Weaving Effects in Metamorphic Animation of Tree-like Fractal based on a Family of Multi-transitional Iterated Function System Code

Tedjo Darmanto
Department of Informatics,
STMIK AMIK Bandung,
Bandung, Indonesia
tedjodarmanto@gmail.com

Iping S. Suwardi
School of Electr.Eng.& Informatics,
Bandung Institute of Technology,
Bandung, Indonesia
iping@informatika.org

Rinaldi Munir
School of Electr.Eng.& Informatics,
Bandung Institute of Technology,
Bandung, Indonesia
rinaldi@informatika.org

***Abstract*— In this paper, the weaving effects in metamorphic animation of tree-like fractals are presented. The metamorphic animation technique is very fascinating, especially when the animation is involving objects in nature that can be represented by fractals. Through the inverse problem process, objects in nature can be encoded into IFS fractals form by means of the collage theorem and affine function as IFS code. Swing direction from left to right or vice versa or weaving effect in metamorphic animation of tree-like fractal can be simulated based on a family of multi-transitional IFS code naturally between the start and target objects by means of IFS randomize algorithm, so can be observed gradually and smoothly.**

Keywords—Fractal; metamorphic animation; tree-like fractal; multi-transitional IFS code; weaving effect; collage theorem; affine function

I. INTRODUCTION

The metamorphic technology is one of the most important technologies in the computer animation. Therefore, the objective of this study is to design a fractal-based algorithm and produce a metamorphic animation based on a fractal idea. Recent studies show that the fractal idea can be effectively applied in the metamorphic animation that can improve the efficiency of animation production by fractal algorithm version and simultaneously greatly reduce the cost that the conventional algorithm version cannot [5,12,13]. One of the contribution of this study is to show a method of metamorphic animation that can produce weaving effects with gradual and smooth visualization.

In this paper, there are five sections. The first and the last sections are introduction and conclusion. In between both sections there are other three sections, those are related works, method and simulation. In this introductory section, the discussion begin with the basic terminology such as fractal and fractal geometry, contractive affine transformation, iterated function system code, and generator algorithm of iterated function system fractal in conjunction with the metamorphic animation in fractal form.

A. Fractal and Fractal Geometry

The term of fractal is first introduced by Mandelbrot, picked from a Latin word: *fractus*, which has a meaning: broken or fractured [1]. One way to generate a fractal object is by L or Lindenmayer systems which is first introduced by Lindenmayer and is suitable for generating the plant-like objects [2]. Another way to generate a fractal is by the iterated function system (IFS) which will be used in metamorphic animation of tree-like fractal in this paper. Barnsley based on Hutchinson's idea as mathematical background introduced IFS for the first time [3, 4]. Since his research, many researchers are following him. In euclidean geometry the range of dimension is in discrete integer number, but in fractal geometry as a superset of the euclidean geometry, the range of dimension can be in fractional numbers continuously.

B. Contractive Affine Transformation

The term of contractive affine transformation (CAT) function is special case of affine transformation function for generating fractal objects which have a self-similarity property, so is called also as self-affine function. The self-similarity as a property of a fractal object means that parts of an object can represent an object as a whole in smaller scale with the different position and orientation. The CAT function in 2D fractal maps the next position of points (**x'**, **y'**) as a vector in an object that depend on the previous ones (**x**, **y**) by a 2 rows and 2 columns matrix which has four coefficients: **a**, **b**, **c** and **d** and a vector (2 rows and 1 column) which has two coefficients: **e** and **f**, so totally there are six coefficients as described in equation (1) below. The coefficients in the matrix represent the form and the orientation of object around the centroid and the coefficients in the vector represent the scale and the position relatively from the centroid.

$$w\begin{bmatrix}x'\\y'\end{bmatrix} = \begin{bmatrix}a & b\\c & d\end{bmatrix} * \begin{bmatrix}x\\y\end{bmatrix} + \begin{bmatrix}e\\f\end{bmatrix} \qquad (1)$$

