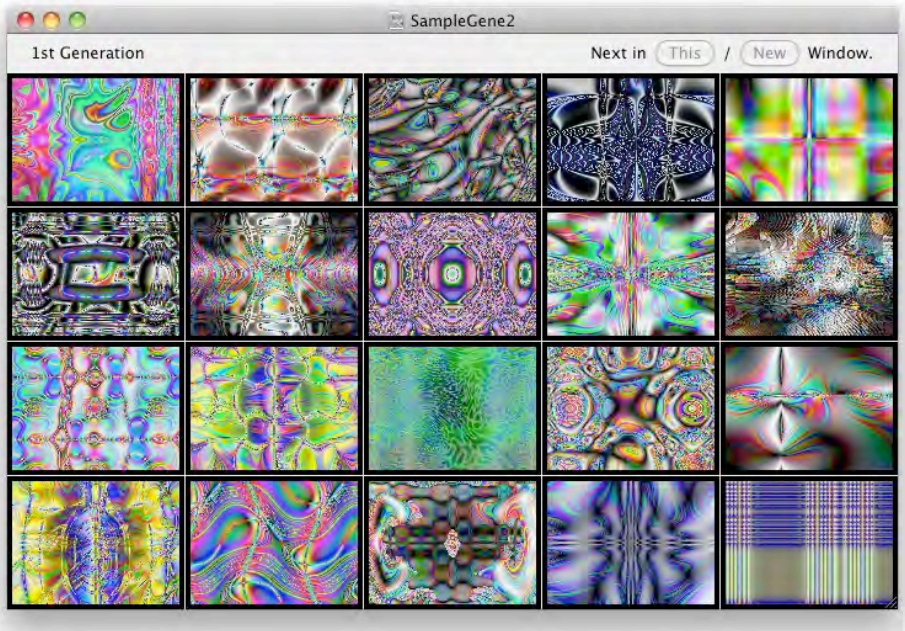
Tatsuo Unemi

Paper: **A Breeding Tool for Abstract Animations and Its Applications**

Abstract: The recent innovation of graphics processing unit (GPU) improved the calculation performance to be fast enough to realize breeding animations in real time on the personal computer. SBArt4 compiles each expression in genotype into a type of shading language, Core Image kernel language, that directly runs on GPU. Even when it renders each frame of the animation in real time, it achieves enough speed for users to evaluate the product of an abstract animation immediately. The compiled code can be exported to another application that utilizes Core Image framework on MacOS X. Four types of video effect plug-ins for Final Cut Pro and an independent application for slide presentation were examined. It is useful not only to create an abstract animation in arbitrary size and duration but also to make a transition effect by deformation and/or discoloration.

Topic: Animation

Author:
Tatsuo Unemi
Soka University,
Department of
Information Systems
Science
Japan
www.intlab.soka.ac.jp/~unemi/



A sample field window of SBArt4

References:
[1] Tatsuo Unemi, "*SBArt4 - Breeding Abstract Animations in Real time*", Proc. of CEC 2010, pp. 4004-4009, Barcelona, Spain, 2010.

Contact:
unemi@t.soka.ac.jp

Keywords:
Interactive evolutionary computing, abstract animation, video effects

A Breeding Tool for Abstract Animations and Its Applications

Associate Prof. T. Unemi, BEng, MEng, DEng.
Department of Information Systems Science, Soka University, Hachioji, Japan
www.intlab.soka.ac.jp/~unemi/
e-mail: unemi@iss.soka.ac.jp

Premise

The recent innovation of graphics processing unit (GPU) improved the calculation performance to be fast enough to realise breeding animations in real time on the personal computer. SBArt4 compiles each expression in genotype into a type of shading language that directly runs on GPU. Even when it renders each frame of the animation in real time, it achieves enough speed for users to evaluate the product of an abstract animation immediately. The compiled code can be exported to another application that utilises Core Image framework on Mac OS X. Exportation of the code into a programmable patch in Quartz Composer, four types of video effect plug-ins for Final Cut Pro and an independent application for slide presentation were examined. It is useful not only to create an abstract animation in arbitrary size and duration but also to make a transition effect by deformation and/or discolouration.

