The Combination of Competitive Evolution and an Interactive Environment

Julie Corder and Ross Messing

May 19, 2003

Abstract

Ecological psychology teaches us that intelligence exists in an environment. We explore the combination of competitive evolution, and an interactive environment. A population of network controller weights are evolved, and robots using those weights compete in a hockey-like game. The behavior produced is neither as consistent nor as complex as hoped for, and a variety of causes, and potential remedies, are discussed.

1 Introduction

In studying intelligence, the environment in which intelligence exists cannot be ignored. The ecological psychologist James Gibson emphasized the importance of the environment in the study of intelligent behavior, stating that understanding of one could not come without understanding of the other(Gibson, 1986). While ecological robotics, a sub-field of the robotics work of second generation cognitive science, is, like Gibson's ecological psychology, focused largely on specific models of visual perception, it too is grounded in a focus on the environment (Duchon, Warren, & Kaelbling, 1995). In the present experiment, we adopt an evolutionary robotics mindset, but take the environmental focus of ecological psychology to heart in our experimental design.

2 Related Work

Nolfi and Floreano have performed several experiments, which collectively encompass a superset of the present one. However, none of their work to date involves both competitive evolution, and environmental interaction. In one experiment, they had two species of robots competitively co-evolve, with one species, a predator, receiving more sensory data, and the other species, the prey, having more speed(Floreano & Nolfi, 1997). This experiment was used to investigate the advantages of co-evolution. While this experiment involves competitive evolution, the only interesting parts of the environment are the competing robots.

All other parts of the environment are just simple walls to bound the experiment, and no direct manipulation of the environment, even the other robot, occurs.

In another experiment, Nolfi created a trash-collecting environment, where a robot wanders through it's environment and collects trash using a gripper module, than deposits the trash outside it's environment (Nolfi, 1997). While the physical morphology of the robot remains constant, its control architecture's weights were evolved using a genetic algorithm, and various sorts of control architectures were compared. The environmental interaction component of this experiment is interesting, because real-world problems almost always involve environmental interaction and manipulation. However, as this experiment only involves one robot, it is necessarily devoid of competition.

Sims has created a system combining competitive evolution with environmental manipulation. In Sims's experiment, evolved simulated individuals from either one or two populations compete in a virtual world for control of an object(Sims, 1994). In Sims's experiment, both the control architectures and the virtual physical morphologies of the individuals are evolved, resulting in a wide variety of evolved individuals. Sims's work could easily be seen as some of the earliest and most revolutionary work in the simultaneous evolution of both physical morphology and control architecture. Sims's experiment, however, required supercomputing power unavailable, even now, to most researchers.

The present experiment attempts to create a framework to investigate competitive evolution in an interactive environment. This combines the features of Floreano's and Nolfi's experiments, without requiring a supercomputer to make the experiment computationally feasible, as Sims did.

3 Experiment

3.1 Simulation Environment

The environment in this experiment was a circular rink, with one goal area at each end, and a puck in the middle. At the beginning of each trial, a robot is placed between it's goal and the puck. The robot, it's opponent, the puck, and each goal are all given different colors, allowing the robot's sensory apparatus to distinguish easily between them. We varied our task from both the predator-prey task, and the trash-collection task, while maintaining their degree of simplicity. This was done in order to ensure that both our findings, and those of Floreano and Nolfi, were generalizable across several problem domains.

Evolved individuals each had the same morphology, consisting of a virtual color-blob detector designed to mimic the functionality of a real robot vision apparatus, a sonar array, and a rectangular body. The individuals were to push the puck into the opponent's goal area, and were awarded fitness based first on their ability to do this, but also, to a lesser extent, on their overall proximity to the puck over the course of the trial, and the puck's proximity to their own goal and their opponent's goal over the course of the trial. These lesser goals

