Multi-Feature Visualisations of Phenotypic Behaviour for Creative Interactive Evolution

Oliver Bown
Design Lab,
University of Sydney, NSW 2006, Australia
oliver.bown@sydney.edu.au

Rob Saunders
Design Lab,
University of Sydney, NSW 2006, Australia
rob.saunders@sydney.edu.au

ABSTRACT

A visualisation method is presented for interactive evolution of interactive software objects, in which multiple outputs of the system are used to construct a two-dimensional shape in a feature space. The method allows multiple phenotypes to be overlaid allowing for quick feedback on the different properties of phenotypes. The properties of the resulting visualisations are discussed.

Categories and Subject Descriptors

D.2.2 [**Design Tools and Techniques**]: Modules and interfaces—*creative interactive evolution*

Keywords

interactive genetic algorithm, creative evolution, visualisation

1. INTRODUCTION

As well as *static* outputs such as images or melodies, interactive aesthetic evolution has been applied to *behavioural* objects, such as an agent that forms part of an interactive music system or artificial painting system, for example. In these cases, we cannot simply look or listen to evaluate the system, but need to interactively explore the output of the system as it responds to different input conditions. In this paper we propose a method for visualising the response of systems to different inputs, in a way that is amenable to interactive evolution and that, we argue, may provide an effective method for interactively evolving complex behavioural objects. This method contributes to a wider goal of seeking to contextualise interactive evolution into creative workflows, which is identified here as a design problem involving constraints defined both by evolutionary computing and creative computing practice, where appropriate visualisation techniques may be important.

We illustrate the visualisation method using a simple example. The objects being evolved are Continuous-Time Recurrent Neural Networks (CTRNNs). These networks have been used in a number of evolutionary robotics experiments (e.g., [1]) and have been applied by the first author to interactive music performance systems [2]. CTRNNs are easily encoded as genotypes that present a smooth open-ended evolutionary space with an extremely rich set of possible

behaviours. At the same time, as connectionist, nonlinear, feedback systems their behaviour is hard to understand by looking at the structure of the systems, let alone design by means other than trial-and-error-based approaches such as evolution.

1.1 Studies

CTRNNs were created with a structure consisting of 1 input node, 5 hidden nodes (fully connected with self-connections) and 1 output node. Output values are in the range [-1,1]. Input values can be any real number but are expected to be in the same range. A CTRNN's interaction with the world therefore consists of a sequence of updates, in which a new real-valued number is fed in from the environment, and a new real-valued output is produced by the CTRNN. CTRNNs have internal state that affects their output, so simulation runs in which the CTRNN is reset at the beginning of the run may have different outputs from simulation runs in which it it not reset[1].

Each CTRNN was run in simulation for 10,000 time steps with 8 different input patterns. The suite of input patterns was fixed throughout the entire process and consisted of: all 0s; all 1s; a ramp from -1 to 1; a ramp from 1 to -1; a sine wave repeating 40 times over the duration of the run; a sine wave repeating 100 times over the duration of the run; a random walk (#1); a random walk (#2). CTRNN outputs for each of the 8 input patterns were recorded. The CTRNNs' internal states were reset at the beginning of each run. Two features were then extracted from each output sequence: (1) a simple correlation measure indicating the similarity between the beginning and end of the pattern[2]; (2) the number of zero crossings.

From these features, 8 2D points are derived for each CTRNN, described here as a phenotypic behavioural representation. The set of points is then represented as shown in Figure 1 (top and middle) [3]. Rather than simply scatterplot the points in the 2D feature space, we want to be able to overlay multiple phenotypes in such a way that a user can distinguish between phenotype representations when superimposed on the same image. We therefore attempt to derive

---

[1]All code (pure Java/Swing) for the experiment can be downloaded from olliebown.com/files/software/MixedMethodCTRNN.zip.

[2]For a total run duration of N time steps, using a window size of $N/10$, the resulting feature was the lowest value of the RMS distance d_i, between the window starting at $N/4$ and each window starting at $N/2 + i$ for $i \in \{0, 1, 2 \ldots (N - N/10)\}$.

