DOI: http://dx.doi.org/10.7551/978-0-262-31709-2-ch083

Evolving Plastic Neuromodulated Networks for Behavior Emergence of Autonomous Virtual Characters

Yuri L. B. Nogueira[1], Carlos Eduardo F. de Brito[1], Creto A. Vidal[1] and Joaquim B. Cavalcante-Neto[1]

[1]Department of Computing, Federal University of Ceará, Brazil
yuri@lia.ufc.br

Abstract

This paper addresses the problem of generating natural behavior of autonomous virtual characters. Inspired by the fields of Embodied and Enactive Artificial Intelligence, we postulate that natural behavior is the result of a coupling between the agent and the world where it lives, which leads to a coherence between its actions and its surroundings. In this work, we present the tools that we have been using to study that idea: a controller based on a plastic neuromodulated neural network, which is capable of molding itself to received stimuli; and a simple novel method for genetic encoding of artificial neural networks. We show the capabilities of the controller in generating interesting foraging behavior of an autonomous virtual robot, and discuss the advantages of its emergent characteristics when compared with traditional approaches.

Introduction

An important requirement for a user's sense of immersion in a virtual world is the way things move in the environment, so that it can be truly perceived as the real world. Zahorik and Jenison (Zahorik and Jenison (1998)) put it in this way: "When the environmental response is perceived as lawful, that is, commensurate with the response that would be made by the real-world environment in which our perceptual systems have evolved, then the action is said to successfully support our expectations".

The term "lawful" is appropriate to describe the behavior of inanimate elements that follow the physical laws. However, what is taken to be natural for the living elements, seems to be the opposite of simply following rules. The behavior of autonomous virtual characters plays an important role in virtual reality environments and obtaining such behavior in a natural and realistic way is still an open problem. We argue that unnatural behaviors are tipically obtained by a lack of connection between the character and the world around it in rule-based implementations.

In this paper we propose:

- A controller that is capable of adapting itself to the character's bodily constitution and to the characteristics of the environment, causing the emergence of appropriate foraging behavior in an autonomous virtual robot; and
- A simple novel method for genetic encoding of artificial neural networks (ANNs).

In the next section, we discuss the traditional methods used by the virtual reality community in generating behaviors of virtual characters, analyze the characteristics of those behaviors, and present some thoughts on the proposition that an emergentist approach could help to overcome the limitations resulting from approaches that attempt to explicitly model the cognitive abilities of the agents.

We also describe the tools that have been employed for studying the emergence of foraging behavior:

- A neuromodulated network for processing sensorial information and generating signals for motor action, which is capable of local modulation of synaptic plasticity; and
- A genetic algorithm (GA) evolution mechanism, which selects the most adapted neural networks.

We show our results with a simulated Khepera-like robot, equipped with sensors that input the robot's distance to a fruit or poison to the neural network. The robot's motor skills are moving forward and backward, and turning left or right. The experiments show that the robot was capable to learn how to make proper use of its sensors and motors in order to catch fruit and avoid poison, displaying a complex navigation control behavior. In conclusion, we present final discussions about the obtained results and future works on our study about natural behaviors of virtual characters.

Behavior Generation of Virtual Characters

Our basic assumption is that the behavior of a virtual character can only be called realistic if it reflects the details of the agent's bodily constitution, and has a close and logical association with the events taking place in the virtual environment. Accordingly, this work supports the idea that the realistic behavior should be obtained as the result of a permanent and intimate dialogue between the agent and its en-

vironment. This position is motivated by the Embodied (Anderson (2003)) and Enactive (Froese and Ziemke (2009)) approaches to AI, where intelligence is perceived as something that emerges from a close interaction among all the components involved in the cognitive phenomenon.

Here, we have a clear contrast with the classical strategies for the simulation of behavior, which concentrate their efforts on embedding the relevant aspects of reality within the agent's mind (in the form of models and "knowledge"), and internally calculating what appropriate action should be performed in a given situation (Shao and Terzopoulos (2007); García-Rojas et al. (2007); Gutierrez et al. (2007); Whiting et al. (2010); Orozco et al. (2011)).The main drawback of that approach is the great difficulty of maintaining a complete and updated description of the relevant aspects of reality, especially in the case of highly dynamical environments. So, a typical result is the strong attachment of the agent's behavior to the rules and facts stored in its mind, rather than an involvement and interaction with what is really happening in its surroundings. That generates a feeling of detachment, which we sometimes classify as robotic behavior.

