created hormone has an empty Path. If a hormone is propagated through a link l(x,y), then its Path = [(x,y) | Path]. If a hormone received by a module does not bond to any receptors, it will be forward to all the output links from that module. Using a hormone, a module can remotely create a new receptor on any module in the network (similar to a remote procedure call). To do so, the caller puts a "receptor(_,_,_)" statement in the content of a hormone, and then send the hormone to the network. When this hormone is bonded to a receptor at a module, that receptor will create a new receptor specified in the content at the local module.

When a self-reconfigurable system is powered on, all modules will start running the basic module process, which runs forever and maintains a Mailbox for received hormones, a Buffer for hormones to be propagated, a LocalTimer, and a variable called ConnectorStatus for monitoring the status of each connector. The module process can be customized by a set of parameters during initialization.

```
module_process(ClockCycle) →
        loop () →
                for each hormone(_, [(x,y)|Path]) in Mailbox, do ConnectorStatus[y] = x,
                execute all receptors to process hormones in Mailbox and create and store hormones in Buffer,
                for each hormone(_, [(z,_)|Path]) in Buffer, do
                        remove the hormone from Buffer and send it through the connector z,
                        if send fails (i.e., there is no receiver at the connector z), ConnectorStatus[z] = 0,
                LocalTimer = mod(LocalTimer+1, ClockCycle);
        end.
```

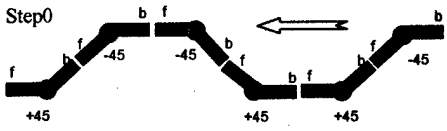
As we can see, the module_process constantly receives hormones, invokes local receptors, and performs local actions. It also monitors the status of each connector so that a module can dynamically adapt to changes in the network and discover its local topology in time. The discovered local topology information is stored in the variable ConnectorStatus. To see this, notice that initially all ConnectorStatus[_]=0. If a module's connector x is in a link l(x,y), then ConnectorStatus[x]=y when the module receives a hormone through x. Since every module attempts to receive hormones from its connectors in every cycle of the program, the ConnectorStatus will be updated correctly whenever there is a change in the local links.

To make a robot perform actions, all that is required is to create a set of proper receptors at the proper modules. The execution of a receptor is triggered and regulated by hormones. The first hormone may come from an external source (such as a human operator through wireless communication) or created by an existing receptor when triggered by an on-board sensor. Interestingly, the execution does not require any module (or the human operator) to know the complete information about the current configuration. All it needs is a single hormone. The

mechanism can execute a complex behavior distributed in many modules in the current configuration or change the current configuration to a new one.

E.4 Distributed Behavior Execution and Control

To illustrate the control of behaviors in a given configuration (item a in our theory), consider a CONRO snake robot and a caterpillar gait shown in Figure 13. To move forward, each module's pitch motor (DOF1) goes through a series of positions and the synchronized global effect of these local motions is a forward movement of the whole configuration (indicated by the arrow). The wavelength of the gait can be flexible, although this example uses a wavelength of four. To specify this gait, one can use a conventional gait control table [26] (also shown in Figure 13) where each row corresponds to the target DOF1 positions for all modules in the configuration during a step. Each column corresponds to the sequence of desired positions for one DOF1. The control starts out at the first step in the table, and then switches to the next step when all DOF1 have reached their target position in the current step. When the last step in the table is done, the control starts over again at step 0. As we can see, the six columns in the table correspond to the six module's DOF1 and the first row corresponds to Step 0.



Module ID for DOF1 actions

M1	M2	M3	M4	M5
+45°	-45°	-45°	+45°	+45°
-45°	-45°	+45°	+45°	-45°
-45°	+45°	+45°	-45°	-45°
+45°	+45°	-45°	-45°	+45°

Figure 13. A caterpillar movement and its control table

The problem of this conventional gait table method is that it is not designed to deal with the dynamic nature of robot configuration. Every time the configuration is changed, no matter how slight the modification, the control table must be rewritten. For example, if two snakes join together to become one, a new control table must be designed from scratch. A simple concatenation of the existing tables may not be appropriate because their steps may mismatch. Furthermore, when robots are moving on rough ground, actions on each DOF cannot be determined at the outset.

