Heterogeneity in the Coevolved Behaviors of Mobile Robots: The Emergence of Specialists

Mitchell A. Potter,[1] **Lisa A. Meeden**[2] **and Alan C. Schultz**[1]
[1] Navy Center for Applied Research in Artificial Intelligence
Naval Research Laboratory, Washington, DC 20375 USA
{mpotter,schultz}@aic.nrl.navy.mil
[2] Computer Science Department, Swarthmore College
Swarthmore, PA 19081 USA
meeden@cs.swarthmore.edu

Abstract

Many mobile robot tasks can be most efficiently solved when a group of robots is utilized. The type of organization, and the level of coordination and communication within a team of robots affects the type of tasks that can be solved. This paper examines the tradeoff of homogeneity versus heterogeneity in the control systems by allowing a team of robots to coevolve their high-level controllers given different levels of difficulty of the task. Our hypothesis is that simply increasing the difficulty of a task is not enough to induce a team of robots to create specialists. The key factor is not difficulty per se, but the number of skill sets necessary to successfully solve the task. As the number of skills needed increases, the more beneficial and necessary heterogeneity becomes. We demonstrate this in the task domain of herding, where one or more robots must herd another robot into a confined space.

1 Introduction

Many mobile robot tasks can be more efficiently solved when a group of robots is utilized. Some tasks cannot be solved at all without multiple robots. The type of organization, and the level of coordination and communication within a team of robots affects the type of tasks that can be solved.

In *swarm* approaches, (usually large) groups of robots execute the same simple strategies with no explicit communication. Complex group behaviors emerge from the simple interactions among the robots. Examples of this include flocking behaviors. In *cooperative* approaches, (usually smaller) groups of robots can have different strategies, allowing some of the robots to become specialists in solving parts of the task—that is, robots can assume roles. Recently, the term *collaborative* has been used to indicate cooperative approaches where the robots explicitly communicate their intent to one another. One important issue in multi-agent robotics is to understand when a particular approach is appropriate for a given task, that is to understand the relative power of each approach.

This paper will examine the tradeoff of homogeneity versus heterogeneity in the control systems by allowing a team of robots to coevolve their high-level controllers given different levels of difficulty of the task. In the homogeneous case, we will restrict the robots to using the same control structure, i.e. only one high-level controller is evolved, which all robots will use. In the heterogeneous case, robots will be allowed to coevolve separate high-level controllers, thus enabling the emergence of specialists. Our hypothesis is that simply increasing the difficulty of a task is not enough to induce a team of robots to create specialists. The key factor is not difficulty per se, but the number of skill sets necessary to successfully solve the task. As the number of skills needed increases, the more beneficial and necessary heterogeneity becomes.

Experiments were conducted within a simulation model of the task domain. The task chosen for these experiments is *herding*, where a group of robots must force another robot into a confined space. The robots are Nomad 200s which are modeled using the TeamBots system [Balch, 1998b]. We show that simply increasing the difficulty of the task alone does not necessarily require heterogeneous control systems, but that introducing a predator into the environment induces the evolution of specialists for defending against the predator.

In the next section we will describe related work. In Section 3, we will describe the task domain and how the complexity of the task can affect coevolved behaviors. The use of a neural network as a high-level controller, and the evolutionary algorithm used for learning will be described in Section 4. We will then describe the experimental methodology and the results in Section 5, followed by our conclusions and a description of future work.

2 Related Work

Evolution of robotic shepherding behaviors was first described in [Schultz *et al.*, 1996], although only one shepherd was involved, and therefore multi-robot coordination was not required. Much has been written about evolution (and co-

evolution) of robot behaviors (see [Mataric and Cliff, 1996; Meeden and Kumar, 1998] for an overview). Several articles have attempted to lay out taxonomies and general issues in multi-robot coordination [Dudek *et al.*, 1993; Cao *et al.*, 1997].

