ANIMATING 3D CREATURES

Rui Paúl Oliveira, Francisco Câmara Pereira, Penousal Machado
Departamento de Engenharia Informática
Faculdade de Ciências e Tecnologia da Universidade de Coimbra
Polo II – Pinhal de Marrocos
3030-290 Coimbra
Portugal
E-mails: ruipaul@student.dei.uc.pt, camara@dei.uc.pt, machado@dei.uc.pt

KEYWORDS

Animating, 3D, creatures, GA, genetic algorithms, neural networks, evolution, movement, artificial intelligence

ABSTRACT

This paper describes a method for animating virtual 3D creatures with any kind of morphology. It is based on genetic algorithms that will evolute a neural network responsible for the movement of a pre-created virtual 3D creature in a 3D application. By using this kind of animation process it is possible to evolute the most "natural" way for any creature, avoiding the tradicional way of creating a specific animation for each specific creature by design artists.

INTRODUCTION

Usually, in the game industry, each time a person wants to develop the movement or animation for a character or creature that is part of a specific game, it is needed a complex animation process.
The idea of doing this work was to avoid this animation process by developing an application capable of generating a movement for a creature despite its morphology.
Furthermore, a previous work in which a 3D creature was created and written to a 3ds file by specifying its morphology in a text script file, had already been done.
The aim was to read this 3D file and automatically create the most adequated and efficient movement for that creature, using a neural network as a control system and a genetic algorithm (GA) for evoluting it.
This work was the final project of a student's degree in Computer Science, supervisionated by two teachers with background in this matter. It lasted for four and a half months but it is expected to be continued as there are still lots of experiences and configurations to test.

SYSTEM

The whole system was divided in 4 main parts:

Control System

Based on a neural network with 3 layers, with the first layer being responsible for receiving the signals from the creature sensors, the second or hidden layer consisting of a custom number of neurons and a third layer responsible for producing the final values to be used by the creature.
The weights in each neuron-to-neuron connection would be the part to be evoluted by the GA so the neural network would produce the desired behavior. These weights would be real numbers and would have a value between -1 and 1.
In figure 1 we can see the topology of the neural network used for a creature with four legs.

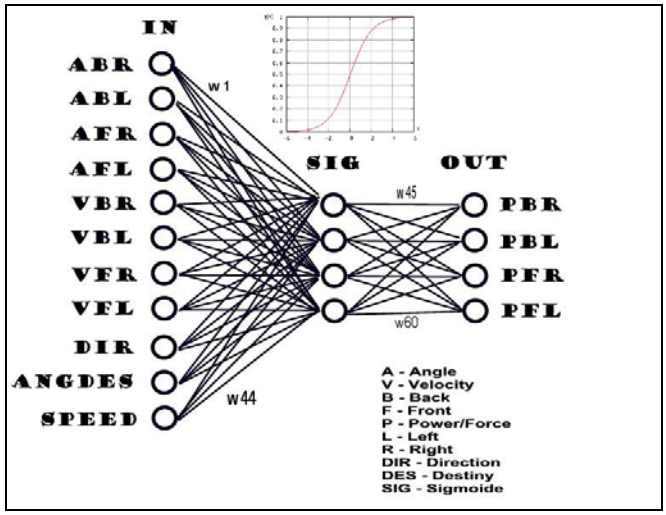

Figure 1: Example of gopology of Neural Network

The inputs for the neural network (in this case) are four angles concerning each leg's angle position, four angle velocity's for each leg, a direction to a goal, a direction to a reference point and a linear speed of the creature.
The outputs are fout forces to be applied to the creature joints.

Genetic Algorithm

Responsible for evoluting a population of the neural network weights. Most of the GA parameters would be changeable and experiences with different values for these parameters would be made in order to find acceptable results. The fitness function would be based on the distance to its final goal: the bigger distance, the worst fitness. The basic structure of the algorithm was the following:

function Used_GeneticAlgorthm
1. Generate a random population of 50 individuals
2. While not_done do
  2.1. Simulate the individuals of the population
  2.2. Evaluate the population
  2.3. Choose the 2 best individuals
  2.4. While new_population_not_full
    2.4.1. Choose 2 individuals by doing 2 tournaments of 5 individuals
    2.4.2. Apply crossover with a 70% (variable) probability in 2 points of the genotype (random points)
    2.4.3. Apply mutation to each gene with a probability of 5% (variiable) using a normal distribution
  end_of_while
2.5. Substitute the old population for the new one
end_of_while

Physics Engine

Used to simulate the real world physics and to determine the performance of each control system. It was used Open Dynamics Engine (open source package) to do this part. In this package a world has to be created (a simple room with four walls and a 9.81m/s^2 gravity value) and bodies and joints to be added (with the same size and configuration as the creature to be simulated).
ODE works by steps. In other words, each time a "nextStep" function is called, ODE will analyse the changes in the world (forces applied by the control system to the creature) and produces the results for these changes (different position of bodies, different angles, etc.)

For example, if we imagine a table with each of its "legs?!?" moving forward and backwards as we can see in figure 2, we have a kind of creature with the possibility of moving and being evoluted.

