Evolving Cooperation: The Butt Lovers Dillema

David Kaczorwski & Andy Zuppan

May 18, 2003

Abstract

In this paper we describe an implementation that evolves cooperation between two robots. Through the use of a competitive fitness function that interrelates the actions of the two robots we generate a first step towards emergent cooperation. The evolution of cooperation is an important step toward developing intelligent robots. Simple forms of cooperation are evident after several runs of our experiments. With more time the conditions necessary for the emergence of cooperative behavior would likely be discovered.

1 Introduction

Cooperative behavior presents a challenge to the standard evolutionary theory of natural selection. Evidence exists that cooperation is an emergent property of natural systems that sufficiently relate the performance of multiple individuals. We intend to test the applicability of emergent cooperation in the domain of robotics.

The next section discusses the motivation behind our experiment and states our hypothesis regarding the emergence of cooperative behavior. The third section summarizes related work. Section four describes the methodology used to implement and test our hypothesis. Section five presents our findings. Section six is a discussion of our results and possible weaknesses of our approach. A range of possible extensions to further test our hypothesis is presented in section seven.

2 Motivation

2.1 Evolution and Cooperation

Evolution is competitive. A key component of the theory of natural selection is the survival of the fittest, which implies the fall of the weakest. Organisms thrive in an evolutionary sense by being better able to spread their reproductive seed to future generations. Creatures with less reproductive ability are considered weaker, and are driven to extinction.

The competitive nature of evolution presents observers of real world phenomena quite a paradox in explaining some animal behavior - cooperation. Humans are a prime example, as we are social creatures who cooperate with both relatives, close friends, and occasionally random strangers. We are not unique in our cooperative, altruistic tendencies however, as several animal species exhibit degrees of cooperation. From relatively complex animals such as monkeys to small insects such as ants, the presence of cooperation seems antithetical to the competitive nature of evolution.

The reason for this anomaly is that cooperation can often lead to increased well-being for both participants in the cooperation. Ants who risk their lives to help their relatives cross rivers may sacrifice their own genes, but if they save enough of their close relatives then their effective gene dispersal could equal or outweigh solely selfish behavior. Another possibility is that competition over resources is not a zero-sum game, that is, working together can yield more total gain than the sum of gains from individuals working alone. By working together, then, both of the cooperating individuals receive higher evolutionary fitness, further propagating the cooperative tendency down the path of evolution [5].

It is this notion of non-zero-sum games that we believe allows cooperation to emerge from evolution. If individuals can both prosper through coordinating actions, then there could certainly be evolutionary pressure to facilitate this coordination. However, finding situations where non-zero-sum games are possible is challenging. In general, where reproductive fitness depends on procreation and staying alive, and in a world with generally finite resources available for consumption, it is unlikely that, say, forgoing eating to provide food for another, will ever lead to increased fitness for the fasting individual. However, when certain actions involve a level of risk and potential reward, and both the risk and reward are contingent not only upon an individual's actions, but on another's actions as well, we believe that the payoff from interaction could easily outweigh the payoff from behaving selfishly.

2.2 Artificial Evolution

The natural phenomenon of evolution has its analogy in the realm of artificial intelligence and robotics in genetic algorithms. Potential solutions are evaluated on their goodness, and the strong can multiply while the weak are eliminated. This paradigm of searching allows an effective, randomized search of a problem's search space. Good solutions are more likely to live on, with mutation allowing varying search states and crossover possibly combining the advantages of two good, but unique individuals.

Given that genetic algorithms are an attempt at mapping a natural phenomenon onto the domain of problem solving and, in particular, robot controllers, we hope to demonstrate the possibility of evolving robot controllers that, although they are superficially competing, nonetheless evolve what human observers would term "cooperation."

More particularly, we believe that the cooperation exists because of an appropriate environments. The environment must have risk and rewards in a manner such that cooperation can lead to an improved outcome for both participants. This should raise the fitness of individuals who attempt to cooperate in our robot world and lower the fitness of those who do not. In general, we hypothesize that cooperative behavior evolves out of an environment that pays off risk and reward properly. To test this hypothesis, we simulate competing robots in an environment where an individual's fitness is dependent on both agent's actions.