A number of researchers have explored heterogeneity at the hardware level by equipping members of a robot team with different sets of sensors or effectors [Cao *et al.*, 1997; Parker, 1999]. Other researchers have utilized teams of similar agents, and have instead explored heterogeneity at the behavior level [Balch, 1998a; Bongard, 2000; Good, 2000]. Our research takes the latter approach in that each herding agent has the same physical capabilities, but can develop unique control strategies through coevolution.

It remains an open question as to what kinds of tasks warrant a heterogeneous approach. Homogeneous teams have one clear advantage—there is built-in redundancy. If a team member fails for any reason, the rest of the team can still go on and be successful. However, as a task increases in difficulty, division of labor or specialization may become essential for success.

In doing reinforcement learning studies, Balch found that diversity within a team is not always desirable [Balch, 1998a]. Certain kinds of tasks, such as foraging, were solved more easily with a homogeneous approach. Balch speculated that any domain in which an individual agent could reasonably perform the task alone, is well suited for a homogeneous approach. Other domains, such as soccer which seems to require a variety of agent types, were more easily solved with a heterogeneous approach.

One result which is somewhat inconsistent with Balch's conclusions, is Luke's experiences developing a genetic programming based softbot soccer team for RoboCup97 [Luke, 1998]. The main goal of his work was to produce the best team possible in a limited amount of time. Luke developed both homogeneous and heterogeneous teams. However, the heterogeneous teams had a larger search space and thus converged much more slowly. By competition time, the homogeneous teams had the advantage and were entered. Luke predicts that given enough evolution time though, the heterogeneous teams might ultimately win out.

Balch has suggested that tasks that cannot be reasonably solved by a single agent should lead to heterogeneity. Bongard focuses more on the domain rather than the agent and argues that decomposable domains should lend themselves more readily to heterogeneity [Bongard, 2000]. Like Balch, he found that a homogeneous team was more successful at foraging than a heterogeneous one. Yet, it is not clear that this supports his hypothesis. Foraging can be easily decomposed as Balch did in his hand-coded control cases. For example, the domain can be decomposed into separate territories or the targets can be decomposed into particular types.

Our hypothesis is that the need for heterogeneity depends on the number of skill sets required by the domain. The herding domain is a good environment for testing this hypothesis as well as those of Balch and Bongard. We can increase the difficulty of the task along a number of dimensions without requiring new skill sets. We can also increase the number of skill sets required by adding a predator to the environment.

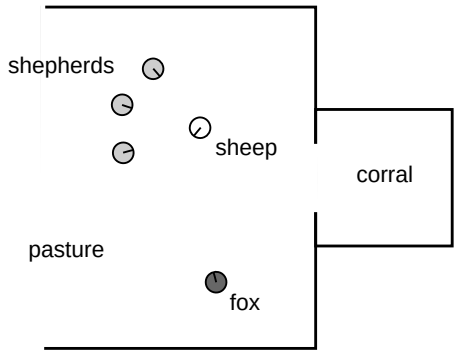

Figure 1: Herding domain

3 Task Domain and Complexity

The herding environment shown in Figure 1 consists of a 37×37 foot pasture that is fenced on three sides, with a smaller enclosed corral on the right. The herding task requires that a group of robots (the *shepherds*) force another robot (the *sheep*) into the corral. The sheep does not want to enter the corral, but instead wants to escape through the unfenced side of the pasture. To further complicate matters, in some experiments there is a predator robot (the *fox*) that attempts to kill the sheep by approaching within a certain distance.

The sheep's behavior is a fixed strategy that causes the sheep to avoid the approach or contact of other robots and obstacles, with additional drives to avoid the corral and to seek escape through the unfenced side of the pasture. The fox's behavior is also fixed, and causes it to attempt to approach the sheep while avoiding the shepherds and other obstacles. If the fox is able to get within a certain distance of the sheep, the sheep dies and the trial is over. The strategies of the shepherds are implemented via high-level neural network controllers which are evolved as described in Section 4.

