Embedding Movie into SBART* – Breeding deformed movies –

Tatsuo Unemi
Department of Information Systems Science, Soka University
1-236 Tangi-machi, Hachioji, Tokyo 192-8577 Japan
unemi@iss.soka.ac.jp

Abstract – *This paper presents a new feature embedded in SBART to breed a deformed movie and an expanded image from movie. To realise these functions, two new non-terminals for GP-style genotype are introduced. Each of them takes one argument of three dimensional vector and returns a colour value picked up from the source movie. The one, named boxel method, interprets a given vector as coordination in cubic volume constructed by expanding the source movie along time axis. It transforms three dimensional boxel data into two dimensional plane. The other one picks up a pixel from a movie frame. By synchronously selecting frames to make a movie, it produces a new movie of deformed frames as a special effect. The recent high-end personal computers have enough computational power to use these functions smoothly.*

Keywords: soft computing, interactive evolutionary computing, media art, special effects for film editing.

1 Introduction

SBART [4] is a breeding tool originally developed as a successor of Kirl Sims's work [2] to breed two dimensional abstract images. It potentially had capability to breed movies and this function was required from a visual jockey in Tokyo four years ago to make a stuff for visual effects at a dance club, but it was difficult because it needed more computational power at that time. Now we obtained faster facility available on our desktop, then the author extended SBART to be able to breed deformed movies to examine its feasibility.

SBART has already had a function to embed external images to produce a deformed image from photograph and illustration. As similar to the case of images, there are a number of possible methods to embed an external movie into SBART. We have to choose and design appropriate methods effective from both computational efficiency and an esthetic point of view.

This paper is to present our approach to embed movie into the drawing program of a functional representation as a gene for breeding, and to show the first results.

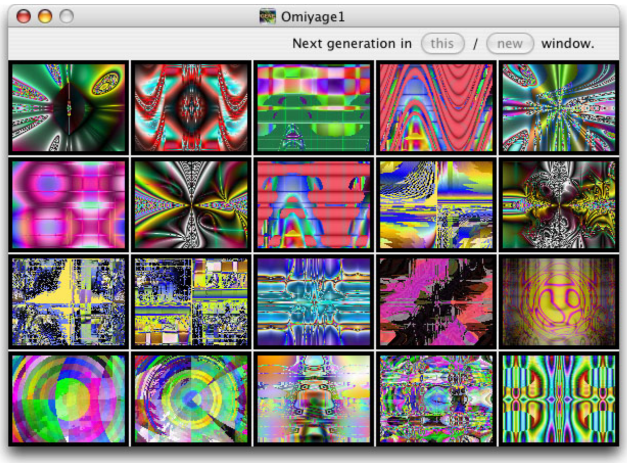


Figure 1: An example of field window of SBART.

2 SBART overview

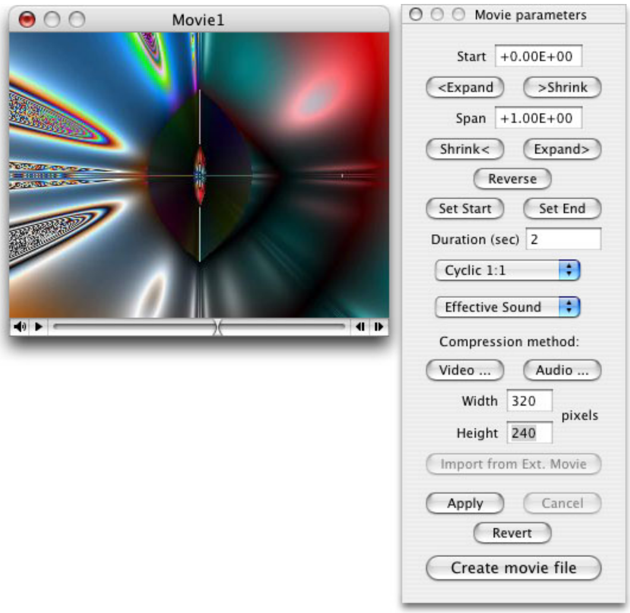
As figure 1 shows an example field window, SBART displays 20 candidates of images generated by functional expression. This section describes some features related on the theme of this paper.

