22

Hunting in an Environment Containing Obstacles: A Combinatory Study of Incremental Evolution and Co-evolutionary Approaches

Ioannis Mermigkis and Loukas Petrou

Aristotle University of Thessaloniki, Faculty of Engineering, Department of Electrical and Computer Engineering, Division of Electronic and Computer Engineering, Thessaloniki Greece

1. Introduction

The field of evolutionary robotics has drawn much attention over the last decade. Using a very general methodology (Evolutionary Computation –EC) and with minimal supervision, it is possible to create robotic controllers that cover a vast repertoire of behaviors, either in simulation or real environments, for commercial, pure research or even entertainment purposes.

The strong point of evolutionary robotics is that if the fitness criterion is defined properly, it is possible to evolve the desired behavior regardless (or at least in a big degree) of other parameters such as Genetic algorithms properties (population size, mutation type, selection function) or even controller specific properties (in case of neural networks, even the architecture can prove irrelevant to the success of the algorithm).

An important feature is the ability of Evolutionary Algorithms (EAs) to find solution simpler than the corresponding hand-made ones. For example, in a garbage collection task, Nolfi (1997) discovered that the Genetic Algorithm (GA) evolved the network to use two distinct modules for a task that hand-crafted controllers would need to define four.

This ability however shows also the limitations of EAs to tasks that are simple in concept. If the problem requires a set of behaviors to be available and switch between one another, a simple GA will not find a successful solution. For this reason, a collection of techniques named Incremental Evolution have been developed to create the possibility of evolving multiple behaviors in one evolutionary experiment.

We shall attempt to evolve behaviors on two competing species in predator-prey setup for simulated Khepera (K-team, 1995) robots, in an area containing obstacles. The robotic controllers will be discrete time recurrent neural networks of fixed architecture and the synaptic weights will be subject to evolution. The evolutionary algorithm will be a standard GA with real value encoding of the neural synapses and mutation probability of 5% per synapse. The experiments will run exclusively on simulation, the Yet Another Khepera Simulator (Carlsson & Ziemke, 2001) respectively. The experimental setup, network architectures and genetic algorithms will be presented in detail in the following sections.

Source: Frontiers in Evolutionary Robotics, Book edited by: Hitoshi Iba, ISBN 978-3-902613-19-6, pp. 596, April 2008, I-Tech Education and Publishing, Vienna, Austria

The chapter’s structure is the following: Incremental evolution is defined in section 2 and the basic guidelines for successful fitness function definition are enumerated in evolutionary and co-evolutionary experiments. In section 3 the problem of hunting (and evading predator) in an environment that requires to avoid obstacles as well is presented. This problem requires combination of techniques as it requires various behavioral elements. Section 4 describes the setup of the experiments, regarding environmental elements, robotic sensors and actuators, robotic neural controllers and the genetic algorithm. It is also analyzed what challenges poses this environment compared to an empty arena and the cases of contradicting readings of the various sensors (the perpetual aliasing (Whitehead & Ballard, 1991) and poverty of stimulus problems).

Sections 5 and 6 present the results of the various experiments defined in section 4. In section 5 the behavioral elements that can be observed by looking at five best individuals are described while in section 6 the data taken from fitness (instantaneous and master) evaluation are presented in order to validate hypotheses made.

Section 7 concludes the chapter and future work is proposed in section 8.

2. Incremental evolution

The first problem that faces every evolutionary procedure is the so-called bootstrap problem: in the first generations of evolution, where best individuals are mainly an outcome of random generation, it is quite unlikely that individuals shall evolve to get an adequate fitness score that can discriminate them from the rest of the population.

Incremental evolution can cope with this kind of problems by describing a process of gradually (incrementally) hardening the evolutionary problem. This can be achieved with various ways

a) By directly linking the generation number to the fitness function, e.g. add more desired tasks to the overall fitness evaluation.

b) By making the environment harder as generations pass, e.g. add more obstacles in a collision free navigation task.

c) In the commonly used case of evolutionary neural training, train the network for one task and then use the final generation of the first pass as the initial generation to train the second task.

Nolfi and Floreano (2000) have stated that while incremental evolution can deal sufficiently with the bootstrap problem, it cannot operate in unsupervised mode and violates their principal of fitness definition being implicit. As the supervisor must define every step of the process, evolution cannot run unsupervised and scale up to better and unexpected solutions.

This argument confines the usability of incremental evolution to small, well-defined tasks. While this is a drawback for the theoretical evolutionary robotics, that visualize evolutionary runs that can go-on for millions of generations and produce complex supersets of behaviors, while being unattended, real robotic problems encourage the incorporation of small behavioral modules in larger, man-engineered schemas. These modules can be produced using several methods and evolutionary algorithms are as good as any. Togelius (2004) invented a method called incremental modular in which he defined modules in subsumption architecture (Brooks, 1999). The interconnection between modules was pre-defined and fitness evaluation proceeded for the whole system, while neural network evolved simultaneously.

2.1 Guidelines to design successful fitness functions

Designing the proper fitness function is fundamental for the success of the evolutionary procedure. While Genetic Algorithms (GAs) and other evolutionary methodologies work as optimization methods for the given fitness function, the definition of the proper function for the task at hand requires a lot of work. In previous article (Mermigkis & Petrou, 2006) we have investigated how the variation in the fitness function can produce different behaviors while the other parameters (network architecture and size , mutation rate, number of generations and epochs) remain the same.

In evolutionary systems, it has been stated that fitness functions should be as simple as possible (implicit) and describe the desired behavior rather than details about how to be achieved (behavioral). It is also better to calculate the fitness based only on data that the agent itself can gather (internal). This allows the evolutionary procedure to continue outside the pre-defined environment of the initial experiment and continue the evolution in real environments where external measurement isn't possible. These three qualities have been summarized by Nolfi and Floreano (2000) in the conception of fitness space.

2.2 Incremental evolution and coevolution

Several research groups have pointed out that evolving two species one against each other is a form of incremental evolution which falls into case (b) of the previous paragraph: If the competing species is considered part of the environment for the one species, then the progress of its fitness is considered hardening of the environment for the opponent and vice versa. This could work flawlessly if there hadn't been the phenomenon of cyclic rediscoveries, both reported in evolutionary robotics (Cliff & Miller, 1995a, 1995b, 1995c, Floreano and Nolfi, 1997) and evolutionary biology (Dawkins, 1996). Cyclic rediscovery, also known as red queen effect, is the tendency of competing species to develop qualities of previous generations in later stages, because these qualities can cope better with the opponent of the current generation. While several methodologies have been proposed to overcome this problem, such as hall of fame tournaments, the problem still exists in nowadays implementations.

3. Hunting in an environment that contains obstacles

The Predator – prey or hunt situation has been explored by different research groups (Mermigkis & Petrou, 2006), (Cliff & Miller, 1995a), (Floreano & Nolfi, 1997), (Buason & Ziemke, 2003), (Haynes & Sen, 1996) using different methodologies. However, in most cases the environment (or arena) of the experiment has been an empty space confined by walls with no obstacle contained within. In previous work (Mermigkis & Petrou, 2006) we explored the possibilities of such an experimental setup and watched the emergence of different kinds of behavior (and behavioral elements such as evasion, pursuit, lurking or pretence). In this paper we shall try to conduct the hunt experiment in an arena that contains square objects (Figure 1) and see how the emerging agents cope with this situation.