To represent this gait using the hormone idea, we notice that the gait has a "shifting" pattern among the actions performed by the modules. The action performed by a module m at step t is the action to be performed by the module (m-1) at step (t+1). Thus, instead of maintaining the entire control table, this gait is represented and distributed at each module as a sequence of motor actions (+45°, -45°, -45°, +45°). If a module is performing this caterpillar gait, it must select and execute one of these actions in a way that is synchronized and consistent with its neighbor module. To coordinate the actions among modules, a hormone can be used to propagate through the snake and allow each module to inform its immediate neighbor what action it has selected so the neighbor can select the appropriate action and continue the hormone propagation. To implement this idea, all we need is the receptors listed in Table 2 below. Specifically, the first four receptors will cause the head module (T1) to generate new hormones [(X,_),b] and send them to the rest of the snake. The creation is triggered based on the local variable LocalTimer and the head generates four hormones per ClockCycle. The last four receptors will cause all the body modules (T16) to execute the actions encoded (as A,B,C,D) in the hormone and propagate hormones to the next module. These modules will receive hormones through the front connector f and propagate hormones through the back connector b. When a hormone reaches the tail

module (T2), the propagation will stop because the tail module's back connector is not connected in any links. The speed of the caterpillar gait is determined by the value of ClockCycle. The smaller the value is, the more frequent the head generates new hormones, and thus faster the caterpillar moves. These receptors can be created easily using the DH2 syntax in Equation (3). For example, the first and the fifth receptor can be created respectively as follows:

```
receptor(*, [=(ModType,T1),=(LocalTimer,0)], [DOF1(+45),hormone([X,A),b]),keep_alive()] ).
receptor(*, [=(ModType,{T16,T2}), =(H,(X,A))], [DOF1(-45),hormone([X,B),b]),keep_alive()] ).
```

The symbol * means that the receptor is created at every module in the configuration, although in general one can specify the type of modules on which the receptor should be created. The action "keep_alive()" means to create a copy of the receptor itself before it is bonded and consumed by a hormone.

TABLE 2: THE RECEPTORS FOR THE CATERPILLAR GAIT

Parameter for Pattern Matching			Actions to Perform	
T (Module type)	V (Local Timer)	H	DOF1	New hormone
T1	0		+45	[(X, A), b]
T1	(1/4)ClockCycle		-45	[(X, B), b]
T1	(1/2)ClockCycle		-45	[(X, C), b]
T1	(3/4)ClockCycle		+45	[(X, D), b]
T16, T2		(X,A)	-45	[(X, B), b]
T16, T2		(X, B)	-45	[(X, C), b]
T16, T2		(X,C)	+45	[(X, D), b]
T16, T2		(X,D)	+45	[(X A), b]

TABLE 3: THE RECEPTORS FOR A LEGGED WALK

Parameters for Pattern Matching			Actions to Perform	
T (Module type)	V (Local Timer)	H	Motors	New hormone
T21, T17, T18	0		Straight	[(L,A),{l,r,b}]
T21, T17, T18	(1/2)ClcokCycle		Straight	[(L,B),{l,r,b}]
T29, T19, T26, T28		(L,A)	Straight	[(L,B),{l,r,b}]
T29, T19, T26, T28		(L,B)	Straight	[(L,A),{l,r,b}]
T5		(L,A)	Swing	
T6		(L,B)	Holding	

Table 3 lists the receptors needed for a legged robot (insects, spiders, or centipedes) to walk. We use the set notation {l,b,r} as a shorthand for the set of connectors to send the hormone, and the module types are illustrated in Figure 3. The action Straight means pitch=yaw=0. The action Swing means to lift a leg module, swing the leg forward, and then put the leg down on the ground. The action Holding means to hold a leg module on the ground while rotating the hip to compensate the swing actions of other legs. These receptors cause the left and right legs to alternate between Swing and Holding so the body will be balanced and move forward. The first two receptors allow a head module to generate hormones for the whole robot twice per CycleClock. The third and fourth receptors allow the module in the spinal cord to alternate the hormones (A and B) and propagate them to the rest of the body. The last two receptors control the Swing and Holding actions of the legs.