C. Iterated Function System Code

Fractal objects in IFS form is represented by IFS code set, which is actually a collection of CAT coefficients. Typically a 2D object in IFS fractal form can be encoded as one or more collections of six coefficients: **a**, **b**, **c**, **d**, **e** and **f**. One IFS code set represents one part of a fractal object that has similarity to the object as a whole as already mentioned in the previous section above. The coefficient-**a**, **b**, **c** and **d** represent and determine the shape of the fractal object in **x** and **y** directions, and the the other two coefficients, **e** and **f** represent and determine the position and scale of the object [4]. The typical IFS code (tree-like fractal in 6 CAT functions) as an example with probability values **p** is displayed in table-I below. The probability values represented by percentage numbers or the probability factors represented by fraction numbers between 0.0 and 1.0 determine the population of pixels in part of object represented by each CAT function.

TABLE I. IFS CODE OF 6 CAT TREE-LIKE FRACTAL (A)

a	b	c	d	e	f	p
0.040	0.000	0.000	-0.440	0.000	0.390	8.0
0.040	0.000	0.000	-0.440	0.000	0.590	8.0
0.389	0.289	-0.389	0.345	0.000	0.220	21.0
0.345	-0.257	0.289	0.306	0.000	0.240	21.0
0.390	-0.275	0.225	0.476	0.000	0.440	21.0
0.408	0.190	-0.190	0.408	0.000	0.480	21.0

The first two rows of function represents the trunk, the second and third represent the middle branches (right and left) and the two last row represent the top branches (left and right). The correspondent figure of Table-I is displayed in Fig.1 (a and b)

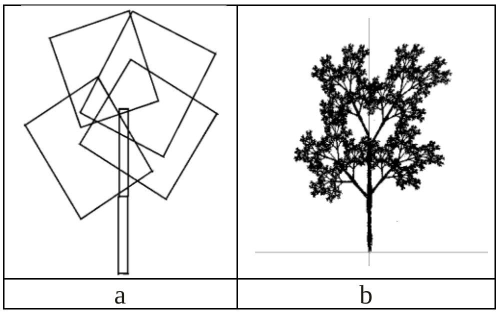


Fig. 1. (a) Collage theorem layout. (b) The correspondent fractal object

D. *Generator Algorithm of Iterated Function System Fractal*

In general there are two major fractal generator algorithms. The first is L systems generator algorithm and the second is IFS generator algorithm. The IFS generator algorithm itself is divided into two IFS generator algorithms. The first is randomize IFS generator algorithm and the second is deterministic IFS generator algorithm. The randomize IFS generator algorithm is also called the random iteration algorithm and the second algorithm is also called the deterministic iteration algorithm [2, 4, 6].

II. RELATED WORKS

A. *Animation and Fixed-Point Algorithm*

As the representation of recent researches in algorithm releated to the fractal construction, two papers are discussed here. First, Chen et al. have presented in their paper a new fractal-based algorithm for the metamorphic animation. The conceptual roots of fractals can be traced to the attempts to measure the size of objects for which traditional definitions based on Euclidean geometry or calculus failed [5]. In his research as the second representation paper, Chang has proposed a hierarchical fixed point-searching algorithm that can determine the coarse shape, the original coordinates, and their scales of 2-D fractal sets directly from its IFS code. Then the IFS codes are modified to generate the new 2-D fractal sets that can be the arbitrary affine transformation of original fractal sets. The transformations for 2-D fractal sets include translation, scaling, shearing, dilation /contraction, rotation, and reflection. The composition effects of the transformations above can also be accomplished through the matrix multiplication and represented by a single matrix and can be synthesized into a complicated image frame with elaborate design [9].

B. *Iterated Function System Model*

Kocic et al. have presented the AIFS (Affine invariant Iterated Function System) that is a slightly modified IFS defined by Barnsley. Instead of the usual Descartes coordinates, the barycentric system has been used. This allows more handy manipulation with the attractors generated by the IFS [7]. Starting from the original definitions of iterated function systems (IFS) and iterated function systems with probabilities (IFSP) in their paper, Kunze et al. have introduced the notions of iterated multifunction systems (IMS) and iterated multi-function systems with probabilities (IMSP). They considered the IMS and IMSP as operators on the space H(H(X)), the space of (nonempty) compact subsets of the space H(X) of (nonempty) compact subsets of the complete metric base space or pixel space (X; d) on which the attractors are supported [10].