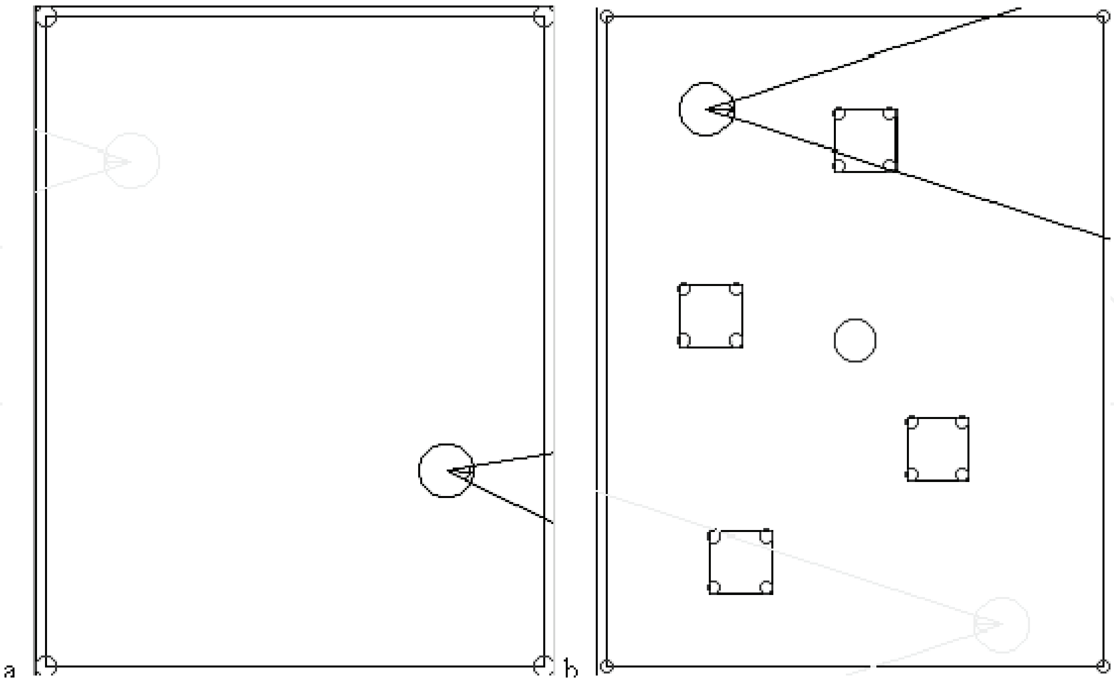


Figure 1. Arena and initial Positions. As every run consists of 4 epochs, the agents switch starting positions. A: Arena without obstacles. B: arena with obstacles

3.1 Need for simulation

The experiments concern the co-evolution of two simulated Khepera robotic vehicles. One vehicle (predator) evolves trying to catch the opponent (prey) while the prey's evolutionary target is to wander around the arena avoiding collisions with obstacles and the predator. YAKS (Yet another Khepera Simulator) (Carlson & Ziemke, 2001) has been adopted to simulate the robotic environment. The reason why simulation has been used is time restrictions: In the following chapter several experiments are conducted that last for 500 generations of 100 individuals. This leads to many hours of experiments that have to be spent and simulation is the best way to a) parallelize the conduction of experiments by spreading to several PCs and b) simulation is in general faster than conducting the experiment with real robots.

On the other hand, various research groups (Carlson & Ziemke, 2001) , (Miglino et al., 1995), (Jacobi et al., 1995) have shown that it is possible to evolve behaviors in simulation that can easily be transferred to real robots in few more evolutionary runs.

4. Experimental setup

4.1 Calculating fitness

Experiments are conducted in the arena depicted in figure 1. Fitness is evaluated in 4 epochs of 200 simulated motor cycles. In every epoch the two agents switch starting positions in order to eliminate any possible advantage by the starting position.

The Evolutionary algorithm (EA) adopted is a simple Genetic Algorithm (GA) applied on Neural Networks (NN) of fixed architecture. Christensen and Dorigo (2006) have shown that other EAs such as the (μ, λ) Evolutionary Strategy can outperform the Simple GA in incremental tasks, however we try to follow the experimental framework of (Mermigkis &

Petrou, 2006) in order to be able to make comparisons. In the same spirit, only mutation is applied on individuals.

```
Main{
  Generation 0:
    Create random populations A,B
    Calculate fitness A against individual B[0]
    Sort pop A (fitnes(A[0])=max)
    Calculate fitness B against A[0]
    Sort pop B
    Hall_of_Fame_A[0]=A[0]
    Hall_of_Fame_B[0]=B[0]
  Main GA Loop:
    for(generation=1;generation<nrOfGenerations;generations++){
    A'=create_new_gen(A)
    calculate fitness A' against B[0]
    Sort A'
    B'=create_new_gen(B)
    calculate fitness B' against A'[0]
    Sort B'
    Hall_of_Fame_A[generation]=A'[0]
    Hall_of_Fame_B[generation]=B'[0]
    A=A',B=B'
  }
}
create_new_generation(pop){
  for (elite=0;elite<nrOfElites;elite++){
    for (mut_ind=0;mut_ind<nrOfIndivids/nrOfElites;mut_ind++){
      pop[rOfElites+20*elite+mut_ind]=mutate(pop[elite])
    }
  }
}
mutate(individual){
  for(synapse=0;synapse<nrOfSynapses;synapse++){
    individual[synapse]+=mutation_probability*gauss_rand()
  }
}
```

Listing 1. Pseudocode of the Genetic Algorithm

The experiments consist of two populations competing against each other for 500 generations. Each population consists of 100 individuals. Fitness of population A is calculated by competing against the best individual of population B of the previous generation or the 10 previous generations.

The Genetic algorithm followed is shown in Listing 1: First two random populations are created and are evaluated one vs. one. From every generation, the 5 best individuals are selected and passed to the next generation. The remaining 95 individuals are produced by mutated copies of the 5 selected ones (19 copies per elite individual). Real-value representation has been chosen since binary encoding constrains synaptic values to predefined min and max levels. Mutation is produced by adding to each synaptic value a random number from a Gaussian distribution multiplied by 0.05 (the mutation probability).

4.2 Agent Hardware and Neural Controllers

The simulated Kheperas originally used the 8 infrared sensors and a rod sensor. The rod sensor is a kind of camera of 10 pixel resolution that can locate other agents equipped with rods. It is assumed that the rods are high enough so that the rod sensor can detect a robot even if there is a wall or other obstacle in the middle.

In order to strike out accidental contacts between the vehicles we define that contact is made if the predator robot touches the prey with the central front part (prey must be in the 4th or 5th pixel of the predator's rod sensor).

The rod sensor however doesn't return any info about how far the other vehicle is, only the relative angle of the two. It is possible that if the two vehicles are in the opposite sides of an obstacle, then rod sensor indicates opponent's presence and infrared sensors indicate contact with something. While there have been studies (see (Nolfi & Floreano, 2000) chapter 5 for a comprehensive review) that show that simple NNs can differentiate between objects based only on IR sensory patterns, it is possible that the agent's controller cannot tell whether there has been contact with the opponent vehicle or an obstacle. For this reason we have conducted another series of experiments in which we have added light sources on top of the simulated vehicles. This way the vehicle can detect the proximity of an opponent by using the 8 ambient light sensors.

Also, since the desired behavior has two distinct elements (collision free movement and evasion-pursuit) we have experimented with a simple NN with a hidden layer of 4 recursively interconnected neurons and with recurrent connection on the output neurons, and a NN that contains a hidden layer of two modules (modular architecture). Each module consists of a decision neuron and 4 value neurons recurrently connected to each other. Hidden neuron values are propagated to the output neurons only if the decision neuron has a positive activation value. Figure 3 shows the architecture of the networks used in this experiment.

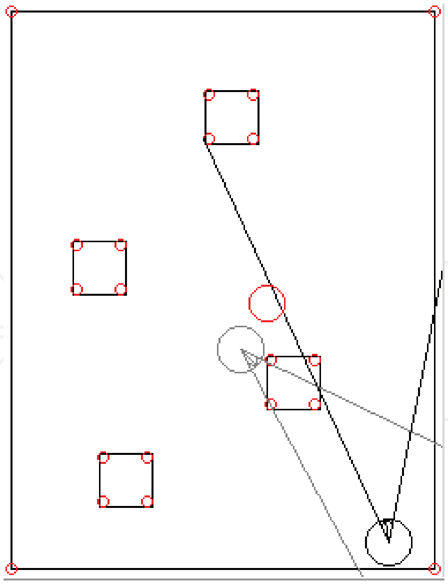


Figure 2. Predator robot (grey) stumbled into an obstacle considering it to be the prey(black)

The input layer of both networks contains one bias neuron that has fixed activation of 1.0 and neurons that map the several sensory inputs scaled so as the minimum value is 0 and the maximum 1.0. Hidden layer and output layer neuron activation function is the sigmoid function while the decision neurons use the step function. Hence, the value y_j of output neuron j at time step t is given by equation (1)

$$y_j[t] = d_M[t] \cdot \left(\frac{1}{1+e^{-A_j[t]}} - 0.5 \right) \tag{1}$$

$$d_M[t] = \begin{cases} 1, A_M[t] > 0 \\ 0, A_M[t] \leq 0 \end{cases} \tag{2}$$

$$A_j[t] = b_j + \sum_{i=0}^{I} w_{ij} x_i[t] + \sum_{k=0}^{K} w_{kj} y_k[t-1] \tag{3}$$

Where d_M is the activation value for the decision neuron of module M (if defined), x_i the value of input i, b_j the bias value for neuron j and w_{ij} the various weights for forward and recurrent connections.

