Growing Effects in Metamorphic Animation of Plant-like Fractals based on Transitional IFS Code Approach

Tedjo DARMANTO[a*], Iping S. SUWARDI[b]& Rinaldi MUNIR[c]
[a]*Department of Informatics, STMIK Amik Bandung, Indonesia*
[b,c]*STEI, Bandung Institute of Technology, Indonesia*
*tedjodarmanto@gmail.com

Abstract: In this paper, the growing effects in metamorphic animation of plant-like fractals are presented. The metamorphic animation technique is always interesting, especially the animation which is involving objects in the 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 self-affine function. Multidirectional, bidirectional and unidirectional growing effects in metamorphic animation of plant-like fractal can be simulated based on a family of transitional IFS code naturally between the source and target objects by an IFS rendering algorithm.

Keywords: Fractal, metamorphic animation, growing effects, collage theorem, transitional IFS code, self-affine function

1. Introduction

In this paper, there are six sections. The first and the last sections are introduction and conclusion. In between both sections there are other four sections, those are related works, transitional IFS code, metamorphic animation and simulation. In this introductory section, the discussion begin with the basic terminology such as fractal geometry, self-affine, IFS code, and IFS rendering algorithms in conjunction with the metamorphic animation in fractal form

1.1 Fractal Geometry

The term of fractal is first coined by Mandelbrot, picked from a Latin word: *fractus*, which has a meaning: fractured or broken (Mandelbrot, 1982). One way to generate a fractal object is by the iterated function system (IFS) which is first introduced by Barnsley based on Hutchinson's idea as mathematical background (Barnsley, 1993 and Hutchinson, 1979). Another way to generate a fractal is by Lindenmayer or L systems which is first introduced by Lindenmayer and is suitable for generating the tree-like objects (Lindenmayer et.al, 1992). The fractal geometry as a superset of the euclidean geometry can have the range of dimension in fractional numbers continuously, and not as a discrete integer numbers like in the euclidean geometry.

1.2 Self-affine

As the next term, self-affine function is special case of affine transformation function of fractal objects which have a self-similarity property. 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 and in the different orientations. The self-affine function (2D) maps the next position of points (**x'**, **y'**) as a vector in an object that depend on the previous ones (**x**, **y**) by a matrix (2 X 2) which has four coefficients : **a**, **b**, **c** and **d** and a vector which has two coefficients : **e** and **f**, so totally there are six coefficients as described in the equation-1 below.

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

1.3 IFS Code

Fractal objects in iterated function systems or IFS form is represented by IFS code set, which is actually a collection of self-affine function 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 (Barnsley, 1993). The typical IFS code as an example with probability factors **p** is displayed in table-1 and the correspondent figure in figure-1 below. The first row of function represents the center part, the second and third represent the right and left side parts and the two last row represent the the trunk (right and left as a pair in opposite orientation to fill the void area complementarily).

Table 1. An example of IFS code set

Tree-like						
a	b	c	d	E	f	p
-.637	0.000	0.000	0.501	0.025	0.570	20.0
0.215	-.537	0.378	0.487	0.047	0.966	20.0
0.510	0.455	-.277	0.397	-.049	1.001	20.0
-.058	-.070	0.453	-.111	0.117	0.728	20.0
-.035	0.070	-.469	0.022	-.105	0.578	20.0

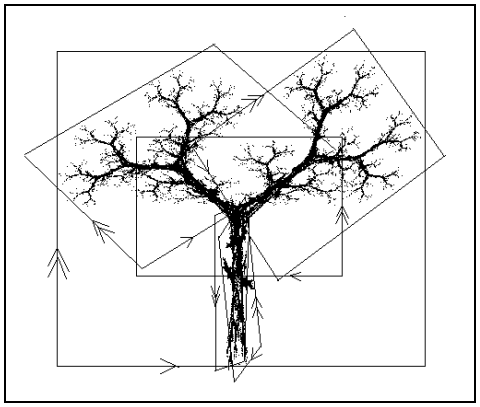


Figure 1. Parts of tree-like fractal

1.4 IFS Rendering Algorithms

In general there are two major IFS rendering algorithms. The first is randomize IFS rendering algorithms and the second is deterministic IFS rendering algorithms. The randomize IFS rendering algorithms is also called the random iteration algorithms, which is divided into : the classic random iteration algorithm, and the colour-stealing random iteration algorithm. The second algorithms is also called the deterministic iteration algorithms, which is divided into : the classic deterministic iteration algorithm, the deterministic iteration algorithm and superfractals, the minimal plotting algorithm, the constant mass algorithm, and the escape-time algorithm (Nikiel, 2008).

