Evolving Virtual Ecology

Gloriya Gostyaeva(✉), Penousal Machado, and Tiago Martins

CISUC - Department of Informatics Engineering, University of Coimbra, Coimbra, Portugal
g.gostyaeva@gmail.com, {machado, tiagofm}@dei.uc.pt

Abstract. Within the field of artificial life it has been possible to create numerous virtual models that have allowed the study of the behaviour of living organisms and their interactions within artificially created ecosystems. Whilst the methods employed in this field have been mostly explored by various researchers in their projects, they had not been broadly applied to the entertainment and art fields.

This paper focuses on a system (digital toy) which contains artificial life agents. These agents learn to interpret external audio commands and adapt to their environment using evolutionary computation and machine learning.

Keywords: Artificial life · Video games · Evolutionary computation Machine learning · Sound recognition

1 Introduction

In this paper, we focus on a 2D virtual ecosystem populated with artificial agents. It utilises Evolutionary Computation (EC) and machine learning (ML) to generate behaviour for agents.

This system is capable of learning the audio spectrum values produced by a user through any external musical instrument and associating actions with them. The sounds can affect the evolutionary process of the agents and can cause either extinction or increase of population. The agents move erratically until they are taught by a user. A user can use the microphone to input sounds and the mouse to indicate correct answers for the neural network. The application can be played with and can be treated as a virtual toy or an artistic experience.

The main inspirations for the project were Polyworld by Larry Yaeger [1] in which he explored the life cycles of artificial agents within a simulated environment, and the evolving creatures by Karl Sims [2]. Additionally Conway's Game of Life [3] and the flocking algorithm by Reynolds [4] showed us that a set of simple rules is capable of generating interesting behaviours.

Steve Grand's Creatures [5] and games such as Nintendogs [6] and Black & White [7] are good examples of artificial life (A-life) within entertainment media and they gave us ideas of how our system could possibly be played or interacted with.

In this paper, we pose a problem of implementation of artificial agents using EC and ML that can be affected by external audio input. The resulting system is directed at

A. L. Brooks et al. (Eds.): ArtsIT 2018/DLI 2018, LNICST 265, pp. 225–230, 2019.
https://doi.org/10.1007/978-3-030-06134-0_26

the entertainment and art purposes and serves as a playground for the virtual ecosystem with audio input.

The two contributions presented are: (i) a virtual ecosystem populated with agents that learn to understand sounds produced by a user and evolve their interaction with the environment; and (ii) an exploration of how an EC and ML approach can be applied in art and entertainment.

The remainder of this paper is organised as follows: Sect. 2 describes the proposed system; Sect. 3 includes a practical analysis of the system; and finally, Sect. 4 presents conclusions and directions for future work.

2 Preliminaries

The proposed system is a 2D world populated with various agents. There are three types of agents: pick-ups, core and particles (see Fig. 1).

A user can input various audio spectrum values using the microphone and the core will respond in a way determined by its Deep Q-Network (DQN) [8, 9]. The movements are random by default, but it is possible for a user to manually backpropagate the expected action by dragging the mouse in the desired direction.

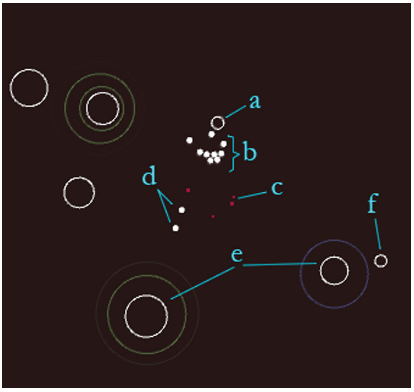

Fig. 1. Screenshot of the system depicting its elements. (a) core; (b) particles; (c) blood visual effect; (d) isolated particles; (e) good and bad pick-ups; (f) pick-up in its growing stage.

Particles are smaller agents clustering around the core. Their interaction with pick-ups is determined by a DQN.

Pick-ups start small but then they grow and start sending audio pings which can be heard by particles and the user if the core is close enough. There are good and bad pick-ups, their sound frequencies differ so that they can be distinguished. When a good pick-up is collected by a particle, this particle produces two similar ones as offspring, however, if it was a bad pick-up, the particle explodes destroying other particles within a certain range.

Because collecting bad objects destroys particles, a low-fitness population reaching for bad pick-ups will soon be exterminated leaving only those particles that ignore or avoid bad pick-ups. When a good pick-up is collected, the particle that collected it will reproduce, creating offspring which copy the weights of their parent's neural network. Sometimes on the reproduction event a mutation of weights can occur which might render a behaviour of an offspring to be different from one of a parent.

In terms of the visual appearance, it was decided to avoid using sharp shapes and instead utilise ellipses because they reflect the biological nature of simulated creatures. The use of circular shapes, in the mind of a user, can relate to the experience of looking through a microscope or observing the night sky. Both cases involve looking at a cosmos, at a micro or macro scale. These experiences are the source of inspiration for the visual appearance. In Fig. 2 the general look of the system is shown.