An earlier version of these and other gaits (such as a rolling track) have been implemented on the CONRO robots [2-5], and they are shown to be scalable (does not matter how long the snake is or how many legs a robot has), efficient (normally one hormone propagation for each gait step), robust (continues to walk if some legs are chop off), and support online bifurcation, unification, and behavior shifting. With the new theory, now the gaits can be dynamically created and switched. Please see these remarkable behaviors in video at our website http://www.isi.edu/robots/.

E.5 The Control of Self-Reconfiguration

The proposed theory can also control the process of changing one configuration to another (i.e., item d in our theory). In such a process, actions are typically organized in hierarchy and

execution is in a cascade fashion. For example, to reconfigure a legged robot into to a snake, the basic action sequence is as follows: the robot first connects its tail to a foot, and then disconnects the connected leg from the body so that the leg becomes a part of the tail. This compound action is repeated until all legs are "absorbed" or "assimilated" into the body. The lower-level actions that implement this compound action are those that enable the tail to find a foot, to align and dock with the foot, and then disconnect the leg from the body. A reverse of this compound action would cause a snake to "grow" a new leg by converting the tail module to a leg module.

This sequence can be implemented using hormones as follows. A module (any one) in the robot is first triggered to generate a hormone LTS (for changing Legs To Snake). This hormone is propagated to all modules, but only the foot modules (T5 or T6) have receptors to react the hormone and generate a "reply" hormone RCT (i.e., Request to Connect to the Tail). Every foot will periodically generate a RCT as long as it is still a foot. However, only the current tail module (T2) has a receptor for RCT, and upon receiving a RCT, this receptor will acknowledge the sender (using the path in the received RCT) with a TAR (Tail Accept Request) hormone, and terminate itself so that no more RCT will be bonded at this tail module. When receiving the TAR hormone, the selected foot module stops its RCT hormones, and generates a new hormone ALT (Assimilate Leg into Tail) to inform all the modules in the path to perform the actions of bending, aligning, and docking the tail to the foot. (The details of these lower-level actions are described elsewhere [13].) When these actions are accomplished, the new tail module will create a new receptor for accepting other RCT hormones, and another leg assimilation compound action will start. This process will repeat until all legs are assimilated, and it is independent to how many legs are in the current configuration. Thus if the reconfiguration sequence were stopped unexpectedly and prematurely, the process can resume itself correctly after the interruption is over.

To implement this sequence, four types of receptors must be in place to accept LTS, RCT, TAR, and ALT, respectively. These receptors can be created as follows:

receptor({T5,T6}, [in(ModType,{T5,T6}),=(H,LTS)], [periodical_hormone([RCT,_])]),
receptor({T2,T5,T6}, [=(ModType, T2),=(H,(RCT,Path))], [hormone([TAR,Path^{-1}])]),
receptor({T5,T6}, [in(ModType,{T5,T6}),=(H,(TAR,Path))], [stop_hormone(RCT), hormone(ALT,Path^{-1})]),
receptor(*, [=(H,ALT)], [assist_ tail_ dock_to_foot(), assist_ leg_ detach_from_body()]),

The first receptor allows a foot to react to LTS and create RCT. The second receptor allows a tail to react to RCT and reply with TAR. The third receptor allows a foot module to react TAR and acknowledge with a ALT. The fourth receptor will enable all modules in the spinal cord to help the compound action for leg assimilation.

Where should these receptors be created? The first and third receptors need only be at the foot modules (T5 and T6). The second receptor must exist on modules of type T2, T5, and T6, because T2 is the current tail and T5 and T6 may become the tail sometime during the process so we want this receptor alive in these modules. The last receptor must be at all modules because every module will become a part of the spinal cord and thus on the path between a leg and the current tail.