2. Related Works

2.1 Algorithm

In their paper, Chen et al. presented 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. Therefore, the objective of this study is to design a fractal-based algorithm and produce a metamorphic animation based on a fractal idea. The main method is to weight two IFS codes between the start and the target object by an interpolation function. Their study shows that the fractal idea can be effectively applied in the metamorphic animation. The main feature of the algorithm is that it can deal with a fractal object that the conventional algorithm cannot. In application, their algorithm has many practical values that can improve the efficiency of animation production and simultaneously greatly reduce the cost (Chen et al., 2006).

In their research, Zhang et al. proposed the general formula and the inverse algorithm for the multi-dimensional piece-wise self-affine fractal interpolation model and presented in their paper to

provide the theoretical basis for the multi-dimensional piece-wise hidden-variable fractal model (Zhang et al., 2007).

In his research Chang 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 (Chang, 2009).

2.2 IFS Model

Starting from the original definitions of iterated function systems (IFS) and iterated function systems with probabilities (IFSP) in their paper, Kunze et al. introduced the notions of iterated multifunction systems (IMS) and iterated multifunction 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 (Kunze et al., 2008).

2.3 Interpolation

In his thesis, Scealy 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 computationally, and may be computed exactly for the {V, k}-variable subclass. In addition he presented a sample application of V-variable fractal interpolation functions to the generation of random families of curves, adapting existing curve generation schemes that are based upon iteration of affine transformations (Scealy, 2009).

2.4 IFS Fractal Morphing

In their paper, Zhuang et al. 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 (Zhuang et al., 2011).

3. Transitional IFS Code

In transitional IFS code the are are two things should be considered. The first one is the number of self-affine function for both IFS code set of the source and target. It is easy to interpolate coefficient of IFS code in between the source and target, if the number of self-affine function in the source and target is the same. If it 's not then the dummy function should be inserted into one of IFS code set either the source or the target, so the number of self-affine function in both sets becomes the same. A dummy function has a probability factor set to zero and the six coefficients are the same as in the counter part (as the source or target). The second one to be considered is the sequence of self-affine function in the source and target should be also the same based on the part of object with the same position and scale represention relatively. If two things mentioned above are satisfied, then a pair of IFS code set as the source and target is in a family of transitional IFS code (Darmanto et al., 2012).

4. Metamorphics Animation

In this paper, there are two kinds of metamorphic animation using the random iteration algorithm with the partial interpolation of IFS code and the whole interpolation of IFS code between the coefficients in functions of the source and target.

4.1 Partial Interpolation IFS Code

There are two types of partial interpolation IFS code. The first one is interpolating all functions of the source to all functions of target in the IFS code set, but is not for all coefficients in a function are interpolated. The second one is interpolating all coeffcients in each function of IFS code set, but is not for all functions of the IFS code set. The pair of IFS code sets example of the first one is displayed in table-2a and table-2b below and the pair of IFS code sets example of the second one are displayed in table-3a and table-3b below. Basically only coefficient-**a** and **c** that are influencing the **x**-direction are interpolated (marked in bold type).

Table-2. A pair IFS code set of grass-like fractal (a. Source; b. Target) : 5 functions

a. IFS code set of grass-like fractal (source)							b. IFS code set of grass-like fractal (target)						
a	b	c	d	e	f	p	a	b	c	d	e	f	P
0.03	0.06	**-0.06**	-0.42	-0.033	0.40	10.0	**0.04**	0.06	**-0.06**	-0.50	-0.033	0.40	10.0
0.10	-0.15	**0.04**	0.37	-0.015	0.20	20.0	**0.24**	-0.15	**0.09**	0.37	-0.015	0.20	20.0
0.08	-0.08	**0.03**	0.19	-0.025	0.30	05.0	**0.18**	-0.08	**0.07**	0.19	-0.025	0.30	05.0
0.15	-0.19	**0.06**	0.46	-0.031	0.40	25.0	**0.36**	-0.19	**0.14**	0.46	-0.031	0.35	25.0
0.23	-0.27	**0.09**	0.65	-0.035	0.40	40.0	**0.54**	-0.27	**0.22**	0.65	-0.033	0.40	40.0