3 Related Work

3.1 Game Theory and Gains from Cooperation

One pertinent field of related works comes from outside the discipline of computer science. Economic research in game theory can go a long way in describing outcomes of agents' behaviors, and predicting patterns of cooperation and selfishness in social situations. The archetypal example of such a model is the Prisoners' Dilemma, a thought experiment on the behavior of two captured felons who are offered the chance to rat on their accomplice.

A standard formulation of the Prisoners' Dilemma payoff matrix is shown in Figure 1. If both agents cooperate, then they both recieve a well-being of three. However, the dominant strategy for this game is for both individuals to defect. To see why this must be true, consider an individual's best action for both cases. If the opponent cooperates, an individual is better off defecting as they will recieve a benefit of five instead of three. If the opponent defects, then it is in the best interest of an agent to also defect, as one is greater than zero. Since the problem is symmetric between agents, both individuals will always find it in their best interest to defect. This defection not only represents a dominant strategy, but it is a socially inefficient equilibrium. Clearly, both agents would be better off if they both cooperated. This potential gain to both individuals illustrates the fact that the Prisoners' Dilemma is a non-zero-sum game. Cooperation permits gains to both individuals; an actor does not have to "beat" its opponent to flourish.

	cooperate	defect
cooperate	3 3	5 0
defect	0 5	1 1

Figure 1: Standard variant of the Prisoners' Dilemma problem

As the Prisoners' Dilemma has an equilibrium dominant strategy of the socially unoptimal defection of both players, it is quite surprising that, if the game is repeatedly played for an unknown number of rounds, cooperation still emerges. In a famous study, Robert Axelrod compared several different strategies for dynamic Prisoners' Dilemma games, and, surprisingly, the "tit for tat" strategy dominated all other approaches [2]. An agent following a "tit for tat" strategy will cooperate in the first round, and then play whatever move the opponent made in the previous round. This style of play encourages cooperation between players as an individual is making a commitment to cooperating, but is willing to immediately punish a defector in the next round. By creating a credible threat of punishment in the future, a "tit for tat" player will force cooperation and shift the game's dominant strategy away from always defecting to always cooperating, thereby achieving the best possible outcome.

This example from game theory can inform upon the evolutionary creation of cooperation. Using an example from the literature, a lone wolf would be unable to take down a large creature, while a pack of wolves can easily kill large prey. However, a single wolf could certainly benefit from not participating in the hunt and only reap the rewards of the dead carcass [5]. As Heylighen points out, "cooperation can create a synergy which strongly extends the set of reachable resources" [5]. Although both the example of the wolves and the Prisoners' dilemma are highly stylized, they nonetheless illustrate a general principle that cooperation can be more beneficial than naive, selfish behavior.

3.2 Angeline and Pollack

Angeline and Pollack's *Competitive Environments Evolve Better Competition* provided the basis for our tournament structure and provided justification for our use of a competitive fitness function. This article argued for the benefits of competitive fitness functions, and implemented a simple tournament as an example.

Angeline and Pollack define "competitive fitness functions" as fitness functions that are calculated so that one or more variables of an agents fitness are dependent on the action of one or more other agents [1]. Sims's evolved morphologies serve as en excellent example of a competitive fitness function. Two robots play a game of "steal-the-bacon." Both attempt to grab an object placed at an equal distance between the two robots. Fitness is determined by the amount of time one spends in possession of "the bacon"[8]. The calculation of fitness in this example is determined by the interaction of two agents.

The problem with competitive fitness functions is their lack of precision. There is no way to specify an optimal solution for the entire population. If two very skillful agents gain a mediocre amount of fitness playing against each other they will gain less fitness than a mediocre agent playing against a terrible agent. What competitive fitness functions lack in precision they make up for in adaptability. As strategies become dominant counter-strategies evolve. A robot's behavior must encode a wide variety of different reactions based on its potential opponents reaction[1]. The possibility of robust, varied behavior based on opponent behavior encouraged us. The potential for ever escalating complexity made a competitive fitness function seem ideal for our project.

Angeline and Pollack also described a number of tournament structures. Our idea for a bi-partite tournament[1] was derived from their work. This tournament structure, which pits every member of the population up against one other member of the population, seemed ideal. It allowed us the benefits of competitive fitness functions while still allowing us to run our simulation in a reasonable period of time.

