Advances in Engineering Research, volume 207
Proceedings of the 2nd International Seminar of Science and Applied Technology (ISSAT 2021)

Artistic Characters Generated by Fractal Model Based on the Composition of Components in Grid

Tedjo Darmanto[1,*]

[1] *Informatics Department, STMIK AMIK Bandung, Indonesia*
[*] *Corresponding author. Email: tedjodarmanto@stmik-amikbandung.ac.id*

ABSTRACT
A set of artistic characters can be generated by the inverse problem method in the fractal model, which has an advantage over the traditional model in recycling the flexible art composition. The code of iterated function systems of a fractal can be generated by composing the interrelated components in the collage composition design iteratively based on the collage theorem which is ruled by a set of self-affine transformations. The fractal result is displayed by means of the partitioned-random iteration algorithm as the decoding process of multi-object of fractals as the collection of the artistic characters. The number of the partition is set based on the number of characters in a composition of the artistic characters accordingly, so the individual self-affine transformation of a character can be modified independently.

Keywords: *Iterated Function Systems, Inverse Problem, Fractal Model, Collage Theorem, Partitioned-Random Iteration Algorithm.*

1. INTRODUCTION

The main problem in generating fractal objects such as a set of artistic characters in a fractal model is how to do the inverse problem process [1] [2]. One of the fractal models is the iterated function systems fractal or abbreviated as IFS fractal. In the IFS fractal model, the inverse problem system is built by implementing the random iteration algorithm to generate any kind of fractal objects. The result of the inverse problem process is a set of the iterated function systems (IFS) code in a table form that consists of seven columns and a number of rows. The particular row in a set of IFS code represents a component of each character that consists of six coefficients in six columns as the representation of the affine function coefficients. There is another column as the last column that represents the probability factor for each component. The number of rows depends on the number of each character component. Once the IFS code has been generated, the modification can be done for adjustment manually. As a visualization process, a single fractal object can be displayed by implementing a random iteration algorithm into the screen pixel by pixel as the representation of the individual character. To display a collection of characters side by side, the implementation of the partitioned-random iteration algorithm is used instead. The number of partitions is set to depend on the number of characters in the collection [3].

One way to design a character is by modelling the composition of components in a grid of n rows by n columns as a rectangle shape, for example, 5 by 5 grid in which the filled or the void areas are existed in any of the 25 components of grid based on the shape of each character accordingly. The comparison between the number of the filled areas and the void areas determines the dense of each character. To keep the slim character such as “I” has dense enough, but its appearance is still as the normal shape compared to the other characters, the dilation process should be accomplished to the thicker shape design as a special modelling trick. To have the appearance of “O” (as an alphabet) and “0” (as a numeric) are different, the manual shifting particular components of the “0” character into the right position should be conducted as another special modelling trick. The last special modelling trick is also needed to generate the “Q” character by manually adding two additional components into the “0” character model. The manual modification of the transformation process can be done by editing the IFS code that is generated by the inverse problem system.

2. LITERATURE REVIEW

2.1. IFS Code

In the IFS fractal model, there is an identity of the fractal object representing the self-affine transformation as the representation of the collage design of the object as a set of IFS codes. One component of collage design is represented by a row in the IFS code. The first two coefficients of the affine transformation determine the shape of the component object in the horizontal direction. The next two coefficients of the affine transformation determine the shape of the component object in the vertical direction. The last two coefficients of the affine transformation determine the position of the object component in horizontal and vertical directions respectively, relative to the absolute centroid as the origin or the fixed-point of the system [3]. The dense of the object component is determined by a probability factor (**p**) as the last factor of each row in the IFS code. The typical IFS code as an example can be seen in Table 1