Table-3. A pair IFS code set of bushes-like fractal (a. Source; b. Target) : 4 functions

a. IFS code set of bushes -like fractal (source)							b. IFS code set of bushes -like fractal (target)						
a	b	c	d	e	f	p	a	b	c	d	e	f	p
0.01	0.06	**-0.004**	0.17	0.00	0.00	5.0	**0.01**	0.06	**-0.004**	0.17	0.00	0.00	5.0
0.01	-0.09	**0.004**	0.26	0.00	0.00	5.0	**0.01**	-0.09	**0.004**	0.26	0.00	0.00	5.0
0.50	-0.30	**0.208**	0.79	-0.09	0.22	45.0	**0.60**	-0.30	**0.238**	0.79	-0.09	0.22	45.0
0.50	0.30	**-0.208**	0.79	0.04	0.12	45.0	**0.60**	0.30	**-0.238**	0.79	0.04	0.12	45.0

4.2 Whole Interpolation IFS Code

The whole interpolation IFS code is interpolating each coefficient of all functions of IFS code set between the source and target. The IFS code set example are displayed in table-4a and table-4b below (almost all coefficients are interpolated). This kind of interpolation will show the multidirectional growing effect as described in the next section.

Table-4. A pair IFS code set of tree-like fractal (a. Source; b. Target) : 5 functions

a. IFS code set of tree-like fractal (source)							b. IFS code set of tree-like fractal (target)						
a	b	c	d	e	f	p	a	b	c	d	e	f	p
-0.64	0.00	0.00	0.50	0.025	0.52	20.0	-0.64	0.00	0.00	0.50	0.025	0.57	20.0
0.19	-0.49	0.34	0.44	-0.007	0.92	20.0	0.21	-0.54	0.38	0.49	0.047	0.97	20.0
0.46	0.42	-0.25	0.36	-0.003	0.94	20.0	0.51	0.45	-0.28	0.40	-0.049	1.00	20.0
-0.06	-0.07	0.45	-0.11	0.110	0.62	20.0	-0.06	-0.07	0.45	-0.11	0.117	0.73	20.0
-0.03	0.07	-0.47	0.02	-0.098	0.48	20.0	-0.03	0.07	-0.47	0.02	-0.105	0.58	20.0

5. Simulation

In this paper, there are three kinds of simulation that have the different types of growing effect resulted, those are the unidirectional growing effect, the bidirectional and the multidirectional growing effects. For the simulation purpose three different plant-like fractal objects are used. The first simulation which shows the unidirectional growing effect, a grass-like fractal object is used. The transitional images as

the result of metamorphic animation of that object is displayed in figure-2 below. The second simulation which shows the bidirectional growing effect, a bushes-like fractal object is used. The transitional images as the result of metamorphic animation of that object is displayed in figure-3 below. Finally the third simulation which shows the multidirectional growing effect, a tree-like fractal object is used. The transitional images as the result of metamorphic animation of that object is displayed in figure-4 below.

5.1 Unidirectional Growing Effect

The eight images below shows the moving direction of top object to the right-below (south-east direction) as long as the object is growing.