The complexity of this domain can be controlled along several dimensions. For example, the degree to which the sheep avoids the corral and seeks an escape from the fenced pasture can be increased, the predatory fox can be included, the radius around the sheep in which the fox kills the sheep can be increased, and we can increase the number of sheep. Given the speed and turning rates of the sheep and shepherds, a single shepherd alone can force the sheep into the corral if the sheep only avoids obstacles (including the shepherds). However, if the sheep aggressively avoids the corral and seeks escape then a minimum of two shepherds are required to accomplish this task. By introducing a fox into the environment, a minimum of three shepherds is required.

As a performance task, success is measured by the ability of the shepherds to get the sheep into the corral. The task has failed if the sheep escapes from the pasture, is killed by the fox, or if a time limit is exceeded. The learning task is for the shepherds to evolve neural controllers that allow them to succeed in the performance task. The fitness measure as used by the evolutionary learning method is given in Section 4.

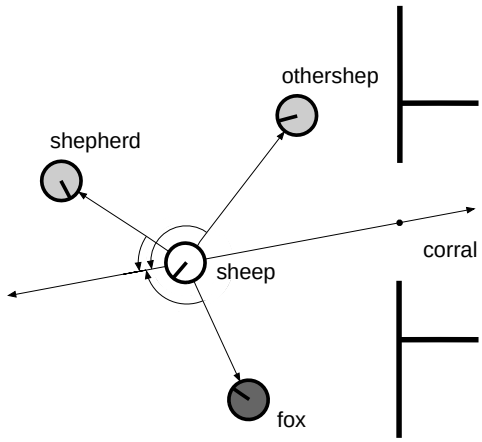

Figure 2: Derivation of sensors used by the neural controller

4 Evolution of Neural Controllers

The high-level controller is implemented with a feed-forward neural network with one hidden layer. The weights of the neural network are evolved using a particular evolutionary algorithm known as an *Evolution Strategy* [Rechenberg, 1964]. As will be seen in a moment, the strategy being evolved represents a high-level behavior, specifically, the neural network generates a goal point at each time step to which the robot will navigate. The lower-level behaviors such as collision avoidance and local navigation are built into the behavior and are not learned.

4.1 Sensors and Actions

Each shepherd is controlled by a neural network that maps its current sensors to a new position to be obtained by the shepherd. This mapping occurs at a 10 Hz rate. Each shepherd has vision sensors that represent the range and bearing to other agents in the environment. In particular, a shepherd can detect the sheep, the closest other shepherd, assuming there is at least one additional shepherd in the environment, and the fox, if present.

These shepherd-based sensor values are translated to ranges and bearing egocentric to the sheep as illustrated in Figure 2. The bearings are relative to the sheep's angle to the corral. Specifically, a polar coordinate system is used in which the pole is centered on the sheep and the polar axis passes through a point at the center of the opening of the corral. All angular measurements are relative to this polar axis. An agent's position is then defined as $(r, \pi - \theta)$, that is, the range is the length of the radius vector from the sheep to the agent, and the bearing is the supplement of the polar angle of the radius vector. The supplement of the polar angle is used so that if an agent is directly behind the sheep with respect to the corral, its bearing will be 0 degrees.

The input to the neural controller includes the following:

1. *shepherd_b:* The bearing from the sheep to the shepherd under control. Present in all experiments.
2. *shepherd_r:* The range from sheep to the shepherd under control. Present in all experiments.
3. *othershep_b:* The bearing from the the sheep to the closest other shepherd. Present in all experiments with two or more shepherds.
4. *othershep_r:* The range from the sheep to the closest other shepherd. Present in all experiments with two or more shepherds.
5. *fox_b:* The bearing from the sheep to the fox. Present in experiments with a predator.
6. *fox_r:* The range from the sheep to the fox. Present in experiments with a predator.

In addition to the vision sensors used by the neural controller, the shepherd also has sonar sensors which are used by the fixed, lower-level behaviors to avoid collisions with other objects, although the distance in which the shepherd will avoid the sheep is less than the distance at which the sheep will avoid the shepherd, allowing the shepherd to be able to herd the sheep.