1. Introduction

The progress of science and technologies has been always providing new media for artworks that sometimes causes revolutionary change in human culture. The improvement of speed and capacity of machineries in both computing and communication in these decades is one of the sources of such a revolution including the birth and the growth of generative art. Especially, the recent innovation on Graphics Processing Unit (GPU) for personal computers brought revolutionary improvement of parallel processing required for real-time responses in computer-based interactive artworks. This technology was originally developed for acceleration of rendering process for 3D graphics typically for gaming, design support system and visualisation, but it has been recognised as a promising technology for any types of parallel computing for simulation and optimisation in large-scale problems.

Some of the researchers and artists have also been seeking new possibilities by an innovative hardware available in reasonable prices in the market. It is attractive because not only it is merely new but also it has possibility to produce quite a new experience for human by an algorithm inspired from natural phenomena and biological adaptation. In the field of Interactive Evolutionary Computation [1], one of the most powerful frameworks for evolutionary art, new experimental works has been always challenged. One of the pioneers of this field is Karl Sims. His early project of artificial evolution system to breed abstract 2D images was utilising CM-2 super computer and 16 graphics workstations in 1990 [2]. The similar system became

possible on the personal computer in some years later, such as SBART by the author [3, 4]. SBART has been extended with embedding external images and movies [5, 6], but it still needed to wait for faster computation power on the personal computer to realise real-time breeding of abstract movies.

The processing speed of CPU in the personal computers reached 3GHz, multi-core, and small-scale parallel calculation. It is still not enough for real time breeding, but by combination with the power of GPU, it achieved the performance fast enough for the next stage in 2009.

The rest part of the paper describes a summary of SBART, the extension to utilise the power of GPU and an introduction of the applications for visual programming, video authoring system and a slide presentation system.

2. Summary of SBART

SBART [4] is a breeding system for abstract 2D images. The fundamental idea was proposed and implemented by Karl Sims in 1990 [2], that is based on a combination of the framework of Interactive Evolutionary Computation [1] and a type of Genetic Programming [7], that is, it is a breeding system using a genotype in a functional expression of tree structure for each individual. This chapter introduces a summary of SBART. The detail can be referred in [4].

The phenotype is an abstract drawing in a rectangle area, each of which pixel is painted with a colour calculated using the function of genotype accompanying with the coordinate. Figure 1 shows an example of the field window in which 20 individuals are displayed. The users are allowed to breed their preferred image by mutation and crossover using this window.