C. *Interpolation and Morphing*

As a representation paper in fractal interpolation subject, in their paper Zhang et al. have proposed the general formula and the inverse algorithm for the multi-dimensional piece-wise self-affine fractal interpolation model for the multi-dimensional piece-wise hidden-variable fractal model [8]. As the other representation paper in fractal interpolation subject, in his thesis, Scealy has studied primarily on V-variable fractals, as recently developed by Barnsley, Hutchinson and Stenflo. He extended fractal interpolation functions to the random (and in particular, the V-variable) setting, and calculated the box-counting dimension of particular class of V-variable fractal interpolation functions. The extension of fractal interpolation functions to the V-variable setting yields a class of random fractal interpolation functions for which the box-counting dimensions may be approximated computation-ally, and may be computed exactly for the {V, k}-variable subclass. [11]. As mentioned in previous sub-sections Chen at.al have proposed a new fractal-based algorithm for the metamorphic animation The proposed main method is to weight two IFS (Iterated Function System) codes between the start and the target object by an interpolation function. The experimental results demonstrate that the animation generated according to their method is smooth, natural and fluent [5]. In their paper, Zhuang et al. have proposed a new IFS corresponding method based on rotation matching and local coarse convex-hull, which ensures both that one IFS's local attractor morph to the most similar local attractor from the other IFS, and the fractal feature is preserved during morphing procedure. The coarse convex-hull and rotation matching is very easy to create. Furthermore, they can be used for controlling and manipulating 2D fractal shapes [12].

III. METHOD

A. Multi-Transitional Iterated Function System

Basically in transitional IFS code there are two things should be considered. The first one is the number of CAT function for both IFS code set of the start and target. It is easy to interpolate coefficient of IFS code in between the start and target, if the number of self-affine function in the start and target is the same. If it ‘s not then the dummy function should be inserted into one of IFS code set either the start or the target, so the number of CAT function in both sets becomes the same. The second one to be considered is the sequence of CAT function in the start and target should be also the same based on the part of object position represented relatively. If two things mentioned above are satisfied, then a pair of IFS code sets as the start and target is in a family of transitional IFS code. If there are more then one pair of IFS code sets as nodes of many transitions, then the collection pair of IFS code sets are in a family of multi-transitional IFS code [13].

B. Metamorphic Animation

Metamorphic animation of tree-like fractal object can be accomplished if there are pairs collection of IFS code set in a family of multi-transitional IFS code by interpolating partially the corresponding CAT coefficients of each the start and target IFS code sets as a node, cyclically node by node. There are two types of partial interpolation IFS code. The first one is interpolating all coeffcients in each function of IFS code set, but is not for all functions of the IFS code set are interpolated. The second one is interpolating all CAT functions of the start to all CAT functions of target in the IFS code set, but is not for all coefficients in functions are interpolated.

The pairs of IFS code sets for the first type of partial interpolation, that has two simulations are displayed in Table-II to Table-V for the first simulation and in Table-VI to Table-IX for the second simulation below and the pairs of IFS code sets for the second type of partial interpolation as the third simulation are displayed in Table-I (type-a) above as the first IFS code example in this paper and in Table-X (type-b) and Table-XI (type-c) below. To minimize the space, Table-III to Table-V, and Table-VII to Table-IX show only the three last CAT functions representing the only parts of object that may contributed in morphing animation (the other CAT functions are the same as in Table-II for the 9 CAT version, and in Table-VI for the 10 CAT version).