2.1 Phenotype and genotype

The drawn images are phenotypes and the corresponding functional expressions are genotypes in a style of Genetic Programming[1]. Each of genotype in the initial population is generated randomly with non-terminals and terminals. A non-terminal is a symbol selected from the pre-defined set of arithmetic functions such as $+$, $-$, $\times$, $\div$, sin, exp, log, and so on, and a terminal is a variable or a constant vector. For example, the genotype of the image at upper left corner of Figure 1 is:

$$0.969/(-(-1.305/\min(\mathrm{XYT}-\cos(\log(-(\mathrm{hypot}(\mathrm{YXT}, \sqrt{\mathrm{XYT}})))), \mathrm{XYT}))-\cos(\log(-(\mathrm{XYT}/\min(\mathrm{YXT}- \mathrm{XYT}, \mathrm{YXT}))))),$$

where a numerical value means a vector of three same values, for instance, the first element 0.969 means (0.969,0.969,0.969), a symbol of three capital letters is a vector of variables, for instance, XYT means (x, y, t). The



Start +0.00E+00

Span +1.00E+00

Duration (sec) 2

Cyclic 1:1

Effective Sound

Width 320

Height 240

pixels

Figure 2: Movie window and parameter panel for making a movie.

variables x and y indicate the co-ordinate of a pixel in drawn image, where the origin is at the centre of the image rectangle, and the values of y at the upper and lower edges are 1 and -1. t is the time variable of which value is 0 for still images, and varies in a range the user specified to make a movie as described later.

For each pixel on the drawn image, the expression of genotype is evaluated with the pixel position in (x, y) co-ordinate and frame position in the time scale. All the intermediate values and the result of calculation are vectors of three scalar values. The resultant vector is finally translated into RGB (red, green, blue) vector by interpreting it as HSB (hue, saturation, brightness) colour vector.

2.2 Breeding process

SBART is based on an idea of Interactive Evolutionary Computing [3], but it is not Interactive Genetic Algorithm but Simulated Breeding[5]. The user is not requested to assign fitness values for each candidate but to select which one should be a seed to produce the offsprings. If only one individual is selected then it produces mutants, and if more than one individuals are selected then it produces combinations to fill the population of the next generation. Mutation is done by replacing randomly selected symbol by another random symbol. If the original symbol has fewer arguments than new one, a new sub-tree randomly generated within a predefined depth is added to fill the new argument. If the number of arguments decreased, some sub-trees are discarded. Combination is done by exchanging randomly selected sub-trees between copies of parents.

Other functions such as individual protection, multi-field user interface and direct genetic operation are facilitated to keep genetic diversity wide and to provide convenience for user's breeding operations.

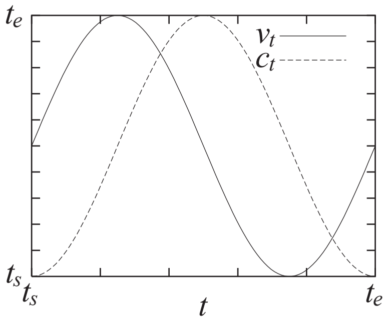


Figure 3: Trajectories of time variables for cyclic mode.

2.3 Making a movie

The user can pick up one individual from the field window to make a movie, and can set up a number of parameters using the parameter panel shown in Figure 2. The seven buttons and two text input fields of upper part of the parameter panel are to set up the range of variable t by indicating the start value and the span. The value of t varies within the specified range during making a movie as described above. This value affects not only to t in a variable vector but also a constant vector by multiplying $t+1$ to each scalar element.

The next text input field is to indicate the movie duration in seconds.

The next pop-up menu button is to indicate how t varies in the duration. The most natural way is to make the value changing linearly from starting value to end value. An alternative method is for making a loop movie that changes through the time but finally return back to the same image of the first frame smoothly. To realise this feature, t for variable vector and t for multiplication for constant vector are replaced by v_t and c_t respectively, that are defined as

$$v_t = (1 + \sin(2\pi t/(t_e - t_s)))(t_e - t_s)/2 + t_s \qquad (1)$$

$$c_t = (1 - \cos(2\pi t/(t_e - t_s)))(t_e - t_s)/2 + t_s \qquad (2)$$

where t_s and t_e are the starting value and the end value of t. Both values are moving within the range between t_s and t_e and the values at $t = t_s$ and $t = t_e$ are same for each, but the trajectories are different as shown in Figure 3. It has other options to make combination of different frequency between v_t and c_t.

The rest part of the parameter panel includes buttons and text inputs for sound, compression method, frame size, and controls for rebuilding and copying the movie. Two of them, sound setting and parameter import, will be explained later because they were newly introduced to import an external movie.