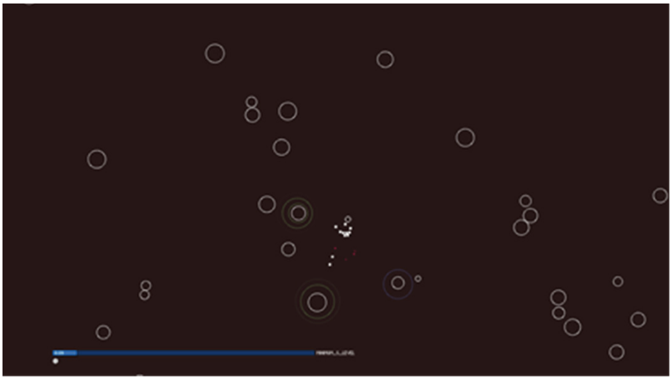

Fig. 2. Screenshot showing the system. The bar at the bottom of the screen controls the audio level input from the microphone.

The following link contains a demonstrational video of the system. It shows a full play through starting with teaching the core and ending with all particles being exterminated: https://cdv.dei.uc.pt/2018/artsit/hive-mind-demo.mp4.

The number of inputs in the DQN of the core is 513. This is the number of values within an audio spectrum transmitted every frame from a microphone. The core receives all spectrum values and then feeds them forward through hidden layers. Hyperbolic tangent (tanh) activation function was used. There are 5 outputs, they correspond to movement directions and a "stand still" action.

Backpropagation of each layer is accomplished by using the Mean Square Error calculation (Eq. 1). Where: Y = vector of the observed values of the variable being predicted; $\hat{Y}$ = vector of n predictions generated from a sample of n data points in all variables.

$$MSE = \frac{1}{n}\sum\nolimits_{i=1}^{n} (Y^i - \hat{Y}_l)^2 \tag{1}$$

There is a list of the last actions performed by an agent, paired with the audio spectrum input in order to deal with nonconsistent data from the microphone and it is important to make sure that backpropagation is done for all variations of one sound.

DQN of the particles is performing feed forward if there is a sound from a pick-up object that is within the hearing range. Learning is accomplished solely through reproduction and survival over generations.

One of the key problems present throughout the development was how the agents would be able to perceive sounds from their own environment. The solution to this was creating an artificial ear in the form of a grid of sensors around an agent (see Fig. 3).

Because the sounds are being transmitted as pings, it was necessary to implement a temporary memory system for each of the sensors. Whenever a sound is produced, a sensor fires and then keeps the value for several frames, unless it is overwritten with a different value.

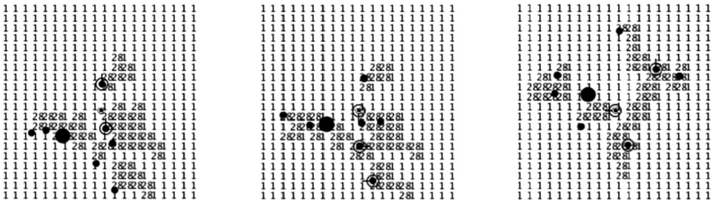

Fig. 3. Each of the three images is a graphical representation of a grid of sensors. Each of the agents on the image is producing sounds. The selected agent has a grid of sensors around it. Numbers on a grid represent detected and memorised audio frequencies. 1 means no frequency detected.

3 Resulting Behaviour

When the application starts a user is faced with a challenge to teach the core agent to respond to sound commands as soon as possible before particles get destroyed by bad pick-ups. Once control is established, the goal of a user shifts into teaching particles to recognise good pick-ups and distinguish them from bad ones. A user will have to rely on particles and their reaction in order to determine which pick-ups are good and which ones are bad.

It is interesting to observe the behaviour of particles when they are reaching towards a pick-up and extending into a line, helping each other to reach further. Sometimes when there are several pick-ups that they want to collect, they can split into two or more “arms” in order to reach them. Some of these interactions are shown in Fig. 4.

The behaviour of particles collecting pick-ups has its own dynamics related to their evolution. Particles that exploded by collecting a bad pick-up will be dead, therefore there will be less particles in the next generation attempting the same action, however, mutations can still cause particles to evolve bad habits. If the particles collect a lot of good pick-ups their population will increase and therefore the chances of lots of particles being caught within the blast radius of an explosion increase. Additionally, more particles can become isolated.

The core itself can learn to respond to audio commands, however, it can often disobey them because of background noise or because a different correct answer was backpropagated through the deep neural network. It can generate unpredictable behaviours that can be interesting to observe.

Particles that were separated from the core remain in place, but whenever a pick-up is moving by, it can be collected by them. As a result, a whole new colony of particles separated from the core can be formed. This can result in an unexpected outcome and emerging behaviour.