2.2.Modelling Methods

Many researchers published their papers dealing with the application of the IFS fractal model. Sarafopuolos et al. proposed the resolution of the inverse problem model for the iterated function system by using the evolutionary algorithm [4]. Li et al. proposed stochastic fractal modelling and process planning for machining using a two-dimensional iterated function system [5]. Xu et al. published the research result on fractal art application for the package decoration design model [6]. Gdawiec et al. worked on a research of the partitioned iterated function system model with division and a fractal dependence graph to recognise two-dimensional shapes [7]. Meng et al. published their research on how to design textile patterns based on the iterated function system model [8]. Xiao et al. studied the application to generate fractal figures based on the iterated function system algorithm model [9]. Darmanto et al. proposed the fractal animation model based on the shifting centroid method on how to simulate the hybrid rotational and revolving of double propellers in an aircraft [10]. Dong et al. accomplished a research study of modelling methods for 3D iterated function systems [11]. Darmanto conducted simulation research on rotating a robot arm model using the non-metamorphic animation method based on the shifting centroid method [12]. Zhang et al. proposed an IFS fractal image generation method based on the Markov random process model [13]. Tao et al. proposed an algorithm of controllable fractal image based on the IFS code model [14].

This paper has two major steps to produce the collection of artistic characters in the fractal model. The first step is the inverse problem process to have the IFS code for modelling a single character as an input for the second step. The second step is the visualization process of several characters side by side independently in a particular position, size and shape for modelling a collection of artistic characters as a final result of the system [3]. In further modelling process as the creative process, once a collection of artistic characters is generated and displayed, an adjustment process of a single character or whole characters as a collection of characters is needed. It can be accomplished independently to fulfil the need to create the artistic characters modelling as shown in Figure 14 (Appendix) in the form of a block diagram as the character modelling lifecycle in the fractal model.

3. DESIGN AND RESULTS

3.1. Model Design

As the result of the inverse problem of the seed model, the IFS code needs to be normalized, as long as the slanted rectangles are just approximations of the perfect rectangles. The name of each component which consists of two digits, refers to the number of rows and column respectively as can be seen in Figure 1 for both the seed and template grids based on the seed model, which looked like the “L” character. To design a grid of the template model in the form of IFS code, the seed model is built first using the inverse problem system consisting of 9 components as shown in Figure 2 in the form of 9 slanted rectangles. After normalization, the IFS code of the seed in the form of a table is displayed in Table 1. The template model has complete 25 components in the 5 rows and 5 columns formation, and the IFS code of the template model can be generated manually based on the IFS code of the seed model which has only 9 components by duplicating the coefficient representing the position of each component horizontally and or vertically for the missing components. The IFS code of the template model can be seen in Table 2. The ‘**p**’ factor in both tables is set to depend on the number of a component accordingly, so the sum of all ‘**p**’ factors equals to 1.0.

The values in the **bold** and *italic* style of Table 2 are duplication of the values in the **bold** and *italic* style of Table 1 based on each component position in the corresponding row and column positions in the grid. The name of cells in the grid is used as the reference of the coefficient duplication in column-5 (**e**) or column-6 (**f**). For example, to fill the coefficients in cell-12 through cell-15 of Table 2, the values in column-5 is duplicated from cell-52 through cell-55 of Table 1. Meanwhile, the values in column-6 is four duplications of cell-11 to cell-12 through cell-15 of Table 2 and repeat the rest accordingly for cell-22 through cell-25, cell-32 through cell-35 and cell-42 through cell-45

explained in the column 'note' of Table 2. As long as the shape of all components in horizontal and vertical directions in both the seed and template are the same, the values of coefficient column-1 through-4 ('a' through 'd') for each column are also the same as values in values cell-11 of Table 1 or Table 2.