TABLE II. IFS CODE OF 9 CAT TREE-LIKE FRACTAL (A)

a	b	C	d	e	f	p
0.010	-0.022	0.001	-0.143	-0.006	0.080	3.0
0.010	-0.024	0.001	0.168	0.000	0.000	3.0
0.010	0.023	-0.001	0.159	-0.015	0.100	3.0
0.009	0.160	-0.010	0.146	0.000	0.200	3.0
0.008	-0.099	0.006	0.130	0.000	0.200	3.0
0.002	-0.117	0.008	0.027	-0.060	0.280	3.0
0.569	**0.267**	**-0.315**	**0.481**	**0.100**	**0.290**	**28.0**
0.531	**0.116**	**-0.142**	**0.435**	**-0.052**	**0.280**	**24.0**
0.420	**-0.125**	**0.161**	**0.327**	**-0.130**	**0.300**	**20.0**

TABLE III. IFS CODE OF 3 LAST OF 9 CAT TREE-LIKE FRACTAL (B)

a	b	c	d	e	f	p
0.607	**0.197**	**-0.233**	**0.513**	**0.100**	**0.290**	**28.0**
0.531	**0.116**	**-0.142**	**0.435**	**-0.052**	**0.280**	**24.0**
0.420	**-0.125**	**0.161**	**0.327**	**-0.130**	**0.300**	**20.0**

TABLE IV. IFS CODE OF 3 LAST OF 9 CAT TREE-LIKE FRACTAL (C)

a	b	c	d	e	f	p
0.607	**0.197**	**-0.233**	**0.513**	**0.100**	**0.290**	**28.0**
0.548	**0.039**	**-0.048**	**0.448**	**-0.052**	**0.280**	**24.0**
0.420	**-0.125**	**0.161**	**0.327**	**-0.130**	**0.300**	**20.0**

TABLE V. IFS CODE OF 3 LAST OF 9 CAT TREE-LIKE FRACTAL (D)

a	b	c	d	e	f	p
0.607	**0.197**	**-0.233**	**0.513**	**0.100**	**0.290**	**28.0**
0.548	**0.039**	**-0.048**	**0.448**	**-0.052**	**0.280**	**24.0**
0.377	**-0.191**	**0.245**	**0.294**	**-0.130**	**0.300**	**20.0**

TABLE VI. IFS CODE OF 10 CAT TREE-LIKE FRACTAL (A)

a	b	c	d	e	f	p
0.010	-0.022	0.001	0.168	0.000	0.000	2.0
0.010	0.021	-0.001	0.159	-0.015	0.100	2.0
0.009	0.146	-0.010	0.146	0.000	0.200	2.0
0.008	-0.090	0.006	0.130	0.000	0.200	2.0
0.008	0.015	-0.001	0.119	-0.060	0.280	2.0
0.002	-0.117	0.008	0.027	-0.060	0.280	2.0
0.007	-0.036	0.003	0.093	-0.138	0.290	2.0
0.607	**0.197**	**-0.233**	**0.513**	**0.100**	**0.290**	**40.0**
0.548	**0.039**	**-0.048**	**0.448**	**-0.052**	**0.350**	**26.0**
0.377	**-0.191**	**0.245**	**0.294**	**-0.160**	**0.350**	**20.0**

TABLE VII. IFS CODE OF 3 LAST OF 10 CAT TREE-LIKE FRACTAL (B)

a	b	c	d	e	f	p
0.607	**0.197**	**-0.233**	**0.513**	**0.100**	**0.290**	**40.0**
0.548	**0.039**	**-0.048**	**0.448**	**-0.052**	**0.350**	**26.0**
0.420	**-0.125**	**0.161**	**0.327**	**-0.160**	**0.350**	**20.0**

TABLE VIII. IFS CODE OF 3 LAST OF 10 CAT TREE-LIKE FRACTAL (C)

a	b	c	d	e	f	p
0.607	**0.197**	**-0.233**	**0.513**	**0.100**	**0.290**	**40.0**
0.531	**0.116**	**-0.142**	**0.435**	**-0.052**	**0.350**	**26.0**
0.420	**-0.125**	**0.161**	**0.327**	**-0.160**	**0.350**	**20.0**

TABLE IX. IFS CODE OF 3 LAST OF 10 CAT TREE-LIKE FRACTAL (D)

a	b	c	d	e	f	p
0.569	**0.267**	**-0.315**	**0.481**	**0.100**	**0.290**	**40.0**
0.531	**0.116**	**-0.142**	**0.435**	**-0.052**	**0.350**	**26.0**
0.420	**-0.125**	**0.161**	**0.327**	**-0.160**	**0.350**	**20.0**