The diagram in Figure 1a illustrates the traditional approach for behavior generation. That approach subdivides the problem into a cognitive level and a motor level, modeling each of those parts separately. In that view, the cognitive level works as a calculator of abstract facts, i.e., the manipulation of symbols already interpreted by the programmer, which correspond to a high-level perception of the world. That calculation should provide the correct (or desired) command to the motor level, which has a built-in set of behaviors fully described, such as walking, sitting or standing in line. The so called reactive agents are typically implemented based on this two-level modelling. To mitigate the problems inherent with this approach, some works (Pina et al. (2006); Schneider and Rosa (2009)) attempt to use techniques of machine learning to automate the process of behavior selection.

To avoid the detachment problem, we propose that the behavior of autonomous virtual characters should be generated through a methodology of emergence. More specifically, we suggest that efforts should migrate from detailed internal representations of reality and explicit descriptions of behavior, to the construction of appropriate conditions that would induce the coupling of the structure and dynamics of mind, body and environment, resulting in the emergence of intelligent behavior. In this way, we expect to obtain the degree of coherence between agent and environment which is required for natural behavior.

The notion of an emergent property of a system refers to a global characteristic of the system that cannot be found in any of its parts (Klaus and Mainzer (2009)). Accordingly, by emergent behavior we mean that the characteristics of the behavior that emerges are not described or encoded in any of the components that define the system: the logic of the controller, the anatomy of the body, or the environment configuration. In fact, the behavior must result from the combination of the properties of all those components.

The works based on the emergentist approach (Sims (1994); Chaumont et al. (2007); Nogueira et al. (2008); Panzoli et al. (2010); Palmer and Chou (2012)) use two basic elements to support the emergence of high-level behaviors: the uninterpreted signals received from the environment, and the elementary movements performed by the body parts. The internal dynamics of the body and mind of the agent should then modulate this information in order to establish a sensorimotor flow that coordinates the low-level movements in order to produce the emergence of high-level behavior. This idea is illustrated in Figure 1b. The key aspect that makes this process possible is that some components have a degree of plasticity, so that their structures and internal dynamics can be modified over time, in response to interactions with other components of the system, in an evolutionary dynamics.

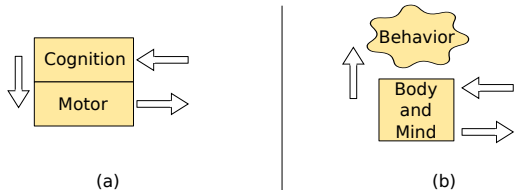

Figure 1: Approaches to behavior generation. The two-level modeling (a), and the emergentist approach (b).

Emergence phenomena tend to produce very specific configurations which are sensitive to small changes in the elements that constitute the system. This observation is consistent with the fact that animals of different species sometimes solve the same problem in radically different ways. This is illustrated by the great diversity of behaviors of locomotion, hunting, breeding, etc. that we find in nature. If the assumption that emergent phenomena are an important aspect of behavior, that would explain, in part, why the problem of realistic simulation has resisted for so long to approaches based on traditional AI techniques: the details of the situation are what determine the behavior of the agent. But these details are precisely what is left out in the process of abstraction inherent to representationalism. In other words, the attempt to simulate the behavior of an agent based solely on a high-level description of the situation in terms of goals and motivations of the agent, complex actions that the body can perform, and qualities of objects and features of the environment, leads almost inevitably to mechanical and unnatural behavior.

The Controller

Here, we present a controller for generating behaviors of virtual characters based on emergence. As we argued, to achieve the emergence of behavior, an essential factor is the

generation of a dynamics which is capable of modifying and adapting the controller to the body of the character and to the environment around it. Aiming at this goal, we chose to evolve a plastic neuromodulated neural network to generate the signals that control the motors of a virtual robot based on its sensory information. The neuromodulatory property brings lifetime adaptation to the network, while a genetic algorithm evolves the network providing adaptation throughout the generations.

The Neural Network

The neural network that we used is essentially a Continuous Time Recurrent Neural Network (CTRNN), whose neurons are modeled in the following general form (Beer (1995)):

$$\frac{dy_i}{dt} = \frac{1}{\tau_i}(-y_i + \sum_{j=1}^{n} w_{ji} f(s_j) + I_i) \quad (1)$$

where t is time, y_i and τ_i are, respectively, the internal state and time constant for each neuron i, w_{ji} is the weight of the jth input synapse of neuron i, s_j is the state of the neuron linked to the jth input synapse, $f()$ is the activation function of a neuron and I_i represents a constant external input to neuron i. In this work we used the same I for all neurons, simulating an external stimulus from a higher center such as the mesencephalic locomotory region of animal brains.