3 New non-terminals

Two types of methods to import an external movie file were designed and implemented. One is to embed boxels

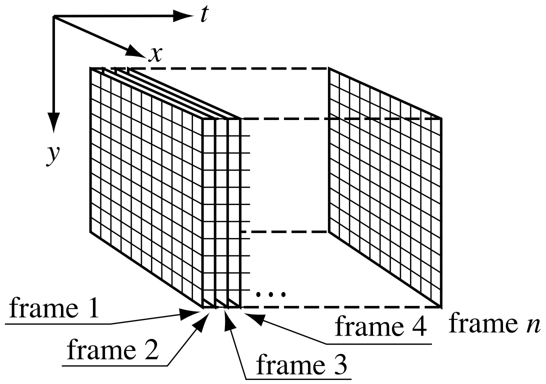


Figure 4: Boxel data from a movie.

in the cubic volume of movie data, assumed to be expanded along time axis as shown in Figure 4. The other one employs each two dimensional frames extracted from the movie. Both of them are implemented by introducing new functions that extract colour values according to the given argument values.

3.1 Boxel method

Boxel method is implemented in a new function named moviec that takes one vector argument. The first, second and third elements of the vector are interpreted as the x, y and t co-ordinate respectively. The values are normalized so that the center of the box is the origin of co-ordinate system and edges are -1 and 1. The values given as the argument are forced to be transformed within the range $[-1, 1]$ by a saw-shaped function illustrated in Figure 5. Then this function returns HSB colour vector at the boxel of the position indexed by the transformed values.

Because the given co-ordinate values are continuous but each boxel is a cube occupying a certain volume, it might produce rough drawing if we had no sophisticated smoothing algorithm. To avoid it, functions to pick up colour value from the external movie should mix colours of neighbour boxels assuming colour is gradually changing over the three dimensional space of movie data. We employed a linear smoothing algorithm to realise it, that is, the resultant colour value for the point is a weighted summation among eight boxels including the boxel the point belong to and seven nearest boxels from the point in terms of Euclidean distance in three dimensional space as shown in Figure 4. The similar method for smoothing two dimensional image was introduced to embed external images to make a type of collage as described in [4]. The new method is an extended version for three dimensional space.

Because this method maps the time axis onto spacial dimensions, a produced image condenses a history of captured scene in one snapshot.

3.2 Frame image method

Frame image method is implemented in a new function named movief that takes also one vector argument, but the third element is ignored to draw a resultant image, that is, only one frame is used as embedded data to draw one image. This method was designed for creating movie. The target frame for extraction changes along the time axis synchronously with produced frames.

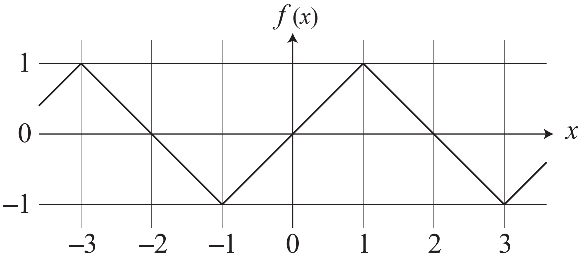


Figure 5: Saw-shaped function to normalise the value within the range $[-1, 1]$.