SEED

11				
21				
31				
41				
51	52	53	54	55

Template

11	12	13	14	15
21	22	23	24	25
31	32	33	34	35
41	42	43	44	45
51	52	53	54	55

Figure 1 The seed and template models in grid as cells

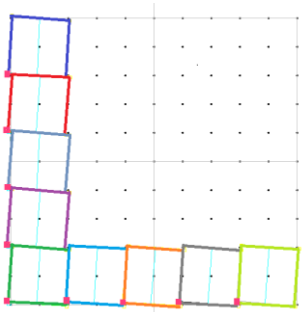

Figure 2 Seed design model in 9 collage components

Table 1 IFS code of the seed model

a	b	c	d	e	f	p	Cell
0.2	0.0	0.0	0.2	-0.4	***-0.8***	0.11	***11***
0.2	0.0	0.0	0.2	-0.4	***-0.6***	0.11	***21***
0.2	0.0	0.0	0.2	-0.4	***-0.4***	0.11	***31***
0.2	0.0	0.0	0.2	-0.4	***-0.2***	0.11	***41***
0.2	0.0	0.0	0.2	-0.4	0.0	0.11	51
0.2	0.0	0.0	0.2	***-0.2***	0.0	0.11	***52***
0.2	0.0	0.0	0.2	***0.0***	0.0	0.11	***53***
0.2	0.0	0.0	0.2	***0.2***	0.0	0.11	***54***
0.2	0.0	0.0	0.2	***0.4***	0.0	0.11	***55***

Table 2 IFS code of the template model

a	b	c	d	e	f	p	cell	Note
0.2	0.0	0.0	0.2	-0.4	-0.8	0.04	11	as in Table I
0.2	0.0	0.0	0.2	***-0.2***	***-0.8***	0.04	12	'**e**' duplication
0.2	0.0	0.0	0.2	***0.0***	***-0.8***	0.04	13	of cell-51 - 55,
0.2	0.0	0.0	0.2	***0.2***	***-0.8***	0.04	14	'**f**' duplication
0.2	0.0	0.0	0.2	***0.4***	***-0.8***	0.04	15	of cell-11
0.2	0.0	0.0	0.2	-0.4	-0.6	0.04	21	as in Table I
0.2	0.0	0.0	0.2	***-0.2***	***-0.6***	0.04	22	'**e**' duplication
0.2	0.0	0.0	0.2	***0.0***	***-0.6***	0.04	23	of cell-51 - 55,
0.2	0.0	0.0	0.2	***0.2***	***-0.6***	0.04	24	'**f**' duplication
0.2	0.0	0.0	0.2	***0.4***	***-0.6***	0.04	25	of cell-21
0.2	0.0	0.0	0.2	-0.4	-0.4	0.04	31	as in Table I
0.2	0.0	0.0	0.2	***-0.2***	***-0.4***	0.04	32	'**e**' duplication
0.2	0.0	0.0	0.2	***0.0***	***-0.4***	0.04	33	of cell-51 - 55,
0.2	0.0	0.0	0.2	***0.2***	***-0.4***	0.04	34	'**f**' duplication
0.2	0.0	0.0	0.2	***0.4***	***-0.4***	0.04	35	of cell-31
0.2	0.0	0.0	0.2	-0.4	-0.2	0.04	41	as in Table I
0.2	0.0	0.0	0.2	***-0.2***	***-0.2***	0.04	42	'**e**' duplication
0.2	0.0	0.0	0.2	***0.0***	***-0.2***	0.04	43	of cell-51 - 55,
0.2	0.0	0.0	0.2	***0.2***	***-0.2***	0.04	44	'**f**' duplication
0.2	0.0	0.0	0.2	***0.4***	***-0.2***	0.04	45	of cell-41
0.2	0.0	0.0	0.2	***-0.4***	0.0	0.04	51	
0.2	0.0	0.0	0.2	***-0.2***	0.0	0.04	52	
0.2	0.0	0.0	0.2	***0.0***	0.0	0.04	53	as in Table I
0.2	0.0	0.0	0.2	***0.2***	0.0	0.04	54	
0.2	0.0	0.0	0.2	0.4	0.0	0.04	55	

From the 5 to 5 grid template, the individual model design of each alphabet and numeric characters can be modelled, as can be seen in Figure 3 through Figure 7 by deleting the components representing the filled area to be the void area based on the shape of each character. There are three characters with "??" in particular cells. The character-"0" can be built manually as modification of character-"O" and character-"Q" can be built manually as modification of character-"0" from each IFS code by editing. The comparison between the IFS code of "O" and "0" characters are displayed in Table 3 and Table 4 and the comparison between the IFS code of "0" and Q" are displayed in Table 4 and Table 5 respectively. To make the comparison between Table 3 and Table 4 and between Table 4 and Table 5 focus at the right one, the corresponding values are displayed in **bold** and italic *styles*. By shifting (subtracting or adding 0.05 to the original value) to the right or left, the appearance of the character-"0" is smoother