3.3 Nolfi

Nolfi's *Using Emergent Modularity to Develop Control Systems for Mobile Robots* describes a robot, trained in simulation, that's designed to pick up plastic pucks and remove them from its environment. The author evolves 5 different neural network control architectures designed to preform the task. Two of the architectures attempt to implement a subsumption-architecture style modularity. The first architecture is designed so that one half of it is responsible for robot locomotion while holding a puck. The other half of the architecture controls the robot when it is seeking a puck. The final architecture lets the network pick between two different output values for one of its effectors. This architecture learns how to break up the task. This architecture also outperforms all of the other architectures in the article, but it only uses the modules for one effector. The use of this module seems to fit no pattern. Nolfi uses this architecture's performance to argue for the priority of proximal descriptions of robot behavior as opposed to distal. Subsumption architectures base the way that they divided up layers of behavior on distal human descriptions of the robot's behavior. Proximal descriptions, which are the view of events from the robots perspective, serve as more accurate indicators of the way robot behavior should be modularized [6].

The method that Nolfi uses to teach his robots served as the source for our method of teaching the controller. Nolfi uses artificial neural networks as his robot controllers, but these controller do not learn. Instead all of the robots received a fitness value after their run. Nolfi then altered the weights of the artificial neural network based on the fitness function with weaker members of the population replaced by mutations of stronger ones [6]. Turning off neural network learning fit our approach for this model because it allowed us to perform the simulations more quickly. The justification for turning off learning came from Nolfi and Floreano's article *Learning and Evolution* which also claimed that learning requires a period of acclimation to the environment [7].

3.4 Grefenstette

Grefenstette's *Learning Decisions Strategies with Genetic Algorithms* uses a production system called SAMUEL that has been outfitted with a genetic algorithm. The system in the article evolves a series of logical rules designed to keep a cat at a fixed distance from a mouse, while still tracking it. The article argues for the use of genetic algorithms in continuous time and sequential problems. Traditional production systems often times have difficulty determining the benefit or demerit of a particular action in time, whereas genetic algorithms don't. The article then goes on to argue, using the cat and mouse example, that outfitting the cat with some initial propositions which it can then refine with the GA leads to optimal performance. It also argues that neural nets are problematic because they don't scale up to larger problems well and because they are difficult to seed with heuristic data that helps jump start the problem solving [4].

Although we disagree with Grefenstette's position on artificial neural networks, his use of genetic algorithms to solve problems with continuous time pay-off matrices intrigued us. Like the author we

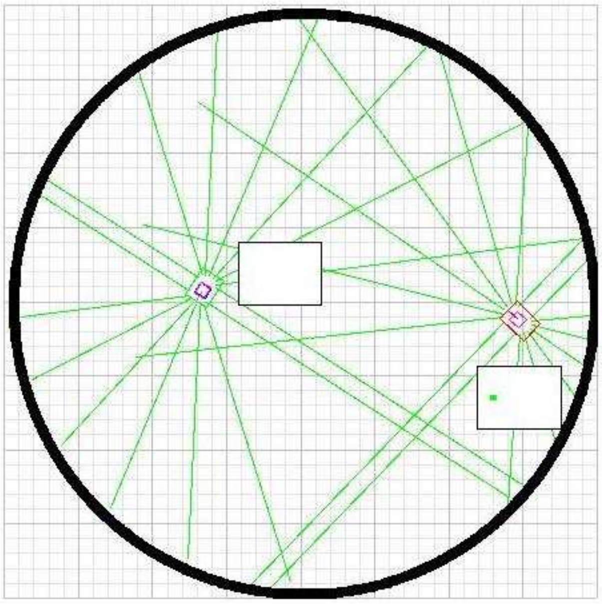

Figure 2: The robots in their world

were interested in exploring games with pay-offs in continuous time. This article made it clear that genetic algorithms were ideal for that class of problems.

4 Setup

4.1 Simulated World

To effectively test for evolution's ability to generate cooperative behavior, creating an appropriate world is critically important. Given both time and robot constraints, we were unable to create a real world environment and to test robots. Instead, we used a simulated world in Stage with Player providing an interface into robots that were functionally similar to Pioneers.