The output of the neural controller is the range and bearing to the new position to which the robot is to navigate in the same coordinate system as described above. These values are translated to a coordinate system egocentric to the shepherd under control, and input into a motor schema (see [Arkin, 1989]) to produce a linear attraction to the target position. This motor schema is combined with motor schema for obstacle avoidance and stochastic noise into an assemblage which controls the robot via turn and translation rate commands.

4.2 Neural Network Controller

We use a simple two-layer feed-forward neural network topology as shown in Figure 3. Nodes are implemented using a standard sigmoid centered at 0 as follows:

$$f(z_i) = \frac{1}{1 + \exp(-z_i)} - 0.5 \tag{1}$$

$$z_i = \sum_j w_{ij} o_j, \tag{2}$$

where o_j is the output of node j, and w_{ij} is the weight on the connection from node j to node i. All input nodes have weighted connections to all hidden nodes, and all hidden nodes have weighted connections to all output nodes. The network shown is the most complex case, which is used in the experiments with two or more shepherds and a predator. In the experiments with two shepherds and no predator, the number of input nodes is reduced to 5, and in the experiment with one shepherd and no predator, the network is further reduced to 3 input nodes and 3 hidden nodes.

The network accepts real-valued inputs corresponding to the sensors described in the previous section. An additional input is clamped to the value 1.0 in order to provide a learnable bias for each node in the hidden and output layers. The target range output is converted to a value between 0.0 and 10.0 units in simulation, which corresponds to the full 37 foot width of the pasture in the real world, and the target bearing output is converted to a value in the range $(-\pi, \pi)$.

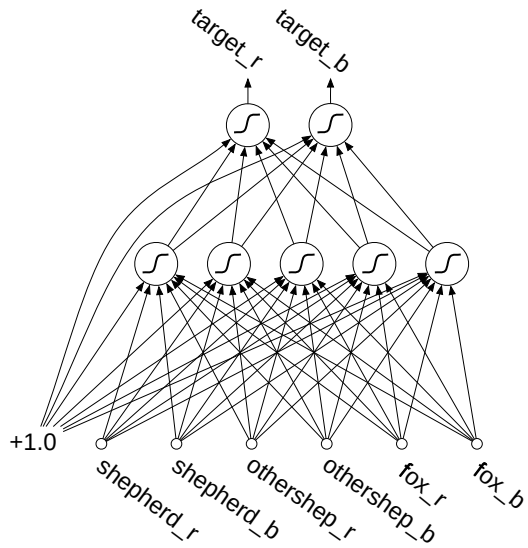

Figure 3: Neural network used as high-level controller

4.3 Evolution of Controllers

The evolutionary algorithm we use to evolve the connection weights for the neural controllers is a $(\mu + \lambda)$ evolution strategy (ES) as described by Bäck and Schwefel [1993], with $\mu = 10$ and $\lambda = 100$. Each individual consists of a real-valued vector of connection weights and a companion vector of standard deviations used by the mutation operator as described below. This class of evolutionary algorithm was introduced in Germany by Rechenberg [1964] for numerical optimization, and variants such as the $(\mu + \lambda)$–ES, originally developed by Schwefel [1981], are a good choice when the problem solution is naturally represented as a vector of real-valued numbers, as is the case when evolving neural network connection weights.

A $(10 + 100)$–ES begins with a population of 10 individuals, and generates 100 children by selecting uniformly from this population and mutating each child. The 100 children and their 10 parents are then evaluated by applying each of them in turn to the target problem, and the 10 individuals with the highest fitness become the next generation of parents. Mutation of an individual $\vec{x}$ with k genes and a companion vector of standard deviations $\vec{\sigma}$ consists of tweaking each gene x_i, for $i = 1, \ldots, k$, according to a normal distribution with mean zero and standard deviation σ_i as follows:

$$x_i' = x_i + \mathrm{N}(0, \sigma_i). \tag{3}$$

Furthermore, the standard deviations $\vec{\sigma}$ are themselves adapted as follows:

$$\sigma_i' = \sigma_i \exp(\tau' \mathrm{N}(0,1), \tau \mathrm{N}_i(0,1)), \tag{4}$$

where

$$\tau = \frac{1}{\sqrt{2\sqrt{k}}} \tag{5}$$

$$\tau' = \frac{1}{\sqrt{2k}}. \tag{6}$$

The elements of $\vec{\sigma}$ are initialized to 1.0 and are restricted to the range $(0.01, 1.0)$. In practice, the standard deviations approach the lower limit of this range over time, which has an effect much like that of simulated annealing.