TABLE X. IFS CODE OF 6 CAT TREE-LIKE FRACTAL (B)

a	b	c	d	e	f	p
0.040	-0.046	-0.004	-0.438	**0.040**	0.390	8.0
0.040	-0.088	-0.008	-0.441	**0.100**	0.590	8.0
0.389	0.289	-0.389	0.345	**0.020**	0.220	21.0
0.345	-0.257	0.289	0.306	**0.020**	0.240	21.0
0.390	-0.275	0.225	0.476	**0.090**	0.440	21.0
0.408	0.190	-0.190	0.408	**0.090**	0.480	21.0

TABLE XI. IFS CODE OF 6 CAT TREE-LIKE FRACTAL (C)

a	b	c	d	e	f	p
0.040	0.046	0.004	-0.438	**-0.040**	0.390	8.0
0.040	0.088	0.008	-0.441	**-0.090**	0.590	8.0
0.389	0.289	-0.389	0.345	**-0.020**	0.220	21.0
0.345	-0.257	0.289	0.306	**-0.020**	0.240	21.0
0.390	-0.275	0.225	0.476	**-0.060**	0.440	21.0
0.408	0.190	-0.190	0.408	**-0.060**	0.480	21.0

IV. SIMULATION

In this section of this paper, there are three kinds of simulation that have the different types of weaving effect resulted. The first and second simulations show partial weaving effect from left to right and vice versa. The third simulation shows total weaving effect form left to right and back. To generate the fractal images in this simulation, the IFS randomize algorithm is used.

In the first simulation, by comparing between two images for an example the images at T1 and T2 in Fig.2 below as a begining of transition event, there is a non-linear change of left branch of the tree-like fractal smoothly, especially the morphing effect of leaves on top of the branch from left to right is changed gradually and decreasingly to the right. For another example the images at T5 and T6 in Fig.2 below as an end of transition event, there is also a non-linear change of left branch of the tree-like fractal smoothly, especially the morphing effect of leaves on top of the branch, but in reverse direction of the previous example from right to left is changed gradually and increasingly to the left. The moderate morphing effect is occured between images at T3 and T4 in Fig.2 below. So the non-linear morphing effect is showed dramatically from one event to other events especially at the begining and at the end of events by metamorphic animation of tree-like object as a fractal object. By contrast the dramatical event occured in the middle event of the second simulation as displayed in Fig.3 below at T3, T4 and T5. The partial weaving effect is exhibited in the first and second simulations as explained later in the next sub-section. The third simulation shows weaving effect by bending the trunk of tree gently from left to right and back, The total weaving effect is exhibited in this simulation as explained later in the next sub-section.