Testing of the robots was accomplished by placing two robots randomly within a circular world with an approximate diameter of 8.0 meters. The robots themselves measure approximately 0.5 meters. The robots are then allowed to run for set period of time. An example of the robots in the world is provided in Figure 2.

4.2 The Robot Controller

To control the robots in our experiment, we used a neural network architecture. The network takes in several sensor values as inputs and outputs motor controls in the form of translation (speed) and rotation (angle to turn). The architecture is described in Figure 3. The sensors used are: four arrays of distance sensors - one each for the front, the left, the right, and the back; the distance to the blob (if visible); the visible area of the blob; and three inputs coded to inform the relative location of the blob (if visible). Our network also uses six hidden nodes.

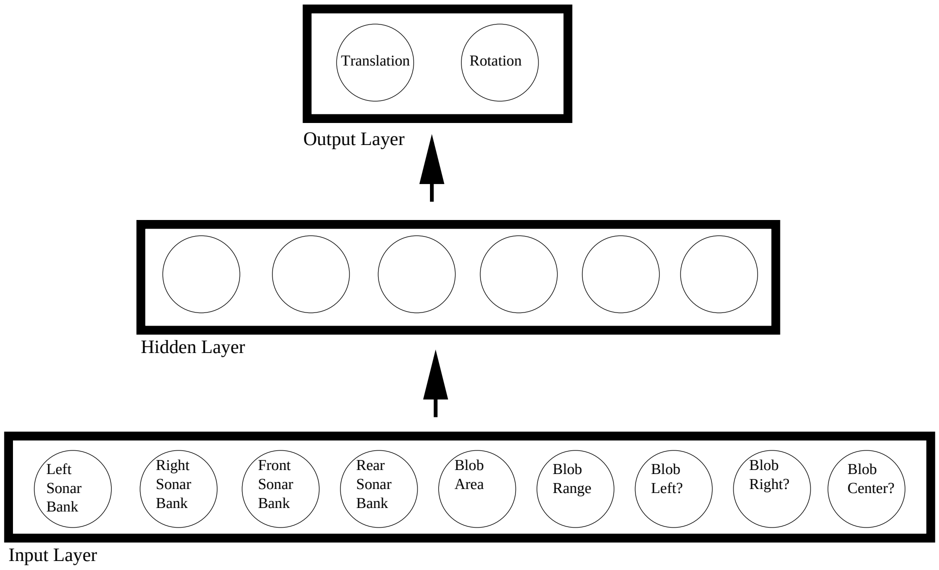

Figure 3: Visual Representation of our fully connected net

4.3 Encoding the Neural Net and GA Parameters

The genes for our genetic algorithm were derived from the neural network controller that ran the robots. These controllers can be represented as a collection of interrelated weights and biases. Our gene was merely an array of all of the weights and biases present in the controller. These weights and biases were floating point values. They were altered based on their controllers performance. Most of the changes we made were made through mutation. We experimented with a .3 rate of mutation with no crossover. Our most successful trials featured a crossover rate of .2 and a mutation rate of .1. Mutation was still the dominant form of change to the network when these new values were used. Every weights and bias had a .1 chance of being mutated, whereas every population member had a .2 chance of being crossed over. With at least 108 genes per population member (9 input nodes * 6 hidden nodes * 2 output nodes) and only 50 population members mutation had more opportunities to work its magic. Mutations of, and crossovers between dominant members of the population replaced poor performers. We used a fairly high selection rate of .8 for the earlier trials, replacing 80 percent of the population each generation. Later we removed the selection rate entirely in favor of elitism. We use elitism on our most successful run as well, maintaining the top 10 percent of the population. So to recap, on the final run:

Population: 50

Mutation Rate: .1

Crossover Rate: .2

Selection Rate: N/A

Elite Rate: 0.1

And for the earlier runs:

Population: 50

Mutation Rate: .3

Crossover Rate: 0

Selection Rate: .8

Elite Rate: 0

4.4 Learning

No learning was used during the runs. We didn't use learning for two reasons. First, our hypothesis placed learning beyond the scope of our project. Our goal was to evolve cooperation, not learn it. Second, we craved speed. Learning is a time consuming task and accordingly we decided it was to be excluded.