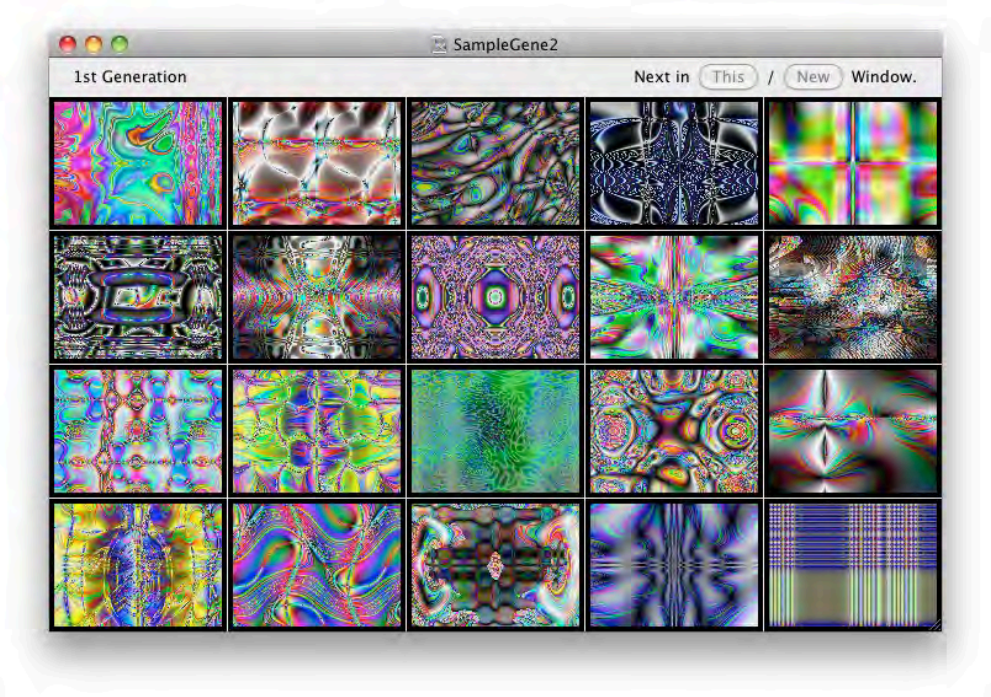


Figure 1. A sample field window of SBArt4.

2.1 Genotype

Each genotype is a functional expression constructed with a pre-determined set of terminal and non-terminal symbols. A non-terminal symbol acts as a function such as addition, subtraction, multiplication, division, exponential function, trigonometric function, and so on. A terminal symbol is a constant or a variable. Because all of the calculations are executed over a domain of vectors of three scalar elements, the constants and the variables are also vectors of this form. A variable vector is a permutation among three scalar variables, *x, y*, and *t*, where (*x, y*) is the coordinate of the pixel in rectangle area, and *t* is a time variable for movie.

Mutation is applied by replacing a symbol by another one based on random selection, and crossover is done by exchanging randomly selected sub-trees between two parents in a same manner of the genetic programming [7].

In addition to the gene of functional expression, we introduced some sets of parametric genes of floating point numbers as follows. Time parameters to determine the range of *t* value for movie, area parameters to specify the scale and offset for each 2D axis, colour parameters to modify the hue value, and scale parameters to magnify for each scalar value in the result vector of functional expression. These genes are effective to expand richness of result images.

2.2 Phenotype

The phenotype is an abstract 2D image drawn by genotype. The result value of calculation of functional expression in the genotype is interpreted as a HSB (hue, saturation and brightness) colour value and then is converted into a RGB (red, green and blue) vector to render the pixel. Each scalar value of the elements is converted so that the value is within a range between 0 and 1 using a saw-shaped function.

The drawing process requires much of computation time because it needs to calculate the value for each pixel. In case of movie, the computation cost is multiplied by the number of frames. For example, if you want a movie of one second with full high-definition and 30 frames per second, the total number of pixels is 30 x 1920 x 1080 = 62 x 10^6. It is difficult to draw each frame in time during movie playback only by CPU since the usual genotype of more than 20 function symbols for an interesting image consumes more than 100 machine cycles to compute the colour of pixel.

2.3 Embedding external images

The old version of SBART has a functionality to embed external images and movies to create a type of collage. It was implemented by introducing a special function named **image** that extract a colour value from the specified image data according to the given argument values as the coordinate in the data. The phenotype becomes a discoloured version of the original image, if the genotype includes this function at the leaf side of the tree. The phenotype becomes a deformed version if this function is at the root of genotype. In addition, the new version of SBArt4 has options for two rendering methods to make deformation and discolouration. In the deformation mode, the result value of genotype is interpreted as the coordinate in the original

image. In the discolouration mode, the result value is interpreted as the modification vector from the original colour.

In both deformation and discolouration modes, it is possible to make a movie that gradually changes the shape or the colour between the original one to modified one by changing a parameter of weighted summation between the original value and the modified value.

3. Compiling a genotype into Shading Language

To take an advantage of the power of GPU, the new version of SBArt4 compiles each genotype into Core Image Kernel Language [8], a subset of GLSL (OpenGL Shading Language) [9]. The language has a similar syntax with C language, and it is compiled again into machine codes for GPU by API provided by the vender of graphics board. It is also possible to use GLSL and API of OpenGL, but we use this subset because it is guaranteed to be executable in any machine officially released with Mac OS X 10.6. The GPU executes the machine code in parallel for each pixel. The throughput time depends on the number of processing units in GPU, communication speed between CPU and GPU for image data, and so on. It is ideal if the GPU has enough capacity to compute all of the colour values simultaneously. You can refer [10] to see the detail of compilation from a genotype to a kernel code and the result of benchmark test among different GPUs. The result showed it was not fast enough before 2009, but recent GPUs have satisfiable power for real-time rendering.