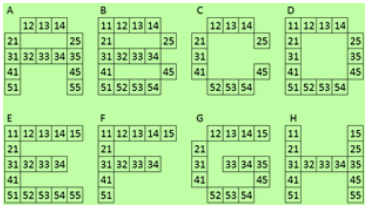

Figure 3 Alphabet characters model part-1

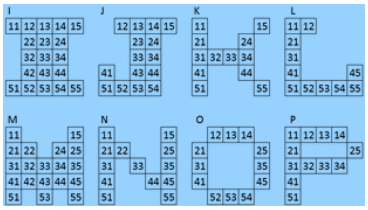

Figure 4 Alphabet characters model part-2

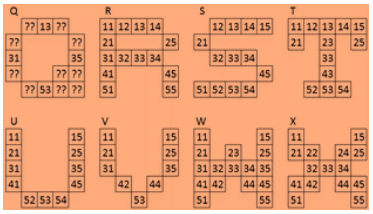

Figure 5 Alphabet characters model part-3

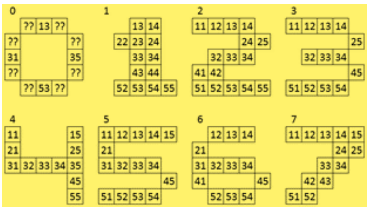

Figure 6 Numeric characters model

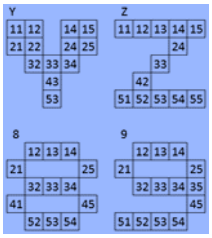

Figure 7 Rest alphanumeric characters model

Table 3 IFS code of character-"O"

a	b	c	d	e	f	p	cell
0.2	0.0	0.0	0.2	-0.2	-0.8	0.083	12
0.2	0.0	0.0	0.2	0.0	-0.8	0.083	13
0.2	0.0	0.0	0.2	0.2	-0.8	0.083	14
0.2	0.0	0.0	0.2	-0.4	-0.6	0.083	21
0.2	0.0	0.0	0.2	0.4	-0.6	0.083	25
0.2	0.0	0.0	0.2	-0.4	-0.4	0.083	31
0.2	0.0	0.0	0.2	0.4	-0.4	0.083	35
0.2	0.0	0.0	0.2	-0.4	-0.2	0.083	41
0.2	0.0	0.0	0.2	0.4	-0.2	0.083	45
0.2	0.0	0.0	0.2	-0.2	0.0	0.083	52
0.2	0.0	0.0	0.2	0.0	0.0	0.083	53
0.2	0.0	0.0	0.2	0.2	0.0	0.083	54

Table 4. IFS code of character-"0"

a	b	c	d	e	f	p	cell	note compared to "O"
0.2	0.0	0.0	0.2	-0.2	-0.75	0.083	12'	shifted 0.05 vertically down
0.2	0.0	0.0	0.2	0.0	-0.8	0.083	13	as "O"
0.2	0.0	0.0	0.2	0.2	-0.75	0.083	14'	shifted 0.05 vertically down
0.2	0.0	0.0	0.2	-0.35	-0.6	0.083	21'	shifted 0.05 horizontally right
0.2	0.0	0.0	0.2	0.35	-0.6	0.083	25'	shifted 0.05 horizontally left
0.2	0.0	0.0	0.2	-0.4	-0.4	0.083	31	as "O"
0.2	0.0	0.0	0.2	0.4	-0.4	0.083	35	as "O"
0.2	0.0	0.0	0.2	-0.35	-0.2	0.083	41'	shifted 0.05 horizontally right
0.2	0.0	0.0	0.2	0.35	-0.2	0.083	45'	shifted 0.05 horizontally left
0.2	0.0	0.0	0.2	-0.2	-0.05	0.083	52'	shifted 0.05 vertically up
0.2	0.0	0.0	0.2	0.0	0.0	0.083	53	as "O"
0.2	0.0	0.0	0.2	0.2	-0.05	0.083	54'	shifted 0.05 vertically up