Note that in this example, we assume each leg is only one module long. If longer legs exist, the foot module would not be T5 or T6, and they must be identified with a more sophisticated

mechanism. One way to do so is to expand and create more topological types in Table 1 by considering not only the immediate neighbors but also those that are within a known distance.

E.6 Detecting Configuration Impasses

To detect the impasse of the current configuration for the current task and environment (item b in our theory), we will take an approach based on the theory of autonomous learning from the environment [52-54]. The main idea is a three step process: (1) before an action is executed, form a prediction about the expected effects based on the existing knowledge or previous experience; (2) observe the actual effects after the action and compare them with the expectation; (3) if the actual and predicted effects do not match, then examine this surprising situation to see if any progress towards the goal has been made. If an action fails to make the expected progress towards the goal many times, then conclude that the current configuration is not competent for the task in the current environment. For example, if a legged robot is walking on a icy and slippery slope, it would notice that its walking does not reduce the distance to the top. Similarly, when a robot is to fetch an object deep in a small crack, it may notice that its arm is too short to reach the object.

Now that a robot can change its configuration, there is always a tradeoff between changing the current configuration and keeping the configuration but trying a different behavior. This tradeoff has never been considered seriously before because most systems (animals or robots) could not change configurations. We hypothesize that when the cost of self-reconfiguration becomes low, in many situations changing configuration would be a much better choice than changing behaviors. In this project, we will consider this tradeoff based on two factors: the cost of changing and the urgency of the task. For example, when the time to get a too-far-to-reach object is short, we will prefer growing a longer arm than making a big detour.
To implement the three-step process, we represent a robot's expectation as a set of "prediction rules":

(the current situation), (configuration, behavior) → predicted consequences

These rules represent (O, A)→P, where O is an observation of the current environment, A is a generalized action (a configuration plus a behavior in our case), and P is the predicted effects. Notice that a prediction rule is different from the knowledge learned by a Reinforcement Learner (RL). There, a RL system learns a map (O) → A from situations to actions, but it cannot predict the expected effects of the action before execution. As a result, a RL will take much longer time to change an already failed behavior.

To correctly assess the current environmental and internal situation, many external, and internal sensors are needed. In a self-reconfigurable robot, the external sensors include vision, rangefinders (laser, sonar or infrared), tactile and proximity sensors (e.g., whiskers), position and orientation sensors (e.g., GPS), accelerometers, seismic and chemical sensors, and so on. The internal sensors include feedback from the motors (such as torque and force), monitoring devices for the connector status, odometers, and sensors for speed (e.g., optical encoders), and so on. One challenge is to integrate all these sensors when they are distributed among many modules, and fuse the data collected from different modules. For example, can an array of small images

collected by cameras on many modules be assembled into a large coherent picture? In this project, we will investigate solutions for such problems.

Ideally, the robot should learn the predication rules through experience. But as an intermediate step, we will first manually construct some rules for common situations in this project. To measure the progress towards a task goal, we will select a fixed sensor for each given task. For example, to reach an object, the robot will measure the progress by the distance to the target object. To climb trees, it will measure the progress by its elevation meter.

E.7 Selecting New Configuration and Behavior

Once the current configuration is deemed not capable of achieving the task, the next challenge (items c and e in our theory) is to select a new configuration/behavior appropriate for the current task and environment. The selection sequence is as follows:

The identified "obstacle features" ➔ (Configuration, Behavior).

When the system notices that the current configuration is not making any progress for the task in the current environment, it must identify a set of "obstacle features" in the environment with respect to the current configuration, and use these features to search for a better configuration. For example, if a spider-shaped robot encounters a passage narrower than its body, then the obstacle feature is the narrow width of the passage.