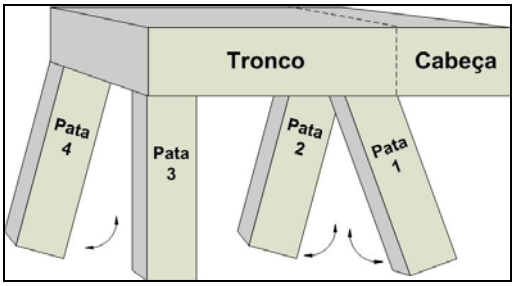

Figure 2. Example of a 3D Creature morphology

3D Engine

The 3D engine used was called Cipher and was used to visualize the behavior of a specific creature with a specific control system.
The weight values of the neural network to simulate were read from a text file created during the evolution.
Cipher reads the 3ds file of the creature to be simulated and puts each model that makes part of the creature in the same position and by the same angle that ODE tells.

RESULTS

The considerable complex space for finding a solution which causes a big simulation time and the lack of computers to run the experiences did not allow us to do the desirable amount of experiences in the month and a half we had left at the end.
The results and the solutions were therefore fewer than the desirable. Even though we achieved to develop experiments with 3, 4 and 6 neurons in the hidden layer of the neural network, and different parameters for the GA. We achieved some interesting results.
Three creatures were created for these experiments:

1. A creature consisting of three parallelepipeds, one consisting of a "tronco" and the other two its two legs, connected to the "tronco" by two joints, one of them having the possibility of moving forward and backwards.

2. A creature similar to the first one. The difference was that its "tronco" was made from two parallelepipeds connected by another joint, allowing the creature to rise the front part of its body.
3. A more complex creature like the one we can see in figure one.

The results observed for creature one were that the creature usually fell forward and would move laid down by making use of strong impulses with its leg (see figure 3). Sometimes we tried to penalize creatures we would fell forward and we observed that creatures would move slower but without falling.
Other approaches were used like beneficiating creatures that would spend less energy, inspirated in what happens in the nature. In this case, the creature moved in the same way as figure 3 shows.

Figure 3: Example of creature 1 movement

The results for creature two (see figure 4) were also interesting. In some configurations we had a creature that would jump backwards, hit the wall and use this impulse to reach the middle of the room. We also had results very similar to the first creature.

Figure 4: Creature 2 aspect

The third creature had even more interesting results. It moved like a "dog" running across the room. We also made experiments by removing the walls of the room and creating wider creatures and the results were also better (faster creatures in less evolution time).
There were also experiences with more and less simulation times.
In figure 5 we can see the aspect of creature 3 movement. It was used a skin of a cow to produce a more attractive look but the morphology is the same of figure 2.

Figure 5: Creature 3 snapshot

Three experiences were repeated thirty times so we could achieve more precise results. We chose the best experiences and we observed that the average was not so good and that in each set of thirty experiments we had very good creatures and very bad creatures depending on the seed of the pseudo random numbers chain.

DISCUSSION

By performing these experiences we reached the conclusion that not only it is possible to develop movement in virtual 3D creatures using this methodology, as also this movement is sometimes very similar to the morphological similar creatures that exist in the nature.
We also observed that less simuation time did not mean worse or better results. This lead us to the conclusion that less simulation times take less time of evolution and take us to same results as more simulation time.
We observed that evolution of smaller neural networks (small space of search) reached faster good results.
By the time we wrote this paper we do not have statisicaly results about the fact that 6 neurons in the middle layer take offer better results but it happened in isolated experiences.
However, the time needed to develop a reasonable movement to a creature with four legs is high (at least half of a day in a Pentium 4) which make us think that creatures with a higher number of joints will take a lot of time to evolute.
Nevertheless it is possible to build a lot of interesting applications and games using this method. This was only the result of some months of work and a month and a half of experiences. Using other kinds of control systems and other parameters for the GA may take us to better results, smaller evolution times and different kind of movements.

FUTURE WORK

The most important goal for us is to keep doing experiences, with different control systems and GA parameters, so we can achieve better results.
One of the ideas, once we have achieved this, is to build an application to create and animate creatures. Idealy it would have a user-friendly interface where the user could build his own creature and animate it by specifying certain parameters or control systems. It could have some features like competitions between creatures.
This project can also be used in other kind of application and games.

REFERENCES

1. http://lancet.mit.edu/~bwall/presentations/IntroToGAs/
2. http://www.cs.stir.ac.uk/~lss/NNIntro/InvSlides.html
3. www.genarts.com/**karl**/
4. www.frams.alife.pl
5. http://ode.org/
6. http://www.cipherengine.com

BIOGRAPHY

Rui Paúl Oliveira was born on the 2nd June 1981 in Coimbra, Portugal. After completing high school in 1999 he started his Computer Science (Informatics Engineering) graduation in the Department of Informatics Engineering of the University of Coimbra, Portugal. He completed it in 2004 with an average of 17 out of 20. His final course project was this work explained in this paper. During the 5 years of university he also did other kind of projects for some subjects, he attended classes in the University of Bologna for 6 months, and worked for about a year in a software development company.