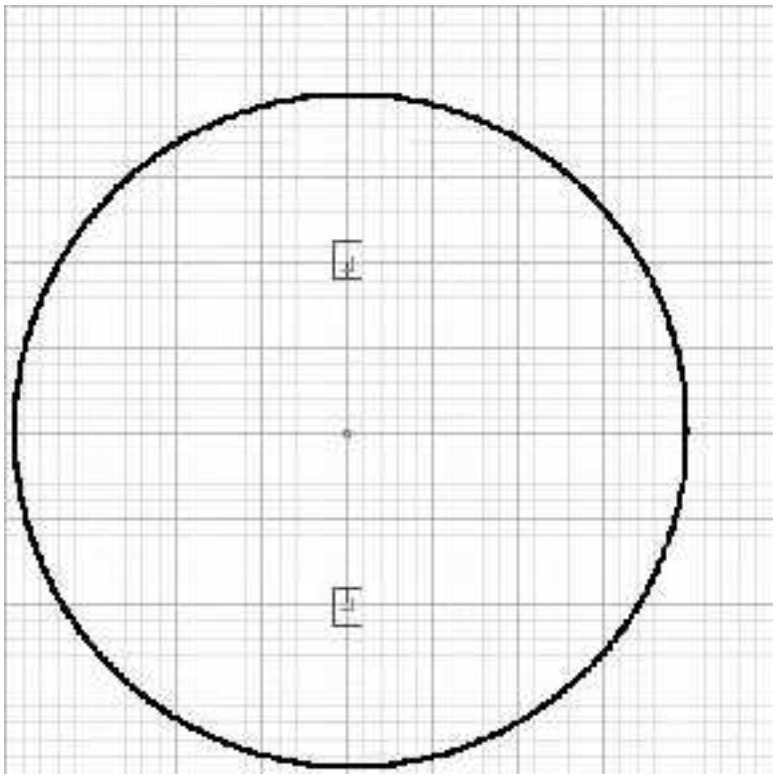

Figure 1: The beginning state of the simulation environment used for this experiment.

were included in order to facilitate the development of more robust behavior in early individuals, so as to give them steps to take on the way to developing goal-scoring behavior. The lesser goals were kept as simple as possible in order to prevent the fitness function from over-engineering the results, but there is still some possibility that the fitness function's complexity limited the solutions.

The simulation was created using the Player/Stage robotics toolkit(Gerkey, Vaughan, & Howard, 2003).

3.2 Robot Controller

The robots in this experiment were controlled by fixed-morphology three-layer feedforward networks, with 11 sensory inputs, 6 hidden nodes, and 2 output nodes. Three of the sensory inputs are sonar readings from the front and the sides, while eight of the remaining nine inputs are used for paired angle-size readings from the color-blob module for each of the puck, the goal to attack, the goal to defend, and the enemy robot. The output nodes control the angle

and speed of the robot's movements for the next time step.