Table 5. IFS code of character-"Q"

a	b	c	d	e	f	p	cell	note
0.2	0.0	0.0	0.2	-0.2	-0.75	0.071	12'	
0.2	0.0	0.0	0.2	0.0	-0.8	0.071	13	
0.2	0.0	0.0	0.2	0.2	-0.75	0.071	14'	
0.2	0.0	0.0	0.2	-0.35	-0.6	0.071	21'	as "0"
0.2	0.0	0.0	0.2	0.35	-0.6	0.071	25'	
0.2	0.0	0.0	0.2	-0.4	-0.4	0.071	31	
0.2	0.0	0.0	0.2	0.4	-0.4	0.071	35	
0.2	0.0	0.0	0.2	-0.35	-0.2	0.071	41'	
0.2	0.0	0.0	0.2	0.155	-0.25	0.071	44'	new insert
0.2	0.0	0.0	0.2	0.35	-0.2	0.071	45'	
0.2	0.0	0.0	0.2	-0.2	-0.05	0.071	52'	as "0"
0.2	0.0	0.0	0.2	0.0	0.0	0.071	53	
0.2	0.0	0.0	0.2	0.2	-0.05	0.071	54'	
0.2	0.0	0.0	0.2	0.4	0.0	0.071	55	new insert

3.2. The object of fractal as a character

As already mentioned in the previous section, the object of the fractal can be visualized to the screen by implementing the random iteration algorithm. The object of fractals visualization of all alphabetical characters can be seen in Figure 8 through Figure 10. As the result of the modification of character-"O" to be character-"0", the latter looks smoother in a curve fashion than the first one. The next modification from character-"0" to be character-"Q", there are just two additional components as the "tail" representation of character-"Q". There is still another minor modification in character-"M" to shift slightly two components in row-4, column-2 and 4 to the right and left, respectively, in order to make a little gap between component-41 and 42 and between component-44 and 45.

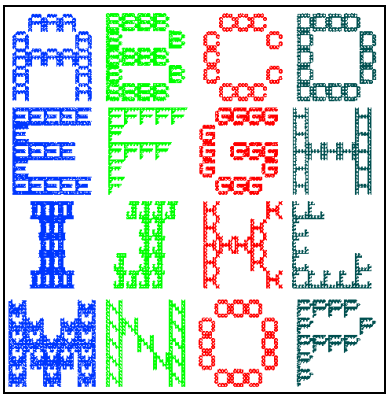

Figure 8 Multi-objects of fractal as the alphabet characters part-1

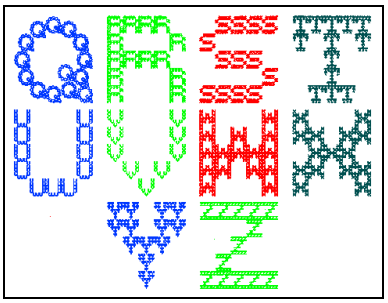

Figure 9 Multi-objects of fractal as the alphabet characters part-2

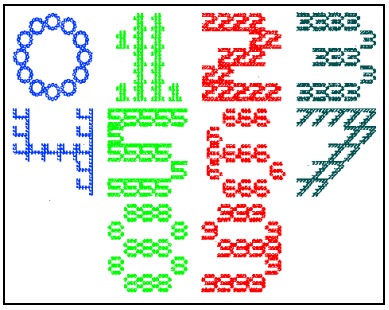

Figure 10 Multi-objects of fractal as the numeric characters

3.3. *Multi-object of fractal as a collection of character*

As already mentioned in the previous section, the multi-object of fractal can be visualized in the screen by implementing the partitioned-random iteration algorithm. The multi-object of fractals visualization of a collection of characters as a word “FRACTALS” as a word example can be seen in Figure 11, 12 and 13.

Figure 11, as the first example of the artistic characters, there is a modification on the collection of characters as two words. It shows the utilization of the shear transformation of affine to have the italic style appearance of the two crossed identical words which can be displayed both in horizontal and vertical fashions.