In addition to this standard dynamics, we chose to include a neuromodulatory feature in the neural network. As discussed by Soltoggio et al. (Soltoggio et al. (2007)), the neuromodulation plays an important role in neural substrates from invertebrates to the human brain, and is related to the induction of Late phase - Long Term Potentiation (L-LTP), a phenomenon of permanent growth of the synaptic contact in brain, causing synaptic stability, being a potential candidate for explaining memory functions involving neural wiring and, consequently, learning. Another mechanism of neural wiring related to neuromodulation is the Long Term Depression (LTD), the permanent decrease of the synaptic contact (Soltoggio et al. (2008)).

Soltoggio et al. proposed the use of a model of neuromodulation in T-Maze (Soltoggio et al. (2008)) and in bee foraging behavior (Soltoggio et al. (2007)). We use the same model in our work, in order to study the effects and advantages of such dynamics in the behaviors of our virtual characters. It consists of a CTRNN with two types of neurons: standard neurons, which are the processing units, and modulatory neurons, which are responsible for modulating the changes of weights in the synapses following the Hebbian rule. Figure 2 shows the modulation mechanism.

Equation 1 defines the activities of the standard and modulatory neurons. The signals of both types of neurons are computed according to that equation, with the activation function $f(x) = \tanh(x/2)$. However, only the input signals that come from standard neurons, and the weights of the respective synapses, are considered for the summation in that equation.

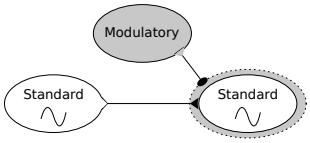

Figure 2: The network processing itself is made only with the standard neurons. The modulatory neuron determines the response of the Hebbian rule, mediating the amount of growth or decreasing of the synaptic weight.

The value of modulatory activation m acting on a neuron i is computed as follows:

$$m_i = \sum_{j \in Mod} w_{ji} \cdot \tanh(s_j/2) \quad (2)$$

where j represents all modulatory neurons connected to neuron i, w_{ji} is the weight associated with the synapse that links modulatory neuron j to neuron i and s_j is the internal state of modulatory neuron j.

Finally, the synapse's weight change is defined by the following equation:

$$\Delta w_{ji} = \tanh(m_i/2) \cdot \eta \cdot [Ao_j o_i + Bo_j + Co_i + D] \quad (3)$$

where Δw_{ji} is the amount of change in the synapse that links the standard neuron j to a neuron i, η is the learning rate, A, B, C and D are tunable parameters, and o_j and o_i are the activation function $\tanh(x/2)$ applied to internal states of neurons j and i respectively. Note that this rule is unstable if the parameters are such that Δw_{ji} is always positive or always negative. To avoid that, we limit the weights of all input synapses of each neuron. Equation 3 differs from the Hebbian rule only on the modulatory term, i.e., $\tanh(m_i/2)$. That is, the modulatory effect changes the learning rate of the Hebbian rule.

The model presented here differs from that of Soltoggio's work in the way we define the input and output neurons. Our work defines some standard neurons as afferent neurons or efferent neurons, which have no internal dynamics. The internal state of an afferent neuron is a sensor value and cannot receive input from other neurons, while an efferent neuron stores the average of the internal states of each neuron connected to it, defining the outputs of the network.

The Evolutionary Algorithm

Evolving neural networks In the literature, we find three main general techniques to evolve neural networks with GA aiming at topology search (Mattiussi and Floreano (2007)): direct encoding, developmental encoding and implicit interaction. In this work, we propose a novel simple method based on the third technique to search appropriate values

to the weights, the time and tuning parameters, the learning rate, the neurons to define as input or output, and the topology of our ANN.

Direct encoding is a straightforward approach, in which the chromosomes describe exactly the graph of the neural network. That requires complex gene and genetic operators specifications oriented to graph evolution, aiming at the maintenance of its consistency, which makes it difficult to simultaneously evolve several characteristics not naturally describable with this type of structure. A well-known algorithm based on direct encoding is the NeuroEvolution of Augmenting Topologies (NEAT) (Stanley and Miikkulainen (2002)).

The developmental encoding genome describes the rules of a developmental process of the network, i.e., the process of constructing and growing the network. This approach resembles genetic programing and allows a compact description of the network. However, it is very difficult to define the genetic operators, and a poor definition can lead to unsatisfactory results in terms of evolution.

Implicit interaction is a biologically inspired technique based on the fact that the interaction between genes in the DNA is the key to the definition of their expressions. The Analog Genetic Encoding (AGE) (Mattiussi and Floreano (2007)) is a well-stablished encoding based on this idea. The genome is represented as a simple string of characters, allowing the use of traditional genetic operators such as crossover and mutation, while keeping the structure and consistency of the network. Since the encoding we propose is based on AGE, next we describe it in more detail.