4. Applications

The compiled code of each genotype is possible to be exported to another application software that utilises Core Image Framework. One typical one is Quartz Composer, Apple's powerful programming system for visual synthesis in a style of graphical programming. Another one is Final Cut Pro/Express, Apple's video authoring tool for professional quality. It also easy to build a new application that has a capability to import the code from SBArt4.

4.1 Visual programming

Quartz Composer is a powerful visual programming environment that allows the user to combine a number of patches connecting their inputs and outputs with well-designed graphical user interface. It is useful to build visual effects for a screen saver, a live performance, a stuff for video authoring, and so on. It includes many types of useful built-in patches of video filters, image generators, math calculator, image importer, and so on. One of the useful categories is *programming patch*. A patch in this category accepts program codes in Java script or Core Image Kernel Language. The code bred in SBArt4 can be exported to this type of patch by copy & paste operation. Figure 2 shows an example of the arrangement of patches in Quartz Composer's editor. It is also possible to make a reactive visual effect against sound inputs and camera inputs by combination with built-in patches managing such audio-visual inputs.

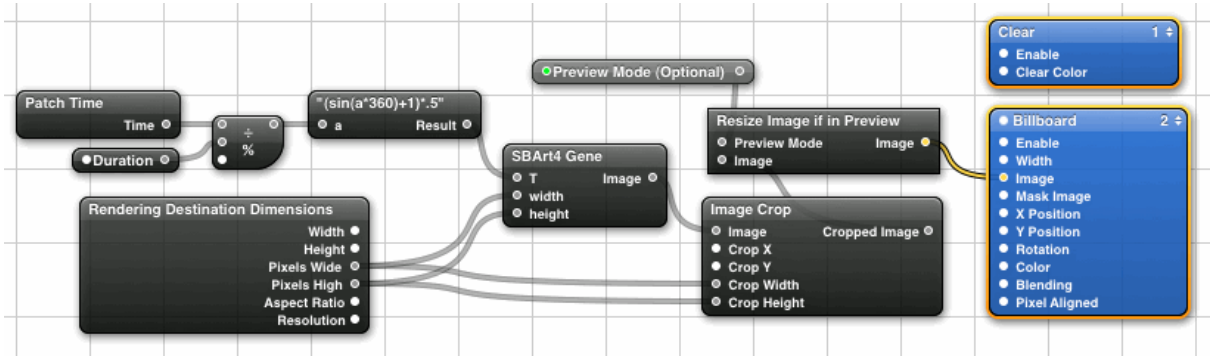


Figure 2. A sample program of Quartz Composer to embed a code generated by SBArt4. The black round rectangle labeled **SBArt4 Gene** at the centre is a programming patch with Core Image Kernel Code.

4.2 Video effects

Apple is providing SDK (Software Development Kit) named FxPlug to build plug-in softwares for Final Cut Pro/Express, a video authoring tool. We tried to develop four types of plug-ins that allows the user to embed a code bred by SBArt4. All of these plug-ins are for video effects, but one is a generator to generate a new video frames of abstract movie, another one is a transition between scenes, and the other two plug-ins are filters to modify video frames. The generator plug-in accepts the code without an external image. The transition and one of the filters accept the code of deformation or discolouration mode. The other filter accepts the code including **image** function.

For the transition, the modification parameter is changing gradually from 0 to 1 and returning back to 0 along the time, that produces a scene continually changing from the original image of the former scene to the later scene mediated with a deformed or discoloured mixed image of both scenes.

In the case of filter with deformation or discolouration mode, the user is allowed to set up a schedule of the modification parameter along the time by a graphical user interface of the plug-in.

4.3 Slide presenter

A new version of slide presentation tool, such as Power Point by Microsoft and Keynote by Apple, always introduces new effects of animation and transition. To use such an effect is helpful to receive attention from audience. However, it easily becomes old-fashioned and less effective soon after audience have many times of experiences of similar types of effects. Especially for the young people in the generation of 3D video games, still images of texts look very boring.