Figure 12, as the second example of the artistic characters, there are two kinds of modification. It shows the utilization of the combination of the shear and rotation transformations of affine to have the italic style appearance with a curved variation of the two crossed identical words which can be displayed in both horizontal and vertical fashions.

In Figure 13, as the third example of the artistic characters, there are two kinds of modification. It also shows the combination of the shear and rotation transformations of affine, but to have the italic style appearance in two modes, so the word is looked like in zigzags fashion vertically, and another word is looked like a bow stretched horizontally crossing the first word.

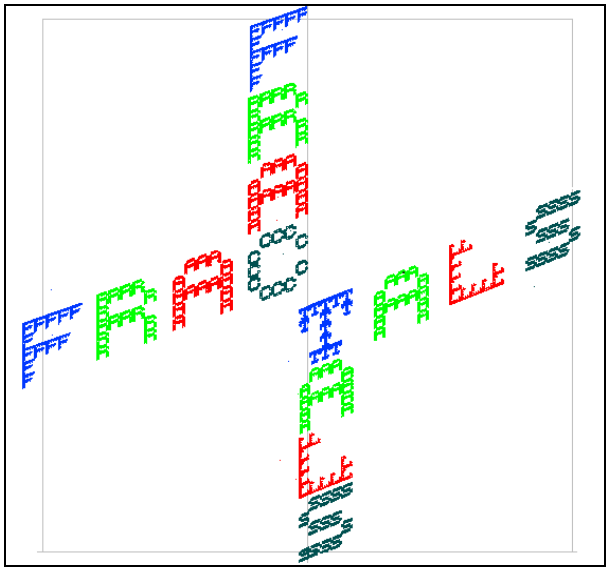

Figure 11 Fractal of artistic characters example-1

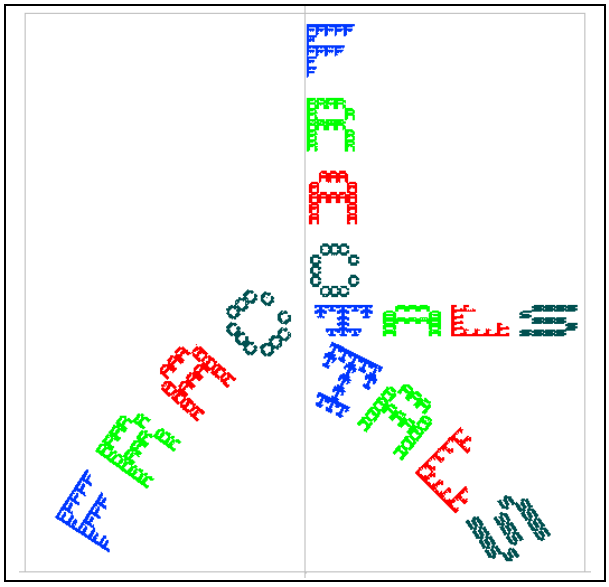

Figure 12 Fractal of artistic characters example-2

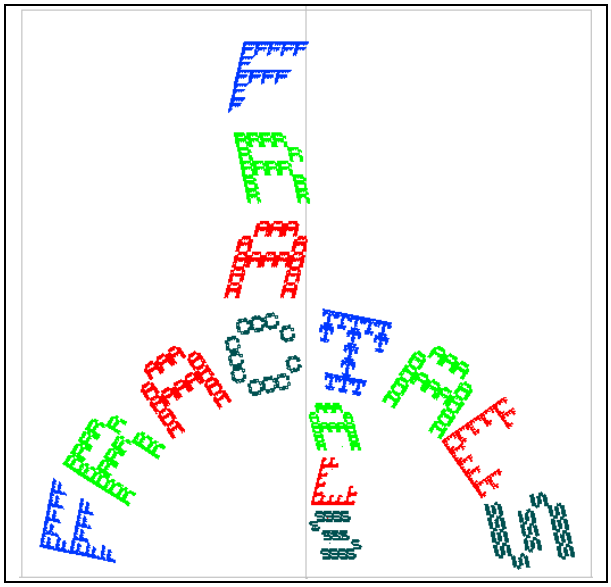

Figure 13 Fractal of artistic characters example-3

4. CONCLUSION

In the IFS fractal model, an identity of the fractal object represented by the IFS code can be modified easily by applying the combination of transformation operation based on two or more affine transformation functions. An artistic character design can be carried out by arranging the collage members in grid composition. The right combination of the functions determines the shape of the individual component object in a group of components, such as the collection of artistic characters in the fractal form in any style and any direction.