In AGE, while neurons are explicitly described in the chromosome, synapses are implicitly defined, since they are formed by the interaction between genes, and not by a gene itself. The genes specify the neurons and their respective terminals, i.e., their inputs and outputs, while interactions between terminals form the synapses. To assemble the synapses, the inputs of all neurons are aligned with the outputs of all neurons and an alignment score is calculated, which indicates the weight of the synapses between the aligned neurons.

Each element of the neural network is encoded in the AGE's chromosomes into substrings called tokens. For example, to encode a neuromodulated network, Soltoggio et al. (Soltoggio et al. (2007)) used the tokens NE to indicate a standard neuron, MO to indicate a modulatory neuron and TE to delimit a terminal sequence.

A terminal sequence is an arbitrary sequence of characters that precede the token TE and defines a neuron's terminal. After a neuron token, all the subsequent characters until a TE token are translated as a neuron's terminal. Each appearance of TE determines a neuron's terminal, and, so, the neuron has as many terminals as the number of TE's appearances after it.

AGE is supposed to accomplish the evolution of any type of analog network, such as electronic networks, neural networks and genetic regulatory networks. To do that, the alignment score is based on a network-specific interaction map that leads to a complex chromosomal representation. Our proposal focuses on ANNs and specifies the chromosome as a binary array, encoding the parameters of the network in a more straightforward way, using a simpler similarity function, and still maintaining the advantageous properties of AGE's interaction maps (Mattiussi and Floreano (2007)) for ANNs evolution. Such an encoding scheme allows us to easily search augmenting topologies of neural networks composed of different types of devices (neurons).

The Proposed Genetic Encoding Our chromosomes are arrays of bits, with each group of 32 bits defining a gene. The first 8 bits (1 byte) of the gene are used to encode an identifier, which indicates the element that the gene represents in the network. The last 24 bits specify a value that indicates a property of the decoded element.

Each individual in the population has two chromosomes: one chromosome stores the global parameters of the neural network, while the other keeps the network itself. The global parameters are the values of the variables in Equation 3 and the external stimulus of the CTRNN (Equation 1). Figure 3 shows the distribution of values in the chromosome.

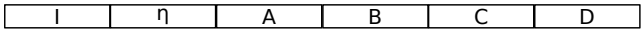

Figure 3: Encoding of the global parameters of the network.

To decode the network chromosome, we have to read each gene and isolate its identifier and value. The 8 bits of the identifier are decoded according to Table 1. The 24 bits that encode the value of a gene are linearly mapped into a floating-point value v in the range $[-1, 1]$, according to the formula:

$$v = \left(\frac{n}{2^{24} - 1} \cdot 2.0 \right) - 1.0 \tag{4}$$

where n is the unsigned integer encoded in the value bits.

Table 1: Genes' identifiers

Byte	Meaning
$0 \leq id \leq 38$	Standard Neuron (SN)
$39 \leq id \leq 51$	Modulatory Neuron (MN)
$52 \leq id \leq 255$	Neuronic Terminal (TR)

With the distribution of identifiers shown in Table 1, we have the following probabilities: $P(SN) = 0.15$, $P(MN) = 0.05$ and $P(TR) = 0.80$, assuming a randomly generated chromosome. This distribution was chosen because we want to have more standard neurons, which actually do the signal generation, than modulatory neurons, which only change the plasticity. At the same time, we need

to have more synapses than neurons, and, for this reason, the probability of creating terminals is greater than that of creating a neuron. The probabilities were empirically chosen.

Suppose that the robot we want to control has s sensors and m motor outputs. To keep some structure of the neural network in the chromosome, we fix the first s genes that encode SN to be afferent neurons, while we set the last m genes that encode SN to be efferent neurons.

The value of a gene (i.e., the last 24 bits) identified as a SN or a MN, encode the time constant τ in Equation 1. The exceptions are the afferent neuron, whose value identifies a stimulus parameter, and the efferent neuron, whose value is ignored, since it has no internal dynamics. The stimulus parameter is -1, if $v < 0$ or 1, if $v \geq 0$, and is multiplied by the sensor signal, generating excitatory or inhibitory sensory signal as input to the neural network.

The genes identified as TR have the value part corresponding to the input or output terminals of the last-read neuron. The first TR read after a neuron's gene is always its input, while the second TR is always its output. Such genes are ignored if no neuron was read before them or the last-read neuron already has its terminals. However, their presence in the chromosome is useful to generate new combinations by the genetic process.