In the experiments where heterogeneous control systems are evolved, we use the architecture for coevolution developed by Potter and De Jong [2000]. This architecture models an ecosystem consisting of two or more species. As in nature, the species are genetically isolated—meaning that individuals only mate with other members of their species. Mating restrictions are enforced simply by evolving the species in separate populations. The species interact within the herding domain as described below and have a cooperative relationship.

To evaluate an individual from one of the coevolving species given heterogeneous control systems, we construct a neural control system using the individual's genes as connection weights, and assign the resulting control system to one of the shepherds. We then select the current best individual from each of the other species and similarly construct neural control systems from them for assignment to the other shepherds. The shepherds are then set to work on the herding task. Alternatively, we could organize the best shepherds into multiagent squads and assign each squad to a different neural control system. In the experiments where purely homogeneous control systems are evolved, the ecosystem consists of only one species. To evaluate one of these individuals, a neural control system is constructed from that individual's genes and assigned to *all* the shepherds, that is, each shepherd will be controlled by an identical neural network.

In this current study, evaluations are done in simulation. Each evaluation consists of 10 trials in which the shepherds herd the sheep until it is corralled, killed by the fox, escapes from the pasture, or 2.5 simulated minutes have expired. The cumulative distance of the sheep from the corral is measured at the rate of 10 Hz throughout the trial. If the trial ends early due to the sheep escaping or being killed, we continue to accumulate the full pasture-width distance until the clock expires. The fitness of an individual is taken to be the final cumulative distance averaged over the 10 trials. The worst possible fitness is 14,768, which would result if the sheep immediately initiated a turn towards the left side of the pasture and escaped without any interference from the shepherds. The best possible fitness is 976, which would result if the sheep set a course directly towards the corral at maximum speed. The ES will seek to minimize this measure.

5 Results

In order to test our hypothesis, we vary the complexity of the task in several dimensions using both homogeneous and heterogeneous multi-agent approaches.

We begin with the simplest case of one shepherd herding one passive sheep that just avoids obstacles. This is essentially a reimplementation (in simulation) of earlier work done by Schultz *et al.* [1996]. We repeat this earlier experiment to validate the design of our neural controller and evolutionary learning method. The results are shown in Figure 4 along with results from a second experiment in which the sheep

