Morphogenesis Analysis for Digital Image Production with L-System

Arie Vatresia[1], Ferzha Putra Utama[2]

[1] *Informatics, Engineering Faculty, Universitas Bengkulu, Indonesia*
[2] *Information System, Engineering Faculty, Universitas Bengkulu, Indonesia*
[1]arie.vatresia@unib.ac.id, [2]fputama@unib.ac.id

Abstract - The process of forming an image requires a correct color composition, location and distance between the lines to produce a good image. Human abilities in both creativity and high imagination are very limited, especially in forming new images by utilizing existing image patterns or images that resemble old images. Here we showed the implementation of L-System to generate new image generations with additional flame as a fire effect/glow on images for image transformation. This research used the L-System algorithm, Iterated Function System, and Voronoi Diagram to improve the result of image transformation. The results of this study indicated that mathematical calculations can be applied in the formation of images and the resulting images can be abstract and symmetrical. The next generation of images produced in this research can be in unlimited numbers as the generation of morphogenesis processes. The process of generating images is carried out randomly by merging the two existing images with morphogenesis analogy. The resulting images can be exported into jpg, png, and svg formats. Furthermore, this research showed that the implementation of the calculation for the variation reach the value of 99.48% while the image variation composition has a value of 99.29%.

Keywords: ***L-System, digital image, morphogenesis, Java Script, Voronoi Diagram***

I. INTRODUCTION

Image is a combination of points, lines, areas, and colors composed to visualize something. Image is a visual language that has been used by humans for a long time, therefore it is a work of art [1–3]. Art is a person's way of expressing everything that is in him. There are many types of art and drawing belongs to the type of printmaking. Printmaking requires a person to have unlimited imagination, creative, and innovative so that the resulting image can be enjoyed by others. Along with the times, art has used technology as a tool so that work becomes faster and gives optimal results. Technology and art are two things that are interconnected [4-5]. Technology requires art to be more accepted by society. On the other hand, art requires technology for optimal results. Digital art is the use of digital technology in the arts. The very rapid development of computer technology has encouraged the development of digital art, one of which is the formation of fractals. Fractal is an object which has similarity with itself (self similarity) at different scales [1], [6-7]. Fractals are said to be a computer graphics application which is a technique of generating images or images by iterating a certain function. Fractals have an irregular shape and through this iteration, a picture that has a similarity to oneself, repeating shapes, and scaling will be obtained. One example of the application of fractals in the fields of technology and mathematics is fractal batik, which is one of the digital works of art that combines art, mathematical patterns, and technology. Beautiful and geometric batik motifs can be produced with mathematical formula patterns.

In the process of digital morphology [8-9], genetic algorithms are heuristic search and optimization algorithms based on the mechanisms of natural and genetic selection. One of the goals of the concept of genetics in general is to form new generations. This algorithm is based on the genetic processes that exist in living things, namely the development of generations in a natural population which gradually follows the principle of natural selection or who is the strongest one who survives [10]. The new generation that was formed eventually gave rise to an ecosystem known in digital graphic arts as a virtual ecosystem. The virtual ecosystem is one of the depictions that show the morphology and movement between populations of groups of individuals in a virtual environment [11-12]. A physical model allows various kinds of individuals to be articulated to explore their potential through simulations in the environment. Not only seeking potential, but also enabling mating between groups of individuals, thus providing more genetic design building blocks for new generations. This simulation can be applied to image changes using sexual reproduction and population methods. Therefore simulation in the form of software needs to be done. In this study, calculations were carried out using Voronoi diagram [13-14]s and l-systems [12], [15] as patterns. Linear Cellular Automata (LCA), which is the first calculation adopted like a social environment (living next door), neighbouring cells are calculated on

the basis of XOR or XNOR. The state at each subsequent time period can be expressed as a linear function of the initial states. The second calculation is based on geometric and recursive shapes using the iterated function system method as a fractal pattern. Furthermore, the Voronoi diagram is used for the case of how an object (the partition space of a cell) is translated into a collection of geometries, each of which consists of points adjacent to one another. The final calculation, namely Lindenmayer Systems (L-Systems), acts as a model for morphology and various organisms such as alphabet symbols that can be used to make strings. IFS and L-Systems calculations are often used by agricultural researchers in studying plant growth patterns through simulation and visualization. Furthermore, using IFS [16-17], Voronoi diagrams and L-Systems can also be used in medicine, art, strategy games and even to draw a map (GIS).

II. METHOD

A. Algorithms

The system implemented the L-System, Iterated Function System (IFS), and Voronoi diagram to support the evolution process of the images. The L-System method is a formal rule structured as grammar in the form of axioma, where the symbols used represent plant growth, there is a parallel and simultaneous change of symbols at each stage[18-19]. Here the important difference between the Chomsky and the L-System grammar lies in production. In Chomsky grammar, production is used as a sequence (sequentiallly) while in production system LS grammar is used as parallel and simultaneous to replace components [20]. This is a result of the reflection of biological motivation, where production is growth, cell differentiation and morphogenesis. The main concept of L-Systems is repetitive writing. Iterative writing is a technique for defining objects in a complex manner by replacing parts of the object by means of rewriting rules or productions. An example of a graphic object defined by rewriting rules is the snowflake curve, in 1905 by Von Koch [21].