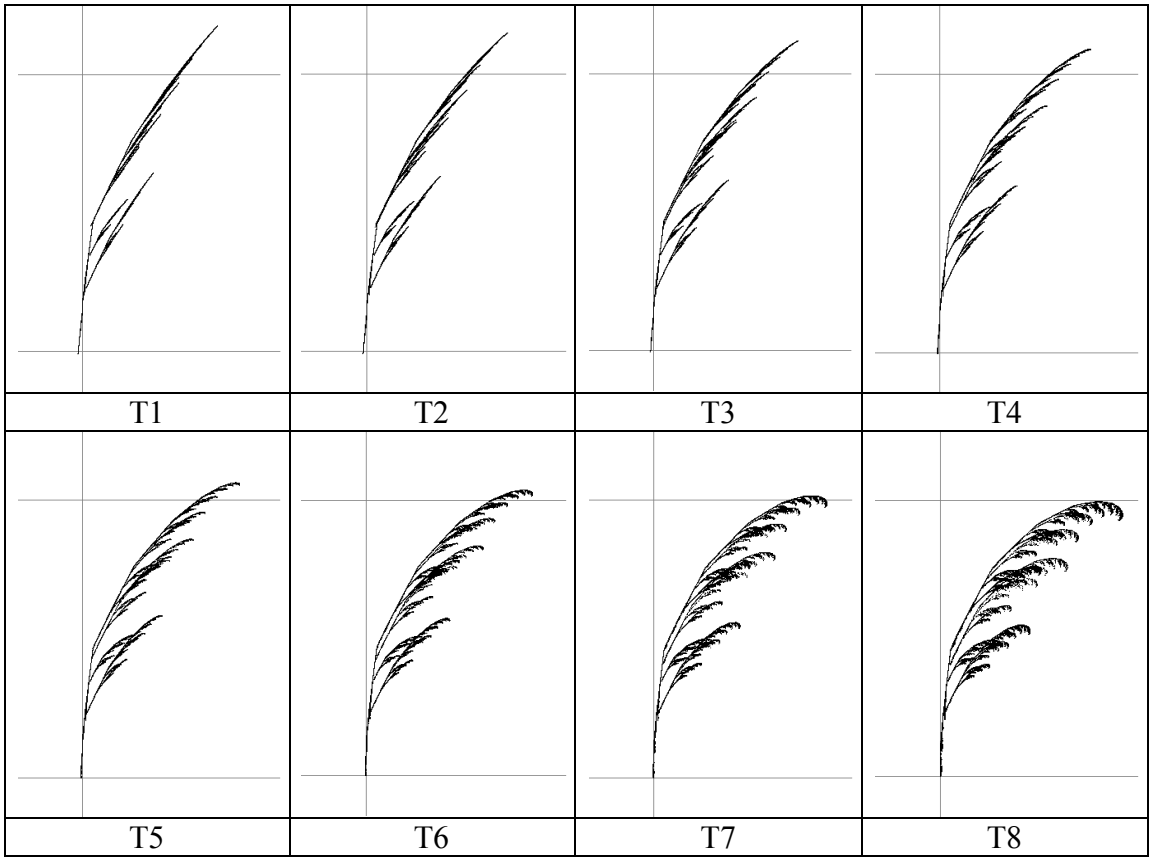
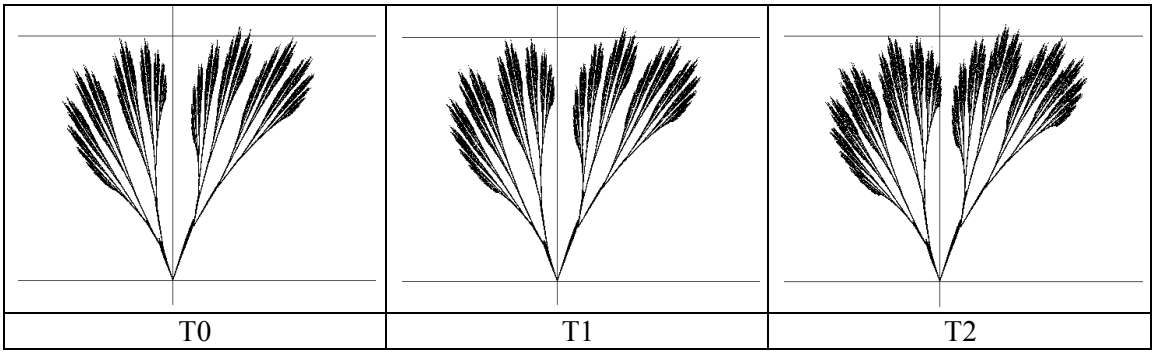


Figure-2. The transitional images in metamorphic animation of grass-like fractal

5.2 Bidirectional Growing Effect

The nine images below shows the growing direction of right side object to the right and left side to the left (horizontally).



