Pro-actively Interactive Evolution for Computer Animation *

Ik Soo Lim and Daniel Thalmann

Computer Graphics Lab (LIG)
Swiss Federal Institute of Technology
CH 1015 Lausanne EPFL, Switzerland
iksoolim@email.com , thalmann@lig.di.epfl.ch

Abstract

This paper addresses the issue of allowing a user more control over the outcomes of *interactive evolution* for computer animation, a variation on genetic algorithms. By using a point-for-point representation for genotype, the control both in selection and variation process is possible while the genotype represented by procedural rules in previous works allows it only in the selection. We experimentally validate these points on generating different walk styles out of a prototype walk motion and also discuss about interactive evolution to be used as a general approach to setting parameters for computer graphics.

Key Words: interactive evolution, genetic algorithm, genotype representation, computer animation.

1 Introduction

Interactive evolution provides a powerful technique for enabling human-computer collaboration. It is potentially applicable to a wide variety of search problems, provided the candidate solutions can be *produced quickly* by a computer and *evaluated quickly and easily* by a human. Since humans are often very good and fast at processing and assessing pictures, interactive evolution is particularly well suited to search problems whose candidate solutions can be visually represented [4][12][15]. While traditional genetic algorithms use an explicit analytic expression for a fitness function to be evaluated by the computer, with interactive evolution the user performs this step based on visual perception.

The beauty of interactive evolution is that the user does not have to state or even understand an explicit fitness criterion: the need is only to be able to apply it. This also frees him from tedious user specifications, design efforts, or knowledge of algorithmic

* In the *Proceedings of Eurographics Workshop on Animation and Simulation '99 (CAS '99)*, September 7 - 8, 1999, Milan, Italy.

details. This feature of interactive evolution is, for example, used very effectively in creating beautiful and abstract color images [12]. An initial population of images generated randomly by the computer is displayed on the screen. From the displayed set the user selects one image for mutation or two images for mating. The mating and/or mutation operations are applied to the selected images to produce a new set of progeny images, that supply the input for the next round of user selection. This process is repeated multiple times, to evolve an image of interest to the user. Evolved images may be saved and later recalled for mating with other evolved images. There are many other notable applications of interactive evolution since the inspiring work of Richard Dawkins [4] (see [3][14] for extensive reviews of it.)

One of the weak sides of interactive evolution, however, is that the user forfeits absolute control over the outcomes: the user, in a *reactive* rather than *pro-active* role, is responsible only for selecting among sets of variations produced by the computer. This is especially inevitable when the genotype is of 'recipe' type [12][15], though a good analogy for DNA [4]. This paper addresses the problem and propose to use the genotype of 'blueprint' type which allows a user control not only in the selection, but also in the variation process. We experimentally validate these points on generating different walk motions for humanoid computer animation. Section 2 explains the differences between the recipe genotype and the blueprint genotype and why the latter is better suited to this pro-actively interactive evolution. The representation of genotype in our work and mutation/mating operations with experimental results are shown in Section 3. Discussion and Conclusions follow it.

2 'Recipe' versus 'Blueprint'

Both biological and simulated evolution involve the basic concepts of genotype and phenotype, and the processes of expression, selection and reproduction with variation. Expression is the process by which phenotype is generated from genotype. For example, expression can be a biological development process that reads and executes the information from DNA strands, or a set of procedural rules that use a set of genetic parameters to create a simulated structure. Usually, there is a significant amplification of information between genotype and phenotype [12]: there is no simple one-to-one mapping between them. This is why recipe is a better analogy for DNA than blueprint: a recipe is a set of *instructions* while a blueprint is a *description*, the first of which is not a point-for-point representation whereas the latter is one [4]. Most applications of interactive evolution use this recipe genotype such as L-systems, cellular automata and LISP expressions [3] which are not in any sense point-for-point representation. The user of this approach has control only in the selection, but not in the variation process which is ruled by the computer.

In the mid of interactive evolution, however, the user would often have intuitive ideas of improvements for the candidate solutions or the candidates might have to satisfy some constraints. Since phenotype is intuitively comprehensible and is shown to the user while genotype is not, it is phenotype where the user's direction and the constraints would be given and then this new change in phenotype has to be faithfully transcribed back into genotype, and hence passed into the next generation. However,

recipe genotype is not suitable for this *reverse* process due to its lack of the necessary one-to-one correspondence. Since a blueprint is, as opposed to a recipe, a point-for-point representation, blueprint genotype is well suited for the inverse mapping from phenotype back to genotype and hence control in the variation process is now possible under both the user's direction and the constraints. We experimentally validate these points on generating different walk styles out of a prototype walk motion.