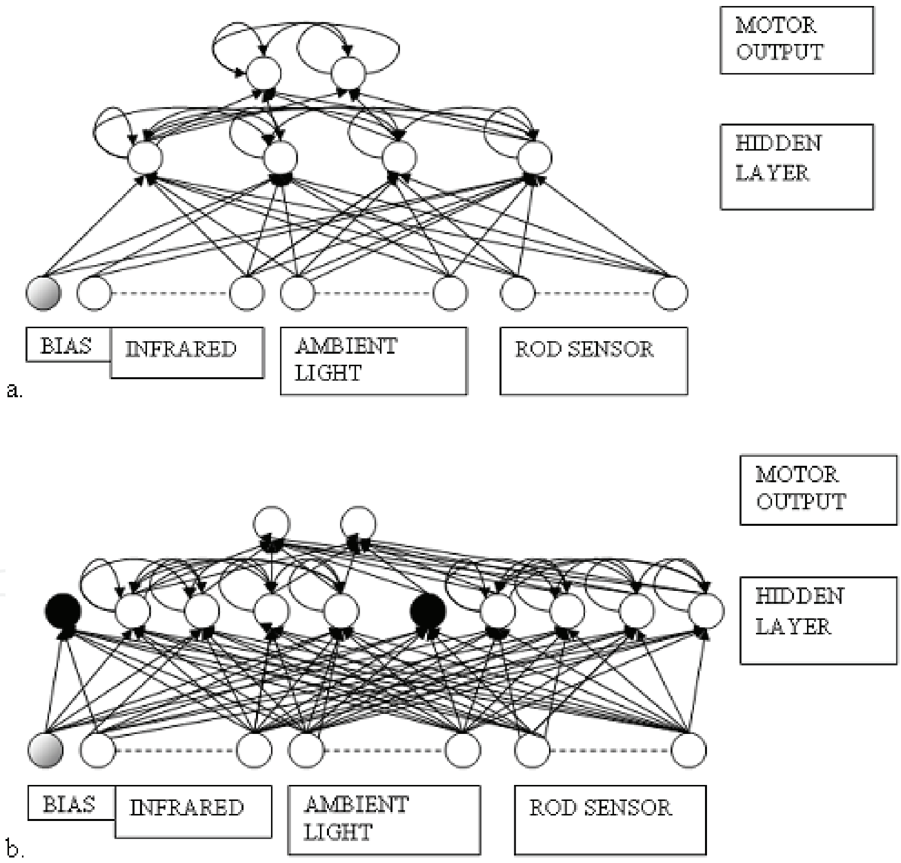


Figure 3. Network architectures tested a. Simple recurrent network with hidden layer of 4 neurons connected to each other. Ambient light input is not present in all experiments b. Hidden layer contains two modules with decision neuron. If decision neuron's activation is >0 then the module neuron's activation is propagated to the output. The output neurons are not recurrently connected

5. Evaluating co-evolution

5.1 Qualitative data in co-evolution

Two elements that are very common in co-evolutionary situation, both in evolutionary biology and evolutionary robotics are the arm-races and the cyclic rediscovery of strategies (a phenomenon commonly known as the red queen effect). The arm-races mean that as generations pass, opposing species constantly alter their strategies in order to beat their opponents. Arm-races can be depicted in instantaneous fitness graphs as oscillations which happen because a strategy x_1 that can beat an opposing strategy y_1 cannot beat strategy $y_2>y_1$. Since evolutionary algorithms slightly change winning strategies, the x_2 strategy that competes against y_2 is more likely to loose.

Cliff and Miller (1995b) validated this phenomenon in robotic simulation experiments and concluded that the instantaneous fitness graphs are not adequate to show the progress of co-evolving populations. Instead, they proposed the CIAO (Current Individual vs. Ancestral Opponents) graphs. A CIAO graph is a grid of pixel where pixel (x,y) contains a color representation of the fitness score of species A generation x competing against Species B generation y.

In an ideal arms-race, an individual $x_2>x_1$ that can beat an individual $y_2>y_1$ should also be possible to beat y_1 as well, leading to CIAO graphs similar to Figure 4a and 4b. However both in nature and robotics this doesn't happen. It is possible that y_2 looses to x_1. This means that it is likely that y_1 will re-appear as $y_3>y_2$ in order to compete against x_1 that reappears as x_3. This way y_2 will reappear again and the circle continues, leading to the phenomenon of cyclic rediscovery of strategies. CIAO graphs that correspond to the emergence of cyclic rediscovery have the tartan pattern similar to figure 4c.

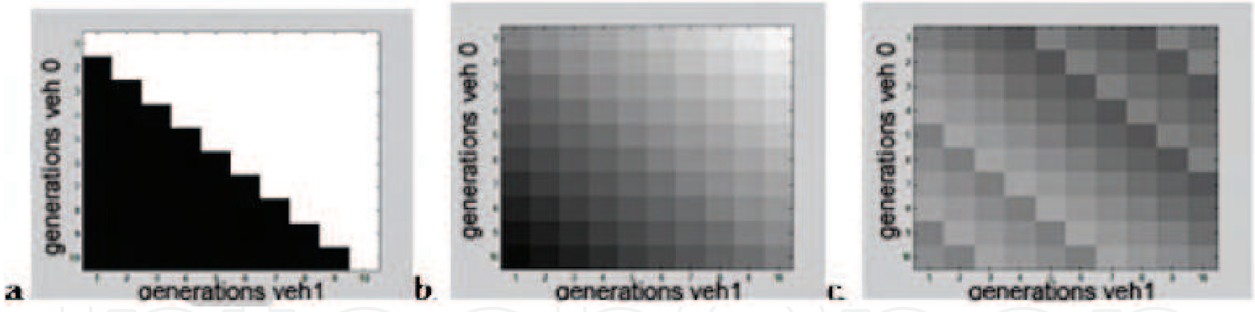


Figure 4 CIAO graphs patterns a: The idealized form for binary fitness function b: the idealized form for proportional fitness function c: the tartan patterns that indicate cyclic rediscovery of strategies

In order to reduce the Red Queen effect's impact, Nolfi and Floreano (1998) proposed the Hall of Fame tournament: Fitness of and individual x of Species A must not only be calculated against opponent of just the previous generation but also against more ancestral opponents. Ideally, fitness should be calculated against all ancestral opponents, in what Floreano and Nolfi call the Master tournament. However, such an evaluation can make the evolved task too hard and paralyze the evolutionary process, as no viable solution can be found. In the experiments presented here, fitness has been evaluated against previous generation best opponent (tournament depth 1) and against the champions of the previous 10 generations (tournament depth 10).

Since CIAO graphs can give only a rough icon of the evolutionary process, Floreano and Nolfi have proposed the Master Fitness measurement, which is the average fitness of an individual against opponents of all generations.

5.2 Methodological approaches of the Predator-Prey experiment

In order to evolve his virtual creatures, Sims (1994) conceptualized the idea of two populations competing against each-other. He experimented with various competition combinations (All vs. All, All vs. best of previous generation, random selection of opponent from previous generation) and concluded that the all vs. best schema could give the best results. Although he didn't analyze the dynamics of the evolutionary process per se, he reported that *"interesting inconsistencies"* could be reported, referring to similar agents evolving again. On the contrary, Cliff and Miller (1995a, 1995b, 1995c) studied co-evolution of predator and prey simulated robotic agents giving emphasis on game-theoretical and methodological aspects.

Floreano & Nolfi (1997), Nolfi and Floreano (1998), conducted similar experiments in real and simulated Khepera robotic miniatures and shown that the Red Queen Effect can be partially anticipated by calculating fitness against more ancestral opponent and not only the last one. This methodology provides more stable behaviors. They also experimented with arenas containing obstacles and drawn conclusions regarding the emergence of static behaviors like obstacle avoidance and how this affects the co-evolutionary process. Floreano et al. (2001) have also studied the possibility of neural networks that modify the synapses in runtime using Hebb rules, instead of the commonly used gene-coded synapses. They concluded that Hebbian modified neurons (or plastic neurons, as they call it) allow the predator to include a wider repertoire of behaviors.

Apart from the coevolutionary methodology (evolving population against the opponents best ancestors), the predator prey experiment has been handled using conventional evolutionary methodology. The most popular variation is that the predator behavior is evolved against prey of standard controller. E.g., Haynes and Sen (1996) used Genetic Programming (GP) to evolve a team of predators against a prey that had a hand-coded controller. Shultz et al (1996) conducted a variation of this experiment where the predator robots (here called the shepherds) tried to lead the prey (here sheep) into a predefined place in the arena (the corral). Potter et al. (2001) extended this experiment by adding another robotic agent (the fox). This experiment has rich dynamics as there are multiple objectives that must be optimize by the evolutionary process. Similar methodology has been used by Nietschke (2003) who evolved neural controllers for simulated Khepera vehicles if predator groups trying to immobilize a prey agent. The prey agent uses static obstacle avoidance controller while the predators could either share the same genotype, or evolve in parallel.