4.5 Fitness

The fitness for a genetic algorithm is crucial. As evolution will tend to gravitate towards the simplest possible solution, then a poorly defined fitness function will fail to generate fully robust solutions to an environment. For our experiment, the most important part is creating significant interdependence between the two agents. If the two competitors' fitness relation is not connected to a sufficient degree, then it is unlikely that cooperation will emerge. In other words, the fitness of an agent should codify an appropriate payoff matrix that will hopefully facilitate competition. Consequently, an agent's fitness will depend not only on its own actions, but that of the opposing agent it is interacting with.

One possibility for assigning dependent fitness functions is to explicitly divide up the total fitness of a run between the two participating agents. However, that style of approach is susceptible to numerous problems. For one, two very disparate performers, i.e., a good and a bad solution, would end up with inaccurate fitnesses. The bad robot would recieve more fitness than it should and the good robot will not recieve enough. A much more desirable approach is to base the fitness solely on an individual's performance, but attempt to tie that fitness explicitly to the other robot's actions.

The method we chose to implement fitness indirectly relates the competitor's actions to an individual by using the visibility and range of the robot's blob sensors. When a robot senses a blob, that is in fact the other robot, and in increase in the blob area indicates that the robots are approaching. Therefore, increases in fitness could result from either the robot approaching its opponent or the opponent approaching the robot. More importantly, when both robots are approaching each other, the fitness of both robots will increase at an even greater rate.

Although this method creates an interdependency on the robots' fitnesses, it is also subject to problems. There is no explicit relation, which means that we cannot guarantee that the robots will correctly realize the payoffs and risks implicit in the system. The interdependency therefore runs the risk of being underdefined, removing any chance at actually evolving cooperation.

To ensure that the robots behaved with adequate complexity, our fitness also encouraged movement, both in speed and rotation. Also, running into objects, which in our world is only the wall or the other robot, is discouraged. Upon contact with an object, the robot continues to run, but it is not given any fitness for the duration of contact. These features ensure that robots will end up both moving and avoiding collisions.

4.6 Tournament Structure

We did runs of up to 200 generations and of as few as 80 generations. Each generation featured n/2 contest between two controllers. Each contest pitted the robots against each other for a total of 300 time steps. On each time step we included a delay of .1 seconds, in an effort to lengthen the contest without raising the number of steps. A greater number of steps would have increased the disparities

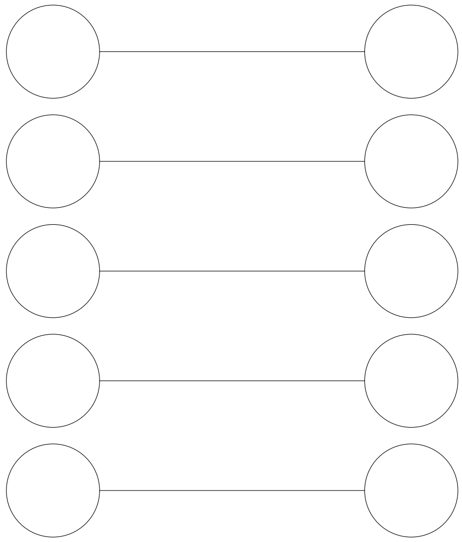

Figure 4: Bi-partite tournament structure for a population of 10

between robots final fitness values as fitness was calculated at each step. Greater disparity makes meaningful comparison more difficult.

The tournament structure we used is called a bi-partite tournament structure (see Figure 4). Each generation every controller faces off against one other controller. The controller match-ups are determined randomly, so that each controller sees a large portion of the population after several generations. This tournament structure does generate a great deal of "noise." Good controllers could meet up against highly specified counter strategies that lower their fitness. These good controllers would then be removed from the population. Still, the speed of this tournament structure recommended it. It enabled us to run simulations very quickly. Also, the noise of this tournament structure seemed minimal. Strategies always stayed dominant for a number of generations, suggesting that powerful strategies were not haphazardly eliminated by this tournament format.

5 Results

Our hypothesis that evolution can lead to cooperative outcomes is particularly difficult to test. As has been described, we set up competing, evolving robots whose fitnesses are dependent on both robots' behavior. Although we believe that such methodology can yield cooperation, the determination of what exactly constitutes cooperation makes quantifying the results a challenge.