3 Pro-actively Interactive Evolution of Walk Motions

Motion control of articulated figures such as humans has been a challenging task in computer animation [2]. Once an acceptable motion segment has been created, either from key-framing, motion capture or physical simulations, reuse of it is important. Much of the recent research in it has been directed towards mixing those selected from a library of example motions to create a new motion [11][18]: for example, a library of walk motions. Though it greatly expands the range of possible motions, it is difficult to acquire the examples in the beginning: it still has to go through key-framing, motion capture or physical simulations. Using the pro-actively interactive evolution as proposed here, however, we can synthesize more example motions from a single prototype motion such as different walk styles out of a normal walk motion and have localized controls over the outcomes in doing it. This can be useful, especially, if it is much easier than animating from scratch.

3.1 Genotype and Phenotype

Articulated objects such as human figures are usually represented as rotation hierarchies parameterized by a whole-body translation, a whole-body rotation, and a set of joint angles [19]. Motion can be described by a set of *motion curves*, $\theta_i(t)$, each giving the value of one of the model's parameters as a function of time. In our experiment, the genotype is represented as a vector of the motion curves $\theta = (\theta_1, \cdots, \theta_N)$, where N is the number of the joint angles: the actual encoding of this is realized using a two dimensional array of sampled data of the joint-angle values over time with the joint indices in row and time in column. The phenotype is the animated human figure based on the description of the motion curves. Obviously, there is an one-to-one mapping between the genotype and the phenotype.

3.2 Mutation

Given a genotype θ^{parent} representing a motion, its mutated versions θ^{child} are generated by

$$\begin{array}{rcl} \theta_i^{child} & = & \theta_i^{parent} \\ & or & \\ \theta_i^{child} & = & \theta_i^{parent} + d_i \end{array}$$

for $i = 1, \cdots, N$, where $d_i(t)$ is a displacement curve. The choice of one or the other depends on a mutation rate indicating the probability that a given motion curve will mutate during reproduction. Notice that the mutation is represented as the difference

between two motion curves of the parent and the child. This kind of decoupling the change from the initial one has a number of advantages [6]. First, it simplifies placing constraints on the changes. Secondly, the decoupling allows a representation for $d_i(t)$ to be freely chosen: even recipe-like procedural rules can be used as a generating function for the displacement curve. In our experiments, a Fourier series of only low frequencies is used in generating the displacement curve: the actual coefficient values are randomly determined. Then, this displacement curve can be modified to satisfy any constraint imposed on it based on warping of the curve.

3.3 Mating

Mating takes two parent motions as inputs and uses them to produce a child motion. The basic approach in mating is to choose a subset of the motion curves from each parent and combine them to form the child. Given two genotypes of parents $\theta^{parent1}$ and $\theta^{parent2}$, their offsprings θ^{child} are generated by

$$\begin{aligned} \theta_i^{child} &= \theta_i^{parent1} \\ & or \\ \theta_i^{child} &= \theta_i^{parent2} \end{aligned}$$

for $i = 1, \cdots, N$, where the choice of one or the other depends on a probability that a given motion curve will derive from the first of the parents.

3.4 Experiments

Our experiments use a web browser with a VRML plug-in as a front end and a humanoid model with 47 degrees of freedom. Figure 1 (a) shows the prototype walk motion whose mutated clones comprise the first generation in the evolution process. Figure 1 (b) and (c) are two different walk motions chosen among those evolved in the process. Figure 2 illustrates the evolution directed by interactively turning on/off parts of the phenotype, which are then mapped back into the genotype so that their corresponding parts are accordingly turned on/off. Notice that the only difference between the two motions of Figure 2 (a) and (b) is in the left forearm's movement: the user can turn on the left elbow joint by mouse-clicking it, and hence new mutations occur only in $\theta_{leftElbow}$, the corresponding part of the genotype. More localized control in the variation process is also possible by giving a value that θ must assume at a specified time and this is illustrated in Figure 3. The user just needs to provide a key-frame which will be inverse-mapped to the genotype and then be treated as the constraint which newly mutated ones have to satisfy: this constraint satisfaction is realized using a standard technique of motion warping [19].