The paradigm of coevolving populations has been shown (Nolfi & Floreano, 1998) to give better solutions than the single evolution of predator behaviors. It also allows for game-theoretical assumptions to be made. However, the main disadvantage is the large number of fitness evaluations that need to take place using this methodology: Regarding the tournament's depth, the method needs 2xdxG evolutionary runs, where G is the number of Generations and D the tournaments depth. Furthermore, if the experiment includes a third robotic species (i.e. the fox) or heterogeneous robotic teams, the fitness evaluation becomes too much complicated.

Other groups have issued the best combination of individuals to test against. Rosin and Belew (1996) have proposed the method of Shared Sampling in which not only the best individual but also individuals with rare genotypes are preserved in each generation.

Stanley and Miikkulainen (2004) and Gomez and Miikkulainen (1997) have conducted co-evolutionary experiments of Battle for food and Predator-Prey in an arena with obstacles respectively. They have used a methodology that evolves not only the network weights but the architecture as well.

6. Results and Findings

We have conducted experiments using various agent setups:

A. recurrent neural network,

B. recurrent network with ambient light sensory input,

C. neural network that contains a hidden layer of two modules (modular architecture) with ambient light inputs.

For the three combinations we have tried the following experimental setups:

1. Fitness function (FF) of hunt experiment, defined by equations (4), where *T* is the total duration of the experiment and T_C the Time-to-Contact. Initial populations are randomly generated.

$$f_{PREY} = \frac{T_C}{T} \quad f_{PREDATOR} = 1 - \frac{T_C}{T} \tag{4}$$

2. Prey’s fitness function is a combination of collision-free navigation and evasion, as seen in equation (5), where v_1,v_2 are the motor velocities normalized in [0,1] (0:full backward, 1:full forward), max(IR) the maximum value of IR sensors and T the total duration of the experiment and T_C the time until contact (with predator) is made. In simple Obstacle avoidance calculations T_C=T. Predator’s FF is the same as in equation (4). Initial population is randomly generated.

$$f_{ov} = \frac{1}{T}\int_{t=0}^{T_C} I(v_1 + v_2 - 1)\sqrt{1 - |v_1 - v_2|}\,dt,\ I = \begin{cases} 1, \max(IR) < \frac{1000}{1023} \\ 0, \max(IR) \geq \frac{1000}{1023} \end{cases} \tag{5}$$

Equation (5) describes obstacle avoidance elements that favour forward movement, since the element (v_1+v_2-1) has value of -1 in case of full backward motion (v_1=v_2=0). We tried another variation where the element v_1+v_2-1 was replaced by its absolute value |v_1+v_2-1| in order to include the possibility of backward motion but didn’t observe the emergence of backwards motion (in prey robot). Hence, we didn’t measure this case (of favouring backwards motion).

3. Using equations (4) , but with initial population being the last population of a previously run GA aiming to evolve collision free navigation with FF given in equation (5).

6.1 Behavioural elements – strategies and trajectories

From the methodological point of view, an easy mistake would be to present strategies categorized by the various fitness functions or vehicle controller. Since the task at hand and the environment is common, the successful solutions will be similar for all cases.

As can be seen by observing the produced behaviours, the problem to solve for both agents is complex and includes several distinct stages. A prey must scan the area avoiding been close to predator. The predator’s problem has 2 distinct phases: First the predator must have a strategy to approach the prey and then adapt the relative position to make the "strike". These two elements are not necessarily combined as can be seen in figure 5 where predator is near the prey but cannot strike it.

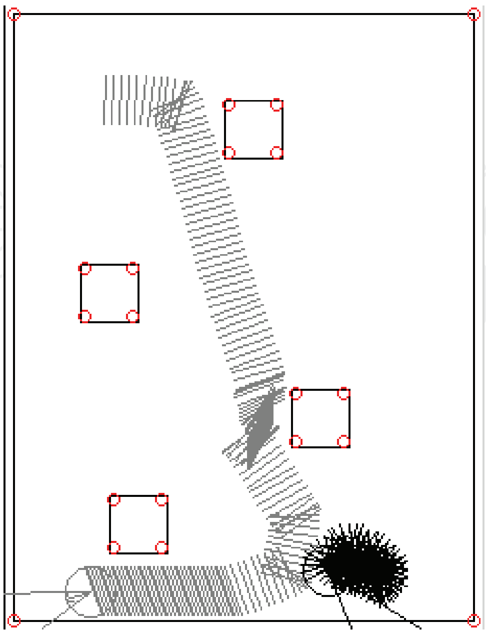


Figure 5 Setup 3A, tournament depth 10, generation 99: Predator approaches prey but fails to strike and turns away

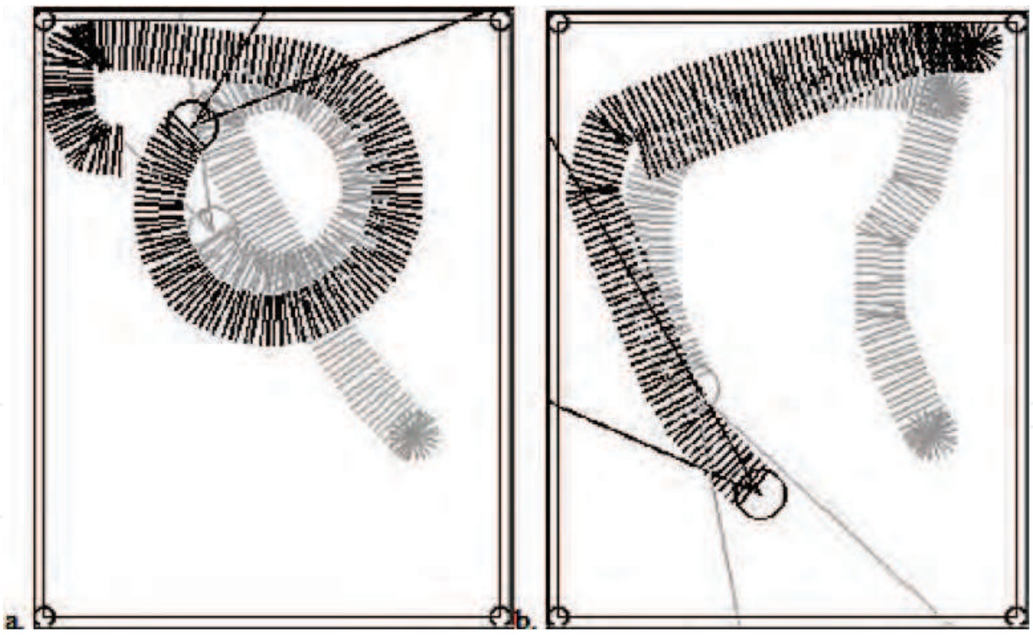


Figure 6 Trajectories detected in obstacle free arena a. Prey (black) moves forward in open circle b. Prey moves backwards trying to escape a predator, Predator(grey) moves against the prey and periodically rotates on spot in order to re-locate the prey robot

When conducting experiments in an empty arena, it was a common strategy for the prey robot to adopt backwards motion. Another dominant strategy was circular motion in wide circles in order to locate the opponent, a strategy adopted by both predator and prey. Figures 6a and b show trajectories corresponding to the above mentioned strategies.

Predator's strategy also always included the rotate-on-spot tactic where the predator rotated until rod-sensor indicated prey in the visual field.

The obstacles of the arena pose new challenges to the evolutionary procedure. Moving in open circles is not a strategy that can consistently be followed as the obstacles will interrupt the open circle.