The parsing of a chromosome produces a list of neurons with their respective parameters and terminals that will compose the network. Similar to what is done in AGE, we pair neurons by their terminals to create the synapses arriving at or leaving the neurons. However, instead of using an interaction map, we define the synapses' weights according to a similarity measure. The idea is to use a proximity function, computing a distance value of two numbers based on the Hamming distance between their binary representations, according to equation:

$$w(i, o) = \frac{eb \cdot (i + o)}{nb \cdot 2}, \tag{5}$$

where w is the weight of a synapse that links an output of value o with an input of value i. The symbol nb indicates the total number of bits that represent the value (24 bits), and eb is the number of equal bits at the same position between the binary representations of i and o. We also defined an existence condition empirically to increase topological diversity: if $\lfloor eb/4 \rfloor \; mod \; 3 = 0$ then $w(i, o) = 0$.

Equation 5 may be interpreted as: the synapse weight is the average of the input and output parameters weighted according to distance. The occurrence of two terminal parameters with zero equal bits, implies a maximum distance from each other, and, therefore, there is no synapse linking them. The idea of the similarity function is that if the number of equal bits is the maximum possible, then $nb = eb$, which implies that the synapse weight is the average of both parameters. Equal parameter values imply a synapse weight with the same value.

In order to understand the decoding process better, take Table 2 as example of an excerpt from a chromosome. Analyzing the identifiers based on Table 1, we have two neurons in the chromosome: an SN on gene G1 with value 15,000,000, and an MN on gene G4 with value 8,500,000. Using Equation 4 with those values, we obtain $\tau_{SN} = 0.79$ and $\tau_{MN} = 0.01$.

Table 2: Sample excerpt from a Network Chromosome

	G1	G2	G3	G4	G5	G6
Identifier	30	120	240	40	125	200
Value	15M	10M	4M	8.5M	11.5M	6M

Genes G2 and G3 are the input and output terminals of SN, while G5 and G6 are the terminals of MN in the same order. From Equation 4, we have the values $TR_{G2} = 0.19$, $TR_{G3} = -0.52$, $TR_{G5} = 0.37$ and $TR_{G6} = -0.29$. Figure 4 illustrates the resulting neural organization.

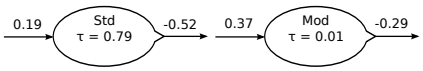

Figure 4: Neurons decoded from Table 2.

Finally, the synapses' weights of the network are computed with Equation 5 applied to each pair input-output, i.e.: $w(TR_{G2}, TR_{G3}); w(TR_{G2}, TR_{G6}); w(TR_{G5}, TR_{G3}); w(TR_{G5}, TR_{G6})$.

Let us compute $w(TR_{G2}, TR_{G3})$ as an example. The binary representation of those parameters, i.e., the way they are stored in the chromosome, are:

bin(0.19) = bin(10M) = 100110001001011010000000
bin(-0.52) = bin(4M) = 001111010000100100000000

From these representations, we have $eb = 13$. Thus, using Equation 5, we would have $w(0.19, -0.52) = -0.09$. However, the existence condition, $\lfloor 13/4 \rfloor = 3$ implies that $3 \; mod \; 3 = 0$, and thus there is no synapse from standard neuron to itself. Similarly computing the other weights, we will have only one synapse, from MN to SN, with $w(TR_{G2}, TR_{G6}) = -0.03$. Figure 5 shows the fragment of the network decoded from Table 2.

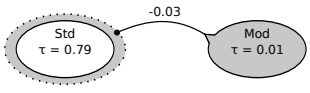

Figure 5: Network decoded from Table 2.

The Simulation of Evolution To evolve the individuals, we simply apply the canonical genetic algorithm. The relative chromosomes of the individuals are paired and the duplication, crossover and mutation operators are applied. Note that, since a chromosome is simply an array of bits,

crossover can break a gene, causing a new gene to appear. The simple mutation of a single bit also can lead to the appearing of a new gene.

Each individual is decoded and made alive for controlling a virtual robot. The robot has an amount of energy that is reduced proportionally to the strength of the generated signals. The evaluation function we used for the emergence of foraging behavior was the number of collected food multiplied by the robot's lifespan.

Case Study

Description of Experiment

To evaluate the proposed controller, we put it to control a Khepera-like virtual robot in a foraging task. The environment consists of randomly distributed fruits and poisons. The simulation was developed with the *Irrlicht 3D Engine*[1], with physics provided by the *Bullet Physics Engine*[2].

The robot is shown in Figure 6. It has a cylindrical body with a black box that plays the roles of eye and mouth. The robot "eats" a fruit or a poison if that box touches them.