To use the obstacle features to find a configuration and a behavior, we assume that every pair of (configuration, behavior) is associated with a set of "required environmental features." For example, in a snake configuration, an undulation behavior requires a minimal space with of width w_1 to allow the maneuver of the undulation, while a sidewinder behavior will require a space with a much wider width $w_2 >> w_1$. Thus, a self-reconfigurable robot would maintain a set of associations such as the following:

[(configuration, behavior), (required environmental feature list)]

[(snake, undulation), (width=w1, height=h1, terrain=uneven, ……)]
[(snake, sidewinder), (width=w2, height=h2, terrain=sand, ……)]
……
[(insect, craw), (width=w3, height=h3, terrain=……)]
……
[(spider, climb), (width=w4, height=h4, terrain=……)]
……
[(octopus, grasp), (width=w5, height=h5, terrain=……)]

When a set of obstacle features are identified in the current environment, the robot will find a set of "required environmental features" that contains or tolerates the "obstacle features", and retrieve the associated configuration and behavior to fit the current environment/task. For example, after a spider notices the obstacle feature of the current environment is the width w_x, it

will select (snake, undulation) as the proper configuration because the associated feature width=w_1 is much smaller than w_x.

The advantages of this proposed technique is that the association memory can be learned through experience. However, such a memory is not easily distributed among many modules in the system. This is an open problem, and deserves careful analysis and more practical solutions.

E.8 Self-Healing by Morphallaxis

Although the CONRO robot has demonstrated some remarkable and unprecedented behaviors, a complete self-healing controller and a full-fledged demonstration has yet been done. In particular, we have to address the basic requirements (ii), (iii) and (iv) listed in the proposed theory. The details of these requirements are: (ii) how to measure damages and detect the lack of capabilities in the current configuration for the task and environment in hand; (iii) how to select a configuration to restore the missing capabilities; and (iv) how to morph into the selected configuration from a damaged configuration. This section presents a set of preliminary ideas for these requirements, and we plan to have a full investigation in future projects.

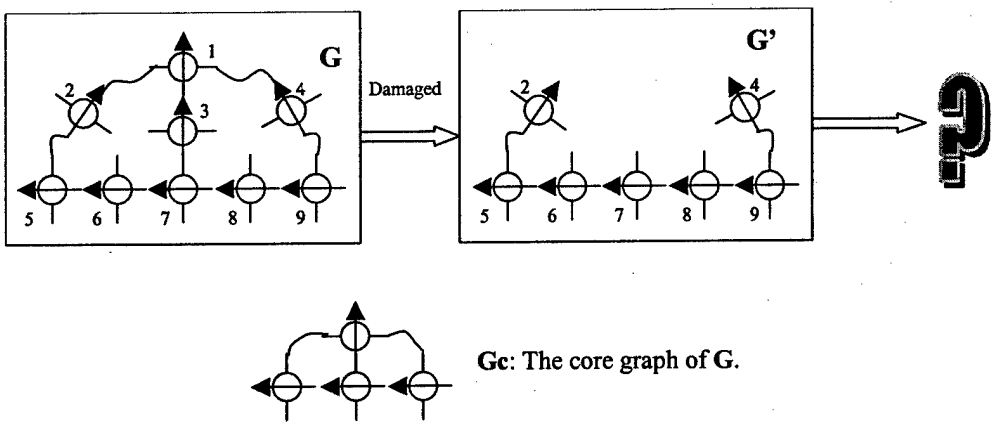


Figure 14: Essential and Trivial nodes/edges, core configuration, and morphallaxis.

Our approach to self-healing is based on morphallaxis and the hormone-inspired methods described above. Given a dynamic network configuration (see Equation 1), we first classify the nodes and edges in the configuration as *essential* and *trivial* depending on the task and environment in hand. For example, if the current task and environment cares not the size but the architecture of the configuration, then the length of limbs and torsos will be considered as "trivial" while any branching points such as shoulders and hips will be "essential". Thus any nodes/edges that are in the middle of a straight line will be trivial and any nodes that are located at branching points will be essential. In our self-healing robot CONRO, this means that the trivial modules are those that use the front and back connectors only (all other modules are essential). For example, in the configuration G shown in Figure 14, the nodes 2, 3, 4, 6, and 8 are trivial, while 1, 5, 7, 9 are essential.