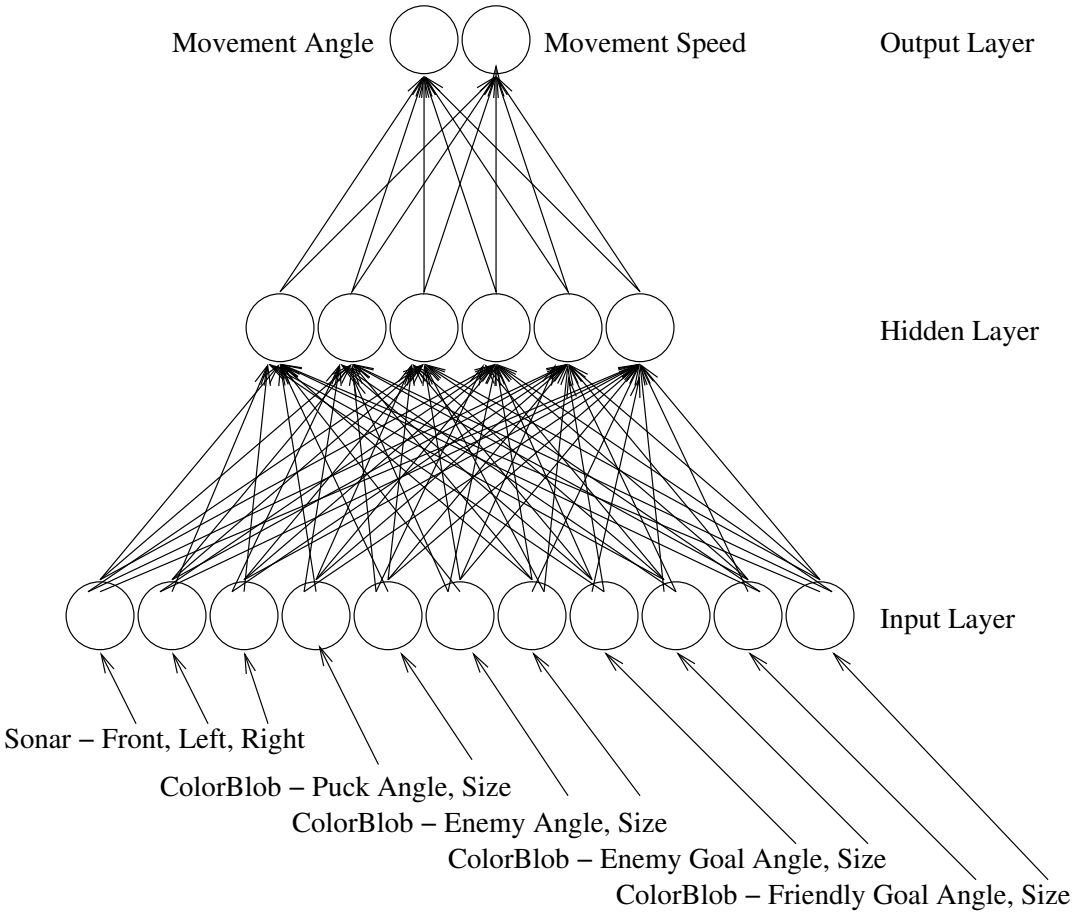

Figure 2: The structure of the controller network

The weights of the individual robot's networks are fixed at the beginning of the trial, and no learning within a trial takes place. The pool of candidate weights is evolved using a genetic algorithm.

The controller weights were evolved using the genetic algorithm library of Pyro, the python robotics platform(Blank, Meeden, & Kumar, 2002). The networks were also implemented using Pyro, this time through Conx, the Pyro's neural network library.

3.3 Methodology

The weight of each of the connections in the neural network brain were evolved using a genetic algorithm. The weights of the individual robot's networks were fixed at the beginning of the trial; no learning took place within a trial. A total of 100 generations were evolved with a population size of 25, a crossover rate of 0.2, and a mutation rate of 0.1. The top 10 percent of each generation was carried over to the next generation with no mutation.

In each generation, each individual in the population was tested against the best individual in the previous generation. The number of time steps in each trial run increased over time. The first generation got only five time steps; at each interval of five generations, five more time steps were added, so that the robots in the final generation were competing for 500 time steps. This allowed early generations to evolve desirable starting behaviors while limiting the testing time for early generations. Fitness was accumulated each time step for getting close to the puck, for getting the puck close to the correct goal, and for keeping the puck away from the opponent's goal. An additional fitness reward was given to a robot for scoring a goal, while fitness was punished if a robot was scored against. A trial run automatically ended if a goal was scored by either robot.

To limit coherency issues, each robot was controlled by a separate thread. Each robot continuously polled its sensors, fed the results through its neural network, and then adjusted its motor speed and direction. A separate thread was used for the genetic algorithm fitness function; at each time step, the fitness function polled the two robots and the puck for their current locations and then assigned fitness based on the location of each of the objects in the world. This approach minimized the time difference between the control of the two robots and minimized any resulting bias towards one of the two robots in terms of fitness scoring.

At the end of each generation, the network weights of the most fit individual were stored for future analysis. At the end of the evolutionary process, the most fit individual from each of the one hundred generations competed against the winner of the final generation for a full 500 time steps. This allowed the winners from each generation to be quantitatively compared since they were competing against the same opponent.

4 Results

Several trials were run, but in all but one, the results were uninteresting (the robots seemed to behave randomly, and did not reach the puck at all). In one trial, however, the robots developed an interesting strategy for accruing fitness. We will report on that trial.

The evolved robots tended to race towards the puck, and get stuck against each other after hitting it between each other several times.

An analysis of the of the network weights of the last (100th) generation's most fit individual yielded the following chart showing the relative importance

of each input to each output.