We evolved three sets of robot controllers using the methods discussed here, and all performed with similar functionality. The general tendency of each robot was to spin back and to the left very quickly. The arc of their movement was such that they would usually avoid contact with the circular wall of the arena and rotate enough so that the opposing robot would come into view for a short, but significant amount of time. Such behavior could reasonably be predicted from our fitness functions, which equally weighs speed, blob sighting, and avoiding contact.

Further evidence for the effect of equal weighting in the fitness function is the pattern of behavior over generations. In general, as evolution progressed, the robots' speed increased along with the angle they were spinning at.

Although all runs generated this similar style of backward spinning, one run generated a much more interesting result, albeit for a very limited evolutionary period of time. Instead of naively spinning

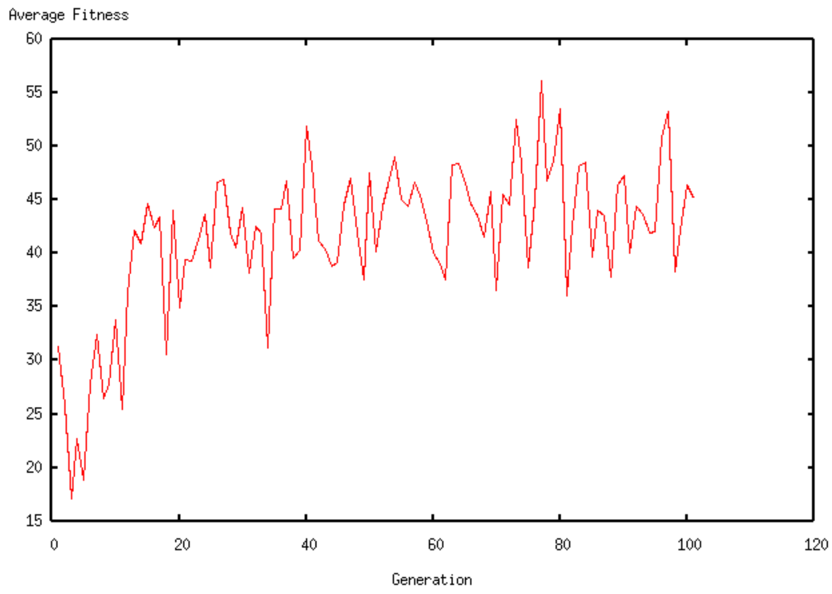

Figure 5: Average fitness per generation on a 100 generation run

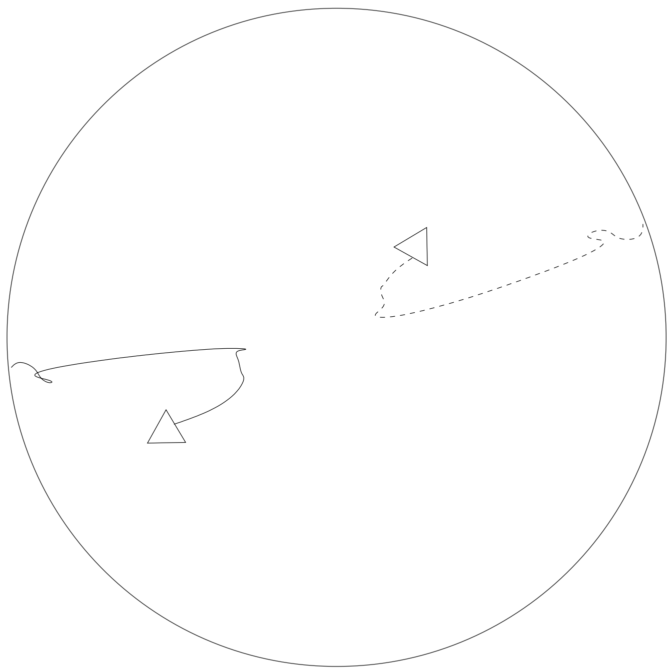

Figure 6: Paths of our best performing controllers

in all conditions, this particular robot controller would exhibit the same pattern of backward motion until the other robot came into view. Upon seeing the other robot's blob, the robot would stop its spinning and switch to moving straight backwards. Although the motion is clearly moving the robot away from the desired goal, it is a clear improvement over naive spinning as the robot can maintain its sight of the other robot. The brain also makes a small attempt at obstacle avoidance by further adjusting its path as it approaches the wall. This functionality is unfortunately underdeveloped, as it tends to plow directly into the wall. Figure 6 details the movement of two robots being controlled by this brain.