Based on the above definitions, for any given configuration G, we can find a *core configuration* Gc by removing all the trivial nodes from G and shortening their connections. We say that two configurations G_1 and G_2 are *structurally equivalent,* denoted as $G_1 \equiv G_2$, if and only if their core

configurations are the same. For example, according to the above definitions, any two snakes are structurally equivalent, but a triangle is structurally different from a rectangle.

Thus, the question of self-healing is: Given a graph G and a damaged graph G', can it recover all the essential nodes/edges in G by reconfiguring the existing nodes and edges in G'? More specifically, given a configuration G and its damaged configuration G', morphallaxis is to find a *continuous transformation* from G' to a new configuration G^* that is structurally equivalent to G. This definition of morphallaxis says that if a configuration is damaged (with broken or losing nodes), then the graph can self-heal the damages, and create a new configuration structurally equivalent to the original one.

To solve this problem, we first construct from G a core graph Gc (this information is remembered before the damage is made), and then manipulate the damaged graph G' towards Gc by restoring each essential node and edge. If the core of G' is already structurally equivalent to Gc, then no healing is needed. If G' is smaller than Gc, then there would be no solution for self-healing because G' does not even have enough nodes to restore all the essential nodes in Gc. Otherwise, self-healing can proceed as follows. We first perform a transformation

$$G' \rightarrow Gf,$$

by reconfiguring G' into a new configuration Gf that is free from any essential nodes and edges. We remember the sequence S of how this transformation is done. We then perform another transformation

$$Gf \rightarrow G^*$$

by restoring all the essential nodes/edges in the reverse order of S so that $G^* \equiv Gc$. For example, to repair a damaged starfish that has only two legs left, the first transformation will reconfigure the damaged starfish into a snake (which has no essential connections), and then the second transformation will reconfigure the snake back into a new starfish with five legs. The new legs may be shorter than the original, but the new structure is equivalent to the original starfish.

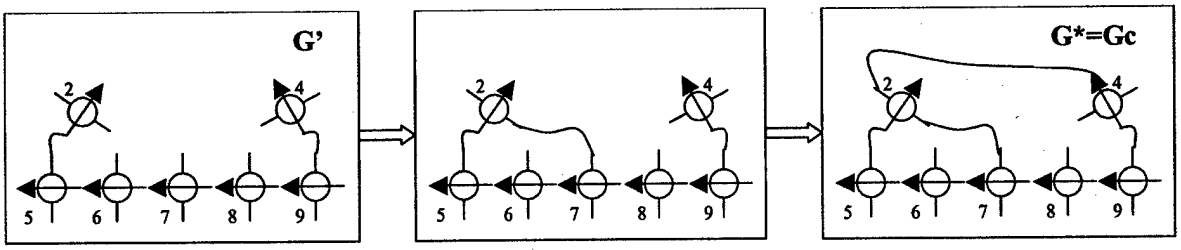


Figure 15: An example of self-healing to restore essential nodes and edges.

Figure 15 shows an example of how to self-heal the damaged graph in Figure 14. When comparing G' with Gc, the system notices that the essential node-1 is missing, and the essential node-7 has become trivial. So they select an existing node, say node-2, to restore the missing essential node-1. To make node-2 essential, the system first connects node-2 to node-7 (see the middle graph) to recover one missing essential edge. And then, it connects the other connector of node-2 to node-4 to recover the other missing essential edge (see the third graph). The self-

healing is completed because the restored graph G* is structurally equivalent to the core graph Gc of the original undamaged graph.

Note that the proposed procedures here still miss several important elements. First, a method is needed to distribute the core graph information onto all the nodes so that damages to any single nodes will result in the loss this critical information forever. Second, algorithms must be developed to select candidates for restoring the missing essential nodes. Third, a computational language must be developed to write these procedures and to prove formal properties (our current candidate is a distributed functional programming language similar to Erlang [8-9]. We plan to fully address these issues in this project.