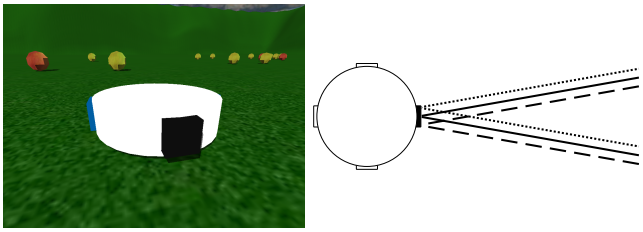

Figure 6: The robot. (a) Robot in the environment. The black box is its eye and mouth. (b) The three sensors distributed in the eye and their fields of sense. We used 20° Field Of Sense (FOS). The maximum sensing distance is six times the robot's diameter.

The neural network is connected with the body and the environment receiving as input the signals of the robot's sensors. There are three sensors aligned side by side at the extent of the eye (Figure 6b), each one able to catch the normalized distance ($[0, 1]$) to the nearest fruit and poison inside its FOS. This implies the generation of six values. There is also a proprioceptive sense of energy, that enables the robot to sense its level of energy spending, which ranges from 0 (the robot is fully energized) to 1 (the robot is totally exhausted). In this way, the strength of the signal allows the robot to perceive when its energy is finishing. This ANN then needs seven afferent neurons.

The robot has two motors, one to move forward or backward and another to make left or right turns, each one controlled by one efferent neuron. When the first motor receives a positive value, the robot moves forward and if it receives a negative value, the robot moves backward. With a positive value, the second motor makes the robot turn right, otherwise it causes the robot to turn left. The robot's energy is reduced every simulation step according to Equation 6, where o_1 and o_2 are the output of the two efferent neurons of the ANN. The constant value 10 is used in order to avoid the evolution of stationary robots.

$$C = (|100 * o_1| + |100 * o_2|)^2 + 10. \quad (6)$$

A robot starts with 50,000 energy units (eu). This value increases 10,000eu whenever a good fruit is eaten (up to a maximum value of 50,000eu) and decreases in two situations: (1) when the robot is alive, its energy is continuously decreasing in proportion to the applied motor signals, (2) whenever the robot eats a poisonous fruit, its energy is reduced to 10,000eu. In the second situation, if the robot's energy level is less than or equal to 10,000eu, the energy is zeroed. If the energy is exhausted, the robot dies.

Each controller decoded from the chromosomes of an individual is assigned the control of a robot, one at a time. Each trial begins at the same position, and the fruits and poisons are always randomly redistributed to prevent the GA from "memorizing" the positions. A trial ends when the robot's energy is exhausted. The GA randomly generates the first population. The following parameters were used:

- Population size: 100 individuals
- Network chromosome size: 100 genes (3200 bits)
- Type of crossover: Monopoint (one break point)
- Crossover probability: 60%
- Mutation probability (per bit): 0.1%

Results

Due to space constraints, we will focus on the analysis of type and quality of the generated behaviors and we will not show the measurement data and comparisons of the several runs we made. However, in all executions we made, the same behavioral result was obtained, except for the neuromodulatory action, that have been developed in rare cases, as we will discuss later. The plots shown in this section represent the tipical results of our experiments.

The GA successfully evolved the neural network to control the robot in the foraging task. Figure 7 shows the evolution of the evaluation averages of all the individuals per generation.

The robot successfully acquired the behavior of catching good fruits only, while avoiding the poisonous fruits, as shown in Figure 8a. Note that, although the evaluation function of the genetic algorithm explicitly selects those individuals that collect the greater number of fruits, there is no direct information about poisons. However, we can observe

[1] http://irrlicht.sourceforge.net/
[2] http://bulletphysics.org/

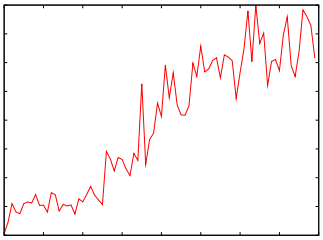

Figure 7: Evolution of the evaluation averages of the populations.

the behavior of moving away from the poisonous fruits (Figure 8b), as a consequence of the fact that individuals who eat poisons may "die" sooner.