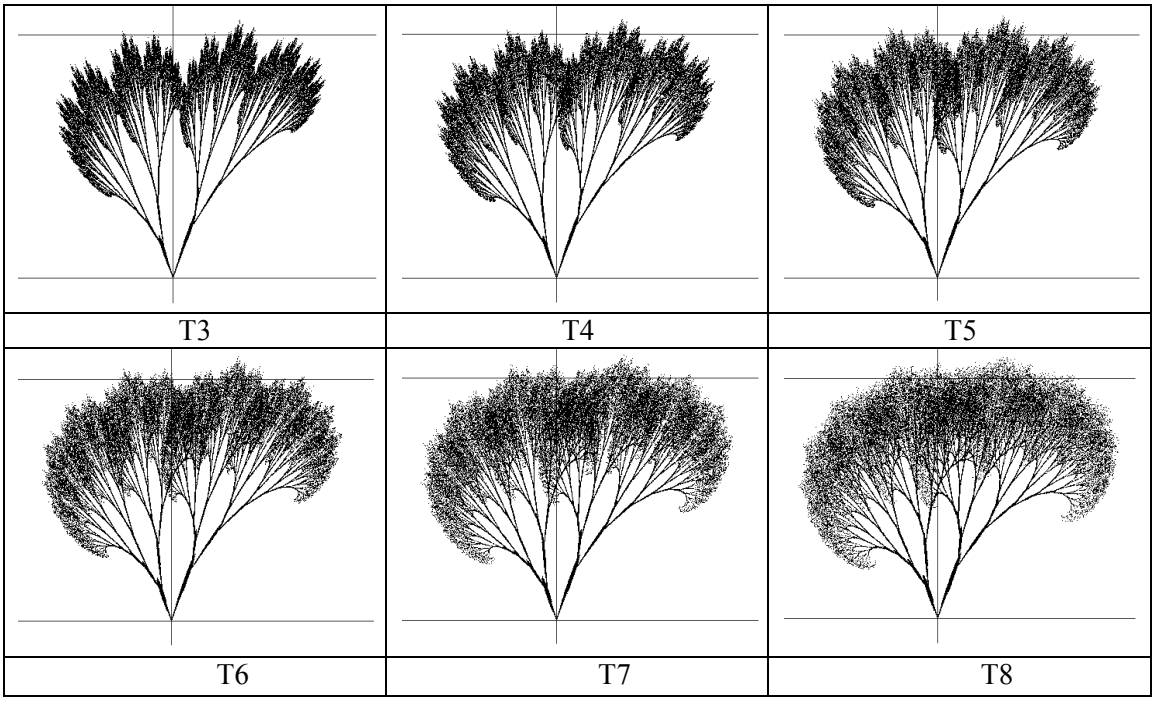
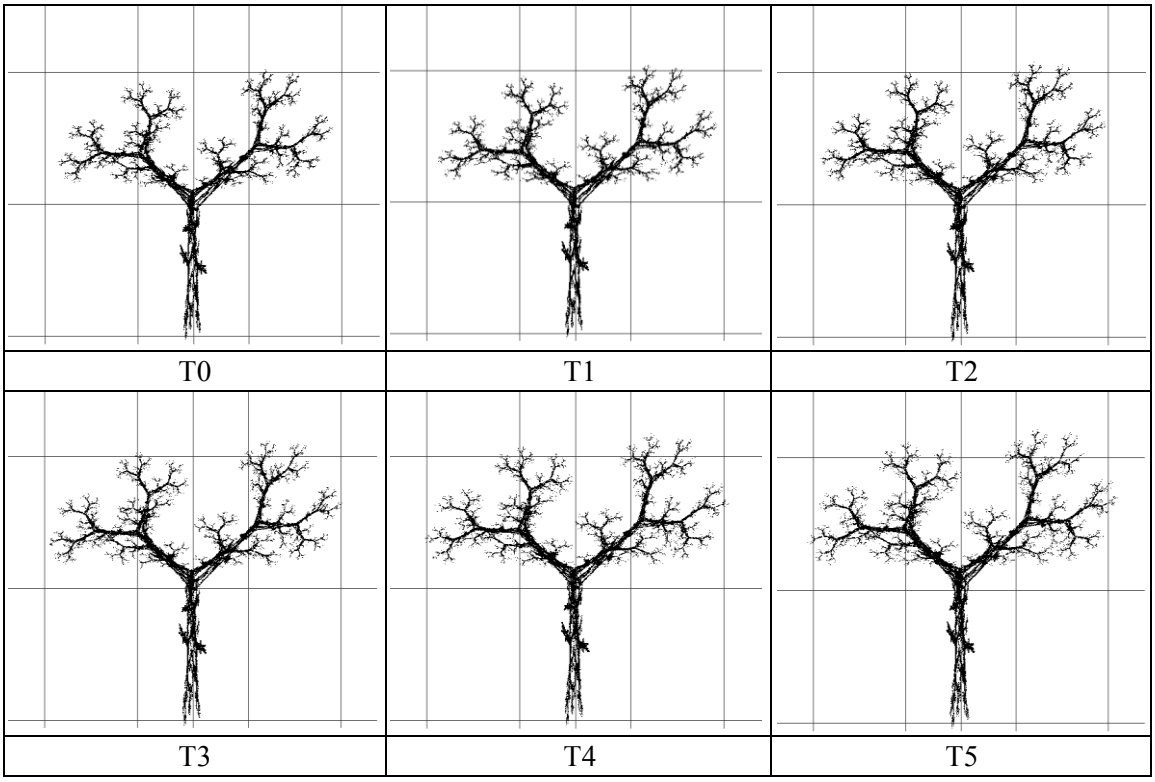
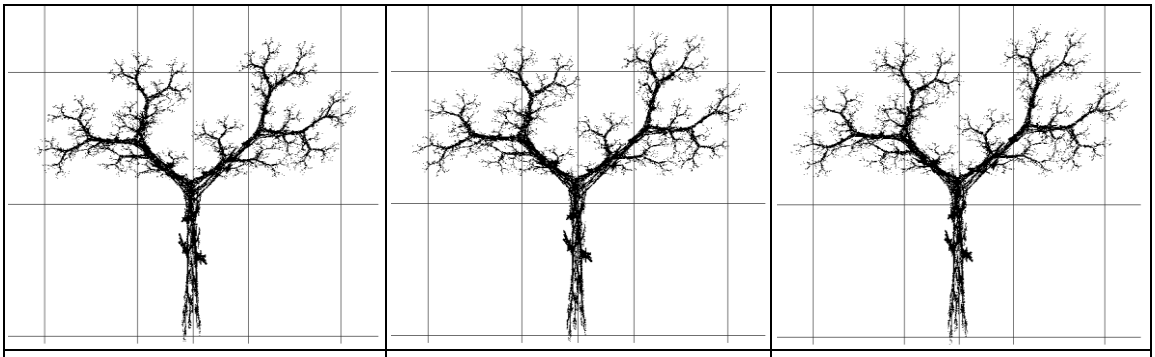