E.9 The Control of Self-Healing Process

As one interesting aspect of self-healing, we can show that the hormone-inspired distributed control can be applied to the control of morphing actions. These actions are typically organized in a hierarchy and a single action in a higher-level can trigger a sequence of lower-level actions. To illustrate this, let us consider the example in Figure 16, where a CONRO robot is to reconfigure from a quadruped to a snake (or vice versa). The robot first connects its tail with one of the feet, and then disconnects the hip from the body so that the leg is "assimilated" into the tail. After this "leg-tail assimilation" action is performed four times, the result is a snake.

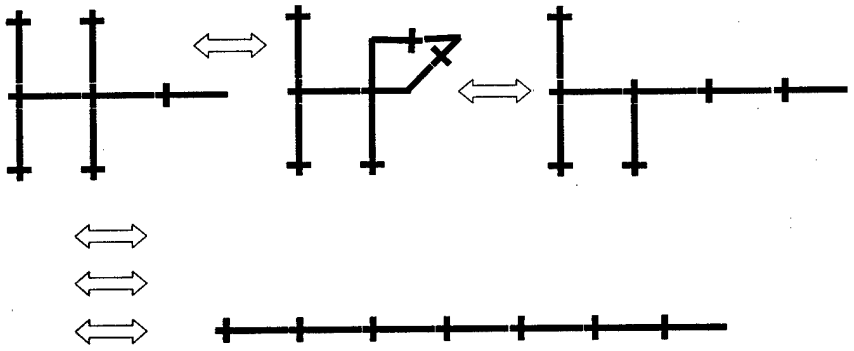


Figure 16: The control sequence from a quadruped to a snake and vice versa.

To control these actions, note that the high-level actions are a sequence of leg-tail assimilations, while the lower-level actions are those that enable the tail to find a foot, to align and dock with the foot, and then disconnect the leg from the body. Using hormones, the control of the reconfiguration can be accomplished as follows. One module in the robot first generates a hormone (called LTS for changing Legs To Snake). This LTS hormone is propagated to all modules, but only the foot modules (which are types T5 or T6) will react. Each foot module will start generating a new hormone RCT to Request to Connect to the Tail. Since there are four legs at this point, four RCT hormones are propagating in the system. Each RCT carries the information about its propagation path (i.e., a concatenation of all the sender connectors and receiver connectors through which the hormone has been propagated so any module along the path can trace back to the original sender). A RCT hormone will trigger the tail module (type T2) to do two things: inhibit its receptor for accepting any other RCT hormones, and acknowledge the sender (using the path information in the received RCT) with a TAR (Tail Accept Request) hormone. Upon receiving the TAR hormone, the selected foot module first terminates its generation of RCT, and then generates a new hormone ALT (Assimilate Leg into Tail) to inform

all the modules in the path to perform the lower-actions of bending, aligning, and docking the tail to the foot. The details of these lower-level actions are described in [102]. When these actions are terminated, the new tail module will activate its receptor for accepting other RCT hormones, and another leg assimilation process will be performed. This procedure will be repeated until all legs are assimilated, regardless of how many legs are to be assimilated. In Table 4, we list one possible sequence of hormone activities for assimilating four legs shown in Figure 16.

TABLE 4: THE HORMONE ACTIVITIES FOR CASCADE ACTIONS

Hormones	**Actions**
LTS	Start the reconfiguration
RCT_1, RCT_2, RCT_3, RCT_4	Legs are activated to generate RCTs
TAR, RCT_2, RCT_3, RCT_4	The tail accepts a RCT, and leg1 stops RCT_1
ALT, RCT_2, RCT_3, RCT_4	The tail and leg1 perform the assimilation process
TAR, RCT_2, RCT_4	The new tail accepts a RCT, and leg3 stops RCT_3
ALT, RCT_2, RCT_4	The tail and leg3 perform the assimilation process
TAR, RCT_2	The new tail accepts a RCT, and leg4 stops RCT_4
ALT, RCT_2	The tail and leg4 perform the assimilation process
TAR	The new tail accepts a RCT, and leg2 stops RCT_2
ALT	The tail and leg2 perform the assimilation process
$\varnothing$	No more RCT, and end the reconfiguration