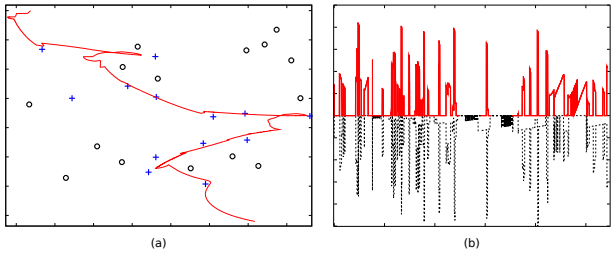

Figure 8: Robot Behavior. (a) The line represents the robot's path, starting at (0,0). Note that its path passes through the fruits (+) while deviates from poisons (○). The line is not always touching the fruits because the graph is showing the center of the objects and, in the simulation, the eye's mesh only needs to touch the fruit's mesh. (b) Motor and poison sensor activities. Note that when a poison is sensed (positive values of solid line), there is a peak of negative signal on motor activity (dotted line), which leads to a backward movement.

An important characteristic to point out about the robot's behavior is how the high level foraging behavior is performed with low level behaviors of direction adjustment. One sensor of the eye alone cannot determine the direction to follow in order to catch the sensed fruit, since it only captures the distance to the fruit. Therefore, the robot needs to change its position to be able to use the three sensors to find out the missing information. This behavior is shown in Figure 9, where we can see the robot turning left to use the right side of a sensor's FOS to follow the fruit. Since the three sensors are slightly displaced with respect to one another, when a fruit leaves the FOS of a sensor, it is possible to determine to what side an adjustment of direction needs to be made. (Figure 6b).

Regarding the modulatory activity, one particular controller evolved with modulatory neurons. With this action, the robot exhibited two ways of search. A local rotation, searching for near food and, when no fruit was caught, it

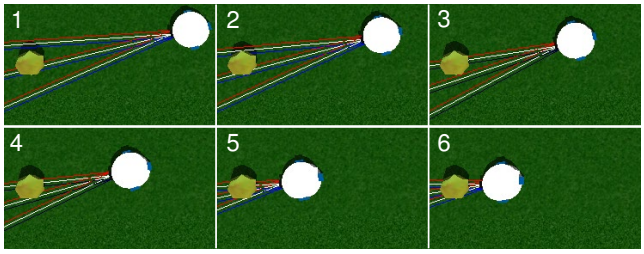

Figure 9: Direction adjustment behavior. Note that, to catch the fruit, the robot approaches it using the right side of the FOS, instead of the FOS's center. The top frames show the moment that it senses the fruit and then turns left. The bottom frames show the robot catching the fruit following its "side sensing".

gradually increased its rotation radius, until it found a fruit, and then passed to the local search again. Figure 10 shows the joint activity of the modulatory neuron, stimulated with the energy sense (signal continuously increasing while no food is collected), and the changing of the motor activity.

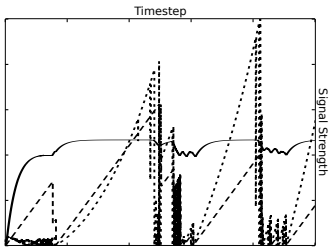

Figure 10: Motor activity (dotted line), energy sense signal (dashed line) and modulation activity (solid line). Note that these three activities are synchronized. When the robot catches a fruit (decrease in dashed line), it searches another one locally (decrease in dotted line). If no fruit is eaten, it gradually increases the search radius (increase in solid and dotted lines).

Conclusion

We described a controller for behavior generation of autonomous virtual characters. We argue that natural behaviors can emerge if the behavior controllers are designed properly, taking into account emergence principles. Such a controller must be capable of adapting itself to the body and to the environment. Thus, it needs to be able of modify itself in contact with the world.

The controller uses neuromodulation for changing its dynamics on the fly, while adapted throughout the generations by the genetic algorithm. In animal brains, the neuromodulators are directly related to memory functions and indirectly to learning. In our experiment, they allowed modifications in behaviors patterns according to environmental changes.

We also presented a simple novel way of genetic encoding ANNs, describing simple arrays evolvable with a canonical genetic algorithm. Those arrays are able to evolve neural networks with growing topologies, and, at the same time, is possible to evolve multiple characteristics of an agent.

We showed the capabilities of our controller through an application involving the foraging behavior of a virtual robot. The robot was able to learn how to use its own movements to compensate for the insufficiency of sensory data in order to accomplish the objective of catching fruits, showing a complex foraging behavior consisting of minor position corrections toward the goal. It is worthwhile to point out that when the plasticity of the controller was increased with modulatory actions, more elaborate strategies have emerged.

The results of our experiments show that the emergentist approach is indeed capable of producing the intimate coupling between agent and environment required for natural behavior. This fact is clearly illustrated by the strategy developed by the virtual agent to compensate for its primitive visual sensory apparatus, showing a high level behavior composed of minimal movements extremely connected with the conditions of the world, rather than a simple and straight "follow the fruit" behavior.