	Output 0 (Movement Speed)	Output 1 (Movement Angle)
Input 0 (Left Sonar)	2.611675326	-3.779960754
Input 1 (Front Sonar)	0.319765879	-3.54299187
Input 2 (Right Sonar)	2.92494285	-7.736563169
Input 3 (Puck Size)	2.507012939	-5.19375695
Input 4 (Puck Angle)	-10.43034984	-1.431591443
Input 5 (Own Goal Size)	-1.850411019	0.262409728
Input 6 (Own Goal Angle)	-8.468315397	5.485205001
Input 7 (Other Robot Size)	20.96254809	6.337365534
Input 8 (Other Robot Angle)	7.801596886	-0.211620431
Input 9 (Other Goal Size)	11.50429228	1.517441805
Input 10 (Other Goal Angle)	0	0

This chart shows that the most important factors for the robot's speed were the other robot, the puck, and the robot's goal, while the most important factors for the robot's angle of motion were some sonar readings, the size (and hence, proximity) of the other robot and the puck, and the angle to it's own goal. However, it is hard to say, without further investigation, what this really means, in terms of the robot's behavior.

In order to ensure a viable comparison, the most fit individual of each generation competed with the most fit individual of the last generation. The results are shown in figure 4.

The chart in figure 4 shows that the fitness of the most fit individual of each generation, when placed in competition with a constant competitor (the most fit individual of the last generation), irregularly cycles between a score of zero, and a score of around 500.

5 Discussion

At a qualitative level, the winners in each generation seem to follow the same basic strategy of moving almost straight towards the puck as quickly as they can. This results in the puck bouncing back and forth between the robots a number of times, and eventually out, away from the robots, towards one of the goals. The two robots then collide with each other and remain stuck for the rest of the trial. An example of the movement of the puck during this behavior is given in figure 5. The somewhat cyclical values for the fitness of each generation's best individual in competition with the last generation's best individual, as shown in figure 4, could be explained by this behavior. Since both robots get to the puck at nearly the same time, it is a matter of slight variations of angle and speed which goal the puck will eventually fly towards. The slight variation which beats one set of angle and speed may lose to another, similar one. The puck's movement towards the opposing goal will raise the winning robot's fitness to about the high levels achieved, while the less fortunate robot will receive almost no fitness, as the puck move towards it's goal, and away from it's enemy's goal. It seems intuitive that in each of the generations in figure 4 where the score

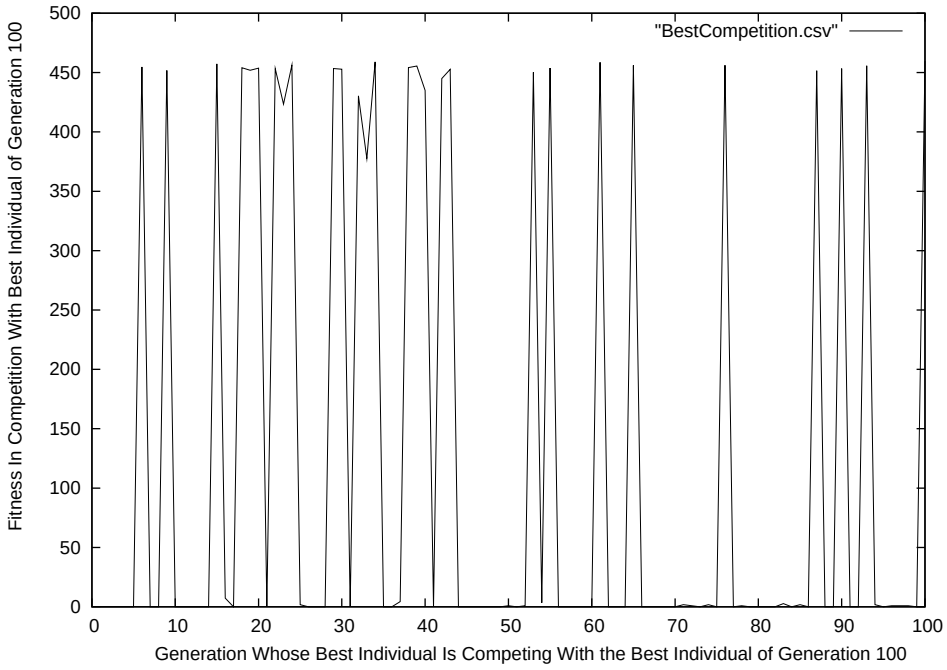

Figure 3: The fitness attained by the most fit individual of each generation in competition with the best individual of the last generation

is around 500, the opposing robot, the most fit individual of generation 100, scored a fitness of 0, while for each of the generations with a score around 0, the competitor would have scored around 500.