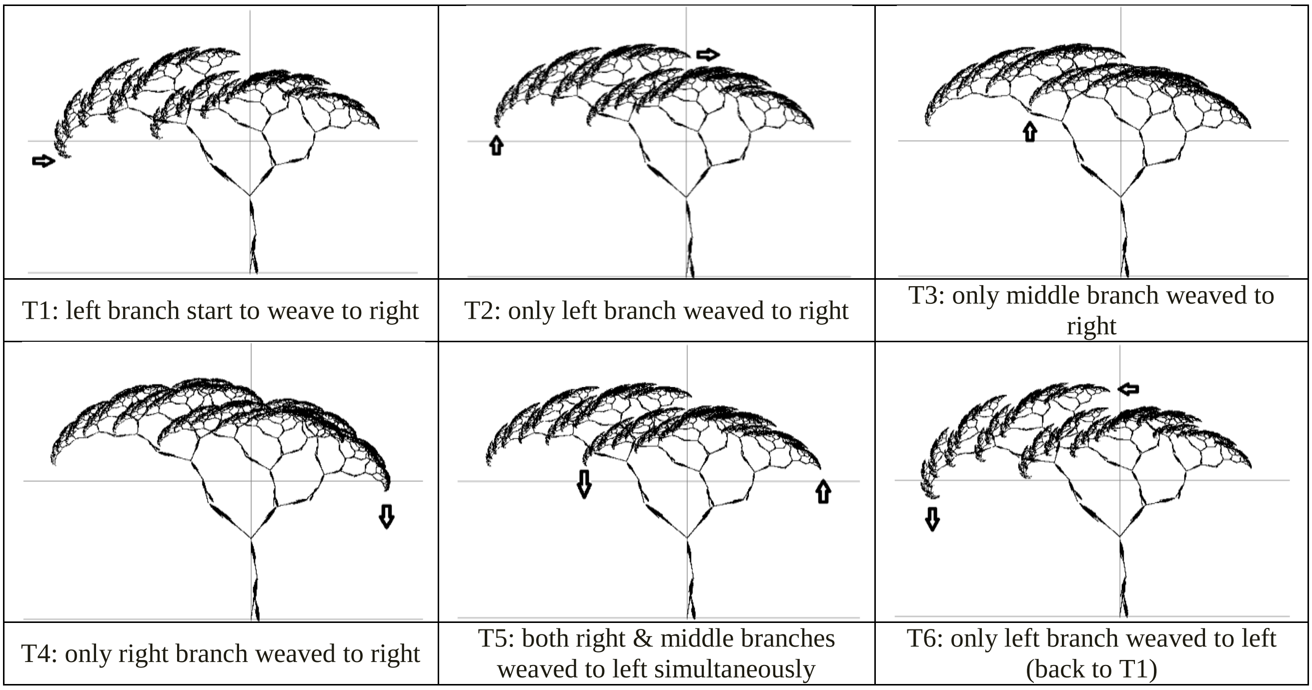
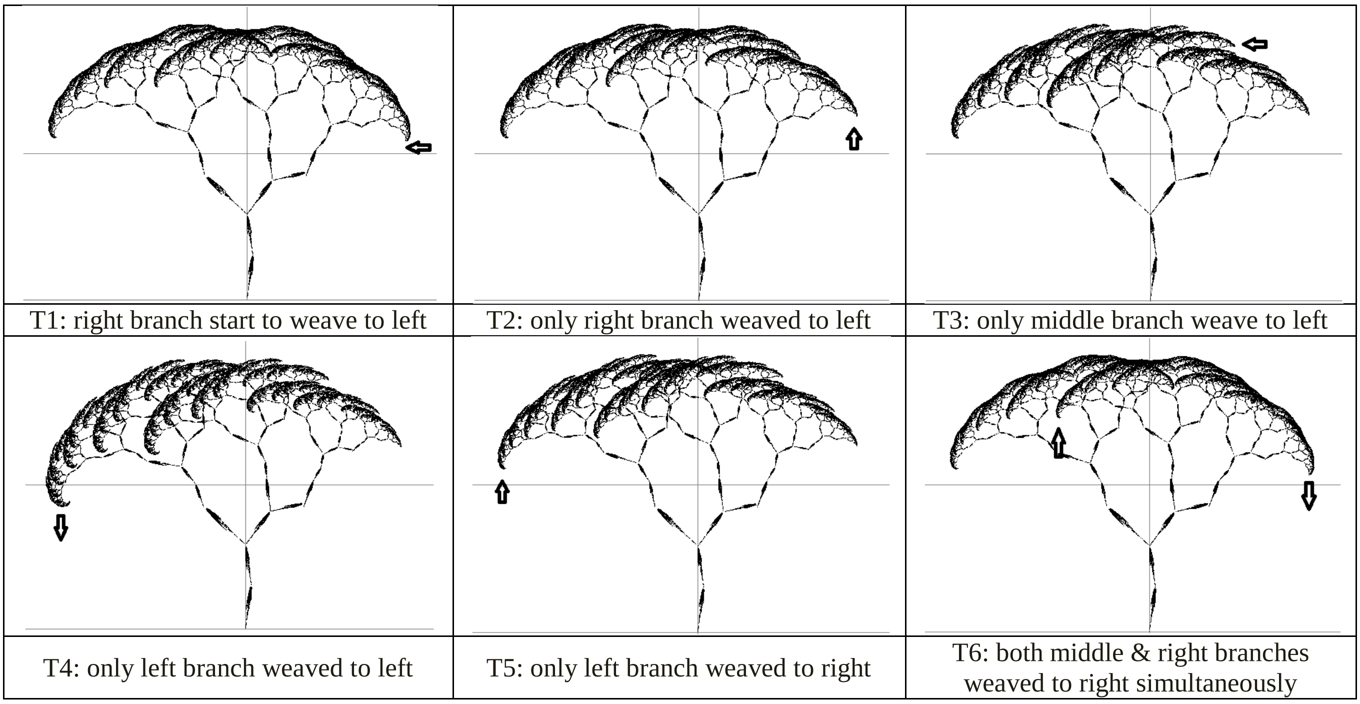