4 Discussions and Conclusions

While our approach is to generate the motion trajectory itself, most of evolutionary computation applications in computer animation are about synthesizing stimulus/response motion control systems [1][7][10][13]: the pro-actively interactive evo-

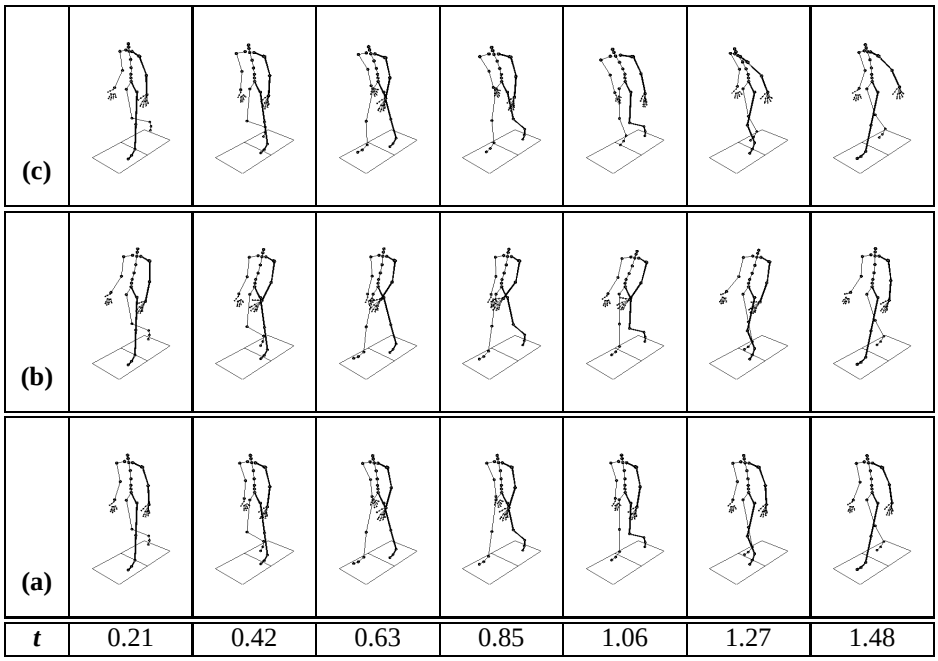

Figure 1: **(a)** The prototype walk motion whose mutated clones comprise the first generation in the evolution process. **(b)** and **(c)** Two different walk motions chosen among those evolved in the process. Time, *t*, is in seconds.

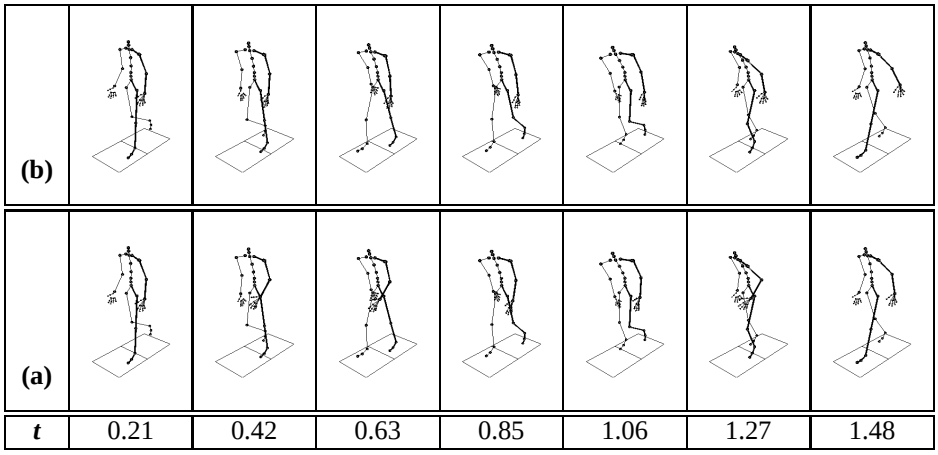

Figure 2: **(a)** and **(b)** Two walk motions which differ only in the left forearm's movement: the user can turn on the left elbow joint by mouse-clicking it and following mutations occur only in $\theta_{leftElbow}$, the corresponding part of the genotype. Time, *t*, is in seconds.