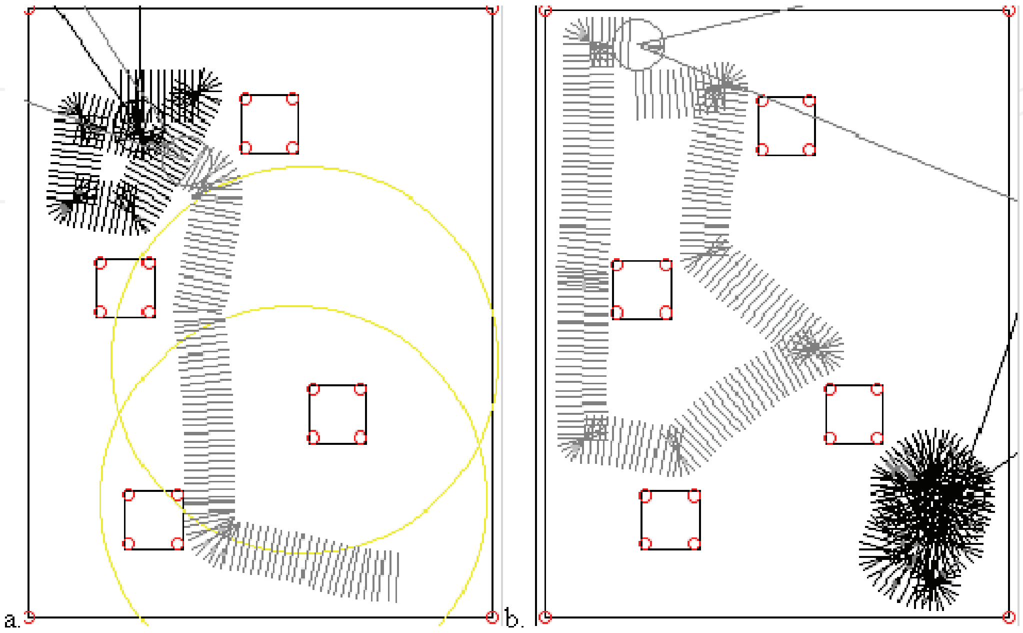


Figure 7. Using obstacles as landmarks: a. Successful approach, b. Unsuccessful guidance

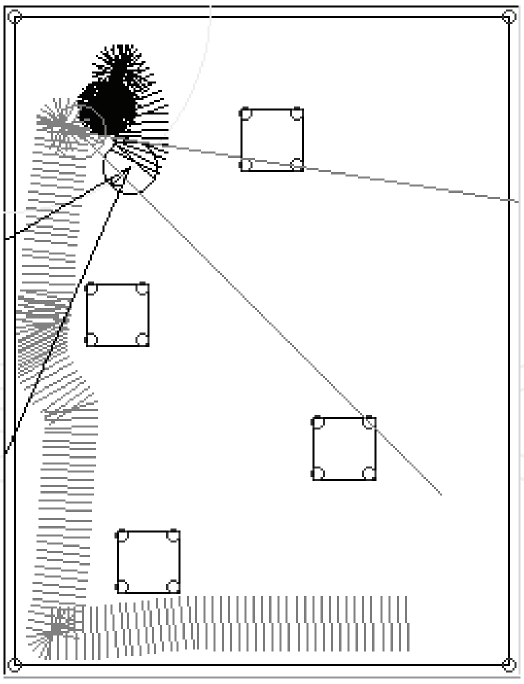


Figure 8. Wall Following approach. When predator is between wall and obstacle, it slides colliding with the wall

Regarding predator strategies, there is another problem: The prey robot is initially outside the sensory range of the predator. In fitness definition of experiment cases 1 and 3, where prey isn't granted fitness for being in motion, prey can achieve fitness simply by staying immobile. And since predator's fitness, in case 1 doesn't include any navigation elements, it is likely that bootstrapping problems will appear as in early generations the fitness of the predator remains 0.

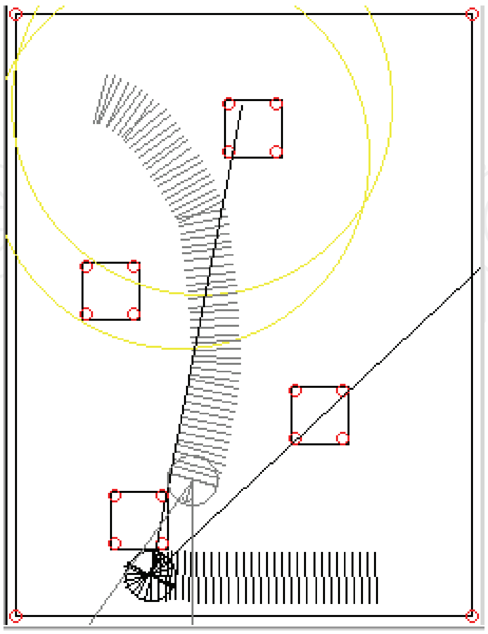


Figure 9. Prey hiding behind the obstacle. When predator tries to approach, stumbles in the upper corner of the obstacle

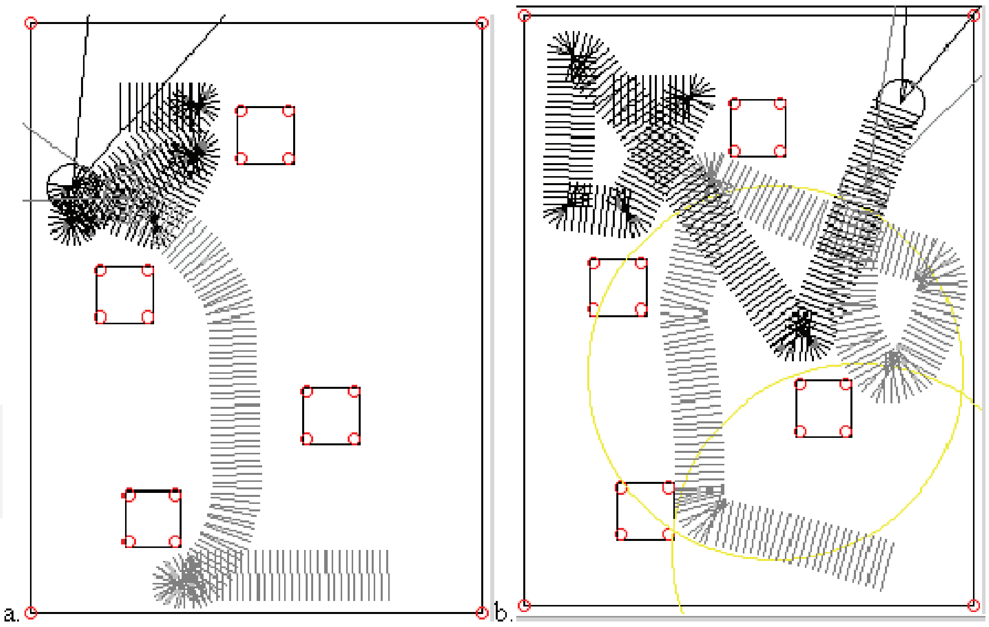


Figure 10. Conventional pursuit – evasion tactics. Contrary to previous examples, predator is based more on the rod sensor to locate the prey than to landmarks and ambient light readings

The evolutionary procedure has however found ways not only to produce obstacle avoiding strategies but also to use the obstacles in various ways. Figure 7 shows how can obstacles be used as landmarks to guide the predator near the prey: As prey wanders in the area confined by the two square objects, predator uses obstacles as landmarks to be guided in the prey chamber. Figure 8 shows a variation of this strategy, where predator follows the outer wall of the arena to encircle the prey.

Another interesting strategy that utilizes obstacles is adopted by the prey: When fitness doesn't explicitly prevents contact with an obstacle, it is quite common to see the prey vehicle collide in the lower corner of an obstacle and monitor the above free area. In this way, the predator robot cannot approach successfully. Figure 9 shows this form of hiding. It is also the only tactic that can prove effective against a predator that has developed collision free navigation. Another parallax is adopted by the predator that sticks to an obstacle's corner waiting for the prey to pass near by. Conventional strategies like seeking or pursuit are shown in figures 10a and 10b.

6.2 Poverty of stimulus

Rod sensor readings are confusing for both agents as they indicate the position of the opponent but doesn't indicate if there are obstacles in the way. It has been mentioned above that this can cause agent to consider collision with opponent while they have collided with and obstacle.

Introducing the ambient light readings changes totally the way agents react. In all variations with ambient light readings, the agents use the rod sensor less, while depending on arena landmarks, internal dynamics and light readings to navigate upon or away the opponent. Also, using ambient light allows the predator to have more options in the "strike" phase of pursuit. It has been observed in cases where predator contacts the prey back-to-back, that using light reading it rotates on spot and strikes the prey. If light sensors are not used, it this case the predator navigated away from the prey.

It has also been observed that the modular neural network architecture evolution produces more awkward behaviours for both agents. This is a common problem is evolutionary robotics: When a certain architecture can solve a problem, adding more elements in the phenotype produces a larger genotype space (Harvey, 1997) that is harder for the evolutionary algorithm to optimize.

6.3 Measuring and evaluating

Observation of the produced behaviours is the most interesting part of an evolutionary experiment and the most revealing concerning the ability to evolve behaviours and solve problems with minimal resources (naive neural architectures, poor and noisy sensory inputs such as the IR sensory inputs). Yet, there is always need to have qualitative metrics of an experiment to be able to monitor the emerging evolutionary dynamics.

Measuring co-evolutionary experiments is, as mentioned before, more difficult than measuring static evolutionary ones. Instantaneous fitness cannot show much about the progress of the experiment, as evolving sophisticated behaviours on one species causes fitness drop on the other. Master fitness, on the other hand is the common denominator of all behaviours, but can be very low (compared to instantaneous fitness levels) when a specific strategy that can beat all opponent strategies cannot be found. Figure 11 shows instantaneous fitness variation for predator and prey in experimental setup 1 (simple prey fitness function form).