Figure-3. The transitional images in metamorphic animation of bushes-like fractal

5.3 Multidirectional Growing Effect

The nine images below shows the growing direction of top object to the north, right side object to the east and left side object to the west (both horizontally and vertically).





T6	T7	T8

Figure-4. The transitional images in metamorphic animation of tree-like fractal

6. Conclusion

From the metamorphic animation and simulation in the above sections, we can conclude that there are three kinds of growing effect as the result of metamorphic animations of plant-like fractals that depend on the kind of interpolation chosen, i.e: unidirectional, bidirectional and multidirectional growing effects.

Acknowledgements

We would like to thank the Orginizing Committee of IC-MCS 2013 who has informed and invited researchers and lecturers in 'Aptikom' society forum through the official invitation letter.

References

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

Hutchinson, John E. 1979. Fractals Self-similarity. *Indiana University Mathematics Journal 30.* http://gan.anu.edu.au/~john/Assets/Research%20Papers/fractals_self-similarity.pdf [visited :2013-07-31]

Barnsley, Michael F. 1993. Fractals Everywhere. 2nd edition. Morgan Kaufmann,. Academic Press

Lindenmayer, A., Fracchia, F.D., Hanan, J., Krithivasan, K., Prusinkiewicz, P. Lindenmayer Systems, Fractals, and Plants. (1992). Springer Verlag. http://www.booksrating.com/Lindenmayer-Systems-Fractals-and-Plants/p149012/ [visited :2013-07-31]

Chen, Chuan-bo, Zheng, Yun-ping, Sarem, M. (2006). A Fractal-based Algorithm for the Metamorphic Animation. *Information and Communication Technologies*, *ICTTA '06. 2nd, Volume: 2, D.O.I: 10.1109 / ICTTA. 2006. 1684885, Page(s): 2957-2962.*

Nikiel, S. (2007).Iterated Function Systems for Real-Time Image Synthesis. © Springer-Verlag London Limited.

Zhang, Tong, Zhuang, Zhuo. (2007). 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*

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

Chang, Hsuan T. (2009).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. http://teacher.yuntech.edu.tw/htchang/geof.pdf [visited :2013-07-29]

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

Zhuang, Yi-xin, Xiong ,Yue-shan, Liu, Fa-yao. (2011). IFS Fractal Morphing based on Coarse Convex-hull. *978-1-4244-8625-0/11/$26.00 ©2011 IEEE*

Darmanto, T., Suwardi, I.S., Munir, R. (2012). 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*