6 Discussion

Eye-balling our robot controllers suggests that they have not achieved particularly complex behavior. One of the reasons for the simplistic behavior may be based on our random placement of the combatants. The principal problem facing the robot may have been locomoting so that it could encounter another colored blob. A robot that could spin about and keep its opponent in view for a reasonable chunk of its run was probably able to defeat robots with nascent wall-avoidance abilities or more dramatic tracking abilities. Merely finding the other robot proved to be a difficult task then. That our controllers can actually perform that task suggests that our fitness function was beginning to generate more interesting behavior.

The robots tendency to back straight up after it caught a glimpse of its opponent suggests that an aspect of the fitness function we considered somewhat trivial may have played a larger part then we thought. A robot with the blob in view, but not at the center of its view had its fitness halved during that time. Our robots strived valiantly to keep their doppelgangers centered in their view. A short period of time with the opponent centered in view made up for a larger chunk of time spent backed up against the wall losing fitness.

The problem of the robot backing into the wall leads to perhaps the most complicated problem that faced this experiment. Despites our emphasis on speed we were unable to substantially tweak the fitness function. Runs still took to long. Our fitness function rewarded a robot for speed, for seeing the opponent, and for maintaining that opponent centered in view. It punished robots for running into the wall. Although only four variables were in play, they needed to be finely tweaked if we hoped to gain more complex behavior then we observed.

Again take the solution of turning till the opponent was centered in view and then backing up. Because we punished hitting the wall as strongly as we punished failing to see the blob the robot developed a strategy that ensured fitness by guaranteeing it came into visual contact with the other robot. Getting some fitness proved more important than getting a lot, so high risk strategies that attempted more complicated means of tracking their opponents failed. They may have received more fitness during some generations, but during some they may have received none. Getting some fitness before hitting the wall worked, reacting interestingly to the other robot didn't.

The robot's backward motion also made sense given the fitness function. Because we rewarded the robot for moving, the only way it could get fitness was with the blob in view while on the go. Backing up again ensured that fitness would be obtained.

Interestingly, the robot's optimal strategy of backing up while turning to get the other robot in view and then keeping it there was replaced in later generations with a much less interesting strategy. This strategy merely circled about, avoiding the walls. This second strategy totally replaced the previous strategy. We believe that the emergence of this strategy served as a counter strategy to the initial one. The total demise of the strategy is probabally an example of a punctuated equillibrium style evoltionary occurence. Instead of the gradual emergence of a new species a new species, slowly taking advantage of weakensses in the old species, leaps to life [3].

Tweaking the fitness function proved quite a difficulty, tweaking other parameters proved equally difficult. Population size, mutation rate and crossover rate as well as the elite rate all were set haphazardly. We tried a number of experiments with different values, but none made a substantial impact on the controller. The difficulty with fine-tuning these values proves to be the lack of strong causation between performance and the parameter. With the fitness function the observed behavior of the robot can be explained quite effectively by simply taking a look at dominant strategies. Other parameters proved more difficult to account for. Should or population have been 50 or 500? A population larger than 50 never occurred to as we performed the experiment. Should we have? These parameters may have hurt the performance of our experiment.

Lastly, the bipartite tournament may have weeded out more specialized strategies by facing them up against erratic opponents. General strategies that "just got some fitness" were able to survive despite a wide array of opponents. More tailored gos at the problem could have been knocked out by controllers that simply couldn't be tracked based on the solution proffered by the given controller.

7 Future Work

Given the problems faced by our experiment we feel that, given more time, we would make the following changes:

7.1 Tournament Structure

A different tournament structure would probably open the door for a greater diversity of behavior. If each controller were to play multiple opponents general solutions would be met with solutions that got a tremendous amount on some runs and very little on others. This altered tournament structure would reward good general solutions and solid specific solutions, injecting elements of both into the population. Chance encounters with rogue robots would also be less likely to knock out strategies that would later become dominant. In short, altering the tournament structure so that each robot faces multiple opponents would undoubtedly increase the complexity of behavior of our controllers.