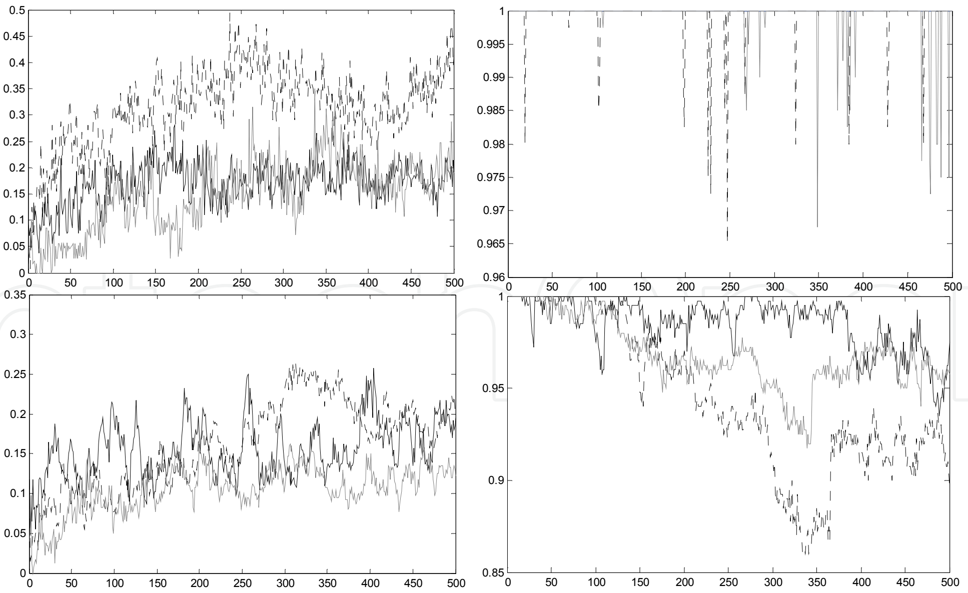


Figure 11. Instantaneous fitness function variation for predator (left hand) and prey (right hand) against number of generations for experiment setup 1 . The top plots are from tournament depth 1 experiments and bottom from tournament depth 10. Black: setup A (simple NN) Dotted: setup B (simple NN, ambient light input) and grey: setup C (Modular architecture, ambient light). All plots are average of 4 experiments with same setup

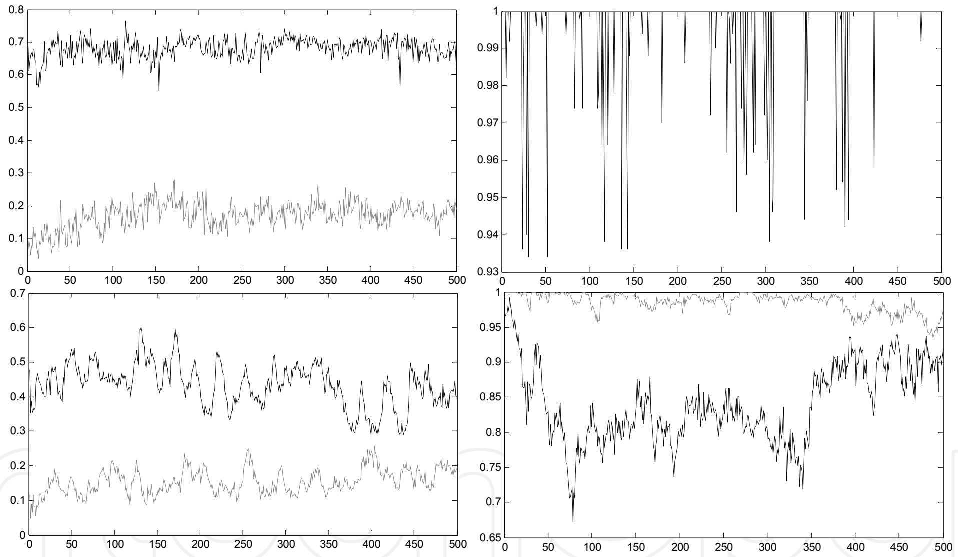


Figure 12. Instantaneous fitness for predator (left part) and prey (right part) in arena with obstacles (grey) and arena without obstacles (black) for experimental setup 1A. Upper Part: tournament depth 1. Lower: Tournament depth 10

Instantaneous fitness shows that the environment is totally favourable for the prey, as best individual's instantaneous fitness seldom is lower that 0.95, especially in tournament 1 depth. Figure 12 shows the instantaneous fitness variation in experiment 1A in an arena with obstacles and another without obstacles.

While the comparison in figure 12 shows a distinctive advantage for the prey robot, the waveforms are similar which seems that the same dynamics that can be monitored in a simple predator-prey setup (no obstacles) can be found when obstacle avoidance must also emerge.

Figure 11 partially proves the poverty of stimulus hypothesis as the predator that used the ambient light sensor gathers more fitness. It also seems that the modular architecture didn't cope well with the problem. The reason is that since the gene space is larger, it takes more generations and bigger population sizes to cope with the problem at hand.

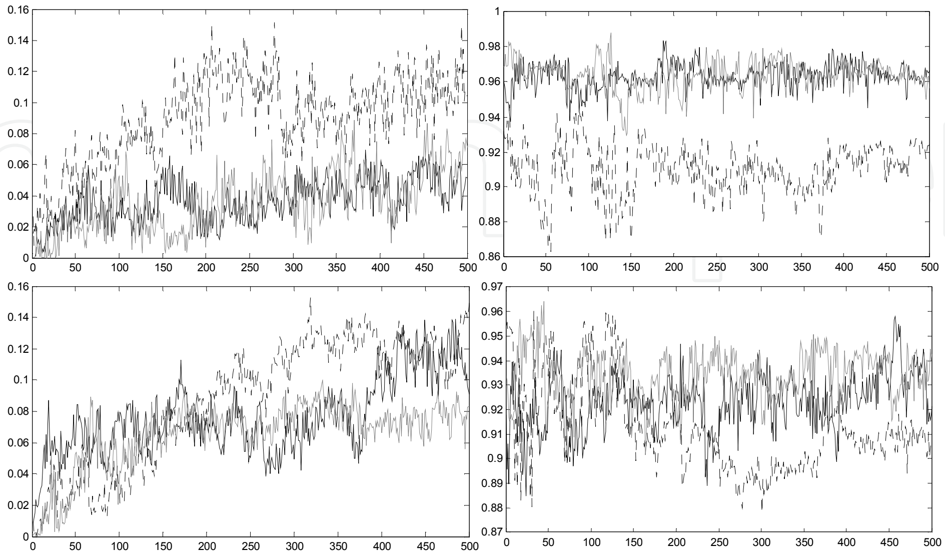


Figure 13. Master fitness comparison for plain fitness function definition using different neural setups (experimental setups 1A-1C). Left side: predator – right side prey, upper part : tournament Depth 1, lower part : tournament Depth 10. Black: setup A (simple NN) Dotted: setup B (simple NN, ambient light input) and grey: setup C (Modular architecture, ambient light). All plots are average of 4 experiments with same setup. Values are average of 4 experimental runs

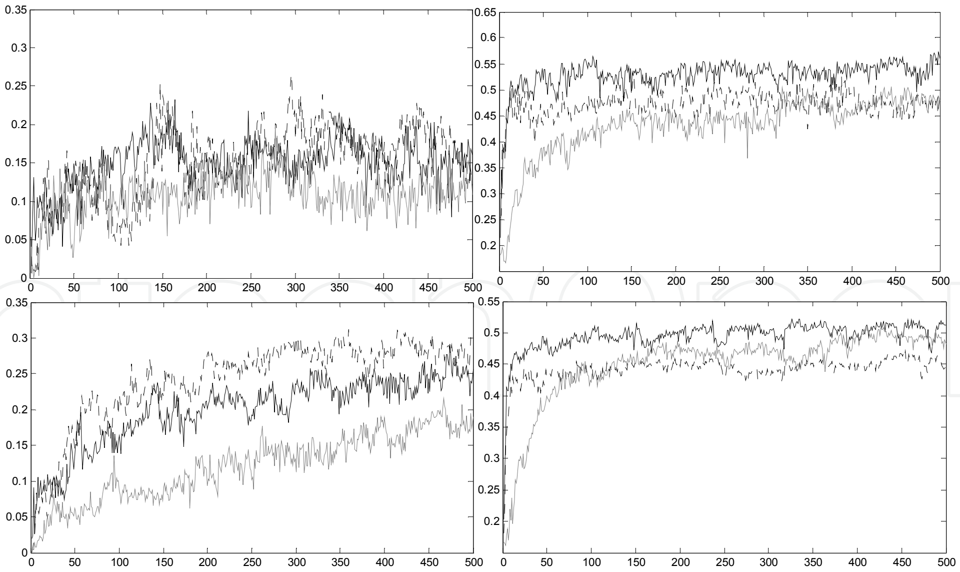


Figure 14. Master fitness comparison for fitness function definition 2. Plot specific details same as figure 13. Values are average of 4 experimental runs

Master fitness function comparison can show whether this is a general case for the co-evolutionary experiment or simply the co-evolutionary dynamics have changed. Figure 13 shows the comparison for experimental setup 1 and figure 14 for experimental setup 2.