REFERENCES

[1] Rinaldo, R., & Zakhor, A., Inverse and approximation problem for two dimensional fractal sets, IEEE Transaction on Image Processing, 3(6), 802-820, 1994, doi: 10.1109/83.336249

[2] Wadstromer, N., An automatization of Barnsley's algorithm for inverse problem of iterated function systems, IEEE Transactions on Image Processing, 12(11), 1388-1397, 2003, doi: 10.1109/tip.2003.818040

[3] Darmanto, T., Animating Fractal of Things based on IFS Fractal Model, Lambert Academic Publishing, copyright © 2016, OmniScriptum GmbH & Co. KG, ISBN 978-3-659-86851-1

[4] Sarafopuolos, A., & Buxton, B., Resolution of the Inverse Problem for Iterated Function Systems using Evolutionary Algorithms, IEEE International Conference on Evolutionary Computation, 2006, doi: 10.1109/cec.2006.1688428

[5] Li., Y.H., Feng, J.C., Li, Y., & Wang, Y.H., Stochastic Fractal Modeling and Process Planning for Machining Using Two-Dimensional Iterated Function System, Key Engineering Materials, 2008, 392-394, 575-579, doi: 10.4028/www.scientific.net/kem.392-394.575

[6] Xu, S., Yang, J., Wang, Y., Gao, J., Application of fractal art for the package decoration design, IEEE 10th International Conference on Computer-Aided Industrial Design & Conceptual Design, 2009, doi: 10.1109/caidc.2009.5375057

[7] Gdawiec, K., & Domanska, D., Partitioned Iterated Function Systems with division and a fractal dependence graph in recognition of 2D shapes, International Journal of Applied Mathematics and Computer Science, 2011, 21(4), 757-767, doi: 10.2478/v10006-011-0060-8

[8] Meng, G.L., Yang, X.H., & Li, D.X., Textile Pattern Design Based on IFS, Applied Mechanics and Materials, 251, 239-243, 2012, doi: 10.4028/www.scientific.net/amm.251.239

[9] Xiao, H.R., Application Study of Fractal Figures Based on Iterated Function System Algorithm, Advanced Materials Research, 694-697, 2886-2890, 2013, doi: 10.4028/www.scientific.net/amr.694-697.2886

[10] Darmanto, T., Hybrid Rotational and Revolving Simulation of Propeller in Aircraft with Two Propellers by Fractal Animation Model Based on Shifting Centroid Method in Double Mode Form From a Fixed Point, International Conference on Computer, Control, Informatics and Its Applications, 2014, doi: 10.13140/2.1.2646.5284

[11] Dong, S.S., Wang, Q., & Rong, X., The studies of Modelling Method for Three Dimension Iterated Function System, Applied Mechanics and Materials, 2014, 513-517, 2569-2572, doi: 10.4028/www.scientific.net/amm.513-517.2569

[12] Darmanto, T., Simulation of Rotating a Robot Arm by Non-Metamorphic Animation Method in IFS Fractal Model Based on Shifting Centroid Technique, International Conference on Information and Communication Technology, 2018, doi: 10.1109/icoict.2018.8528742

[13] Zhang, l., Chang, T., Mao, Y., A kind of IFS fractal image generation method based on Markov random process, IEEE International Conference on Computation, Communiaction and Engineering, 2019, doi: 10.1109/ICCCE48422.2019.9010799

[14] Tao, X., Bai, S., Liu, C., Zhu, C., Chen, H., Yan, L., Algorithm of Controllable Fractal Image Based on IFS Code, IEEE International Conference on Power Electronics, Computer Applications, 2021, doi: 10.1109/ICPECA51329.2021.9362569