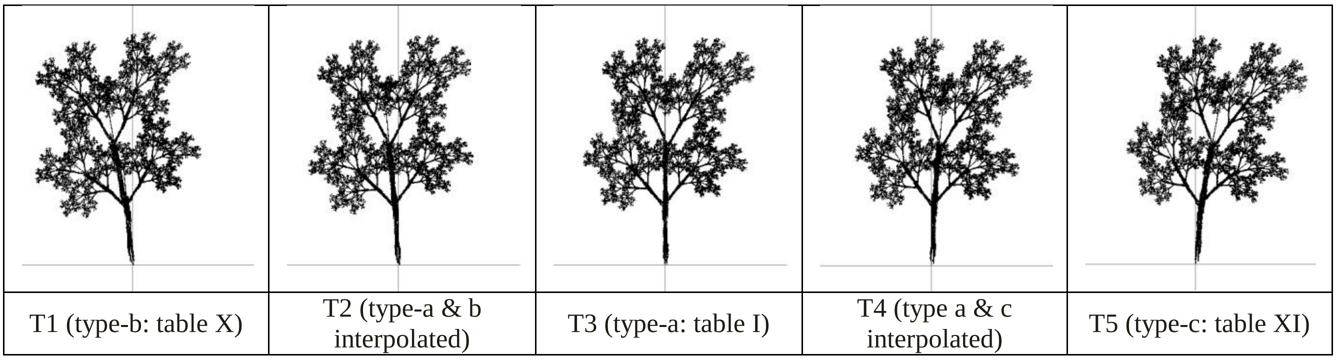
Fig. 2. Transitional Images as the result of Metamorphic Animation of the 9 CAT tree-like fractal showing partial weaving effect (left to right and back)



T1: right branch start to weave to left | T2: only right branch weaved to left | T3: only middle branch weave to left

T4: only left branch weaved to left | T5: only left branch weaved to right | T6: both middle & right branches weaved to right simultaneously

Fig. 3. Transitional Images as the result of Metamorphic Animation of the 10 CAT tree-like fractal showing partial weaving effect (right to left and back)



T1 (type-b: table X) | T2 (type-a & b interpolated) | T3 (type-a: table I) | T4 (type a & c interpolated) | T5 (type-c: table XI)

Fig. 4. Transitional Images as the result of Metamorphic Animation of the 6 CAT tree-like fractal showing total weaving effect left to right and back from T6 to T9 like in reversal order from T4 to T1 (that are not displayed here)

A. Partial Weaving Effect

To simulate the metamorphic animation that is showing a weaving effect partially, the 9 CAT and 10 CAT tree-like fractals are used as two examples. To prepare this kind of animation, all coefficient of the last three CAT functions (marked in bold type) are modified one function at a time consecutively as can be seen in Table-II to V (for the 9 CAT tree-like fractal version) and Table-VI to IX (for the 10 CAT tree-like fractal version) below. The first and second of the six transitional images sets as the results of those animations that show the partial weaving effect of top branches from left to right and vice versa, can be seen at Fig.2 and Fig.3 above. To clarify the sequence of the partial weaving effect which are occured, please examine the notes at the bottom of each image from time sequence: T1 to T6 in both figures.