Although the above descriptions have only scratched the surface of a complete self-healing theory, we are confident that they have provided sufficient evidence for launching a new and larger scaled investigation in the subject. In software, we will develop a new distributed functional programming language to implement the theory outlined above. In hardware, we will integrate the leading technologies in the field of self-healing robotics, exploit the state of the art engineering methods, and improve the capabilities of our existing CONRO modules. The existing modules already have the basic required devices on board, and we will improve their scalability, reliability, flexibility, and robustness. We plan to make them water prove and extend their locomotion to running, slithering, swimming, climbing, digging, and grasping. We will make the new modules capable of automatic selecting and executing the proper configuration based on the given task and environment. They will also be able to perform self-healing by discarding the damaged modules and reorganizing the existing module connections as we outlined here.

F Personnel Supported

Professor Wei-Min Shen (estimated 15% of his effort has been supported by this project).
Dr. Behnam Salemi (estimated 50% of his effort has been supported by this project. He has completed his PhD thesis in November, 2003.)

G Publications

Shen, W.-M., A. Galystan, P. Will, C.-M. Chuong, Self-organization and Distributed Control for Massive Robot Swarms, *Autonomous Robots*, 2004 (in press).

Salemi, B., P. Will, and W.-M. Shen, Distributed Task Negotiation in Modular Robots, Special Issue on "Modular Robotics", *Journal of the Robotics Society of Japan* (RSJ), 21(8)32-39, 2003.

Shen, W.-M., B. Salemi, and P. Will, Hormone-Inspired Adaptive Communication and Distributed Control for CONRO Self-Reconfigurable Robots, *IEEE Transactions on Robotics and Automation*, 18(5), October, 2002

Shen, W.-M., Self-Organization through Digital Hormones (invited), *IEEE Intelligent Systems*, 81-83, 8/2003.

Behnam Salemi, Peter Will, Wei-Min Shen, Distributed Task Negotiation in Self-Reconfigurable Robots, International Conference on Intelligent Robots and Systems. Las Vegas, October 2003.

Shen, W.-M., C.-M. Choung, P. Will, *Simulating Self-Organization for Multi-Robot Systems*, International Conference on Intelligent and Robotic Systems, Switzerland, 2002.

Shen, W.-M. and B. Salemi, Distributed and Dynamic Task Reallocations in Robot Organization, IEEE Conference on Robotics and Automation, Washington DC, 2002.

Shen, W.-M., C.M. Choung, P. Will. *The Digital Hormone Model for Self-Organization.* International Conference on Simulation of Adaptive Behaviors (From Animals to Animats 7). 2002. Edinburgh, Scotland.

Shen, W.-M., C.M. Choung, P. Will. *The Digital Hormone Model for Self-Organization.* The 8th International Conference on Artificial Life. 2002. Sidney, Australia.

H Interactions and Transitions

1. Naval Research Laboratory, Invited AI Seminar on self-reconfigurable robots, Washington, DC, June 2002.
2. UCLA CS Seminar, Self-Reconfigurable Robots and Digital Hormones, Los Angeles, January 2002.
3. NASA Workshop on Human and Robotic Space Exploration, NASA Langley Research Center, November 2001.
4. NASA Ames Research Center, Hormone-Inspired Control for Self-Reconfigurable Robots, September 2001.

5. Invited to Australian Center for Field Robotics Seminar, University of Sydney, Self-Reconfigurable Robots, 7/24/2003
6. Invited to UC San Diego AI Seminar, Self-Reconfigurable Robots and Digital Hormones 2/22/2003.

I New Discoveries, Inventions, or Patent Disclosures

US Patent Award #6636781: Distributed Control and Coordination of Autonomous Agents in a Dynamic, Reconfigurable Systems, 2003.

J Honors and Awards

Dr. Wei-Min Shen has received a 2003 Phi Kappa Phi Faculty Recognition Award from the University of Southern California.