Figures 13 and 14 don't show a clear supremacy of setup B (plain NN – ambient light sensors) by comparing predator's master fitness in each case. In prey's master fitness we see that the corresponding deterioration is clearer, which means that this particular setup is more favourable for the predator agent. Modular architecture seems to have the worst score in both agents.

Figure 14 shows also that predator's fitness is higher in general when prey's fitness incorporates the obstacle avoidance element. The explanation for this is a common problem of evolutionary robotics: When separate behaviours must evolve in parallel in an agent, evolution can paralyze in early generations by lack of useful lifetime experiences. In garbage collecting task Nolfi (1997) and Ziemke et al. (1999) faced this problem since in early generations the Khepera agent didn't encounter objects in order to evolve the gripper handling behaviour. The solution to that was to program the simulator to put an object in front of the agent in the first generations.

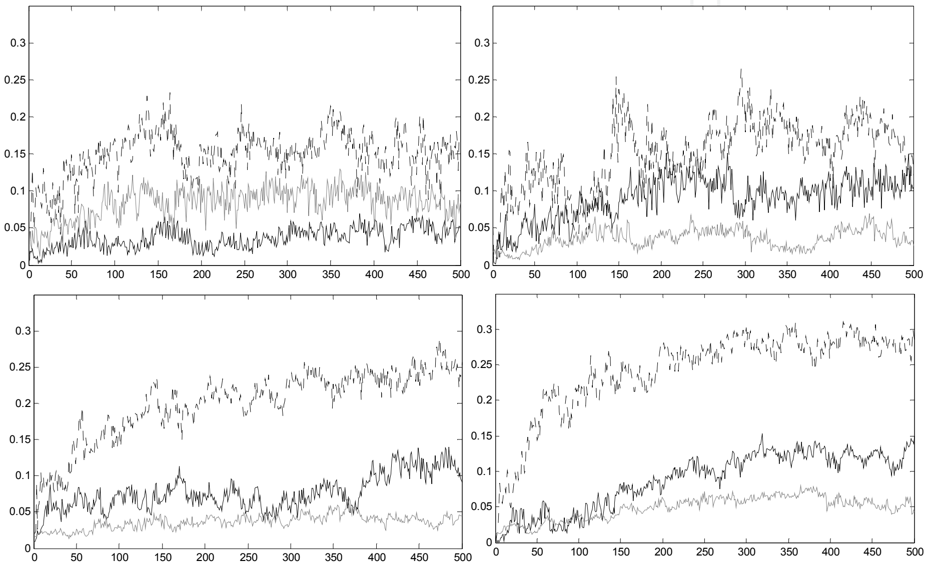


Figure 15. Comparison of the predator's master fitness for the 3 fitness function definitions setups left: Plane NN architecture, right : Plain with ambient light. Top : tournament 1 depth, bottom: tournament 10 depth. Black: fitness definition 1 (plain prey fitness), dashed: fitness definition 2, grey: Fitness definition 3 (plain with defined start generation)

In the previous section we saw that when prey's fitness doesn't favour obstacle avoidance and navigation in the arena, the favourable strategy for the prey was to stay immobile in the initial location or rotate on spot. This way it evaded being detected by predator agent's rod sensor. By forcing the prey to move around, useful rod sensory reading are generated for the predator causing master (and instantaneous) fitness to rise. The ascending trend in both predator's and prey's master fitness can be explained by the emergence of more refined obstacle avoidance.

Figure 15 seeks to give an answer to the initial question of the experiments presented in this chapter, which incremental evolution method is the best for the predator-prey task in the arena with obstacles. By comparing the master fitness variation of all setups (excluding the modular architecture which was proved inappropriate for the task) it seems that regarding predator emerging behaviours, fitness definition 2 gives the best results.

7. Conclusions

The possibility to provide an experimental framework for evolutionary biology and evolutionary game theory are the main strengths of the coevolutionary methodology. E.g., Cliff and Miller (1995a) rationalized the usability of co-evolutionary experiments with robotic agents in order to explain natural phenomena such as the emergence of protean behaviours in animals that usually are prey to others. By conducting experiments with simulated robots they were able to reproduce the phenomenon as it progresses and proceed with a game-theoretical analysis. As can be seen by the large number of corresponding publications in artificial life and evolutionary biology, evolutionary robotics greatly interacts with these areas.

The question whether coevolutionary methodology is capable of providing better robotic controllers than conventional evolutionary methods is quite hard to answer. First and foremost, the motivation behind coevolutionary experimentation is more to mimic biological procedures than produce competitive robotic behaviours. However, even when there are no optimistic results, co-evolution has the methodological advantage of being open-ended by achieving gradual complexification of the competing agents.

The experiments presented in this chapter proved that combination of behaviours can be done in competitive coevolutionary situations. The results (for the predator agent) were poor, something that can be explained by the fact that the two evolving strategies were controversial: One sought maximization of IR readings at some point (when contact with prey was made) while the other sought minimization of IR readings (collision avoidance). The predator's fitness has been extensively used as it shown variation across generations, something that was not the case for prey's fitness, which was high due to environmental advantage. Adding more sensory input improved the performance of the predator and the coevolutionary dynamics in general, while the hand made modular architecture proved poorer than the simple recurrent network.

The architectural evolution of neural controllers is something that must be further investigated: the experiments have shown that the architectures shown were under qualified to cope with the complexity of the combined tasks. However, in the experiments presented here, it was possible to see strategies like hiding or lurking emerge. In such strategies, the agent collided with an obstacle and then escaped by changing the direction of movement. Similarly, in many cases it was possible to observe the obstacles being used as landmarks to help the predator navigate towards the prey's initial location. Using landmarks, the predator didn't need to use the rod sensor except for the final phase before contacting the prey. This way the rod sensor reading that were confusing (as seen in section 4.2) when the prey was behind an obstacle were negated.

8. Future Work

This line of experiments has left open the point of what could be an optimal network architecture for problems of increasing complexity. Several researchers have conducted experiments in which the neural architecture was subject to change, either in genotypic (variable length GAs) (Harvey, 2001), (Husbands et al., 1998), or phenotypic (by using some developmental process during lifetime) level (Michel, 1997). Also, the ability to train the neural network during lifetime use delta or Hebb rule must be investigated.

Apart from the robotic controller internals, it is of great interest to study co-evolutionary experiments that include more that two species of species teams, clonal or aclonal. Experiments in these areas have shown the emergence of communication with or without dedicated channels (Quinn, 2001) or stigmergic collaboration (Caamano et al., 2007). By combining this feature with the multi-objective nature of multi-team situations, many interesting features can be studied, especially in the behavioural level.

9. References

Brooks, R.A. (1999). *Cambrian Intelligence*, MIT Press, ISBN 0262522632, Cambridge, MA

Buason G. & Ziemke T. (2003). Explorations of Task-Dependent Visual Morphologies in Competitive Co- Evolutionary Experiments, In: W. Banzhaf et al. (Eds.) *Advances in Artificial Life - The 7th European Conference on Artificial Life* (pp. 763-770). Springer-Verlag, Berlin,Heidelberg

Caamano P, Beccera, J.A., Duro , R.J, Bell F. (2007). Incremental Evolution of Stigmergy-Based Multi Robot Controllers Through Utility Functions. *KES2007/WIRN 2007*, Part II, Springer-Verlag LNAI 4693, pp. 1187-1195, ISBN 1889994606

Carlsson, J. & Ziemke, T. (2001). YAKS - Yet Another Khepera Simulator, In: Rückert, Sitte & Witkowski (eds.), *Autonomous minirobots for research and entertainment* (pp. 235-241), Paderborn, Germany: HNI-Verlagsschriftenreihe, ISBN 3935433069

Christensen A.L. , Dorigo M. (2006). Incremental Evolution of Robot Controllers for a Highly Integrated Task, S. Nolfi et al. (Eds.): *SAB 2006*, LNAI 4095, pp. 473–484. Springer-Verlag, ISBN 1595930108, Berlin Heidelberg.