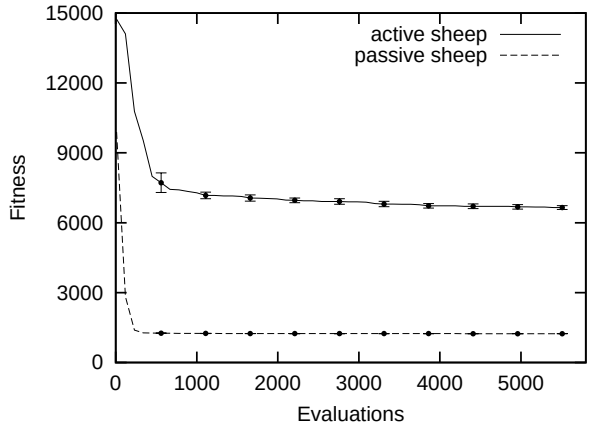

Figure 4: One shepherd herding one sheep

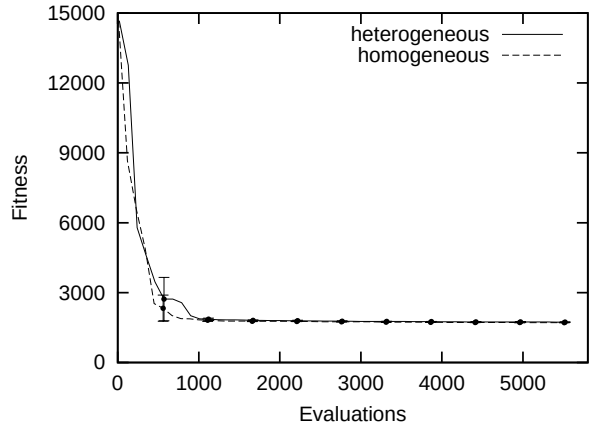

Figure 5: Two shepherds herding one sheep

actively seeks escape from the pasture and avoids the corral. The curves in the graph represent the mean fitness of the best individual seen so far averaged over 10 separate runs. Overlaid on the curves at increments of 5 generations (550 evaluations) are 95-percent confidence intervals on the mean. Although these control systems were evolved for 50 generations (5,500 evaluations), a controller with near optimal behavior on this simple task was easily evolved in as little as 2 generations. We also observed simulated robots solving this task using the best neural control system from the final generation of evolution, and it is clear that they are exhibiting behavior very much like the behavior evolved in the earlier work by Schultz *et al.*. Specifically, the robot positions itself directly behind the sheep with respect to the corral, and herds the sheep by moving towards it and triggering its obstacle avoidance behavior. The shepherd makes subtle swings from left to right to counter irregularities in the movement of the sheep. In contrast, the more complex case involving the sheep seeking freedom (hereby referred to as the *active* sheep) does not appear solvable with a single shepherd, as indicated by the learning curve flattening out at a very poor level of fitness.

The result of adding a second shepherd to the more difficult task of herding an active is shown in Figure 5. We compare both the evolution of two shepherds using homogeneous control systems, and two shepherds using heterogeneous control systems. Although it took slightly fewer evaluations to evolve good homogeneous control systems, adequate behavior was also evolved in the case of heterogeneous control. Due to the significantly smaller search space of homogeneous controllers, it is not surprising that when good homogeneous solutions exist it is easier to find them. When observing robots solving this task using the best neural control systems from the final generation of evolution, we see that in both cases (heterogeneous and homogeneous) the pair of robots assume positions behind the sheep, but slightly to the left and right. This counters the strong tendency of the sheep to slip by the shepherds and escape from the pasture. This is clearly a cooperative approach on the part of the shepherds, and it shows

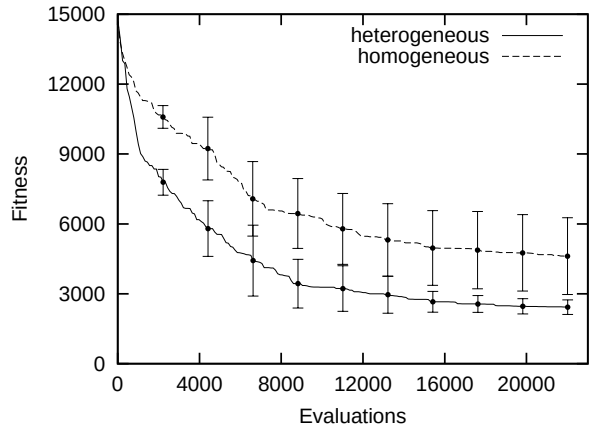

Figure 6: Three shepherds herding one sheep with fox

the need for cooperation is not alone sufficient to warrant the use of heterogeneous control.

Finally, we add a predator to the environment, in the form of a fox that seeks to kill the sheep. Now the shepherds require two skills—the ability to herd the sheep and the ability to keep the fox a safe distance away from the sheep. To encourage specialization, we initially position two shepherds behind the sheep, and a third shepherd is positioned between the sheep and the fox. As before, we compare the evolution of shepherds using homogeneous control systems with the evolution of shepherds using heterogeneous control systems. In the case of heterogeneous control, the two shepherds positioned behind the sheep are controlled by one neural network and the shepherd positioned closer to the fox is controlled by a second neural network. The results from this experiment are shown in Figure 6. This task is much more difficult than the previous ones, so we evolved these control systems for 200 generations (22,000 evaluations). The graph clearly shows the superiority of heterogeneous control on this task.