APPENDIX

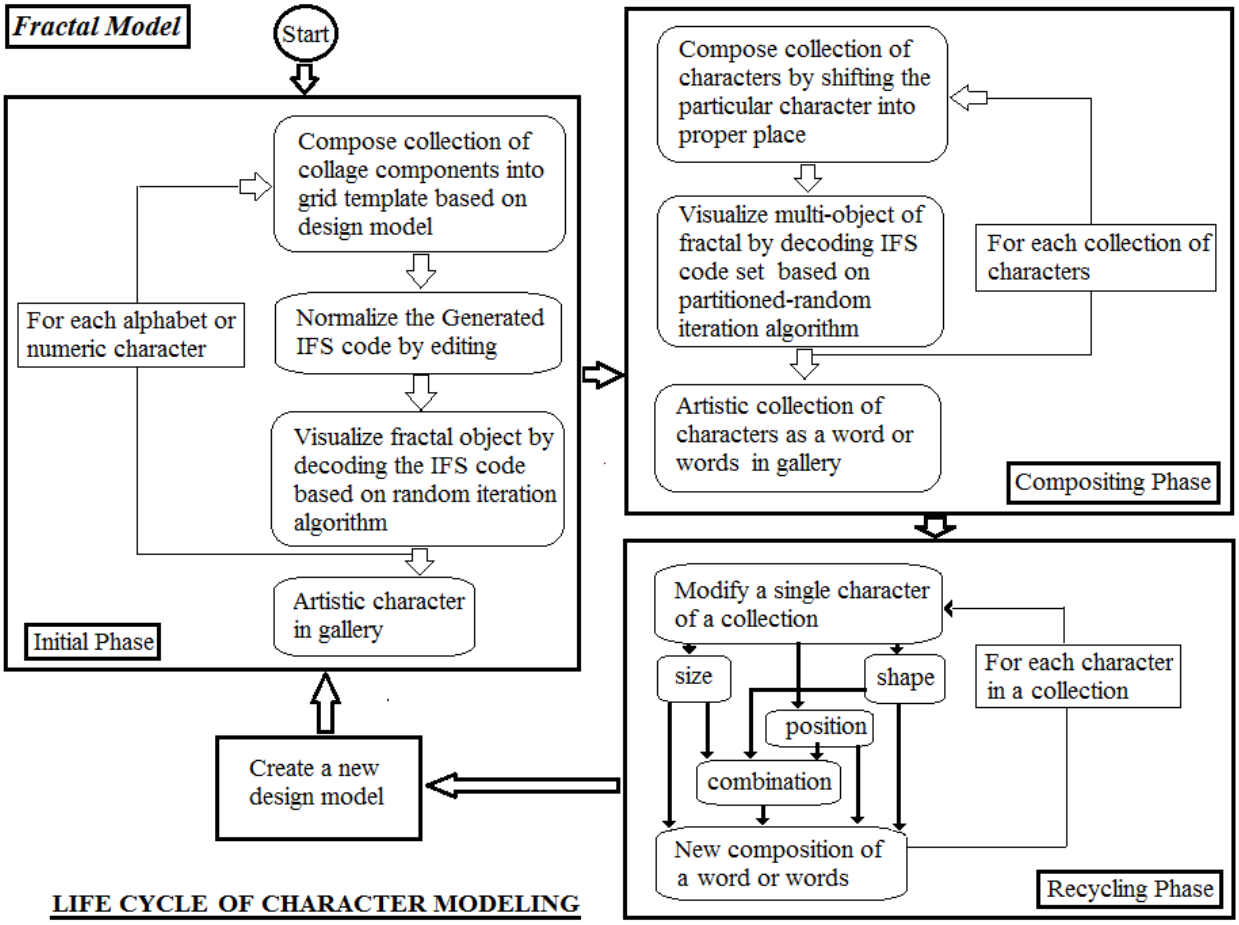

Figure 14 Lifecycle of Character Modeling in Fractal Model