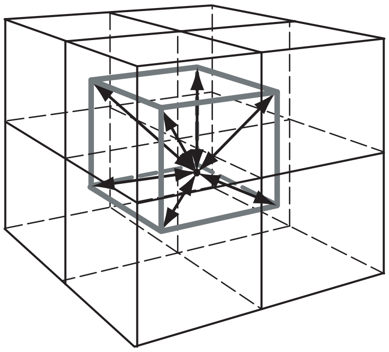


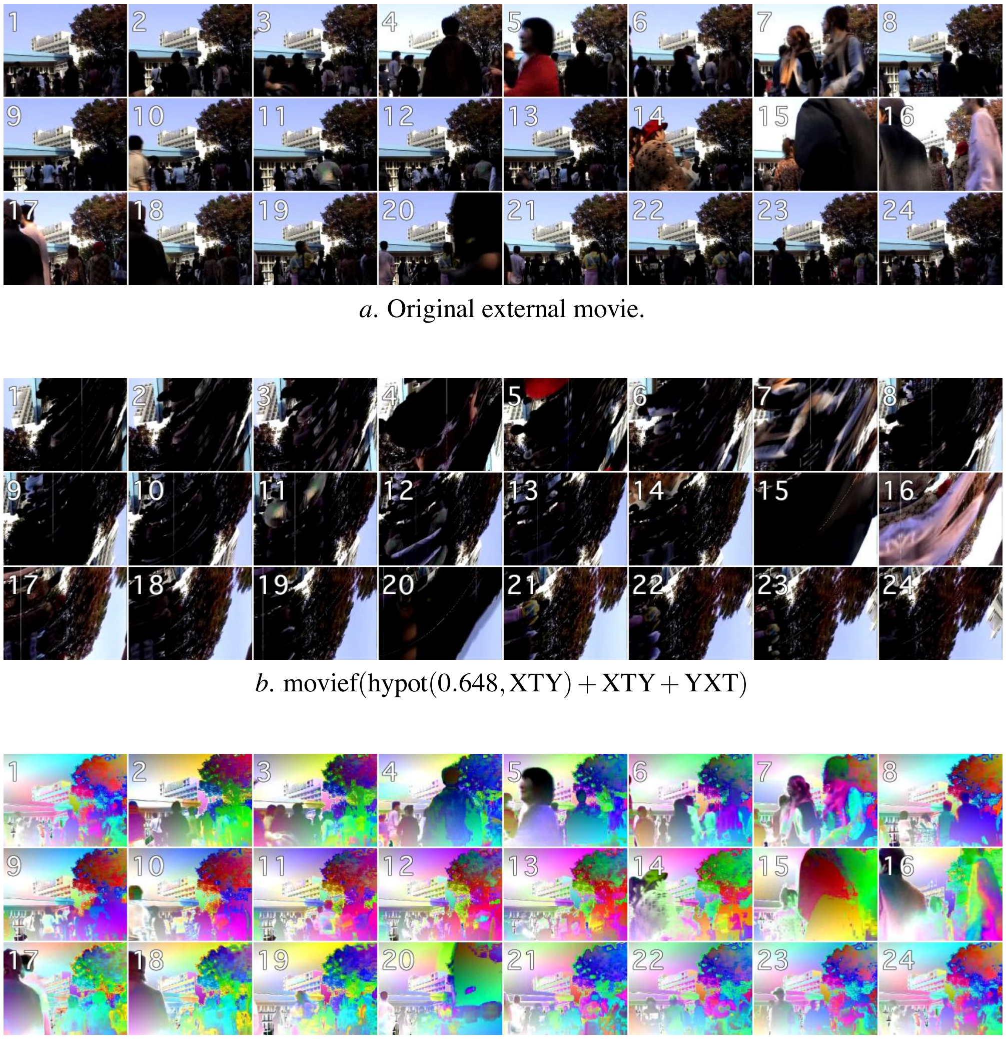
Figure 6: Smoothing for boxels. The colour value at the point inside the central cube is calculated as a weighted summation among colours of nearest eight boxels.

By combination with a new operation to attach the audio track of original movie into the created movie, it provides an effect to deform each frame. The user can choose audio track as the sound of created movie by operating a pop-up menu button in parameter panel shown in Figure 2. To guarantee the same quality of the new movie as the original one, parameters of movie, such as frame size in pixels, movie duration, frame rate, compression method, and so on, can be imported from the original one by clicking a button in parameter panel.

This function also utilises the smoothing algorithm to pick up colour value from a frame image. It is not three dimensional but two dimensional space.

4 Sample productions

We implemented the functions described above and examined them with some short movies. The upper sequence of movie frames in Figure 7 is one of the movies used for external movie data, in half size of NTSC DV format, 320×240 pixels of frame size, 29.97 frames per second, and 30 seconds of duration. It includes $29.97 \times 30 = 899$ frames, then totally $320 \times 240 \times 899 = 69{,}043{,}200$ boxels. It was taken by a handy DV camcorder in the campus during a campus

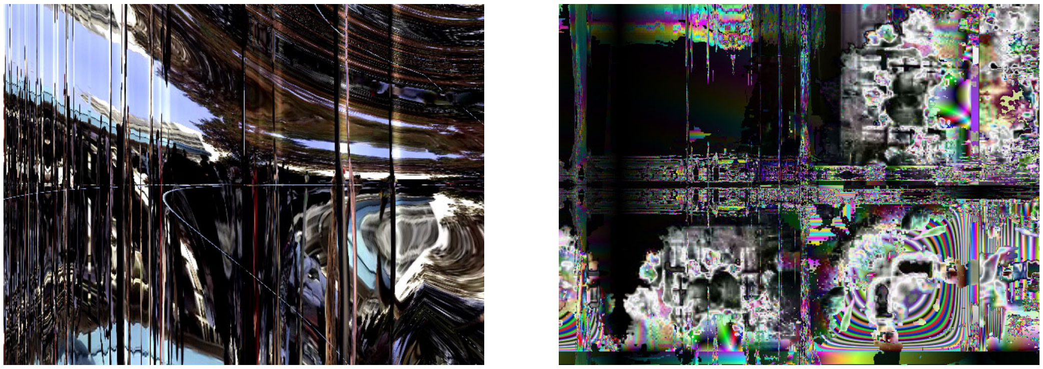


a. Original external movie.

b. $\mathrm{movief}(\mathrm{hypot}(0.648, \mathrm{XTY}) + \mathrm{XTY} + \mathrm{YXT})$

c. $-0.203 + (0.648 + \mathrm{YXT} + \mathrm{movief}(\mathrm{XYT}))$

Figure 7: Sample frame sequences produced with *frame image method*.