Control	*Shepherds*	*Sheep*	*Fox*	*Fitness*			*Success Ratio*
				Mean	*Minimum*	*Maximum*	
homogeneous	1	passive	no	1273.9 ± 2.7	1205.3	1356.0	1.000
homogeneous	1	active	no	7592.5 ± 127.3	5865.9	14548.7	0.000
homogeneous	2	active	no	1808.9 ± 60.4	1636.8	14621.8	0.996
heterogeneous	2	active	no	1865.5 ± 81.4	1645.7	14573.0	0.992
homogeneous	3	active	yes	4336.0 ± 372.8	1669.9	13966.6	0.758
heterogeneous	3	active	yes	3149.2 ± 305.5	1579.8	14153.8	0.874

Table 1: Comparison of the mean fitness and success rate of the best individual from each experiment, evaluated over an additional 500 trials

When we observe the heterogeneous control systems performing this task, we see that the shepherd that is initially positioned between the sheep and fox does indeed exhibit specialized blocking behavior, while learning to herd as well. It begins by moving towards the sheep, keeping its body between the sheep and fox. When it nears the sheep, it sometimes turns to face the fox and performs a quick deflection maneuver before it joins the other two shepherds begins herding. However, most of the time its blocking behavior is more subtle. Close observation reveals that the blocking shepherd maintains slightly more distance between itself and the sheep than the other two shepherds, which enables it to keep the fox at a safe distance from the sheep while still providing some help in herding. Without having to concern themselves as much with the fox, the other two shepherds are able to herd the sheep more directly towards the corral. The homogeneous control systems rely much more on herding the sheep away from the fox, which is not as effective as blocking because it gives the sheep more opportunities to escape from the pasture.

It should be noted that there is a difference in complexity between the heterogeneous and homogeneous control systems. Since the heterogeneous control system utilizes two separate neural networks—one for the pair of shepherds initially positioned behind the sheep, and one for the shepherd positioned between the sheep and fox—a total of 94 connection weights are being evolved, while only 47 connection weights are evolved for the single-network homogeneous control system. To verify that the observed difference in performance is not simply due to this difference in complexity, we evolved a homogeneous control system with a complexity on the order of the heterogeneous system by increasing the number of hidden units to 10, which produced a neural network with 92 connection weights. As expected, this homogeneous control system performed much more poorly than the homogeneous control system with only 5 hidden units. Specifically, averaged over 10 runs to 200 generations, the final mean fitness of the more complex homogeneous control system was 7,918, compared with an average fitness of 4,614 for the simpler homogeneous control system.

The results from the previous three graphs are summarized and further supported by Table 1. Here we take the single best individual from the 10 runs of each experiment and apply it to the herding task for an additional 500 trials. We report the mean fitness, along with 95-percent confidence intervals on this mean, the maximum and minimum fitness achieved, and the success ratio, that is, the percentage of trials in which the sheep was actually corralled. The table reinforces our earlier observation that a single shepherd is not capable of herding an active sheep into the corral. However, two cooperating shepherds are sufficient to accomplish this mission. Heterogeneous control systems are not an advantage here, in fact, they perform slightly worse, although a *t*-test on the means of the two-shepherd homogeneous and heterogeneous trials produced a *p*-value of 0.2729, indicating a lack of statistical significance in their difference. Only when the task requires multiple skills (e.g., herding the sheep and blocking the predator) does heterogeneous control perform better than homogeneous control, as indicated by the trials with three shepherds. A *t*-test on the means of the three-shepherd trials produced a *p*-value of 0.0000, clearly showing a statistically significant advantage to using heterogeneous control.

6 Conclusion

In this paper, we have tried to demonstrate that simply increasing the difficulty of a task is not enough to induce a team of robots to create specialists. The key factor is not difficulty per se, but the number of skill sets necessary to successfully solve the task. As the number of skills needed increases—in this study by adding the response to a predator—the more beneficial and necessary heterogeneity becomes.