On the other hand, the simulations also show that the type of behaviors which we were able to obtain are relatively simple and, at this point of the investigation, it is not clear how to incrementally increase the complexity of the system in an emergent way. However, the traditional techniques can produce behaviors with arbitrary complexity, by using more detailed models and facts about reality, but paying the price of some level of detachment with respect to the environment. So, a natural question is whether it is possible to combine ideas of the traditional and emergent approaches to obtain the advantages of both sides.

Acknowledgements

This work was supported by Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES).

References

Anderson, M. L. (2003). Embodied cognition: A field guide. *Journal of Artificial Intelligence*, 149:91–130.

Beer, R. D. (1995). On the dynamics of small continuous-time recurrent neural networks. *Adaptive Behavior*, 3(4):469–509.

Chaumont, N., Egli, R., and Adami, C. (2007). Evolving virtual creatures and catapults. *Artificial Life*, 13:139–157.

Froese, T. and Ziemke, T. (2009). Enactive artificial intelligence: Investigating the systemic organization of life and mind. *Journal of Artificial Intelligence*, 173:466–500.

García-Rojas, A., Vexo, F., and Thalmann, D. (2007). Semantic representation of individualized reaction movements for virtual humans. *IJVR*, 6(1):25–33.

Gutierrez, D., Frischer, B., Cerezo, E., and Serón, F. (2007). AI and virtual crowds: Populating the colosseum. *Journal of Cultural Heritage*, 8(2):176–185.

Klaus and Mainzer (2009). From embodied mind to embodied robotics: Humanities and system theoretical aspects. *Journal of Physiology-Paris*, 103(3-5):296 – 304.

Mattiussi, C. and Floreano, D. (2007). Analog Genetic Encoding for the Evolution of Circuits and Networks. *IEEE Trans. Evol. Comput.*, 11(5):596–607.

Nogueira, Y. L. B., Vidal, C. A., and Cavalcante-Neto, J. B. (2008). A nervous system model for direct dynamics animation control based on evolutionary computation. In *SAC '08: Proc. of the 2008 ACM symposium on Applied computing*, pages 1793–1800. ACM.

Orozco, H., Ramos, F., Ramos, M., and Thalmann, D. (2011). An action selection process to simulate the human behavior in virtual humans with real personality. *The Visual Computer*, 27:275–285.

Palmer, M. E. and Chou, A. K. (2012). An artificial visual cortex drives behavioral evolution in co-evolved predator and prey robots. In *Proc. of the 14th intl. conf. on Genetic and evolutionary computation conference companion*, GECCO Companion '12, pages 361–364. ACM.

Panzoli, D., de Freitas, S., Duthen, Y., and Luga, H. (2010). The cortexionist architecture: behavioural intelligence of artificial creatures. *Vis. Comput.*, 26:353–366.

Pina, A., Serón, F., Cerezo, E., and Gutierrez, D. (2006). ALVW: an alife behaviour modelling system. *Kybernetes*, 35(9):1431–1451.

Schneider, M. O. and Rosa, J. L. G. (2009). Application and development of biologically plausible neural networks in a multiagent artificial life system. *Neural Comput. Appl.*, 18(1):65–75.

Shao, W. and Terzopoulos, D. (2007). Autonomous pedestrians. *Graph. Models*, 69(5-6):246–274.

Sims, K. (1994). Evolving 3D morphology and behavior by competition. *Artificial Life*, 1(4):353–372.

Soltoggio, A., Bullinaria, J., Mattiussi, C., Dürr, P., and Floreano, D. (2008). Evolutionary advantages of neuromodulated plasticity in dynamic, reward-based scenarios. In *Artificial Life XI: Proc. of the 11th Intl. Conf. on the Simulation and Synthesis of Living Systems*, pages 569–576. MIT Press.

Soltoggio, A., Dürr, P., Mattiussi, C., and Floreano, D. (2007). Evolving neuromodulatory topologies for reinforcement learning-like problems. In *In: Proc. of the IEEE Congress on Evolutionary Computation, CEC*.

Stanley, K. O. and Miikkulainen, R. (2002). Evolving neural networks through augmenting topologies. *Evolutionary Computation*, 10(2):99–127.

Whiting, J. S., Dinerstein, J., Egbert, P. K., and Ventura, D. (2010). Cognitive and behavioral model ensembles for autonomous virtual characters. *Comput. Intell.*, 26(2):142–159.

Zahorik, P. and Jenison, R. L. (1998). Presence as being-in-the-world. *Presence-Teleop. Virtual Env.*, 7:78–89.