One of the reasons for the relatively uninteresting behavior shown by the evolved individuals might be the time step constraints placed on the early generations. These constraints may have led to the population consisting of controllers that will always immediately move forward, since that is always where the puck is from the starting condition, and any other actions, in a simulation with a severely limited number of time steps, will lead to zero fitness. One solution would be to length the early trials, and increase the number of time steps per trial more gradually. This was attempted, but due to time limitations, the results were inconclusive.

Another problem may have been the fitness function itself, which was complicated enough that it tried to emphasize certain sorts of behaviors, but failed to punish others. For example, one change to the fitness function that may have resulted in more interesting behavior might have been a punishment for not moving. This was also attempted, by preventing any further fitness scoring if both the robots and the puck are immobile for five seconds. However, due to the aforementioned time limitations, only inconclusive results were obtained.

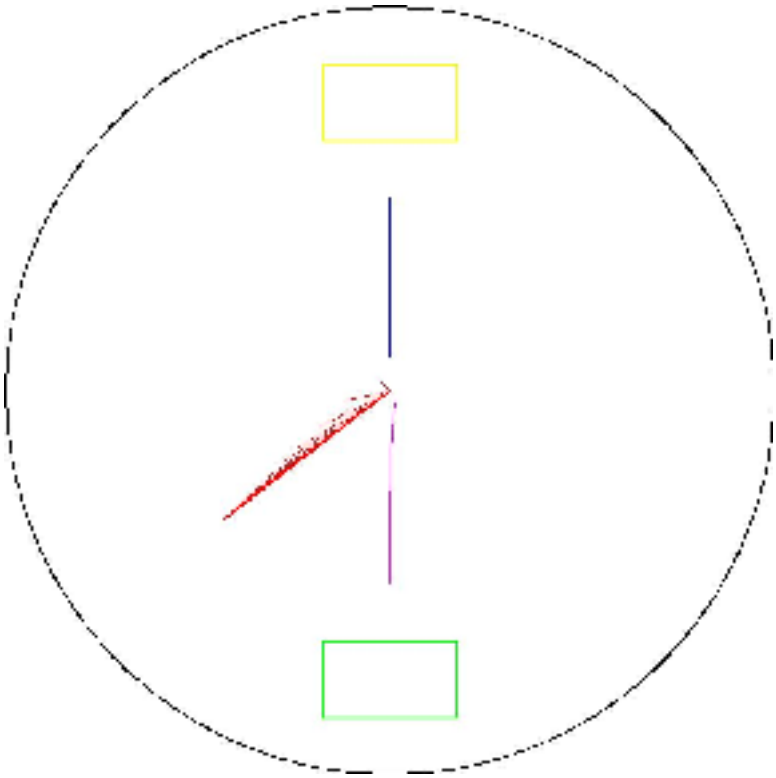

Figure 4: The behavior demonstrated by the best evolved individuals was a race for the puck, at a very slight angle. This results in the puck bouncing back and forth from robot to robot, eventually bouncing away towards one of the goals. Subsequently, the robots would collide and get stuck, and the puck would stop.

Some of our results may be at least partially suspect, as the simulator used in the generation of all data is not entirely deterministic. On several occasions, individual competitions run with identical parameters have produced different results. This manifests itself with slightly different trajectories for the puck after it leaves the collision. An example of this can be seen in the puck lines in figure 5. While this is an important potential problem, if it is consistent with our qualitative analysis of the evolved individuals's behavior, then it just says that the specific numbers given in figure 4 might not be fully trustworthy.