We developed an application software for a slide presentation that shows an animation in the background of each slide and applies transition effect. Both animation and transition are generated by codes bred with SBArt4. The current version of this software reads a PDF file as a series of slides, accepts codes without external image as background animations, and accepts codes of deformation mode

as transition effects. By using a variation of different codes, it is effective to catch attention.

5. Concluding Remarks

The technological innovation on GPUs enabled breeding of abstract movies in real-time on the personal computer. Due to the portability of compiled code in Core Image Kernel Language, a bred code is useful by exporting another software, such as Quartz Composer, Final Cut Pro, and the author's original application for slide presentation. Because the complexity of abstract movies and video effects are much larger than the case of 2D image, there must be rich domain of applications of generative productions. This also means there are a number of new technical issues for practical usage for any fields such as engineering, entertainment, and art. To reduce the complexity of search domain, we implemented a functionality of partial breeding [11] in SBArt4 to lock mutability for each parametric gene. It is useful for trained users. The breeding process of movie has a similar problem with the case of music [12] because the breeder needs to waste a time to evaluate each individual phenotype. By this reason, it is required to develop an effective method to reduce the user's fatigue by combining with some intelligent algorithm for automated evaluation.

The author hope that this challenge of real-time rendering for breeding an animation inspire another idea in the domain of generative art. The binary code of SBArt4 is available from the following URL on the internet.

http://www.intlab.soka.ac.jp/~unemi/sbart/4/

Acknowledgement

This work was supported by Soka University and Grant-in-Aid for Scientific Research (C) No. 21500224 from the Ministry of Education, Culture, Sports, Science and Technology in Japan. The author would like to thank Daniel Bisig, John Flury, Hideyuki Takagi, Hitoshi Iba, Isao Ono, Shigenobu Kobayashi for their helpful comments to develop and improve SBArt4.

References

[1] H. Takagi, Interactive Evolutionary Computation: Fusion of the Capabilities of EC Optimization and Human Evaluation, Proceedings of the IEEE, Vol. 89, No. 9, pp. 1275-1296, 2001.

[2] K. Sims, Artificial Evolution for Computer Graphics, Computer Graphics, Vol. 25, pp. 319-328, 1991.

[3] T. Unemi, A Design of Multi-Field User Interface for Simulated Breeding, in Proceedings of the third Asian Fuzzy Systems Symposium, Masan, Korea, pp. 489-494, 1998.

[4] T. Unemi, Simulated Breeding: A Framework of Breeding Artifacts on the Computer, in Artificial Life Models in Software - Second Edition, (Chapter 12), Edited by M. Komosinski and A. Adamatzky, London, UK, Springer-Verlag, 2009.

[5] T. Unemi, SBART 2.4: an IEC Tool for Creating 2D Images, Movies, and Collage, Leonardo, Vol. 35, No. 2, pp 171, 189-191, MIT Press, 2002.

[6] T. Unemi, Embedding Movie into SBART - Breeding Deformed Movies, in Proceedings of the IEEE Conference on Systems, Man and Cybernetics, The Hague, Netherlands, 2004.

[7] J. R. Koza, Genetic Programming: On The Programming of Computers by Means of Natural Selection, Cambridge, MA: MIT Press, 1992.

[8] Apple Corp., Core Image Kernel Language Reference, in Mac OS X Reference Library, http://developer.apple.com/mac/library/ , 2008.

[9] J. Kessenich, The OpenGL Shading Language - Language Version: 1.50, The Khronos Group Inc., http://www.opengl.org/documentation/glsl/, 2009.

[10] T. Unemi, SBArt4 - Breeding Abstract Animations in Real time, Proceedings of the Conference on Evolutionary Computation 2010, pp. 4004-4009, Barcelona, Spain, 2010.

[11] T. Unemi, Partial Breeding - a method of IEC for well-structured large scale target domains, in Proceedings of the IEEE Conference on Systems, Man and Cybernetics, TP1D4, Yasmine Hammamet, Tunisia, 2002.

[12] T. Unemi, A Tool for Multi-part Music Composition by Simulated Breeding, in Proceedings of the Eighth International Conference on Artificial Life, Sydney, Australia, pp. 410-413, 2002.