Cliff & Miller (1995a). Co-evolution of pursuit and evasion I: Biological and Game-theoretical foundations, *Technical Report CSRP311*, University of Sussex, School of Cognitive and Computing Science

Cliff & Miller (1995b). Tracking the red Queen: Measurements of adaptive progress in coevolutionary simulations, In: Moran, F., Moreno, A., Merelo, J. J., and Chacon, P. (eds.) *Advances in Artificial Life,3nd European Conference on Artificial Life,* pp. 200-218, ISBN 978-3540594963

Cliff & Miller (1995c). Co-evolution of pursuit and evasion II: Simulation methods and Results, In: *From Animals to Animats 4: Proceedings of the Fourth International Conference on Simulation of Adaptive Behavior* , pp. 506--514. MIT Press, Cambridge, USA.

Dawkins, Richard (1996). *The Blind Watchmaker*. New York: W. W. Norton & Company, Inc.. ISBN 0-393-31570-3.

Floreano D. & Nolfi S. (1997). God save the red Queen: Competition in evolutionary robotics, In J.Koza et al. (Eds.), *2nd Conference on Genetic Programming,* San Mateo, CA: Morgan Kaufmann, pp.398-406.

Floreano D., Nolfi S., Mondada F.(2001). Co-evolution and ontogenetic change in Competing Robots, In M. Patel, V. Honavar, and K. Balakrishnan (eds.), *Advances in the Evolutionary Synthesis of Intelligent Agents,* Cambridge (MA): MIT Press, pp. 273-306, ISBN 978-0262162012

Gomez F, Miikkulainen R. (1997). Incremental evolution of complex general behavior. *Adaptive Behavior,* 5(3--4), pp. 317--342

Harvey I. (1997)Artificial Evolution for Real Problems, *5th Intl. Symposium on Evolutionary Robotics*, Tokyo April 1997. Invited paper. In: Evolutionary Robotics: From Intelligent Robots to Artificial Life (ER'97), T. Gomi (ed.). AAI Books

Harvey I. (2001). Artificial evolution :A continuing saga. In Takashi Gomi, editor, *Proc. of 8th Intl. Symposium on Evolutionary Robotics (ER2001).* Springer-Verlag LNCS 2219, pp. 94-109.

Haynes T. and Sen S. (1996). Evolving behavioral strategies in predators and prey. In Gerhard Weißand Sandip Sen, editors, *Adaptation and Learning in Multiagent Systems*, pp. 113--126. Springer Verlag, LNAI 1042, Berlin, ISBN 3540609237

Husbands, P. and Smith, T. and Jakobi, N. and O'Shea, M. (1998). Better Living Through Chemistry: Evolving GasNets for Robot Control, *Connection Science*, 10(4), pp185-210

Jakobi N., Husbands P., and Harvey I. (1995). Noise and the reality gap: The use of simulation in evolutionary robotics. In F. Moran, A. Moreno, J. Merelo, and P. Chacon, editors, *Advances in Artificial Life: Proc. 3rd European Conference on Artificial Life,* pp. 704--720. Springer-Verlag, Lecture Notes in Artificial intelligence 929

K-Team, S. A. (1995). Khepera: User Manual. Switzerland *http://ftp.k-team.com/khepera/documentation/KheperaUserManual.pdf*

Mermigkis I, Petrou L. (2006). Exploring Coevolutionary Relations by Alterations in Fitness Function: Experiments with Simulated Robots, *Journal of Intelligent and Robotic Systems*, Volume 47, Number 3, November 2006 , pp. 257-284.

Michel O., Clergue M., Collard P. (1997). Artificial Neurogenesis: Applications to the Cart-Pole Problem and to an Autonomous Mobile Robot. *International Journal on Artificial Intelligence Tools* 6(4):pp. 613-634

Miglino O., H. H. Lund, and S. Nolfi (1995). Evolving Mobile Robots in Simulated and Real Environments. *Artificial Life*, Vol. 2,4: pp. 417--434

Nietschke G(2003). Emergence of Cooperation in a Pursuit-Evasion Game, *International joint conference on Artificial Intelligence,* 2003, vol. 18, pp. 639-646

Nolfi S. & Floreano D (1998). Co-evolving predator and prey robots: Do 'arm races' arise in artificial evolution?, *Artificial Life,* 4(4), pp. 311-335

Nolfi S. & Floreano D.(2000). *Evolutionary robotics*, Cambridge, MA: MIT Press, ISBN 978-0262140706

Nolfi, S. (1997). Using Emergent Modularity to Develop Control Systems for Mobile Robots, *Adaptive Behavior* 5, 3-4, pp. 343-363

Potter M.,Meeden L. A., Schultz A. (2001). Heterogeneity in the coevolved behaviors of mobile robots: The emergence of specialists, *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence,* pp.1337-1343. Morgan Kaufmann

Quinn M.(2001):Evolving Communication without dedicated communication channels, In J.Kelemen and P. Sosvk, editors, *ECAL01*, pp. 357--366. Prague: Springer.

Rosin, C., Belew, R. (1996). New Methods for Competitive Co-evolution. *Evolutionary Computation* 5:1 pp.1-29

Schultz, A. C., Grefenstette, J. J., and Adams, W. L. (1996), RoboShepherd: Learning a Complex Behavior. *Proc. of the Robots and Learning Workshop (RoboLearn '96),* May 1996, Key West,Florida, pp. 105-113. NCARAI Report AIC-96-030

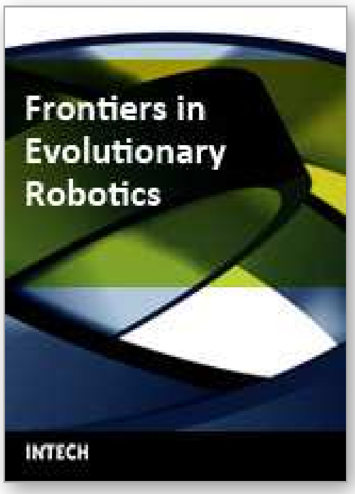
Sims K. (1994). Evolving 3D Morphology and Behavior by Competition, *Artificial Life IV Proceedings*, ed.by Brooks & Maes, pp. 28-39, MIT Press, ISBN 0262521903

Stanley K.O, Miikkulainen R. (2004). Competitive Coevolution through Evolutionary Complexification, *Journal of Artificial Intelligence Research* 21 pp. 63-100

Togelius J. (2004).Evolution of a subsumption architecture neurocontroller. *Journal of Intelligent and Fuzzy Systems*, Vol. 15, 1 : pp.15--20

Whitehead S.D and Ballard D.H (1991). Learning to perceive and act by trial and Error, *Machine Learnining,* Vol. 7, 1 : pp. 45-83.

Ziemke, T., Carlsson, J. and Bodén, M. (1999). An experimental comparison of weight evolution in neural control architectures for a 'garbage-collecting' Khepera robot, in Experiments with the Mini-Robot Khepera - *Proceedings of the 1st International Khepera Workshop*, Löffler, A., Mondada, F., and Rückert, U. (eds.), pp. 31 - 40. Paderborn, Germany. HNI.



Frontiers in Evolutionary Robotics
Edited by Hitoshi Iba

ISBN 978-3-902613-19-6
Hard cover, 596 pages
Publisher I-Tech Education and Publishing
Published online 01, April, 2008
Published in print edition April, 2008

This book presented techniques and experimental results which have been pursued for the purpose of evolutionary robotics. Evolutionary robotics is a new method for the automatic creation of autonomous robots. When executing tasks by autonomous robots, we can make the robot learn what to do so as to complete the task from interactions with its environment, but not manually pre-program for all situations. Many researchers have been studying the techniques for evolutionary robotics by using Evolutionary Computation (EC), such as Genetic Algorithms (GA) or Genetic Programming (GP). Their goal is to clarify the applicability of the evolutionary approach to the real-robot learning, especially, in view of the adaptive robot behavior as well as the robustness to noisy and dynamic environments. For this purpose, authors in this book explain a variety of real robots in different fields. For instance, in a multi-robot system, several robots simultaneously work to achieve a common goal via interaction; their behaviors can only emerge as a result of evolution and interaction. How to learn such behaviors is a central issue of Distributed Artificial Intelligence (DAI), which has recently attracted much attention. This book addresses the issue in the context of a multi-robot system, in which multiple robots are evolved using EC to solve a cooperative task. Since directly using EC to generate a program of complex behaviors is often very difficult, a number of extensions to basic EC are proposed in this book so as to solve these control problems of the robot.

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820

www.intechopen.com

Fax: +385 (51) 686 166
www.intechopen.com

Fax: +86-21-62489821