$\mathrm{moviec}(\mathrm{XYT} - \cos(\log(\mathrm{hypot}(\mathrm{YXT} +$
$-1.227, \sqrt{\log \mathrm{XYT}}) + \mathrm{TYX})))$

$\min(\log(\max(\mathrm{TXY}, \mathrm{YTX}/\log(\max(\mathrm{TXY},$
$\mathrm{YXT})))), -1.422 - \min(\log(\max(\mathrm{TYX}, \sin(\mathrm{YTX}))),$
$\mathrm{moviec}(\mathrm{YXT})^{\mathrm{XTY}}/-0.805)) \times (\min(\log($
$\max(\mathrm{TYX}, \mathrm{moviec}(\log(\max(\mathrm{TYX}, \mathrm{XYT}))))),$
$-1.422 - \min(\log(\max(\mathrm{TXY}, \sin \mathrm{XTY})),$
$\mathrm{moviec}(\mathrm{XYT})^{\mathrm{YTX}}/-0.805)) \times -1.906)$

Figure 8: Sample images produced from the external movie in Figure 7 by boxel method.

festival in the fall of 2003.

Figure 8 shows two typical productions using boxel method. The genotype of left side production has the function moviec at the root of tree structure. This makes nonlinear transformation of the shape of original pattern. The time axis was expanded along horizontal axis. Some vertical lines included in the resultant image indicate rapid adaptation of automatic adjustment of exposure amount by the camcorder. Small vertical vibration along horizontal axis means it was difficult to keep the handy gadget stable in my hand.

The right one has more complex genotype that includes moviec at the intermediate nodes. It produces not only deformation of the shape but also modification of colours. As shown in both drawings, boxel method brings us a new effect to produce more complicated and unpredictable results from some types of external movies by inheriting the complexity of three dimensional pattern involved in the movie data.

The middle and lower frame sequences in Figure 7 are sample movies produced from genotypes that includes movief. The middle one includes the function at the root, and the lower one includes it at just above a leaf of variable XYT. The former one produces shape deformation, and the latter one produces colour modification, as similarly to the two examples for boxel method.

We also examined another film taken at a concert of student big band. It can produce a very interesting and effective movie clip that seems useful as a stuff to create a music video or so.

5 Concluding remarks

Two methods to embed external movies into SBART were proposed. Boxel method has ability to produce interesting images by mapping the time axis onto spacial dimensions. But the problem is that it needs more computation power to make a smooth image from boxels of movie including comparable number of frames with the spacial resolution. It is much better to gather colour values of all of boxels involved in the movie data beforehand for keeping them in the memory than to extract them when needed. Recent personal computers have more than some hundreds megabytes of memory on board, so this technique can be employed when the total number of boxels is smaller than one hundred million. The example original movie described above was in a proper size to use this method, and it needs not so much of CPU time to breed the examples shown in Figure 8.

Frame image method is useful to make a type of special effects on a film for promotion and artworks. This is not so heavy for calculation during breeding process because it uses only one frame for still image, but it also wastes much of CPU time to produce a movie. This task might be one of good example for high performance computation such as a grid computing, because it is possible to calculate each of colour values separately.

This experimental system can be downloaded from *SBART Cocoa* web site:

`http://www.intlab.soka.ac.jp/~unemi/sbart/Cocoa/`

and can be tried on MacOS X. We hope this system and idea will be helpful for any type of creative works of many persons.

References

[1] J. R. Koza, *Genetic Programming: on The Programming of Computers by Means of Natural Selection*, MIT Press, 1992.

[2] K. Sims, “Artificial Evolution for Computer Graphics,” *Computer Graphics*, Vol. 25, No. 4 (SIGGRAPH 91), pp. 319–328, 1991.

[3] H. Takagi, “Interactive Evolutionary Computation: Fusion of the Capacities of EC Optimization and Human Evaluation,” *Proceedings of the IEEE*, Vol. 89, No. 9, pp. 1275–1296, 2001.

[4] T. Unemi, “SBART 2.4: An IEC Tool for Creating Two-Dimensional Images, Movies and Collages,” *Leonardo*, Vol. 35, No. 2, pp. 53, 71–73, 2002.

[5] T. Unemi, “Simulated Breeding – a Framework of Breeding Artifacts on the Computer,” *Kybernetes*, Vol. 32, No. 1/2, pp. 203–220, 2003.