Although heterogeneous control systems can promote better solutions for many tasks, there is a trade off. Learning (coevolving) a team of homogeneous agents can take much less time, since each evaluation of an individual in the population goes towards all individuals' progress and the search space is smaller. In a heterogeneous group, the available CPU time during evolution must be divided among the different skill sets.

Ongoing experiments are attempting to more generally determine the properties that dictate the type of approach that is appropriate. In addition, results will be duplicated on the physical Nomad 200 robots to show that the simulation results hold on the actual robots.

Acknowledgments

This work was supported by the Office of Naval Research under work request N0001401WX20073.

References

[Arkin, 1989] Ronald C. Arkin. Motor schema-based mobile robot navigation. *The International Journal of Robotics Research*, 8(4):92–112, 1989.

[Bäck and Schwefel, 1993] Thomas Bäck and Hans-Paul Schwefel. An overview of evolutionary algorithms for parameter optimization. *Evolutionary Computation*, 1(1):1–23, 1993.

[Balch, 1998a] Tucker Balch. *Behavioral Diversity in Learning Robot Teams*. PhD thesis, Georgia Institute of Technology, 1998.

[Balch, 1998b] Tucker Balch. Integrating robotics research with javabots. In *Working Notes of the AAAI-98 Spring Symposium, Stanford, CA*, 1998.

[Bongard, 2000] Josh C. Bongard. The legion system: A novel approach to evolving heterogeneity for collective problem solving. In *Genetic Programming: Third European Conference*, pages 25–37. Springer-Verlag, 2000.

[Cao *et al.*, 1997] Y. U. Cao, A. S. Fukunaga, and A. B. Kahng. Cooperative mobile robotics: Antecedents and directions. *Autonomous Robots*, 4:7–27, 1997.

[Dudek *et al.*, 1993] G. Dudek, M. Jenkin, E. Milios, and D. Wilkes. A taxonomy for swarm robots. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 441–447, 1993.

[Good, 2000] Benjamin McGee Good. Evolving multi-agent systems: Comparing existing approaches and suggesting new directions. Master's thesis, University of Sussex, 2000.

[Luke, 1998] Sean Luke. Genetic programming produced competitive soccer softbot teams for RoboCup97. In John Koza, editor, *Proceedings of the Third Annual Genetic Programming Conference*, pages 214–222. Morgan Kaufmann, 1998.

[Mataric and Cliff, 1996] M. Mataric and D. Cliff. Challenges in evolving controllers for physical robots. *Robotics and Autonomous Systems*, 19:67–83, 1996.

[Meeden and Kumar, 1998] Lisa A. Meeden and Deepak Kumar. Trends in evolutionary robotics. In L.C. Jain and T. Fukuda, editors, *Soft Computing for Intelligent Robotic Systems*, pages 215–233. Physica-Verlag, New York, NY, 1998.

[Parker, 1999] L. E. Parker. Adaptive heterogeneous multi-robot teams. *Neurocomputing*, 28:75–92, 1999.

[Potter and De Jong, 2000] Mitchell A. Potter and Kenneth A. De Jong. Cooperative coevolution: An architecture for evolving coadapted subcomponents. *Evolutionary Computation*, 8(1):1–29, 2000.

[Rechenberg, 1964] Ingo Rechenberg. Cybernetic solution path of an experimental problem. Library Translation 1122, August 1965. Farnborough Hants: Royal Aircraft Establishment. English translation of lecture given at the Annual Conference of the WGLR, Berlin, 1964.

[Schultz *et al.*, 1996] A. C. Schultz, J. J. Grefenstette, and W. Adams. Robo-shepherd: Learning complex robotic behaviors. In M. Jamshidi, F. Pin, and P. Dauchez, editors, *Robotics and Manufacturing: Recent Trends in Research and Applications, Volume 6*, pages 763–768. ASME Press, 1996.

[Schwefel, 1981] H.-P. Schwefel. *Numerical Optimizaion of Computer Models*. Chichester: Wiley, 1981.