7.2 Two Evolutionary Tracks

The use of two evolutionary tracks, one for the green robot and one for the red robot might also generate more complex behavior. Counter-strategies might emerge more readily under this model. As it stands our entire population converges toward one solution. Controllers generate strategies where both robots will get fitness in part because the best solution for the population includes cooperative behavior. Using two different evolutionary tracks will generate a more interesting history of behavior. As one population converges toward a solution, the other must react to that convergence by adapting to the converging population in a style of growth similar to puncuated equillibrium [3].

7.3 Explicit Payoff Matrix

One promising area of future work would depend on a more explicit creation of a payoff matrix between two competing agents. Although our paper represents an important step towards evolving cooperation in a dynamic world, inclusion of an explicit game theoretic scenario could easily be designed to further facilitate coordination of actions between agents.

As discussed previously, the Prisoners' Dilemma is the quintessential game theory problem. Furthermore, several attempts have been successful at evolving agents who are effective at playing that game. An interesting experiment could be made from encoding the cooperate/defect payoffs into an actual physical environment. One method of creating a world that represents the Prisoners' dilemma would be to create a large grid structure within the world. Certain squares would be associated with "cooperating," where the robot's fitness would be increased if its opponent was also in a cooperating grid, while it would be decreased if the opponent were in a cheating grid. Such a layout would allow agents to attempt to solve the problem in a physical, real world setting. As the problem can already be solved satisfactorily in an abstract environment, the possibilities presented by a real implementation could be very exciting.

Another possibility for explicitly encoding payoff functions into the robot's fitness could be through IR messages passed between the robots. The robots could be engaged in a Prisoners' Dilemma situation at every time step, and could broadcast their response to their opponent. The results of a round in the game could then be linked to the behavior of the robot, such as overriding their network output and motor control. For example, a robot who 'loses' a round may find themselves turning away from the other robot or slowing down - behavior that would lower their overall fitness. Combining the explicit interdependency with the implicit relationship presented here could lead to very interesting results. Robots would have to learn both how to move and "communicate" in a cooperative fashion to maximize fitness.

8 Conclusion

Our experiment attempted to evolve cooperative behavior. We implemented cooperative behavior through the use of a competitive fitness function. This fitness function was dependent upon the interrelationship of two robots. Through the use of the fitness function we evolved robot controllers.

Our evolution was mildly succesful. Although the behavior we obserbed could be termed cooperative, the simplicity of the robots' behavior suggests that an explicit interrelationship between the two robots' fitness is necessary. The evolution of cooperation will prove to be an interesting area for further research. Uncovering the enviromental conditions necessary for the emergence of cooperative behavior will be a difficult task. Through it we may learn a good deal about the nature of intelligence in general.

References

[1] Peter J. Angeline and Jordan B. Pollack. Competitive environments evolve better solutions for complex tasks. In *Proceedings of the 5th International Conference on Genetic Algorithms (GA-93)*, pages 264–270, 1993.

[2] Robert M. Axelrod. *The Evolution of Cooperation.* Basic Books, 1984.

[3] P. J. Darwen and X. Yao. The exploitation of cooperation in iterated prisoner's dilemma.

[4] John J. Grefenstette. Learning decision strategies with genetic algorithms. In Klaus P. Jantke, editor, *Analogical and Inductive Inference (AII'92) Proc. 3rd International Workshop, Dagstuhl Castle, GDR, October 5–9 , 1992*, volume 642, pages 35–50. Springer-Verlag, 1992.

[5] Francis Heylighen. Evolution, selfishness, and cooperation: Selfish memes and the evolution of cooperation. *Journal of Ideas*, 2:70–84, 1991.

[6] S. Nolfi. Using emergent modularity to develop control system for mobile robots, 1997.

[7] S. Nolfi and D. Floreano. Learning and evolution, 1999.

[8] K. Sims. Evolving 3d morphology and behavior by competition. In *Proceedings of the Artificial Life IV Conference*, 1994.