B. Total Weaving Effect

To simulate the metamorphic animation that is showing a weaving effect totally, the 6 CAT tree-like fractal is used as an example. To prepare this kind of animation, all coefficient-**e** (scale and position factor in axis-**x**) of all functions in Table-I (as type-a) are modified by shifting to the right and left (as type b and c), and the results can be seen in Table-X and XI (marked in bold type). By putting IFS code set in Tabel-X as the start and IFS code set in Table-I as target of node-1 of animation and IFS code set in Tabel-I as the start and IFS code set in Table-XI as target of node-2 of animation etc., then the total weaving effect will be exhibited by the animation. The five transitional images as the result of this animation that shows the total weaving effect of top and middle branches from left to right and back and also shows the bending effect of the trunk, can be seen at Fig.4 above. Please also examine the notes at the bottom of each image from time sequence: T1 to T5.

V. CONCLUSION

From the discussion of the metamorphic animation of tree-like fractal based on a family of multi-transitional IFS code and it's simulation in the above sections, we conclude that there are two kinds of weaving effect as the results of metamorphic animations of tree-like fractals that depend on the kind of interpolation chosen. The metamorphic animations of the 9 and 10 CAT tree-like fractals show the partial weaving effect and the 6 CAT tree-like fractal shows the total weaving effect. There is a non-linear change occured in morphing effect especially in partial weaving simulations as another conclusion.

REFERENCES

[1] Mandelbrot, Benoit B., "The Fractal Geometry of Nature", W.H. Freeman and Company, 1982.

[2] Lindenmayer, A., Fracchia, F.D., Hanan, J., Krithivasan, K., Prusinkiewicz, P., "Lindenmayer Systems, Fractals, and Plants", Springer Verlag, 1992.

[3] Hutchinson, John E., "Fractals Self-similarity", *Indiana University Mathematics Journal 30*, 1979.

[4] Barnsley, Michael F., "Fractals Everywhere", 2nd edition., Morgan Kaufmann,. Academic Press, 1993.

[5] Chen, C., Zheng, Y., Sarem, M., "A Fractal-based Algorithm for the Metamorphic Animation", *0-7803-9521-2/06 ©2006 IEEE, Page(s): 2957 – 2962*, 2006

[6] Nikiel, S., "Iterated Function Systems for Real-Time Image Synthesis", © Springer-Verlag London Limited, 2007.

[7] Kocic, L., Stefanovska, L., Babace, E. "Affine invariant iterated function systems and the minimal simplex problem", *BSG Proceedings 15. The International Conference "Differential Geometry - Dynamical Systems" DGDS-2007, October 5-7, 2007, pp. 119-128*. © Balkan Society of Geometers, Geometry Balkan Press 2008.

[8] ZHANG Tong; ZHUANG Zhuo, "Multi-Dimensional Piece-Wise Self-Affine Fractal Interpolation Model", *Tsinghua Science and Technology; ISSN 1007-0214 02/18 pp244-251; Volume 12, Number 3, June 2007; D.O.I: 10.1016/S1007-0214(07)70036-6*

[9] Chang, Hsuan T., "Arbitrary Affine Transformation and Their Composition Effects for Two-Dimensional Fractal Sets", Photonics and Information Laboratory Department of Electrical Engineering of Taiwan National Yunlin University of Science and Technology, 2009. http://teacher.yuntech.edu.tw/htchang/geof.pdf [visited :2013-07-29]

[10] Kunze, H.E., La Torre, D., Vrscay, E.R., "From Iterated Function Systems to Iterated Multifunction Systems", *Communications on Applied Nonlinear Analysis {\bf 15} (4), 1-14.*, 2008.

[11] Scealy, Robert, "V-Variable Fractals and Interpolation", Philosophy doctoral thesis of the Australian National University, 2009. http://www.superfractals.com/pdfs/scealyopt.pdf [visited :2013-07-29]

[12] Zhuang, Yi-xin, Xiong ,Yue-shan, Liu, Fa-yao, "IFS Fractal Morphing based on Coarse Convex-hull", *978-1-4244-8625-0/11 ©2011 IEEE,* 2011

[13] Darmanto, T., Suwardi, I.S., Munir, R., "Cyclical Metamorphic Animation of Fractal Images based on a Family of Multi-transitional IFS Code Approach", *Control, Systems & Industrial Informatics (ICCSII), IEEE Conference on, D.O.I: 10.1109/CCSII.2012.6470506, Page(s): 231 – 234*, 2012.