One unexpected outcome of the experiment was a repudiation of our choice of the previous generation's most fit individual as the competitor. This decision lead to the evolution of individuals that would defeat themselves, which obviously cannot be done more than half the time. In practical terms, this meant that individuals exhibited a strategy where each robot would defeat it's opponent (whose control network, and hence strategy, was nearly identical) roughly

half the time, and lose the other half of the time, in the battle for where the puck would go after bouncing between the colliding robots.

6 Future Work

One element missing from this experiment found in most other competitive evolutionary experiments is co-evolution. It was excluded in the present work because of time constraints, but we hypothesize that co-evolution would lead to species with varying strategies, each optimized against it's co-evolutionary partner. Such findings would be consistent with those of Floreano and Nolfi (Floreano & Nolfi, 1997), and of Sims (Sims, 1994). However, these results are predicated on robots with different, and in the case of Sims's experiment, evolving, morphologies. In a situation with morphologically identical robots, facing almost identical tasks, single-species evolution may make more sense, even with a competitive task. One thing is certain, however - as Floreano and Nolfi suggest, the fitness function could be much simpler in a co-evolutionary system, because exploiting holes in the fitness function that fail to really aid task performance, which often leads to stagnation and necessitates a more complicated fitness function in single-species evolution, is taken care of by the competing species in co-evolution (Floreano & Nolfi, 1997).

The major function of the addition of co-evolution in this experiment would have been to evolve multiple solutions to a similar, but not quite identical problem (since the evolving competitor is unique to every generation of each evolved species, only evolved individuals of the same species and generation share the same task). Some of the other advantages of co-evolution might not be as applicable to the present task, however, because they are already accounted for in other ways. For example, the task evolves with the population, not just in that each generation must face the best competitor from the last generation, but also in that the number of time steps that each competition is allowed to take increases with the number of generations. Even the fitness function, which doesn't actually change, has different levels of success, as individuals must first seek the puck, then try to get it towards their opponent's goal, and away from their own, and finally score. This multilevel fitness is almost like having a changing fitness function, because it adapts to the success of each individual, by offering radically increased fitness for accomplishing higher level goals. In effect, the present fitness function provides for what Nolfi calls "incremental evolution", where a population is trained to do progressively more and more complex tasks (Nolfi & Floreano, 1998). While the present fitness function could be argued to include the engineer's assumptions about how to accomplish the task, it could also be argued that any good solution to the task must involve the robot seeking the puck (as it otherwise has no control over the puck), and trying to herd it into the goal (otherwise, it cannot score).

While co-evolution tends to generate interesting divergent strategies for a task, this can also be a weakness, as such strategies can lead to a "cycle" effect, where one population will develop one strategy, the other population will develop

a strategy specifically designed to beat the first strategy, the first population will develop a new strategy to counter the other population's strategy, and the other population will revert to it's initial strategy, if that beats the new one strategy of the first population (Nolfi & Floreano, 1998). That said, this could also be a weakness in the present single-species evolution model, though only if alternative strategies could be evolved with sufficient speed, because a competition against the previous best leaves plenty of room for the population to cycle between three or more strategies, rather than developing new ones.

One way to counter the cycle effect would be to test individuals against several previous generations' best individuals(Nolfi & Floreano, 1998). This would encourage what Nolfi calls an "arms race", where each population races to build better and better individuals to compete against each other, rather than cycles, which are a lot like local maxima over the fitness landscape of both populations. However, this approach can be very expensive, as it requires each tested individual to compete against several opponents, rather than just one.

Another way to create more interesting behavior might be to add learning to our evolved controller networks. If learning could be added, the results might be even more impressive than with co-evolution, because learning allows a search for small, steep spikes of genetic fitness that any form of evolution is much less likely to find. Learning would be difficult to implement, as a teacher must be available at every step for most learning algorithms. However, a partial solution is offered by complementary reinforcement back-propagation, or CRBP (Ackley & Littman, 1990). CRBP is not as fast a learning mechanism as the supervised learning in normal back-propagation, but it is more generalizable, as it allows the translation of general feedback, like reward and punishment, into the very specific error feedback needed by back-propagation. In addition, CRBP allows variable biases towards it's different feedback options. For example, Meeden used CRBP to facilitate learning in an alternatively photophilic and photophobic robot, without specifying how the robot should achieve it's goals (Meeden, 1994).