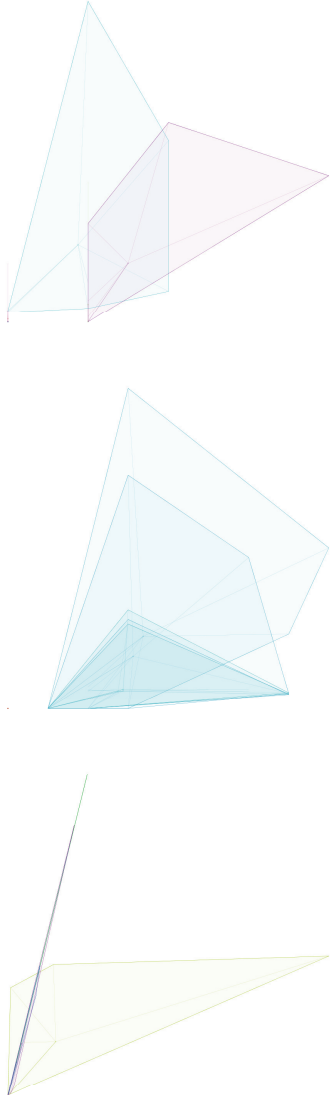[3]Images are best viewed digitally and in colour.

the centroid, the more the network is responsive to different forms of input.

Figure 1 (top) shows a comparison of two randomly generated CTRNNs whose convex hull representations overlap slightly and are of a similar scale. More often than not the representations of randomly generated CTRNNs are of radically different scales and drawn out along different axes, providing a clear indication of phenotypic difference in multiple respects.

Figure 1 (middle) shows several mutations of the same parent CTRNN overlaid. This shows how the representation can act both as a typical scatter-plot in 2D and as an abstract representation of system behaviour. From this representation we can see not only the relative properties of different agents but also get an intuitive sense of how random mutation in the genotype space translates to distribution in this phenotypic representation space.

In an additional example, each run was repeated twice without resetting the CTRNN in between, resulting in 8 pairs of output sequences (Figure 1, bottom). The second pattern in each set can be used to indicate the extent to which the network is influenced by its own internal state. If the first and second patterns are very different, this is an indication of internal rather than external causation. Representations were produced in the same way as above, but the zero-crossing feature was replaced with the correlation feature applied to the second series in each pair of outputs. In cases where the resulting shape forms a straight line emanating from the origin (no area), we can infer that the network is externally responsive but not internally causal. Where the resulting representation has area we can infer internal causality.

2. CONCLUSION

The visualisation provides a simple and potentially informative way for interactive evolution to be applied to objects that respond with complex behaviour to a variety of inputs. In future work we intend to conduct user-studies to investigate the usability of such methods. Further examples can be found by following the link from this paper in the publications section at olliebown.com.

3. REFERENCES

[1] R. Beer. Toward the evolution of dynamical neural networks for minimally cognitive behavior. In *From animals to animats 4: Proceedings of the Fourth International Conference on Simulation of Adaptive Behavior*, pages 421–429. MIT Press, 1996.

[2] O. Bown and S. Lexer. Continuous-time recurrent neural networks for generative and interactive musical performance. In F. Rothlauf and J. Branke, editors, *Applications of Evolutionary Computing, EvoWorkshops 2006 Proceedings*, 2006.

Figure 1: (Top) two different randomly generated phenotypes. (Middle) A series of siblings. (Bottom) Three phenotypes represented in the 'phase' space.

a shape from each set of points. We do this first by deriving the convex hull of the point set and drawing that as a transparent fill. The hull may conceal any number of points, so we also find the centroid of the points and draw lines to all points from the centroid. This reveals the original points in such a way that they can be visually related.

The benefit of this form of representation is that it illustrates both specific features of the system's output and its range of behaviours. For many CTRNNs, the output pattern of the network from its initial reset will be entirely or almost entirely unaffected by the input signal. This may be true even if the output is itself a rich temporal pattern. In that case all points will be in the same place. The appearance of area indicates a network that is responsive to inputs. We can see in which dimensions that variance occurs. Furthermore, the more lines that we can see emanating from