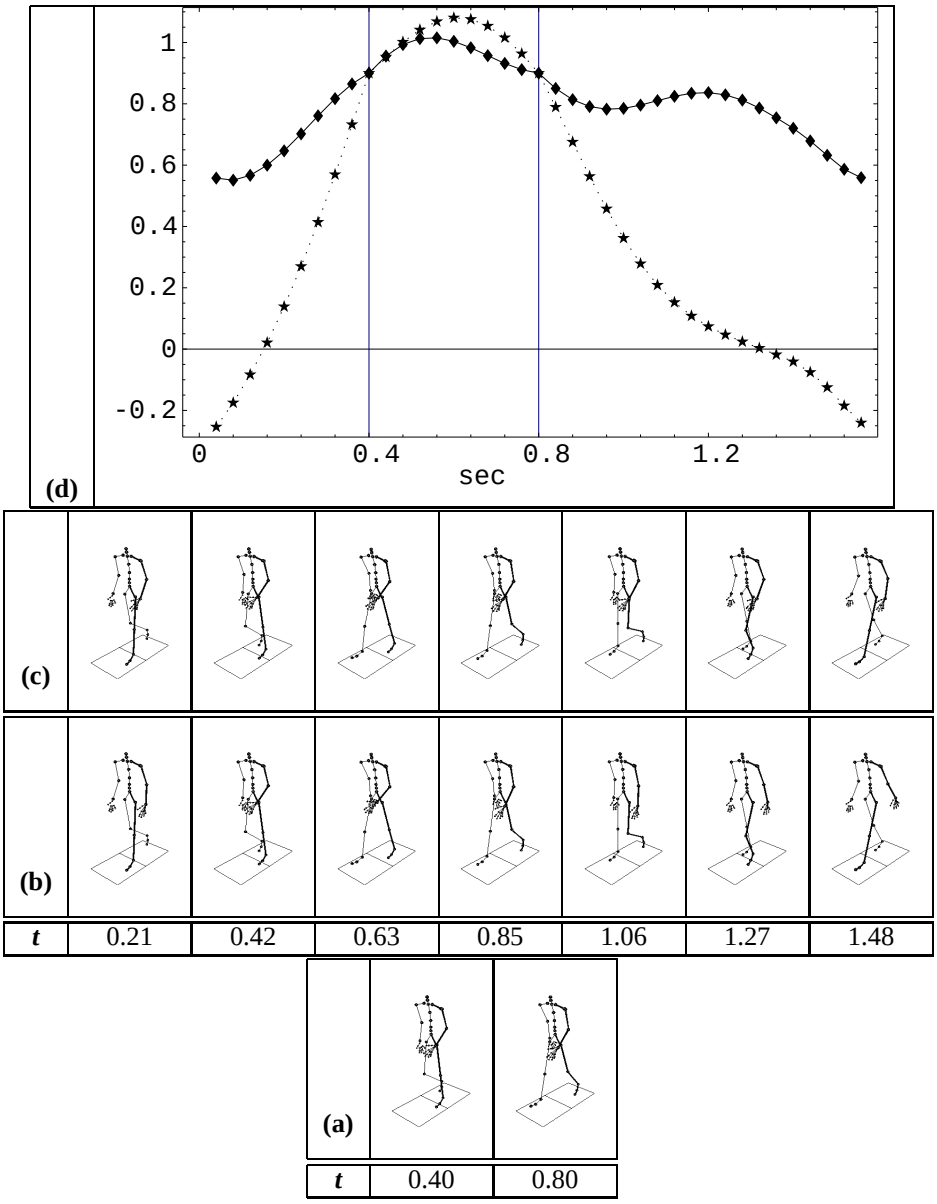

Figure 3: **(a)** Two constraint key-frames which are given by the user. **(b)** and **(c)** Two walk motions mutated but satisfying the constraints: *pro-actively controlled mutations!* **(d)** The motion curves of the left elbow joint, *i.e.*, the joint-angle values over time: notice that they are similar to each other between t = 0.40 and 0.80 where the constraints are given. Time, *t*, is in seconds.

lution would not be possible due to their recipe genotypes. Even the turning a joint on/off for mutation would not be allowed though a motion control program for each joint is evaluated independently: as depending on state and sensor variables, the control programs for different joints may produce coupled actions [7][13]. An exception is to use sine wave motors for articulated stick figures where both automatic evolution and reactively interactive evolution are employed together [16][17]. Rather than the motion itself, however, only the amplitude and the phase offset of the sine wave motors are encoded as genotype so that the localized control of motion as shown in Figure 3 would not be possible although the turning a joint on/off would be so.