Another potential place where the present model could be improved is the morphology of the control network used. The network used here was a simple three-layer feedforward network. In a comparison of various evolved network structures, Nolfi found that networks of this type were significantly less robust than Elman-style recurrent networks, which have the previous activations of their hidden nodes fed back into the network as additional inputs, and his own emergent modularity networks, which could learn to use or not use parts of their architecture, and therefore prevent themselves from being too complex(Nolfi, 1997). While there has been some objection to Nolfi's work, mostly focusing on the relative unfairness of his comparisons between networks of vastly different sizes, further work by Olsen and Fowles corroborates Nolfi's finding that a simple three-layer-architecture is vastly inferior to either recurrent Elman-style network, or a network using Nolfi's emergent modularity technique(Olsen & Fowles, 2003).

Lastly, the argument could be made that since this experiment takes place entirely in a simulator, any results and conclusions from it are inherently sus-

pect. This is a hotly debated subject in general, and, as reviewed by Meeden and Kumar(Meeden & Kumar, 1998), arguments from both sides apply to the present experiment. On the one hand, the environment seems relatively simple, and our robots' inputs are at least modelled on real sensors, such that the experiment, if not the evolved brains themselves, could easily be translated into reality. This is in sharp contrast to the previously discussed work by Sims(Sims, 1994), which would prevent be extremely difficult to translate into reality, because it evolves changing physical morphology, in addition to it's changing controller. Still, while the present experiment could be embodied, it might face unforseen difficulties. This is because we approached the design of this system without intending to embody it, and so took no efforts to introduce noise into the simulation, a necessary step in attempting to use simulation to evolve a network that will perform in the real world.

References

Ackley, D. H., & Littman, M. S. (1990). Generalization and scaling in reinforcement learning. In D. S. Touretzky (Ed.), *Advances in neural information processing systems* (Vol. 2, pp. 550–557). Denver 1989: Morgan Kaufmann, San Mateo.

Blank, D., Meeden, L., & Kumar, D. (2002). Python robotics: An environment for exploring robotics beyond legos. In *Acm special interest group: Computer science education conference.* (SIGCSE 2003 - Reno, NV.)

Duchon, A., Warren, W., & Kaelbling, L. (1995). Ecological robotics: Controlling behavior with optical flow. In *Proceedings of the 17th annual conference of the cognitive science society* (pp. 164–169).

Floreano, D., & Nolfi, S. (1997). God save the red queen! competition in co-evolutionary robotics. In J. R. Koza, K. Deb, M. Dorigo, D. B. Fogel, M. Garzon, H. Iba, & R. L. Riolo (Eds.), *Genetic programming 1997: Proceedings of the second annual conference* (pp. 398–406). Stanford University, CA, USA: Morgan Kaufmann.

Gerkey, B., Vaughan, R. T., & Howard, A. (2003). The player/stage project: Tools for multi-robot and distributed sensor systems. In *11th international conference on advanced robotics.* (to appear)

Gibson, J. J. (1986). The ecological approach to visual perception. In (p. 7-8). Lawrence Erlbaum Associates.

Meeden, L. (1994). *Towards planning: Incremental investigations into adaptive robot control.*

Meeden, L., & Kumar, D. (1998). Trends in evolutionary robotics. *Soft Computing for Intelligent Robotic Systems*, 215–233.

Nolfi, S. (1997). Using emergent modularity to develop control system for mobile robots. *Adaptive Behavior*, *5*.

Nolfi, S., & Floreano, D. (1998). How co-evolution can enhance the adaptive power of artificial evolution: Implications for evolutionary robotics. In *Evorobots* (p. 22-38).

Olsen, S., & Fowles, M. (2003). *Giving nolfi a fair fight: A comparison of elman recurrence and emergent modularity.* (Developmental Robotics undergraduate final project)

Sims, K. (1994). Evolving 3d morphology and behavior by competition. In R. Brooks & P. Maes (Eds.), *Proceedings of the artificial life iv conference.*