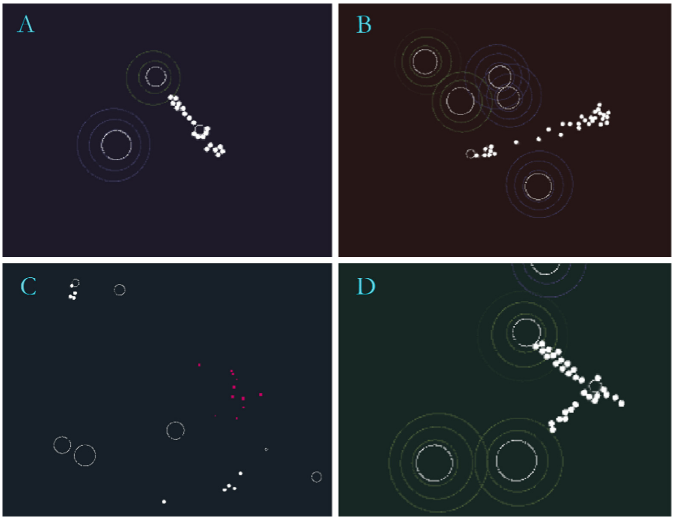

Fig. 4. Screenshots of the system showing interaction between particles and pick-ups. (A) particles are reaching towards a pick-up, trying to collect it; (B) particles are forming a line as they are avoiding two pick-ups on both sides of the line, while trying to get to the core; (C) some particles (bottom of the image) have become isolated from the core and are behaving autonomously, additionally, near the centre of the image, blood can be seen as a result of an explosion of a bad pick-up; (D) particles are splitting into two "arms" as they are trying to reach towards two distinct pick-ups.

4 Conclusion

We have described and tested a virtual environment that contains A-life agents. These are the main contributions: (i) a virtual ecosystem populated with agents that learn to understand sounds introduced by a user and evolve their interaction with the environment; (ii) A-life agents used as key elements in a virtual toy; (iii) use of ML and EC for the behaviour of virtual creatures and their interaction with the environment and a user; (iv) the perception of sounds and the evolution of neural networks through their perception of sound; (v) agents are taught to respond to sounds produced by a user using neural networks, resulting in them being completely controlled by sounds; and (vi) a hierarchy that allows the particle agents to alter their behaviour from the learning that was accomplished by the core agent.

It is not expected from this approach to replace existing techniques used in programming AI behaviour for video games and interactive experiences. It is an exploration of how a ML, EC and sound-based approach can be used in designing a unique behaviour and interaction for virtual agents within art and entertainment directed projects.

The resulting system can be considered a toy because a user can interact (play) with it and the actions of a user affect the state of the environment and the artificial creatures within it.

Future work will focus on: (i) increasing the playability of the system by adding more goals for the user, increasing difficulty and adding more fun interactions; (ii) adding more variants of pick-ups, particle types and effects for them in order to increase complexity; (iii) adding a tutorial for the user and exploring various user interface designs in order to increase usability; (iv) experimenting with the behaviour of agents, adding more possible actions and promoting autonomous behaviour in order to see more emerging mechanics/behaviour as a result; and (v) exploring the possibility of using this project as an interactive experience with more real-life interactions, such as agents reacting to touch, movement and user position by using a tool such as a Microsoft Kinect and projecting the environment of the system onto a wall or floor.

References

1. Yaeger, L.: Computational Genetics, Physiology, Metabolism, Neural Systems, Learning, Vision and Behavior or PolyWorld: Life in a New Context. Artificial Life III, Vol. XVII of SFI Studies in the Sciences of Complexity, Santa Fe Institute, (1), 1–25 (1993). https://doi.org/10.1.1.38.6719
2. Sims, K.: Evolving virtual creatures. In: Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques - SIGGRAPH 1994, pp. 15–22 (1994). https://doi.org/10.1145/192161.192167
3. Conway's game of life on rosettacode. http://rosettacode.org/wiki/Conway%27s_Game_of_Life
4. Reynolds, C.W.: Flocks, herds and schools: a distributed behavioral model. ACM SIGGRAPH Comput. Graph. **21**(4), 25–34 (1987). https://doi.org/10.1145/37402.37406
5. Grand, S., Cliff, D.: Creatures: entertainment software agents with artificial life. Auton. Agents Multi-agent Syst. **57**, 39–57 (1998). https://doi.org/10.1023/A:1010042522104
6. Nintendogs. http://nintendogspluscats.nintendo.com/
7. Wexler, J.: Artificial intelligence in games: a look at the smarts behind lionhead studio's "black and white" and where it can and will go in the future. In: Spring Simulation Multiconference (2008). https://doi.org/10.4018/978-1-60960-567-4.ch007
8. Mnih, V., Silver, D., Riedmiller, M.: Playing atari with deep reinforcement learning (DQN). In: Nips, pp. 1–9 (2013). https://doi.org/10.1038/nature14236
9. Mnih, V., et al.: Human-level control through deep reinforcement learning. Nature **518**(7540), 529–533 (2015). https://doi.org/10.1038/nature14236