Besides motions, the pro-actively interactive evolution proposed in this paper can be applied to generating texture and geometry if their genotypes are also chosen to be of blueprint type: a set of pixel values and geometric primitives, respectively, rather than procedural rules for generating them. These blueprint representations for genotype also fit well into the familiar computer graphics paradigms for animation, texture and geometry, allowing a wide range of existing tools, techniques and skills to be brought to bear [5].

In summary, we have attempted to allow the user more control in interactive evolution and this is realized by using a point-for-point representation for the genotype. Another significant weakness of interactive evolution to be overcome is that, if the process cannot be computed in near real time, it becomes unusable: unfortunately, there are many interesting and important graphics processes which cannot be done in near real time [9]. We are working on fixing this pitfall so that interactive evolution can be used as a general approach to setting parameters for computer graphics and animation.

5 Acknowledgments

We thank Paolo Baerlocher and Christophe Bordeux for their helpful discussions and Christian Babski for the humanoid model and its prototype walk motion. This work is supported in part by the Swiss National Foundation.

References

[1] J. Auslander, et al. Further Experience with Controller-Based Automatic Synthesis for Articulated Figures. *ACM Transactions on Graphics*, v14 n4, pp.311-336, 1995.

[2] N. Badler, C. Phillips and B. Webber. *Simulating Humans: Computer Graphics, Animation, and Control*. Oxford University Press, 1993.

[3] W. Banzhaf. Interactive Evolution. *Handbook of Evolutionary Computation*. T. Back, D. B. Fogel and Z. Michalewiez, editors. IOP Publishing Ltd and Oxford University Press, 1997.

[4] R. Dawkins. *The Blind Watchmaker*. Norton: New York, London, 1987.

[5] D. Foley, et al. *Computer Graphics : Principles and Practice, Second Edition in C*. Addison-Wesley, 1990.

[6] M. Gleicher. Retargetting Motion to New Characters. *Proceedings of SIGGRAPH 9*8, pp.33-42, 1998.

[7] L. Gritz and J. K. Hahn. Genetic Programming for Articulated Figure Motion. *Journal of Visualization and Computer Animation*, v6, pp. 129-142, 1995.

[8] L. Gritz and J. K. Hahn. Genetic Programming Evolution of Controllers for 3-D Character Animation. *Proceedings of Genetic Programming 97*. pp. 139-146, 1997.

[9] J. Marks, et al. Design Galleries: a General Approach to Setting Parameters for Computer Graphics and Animation. *Proceedings of SIGGRAPH 97*, pp.389-400, 1997.

[10] J. T. Ngo and J. Marks. Physically Realistic Motion Synthesis in Animation. *Evolutionary Computation*, v1 n3, pp.235-268, 1993.

[11] C. Rose, M. F. Cohen and B. Bodenheimer. Verbs and Adverbs: Multidimensional Motion Interpolation. *IEEE Computer Graphics and Applications*, pp. 32-40, September/October 1998.

[12] K. Sims. Interactive Evolution of Equations for Procedural Models. *The Visual Computer*, v9, pp. 466-476, 1993.

[13] K. Sims. Evolving 3D Morphology and Behavior by Competition. *Artificial Life*, v1, pp. 353-372, 1994.

[14] H. Takagi. Interactive Evolutionary Computation - Cooperation of Computational Intelligence and Human. *Proceedings of 5th International Conference on Soft Computing*, pp.41-50, World Scientific, Iizuka, Fukuoka, Japan, October 1998.

[15] S. Todd, W. Latham and P. Hughes. Computer Sculpture Design and Animation. *Journal of Visualization and Computer Animation*, v2, pp. 98-105, 1991.

[16] J. Ventrella. Explorations in the Emergence of Morphology and Locomotion Behavior in Animated Characters. *Proceedings of Artificial Life IV*, pp. 436-441, 1994.

[17] J. Ventrella. Disney Meets Darwin: the Evolution of Funny Animated Figures. *Proceedings of Computer Animation 95*, pp. 35-43, April 1995.

[18] D. Wiley and J. K. Hahn. Interpolation Synthesis of Articulated Figure Motion. *IEEE Computer Graphics and Applications*, pp. 39-45, November/December 1997.

[19] A. Witkin and Z. Popovic. Motion Warping. *Proceedings of SIGGRAPH 95*, pp. 105-108, 1995.