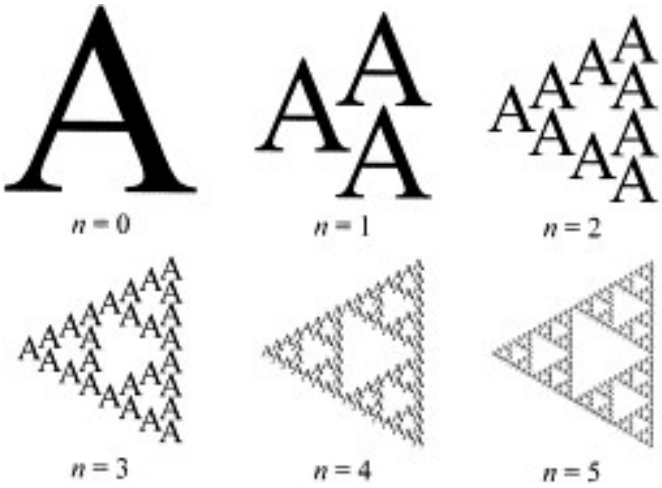


Fig. 1 Fractal multiple reduction [21]

Iterated function system is a fractal formation method which is visualized into its present form so that it resembles itself and was popularized by Michael Barnsley in the book Fractals Everywhere. Fractals formed from IFS can exist in any spatial dimension (in any dimension). But usually IFS fractals are usually computed and drawn on two dimensions. An IFS fractal is the result of a recursive set of equations. IFS fractals consist of unions or combinations of copies of themselves, like can be seen in Figure 1. So that IFS is metaphorized as a photocopier called the Multiple Reduction Copy Machine. MRCM has many lenses and each lens performs a large amount of image reduction. The image generated from the copier is re-operated as input for making the next copy.

Voronoi diagram is the division of an area into *n* regions where each point in the area is closer to the point forming the area than the point forming the area. Voronoi diagrams are one of the branches of science studied in the field of computational geometry that emerged in the 17th century. Voronoi diagrams have been used in many fields to determine the division of regions. Furthermore the system would add flame into the design to improve the evolution process of the image. Flame is a library or external engine that functions to translate the basic computed texture layer parameters, this library will act in the background and produce simulated flames that vary like a fire effect mixed with brush effects. This library was written in 1992 and released as open source.

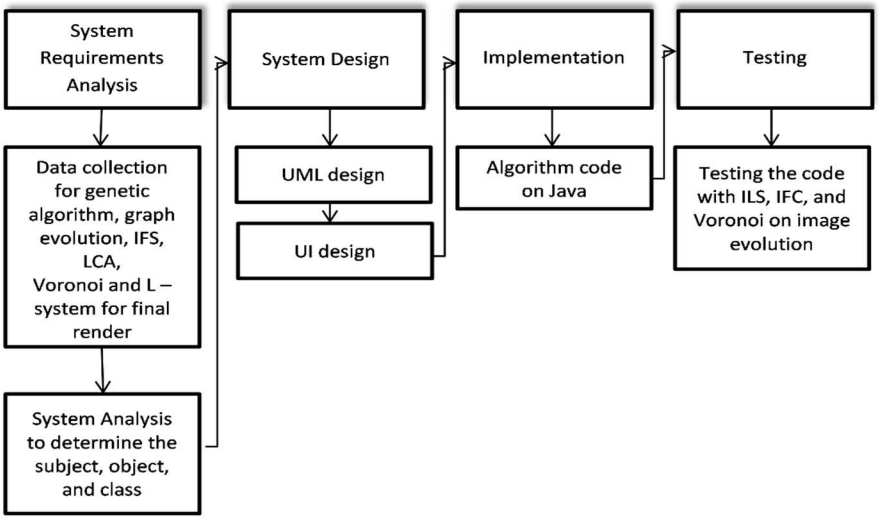


Fig. 2 System development process

B. System Development

The system would be developed using the method of Object oriented programming using Java language [22-23]. The UML model would help to visualize the implementation of the algorithm into image evolution process. It can be seen in Figure 2. The system developed in this study uses the SDLC (system development life cycle) development model [24-25]. One model is the Waterfall model, or it is often called a linear sequential model which is systematic, sequential in building software. The following is an explanation of the stages of the linear sequential model in this study:

1) Systems Engineering and Modelling. The first thing to do in this research is to identify the existing problems. After the identification process is complete, the researcher analyses the system to be made as a solution to existing problems.

2) Software Requirements Analysis. At this stage, analysis and definition of needs will be carried out software to be created.

3) System Design. This system design stage is carried out after the analysis stage and the definition of requirements have been completed. At this stage the results of the analysis that have been carried out in the previous stage are translated into the form of interface design, data structure design and algorithmic procedure design.

4) Code Generation. After the system design stage is complete, the design results will be converted into a form that is understandable by the machine, namely into a programming language that has been determined through the program writing process. In this study, Java programming language was used.

5) System Testing. At this stage, testing will be carried out to see if the system is in accordance with the design.

6) Maintenance. The maintenance stage is the final stage of system development and implementation. In order for the system to be used in the long term, it is necessary to maintain the system. System maintenance is not only a process of fixing program errors but a process that has the characteristics of fixing errors that were not found in the previous stages or adding new functions that do not exist in the program.

III. RESULTS AND DISCUSSION

The work procedure includes the arrangement of the initial chromosomes, the use of genetic algorithms and uses basic calculations which include LCA, iterated function systems, voronoi diagrams and L-System in producing new images. In addition, it also involves color as a type of chromosome used in image formation. The color chromosomes used include binary image, grayscale image, RGB image, HSB, lookuptable color, and gradient. A binary image is an image that has only two possible pixel values, namely black or white. A grayscale image is an image that only has one known value for each pixel. The colors that are owned are the colors from black, gray and white. RGB image consists of three main colors, namely red, green, and blue. Each color has 8 bits in one pixel. So the color palette contains $(2^8)^3 = (256)^3 =$ 16,777,216 colors. HSB (hue, saturation, brightness), hue is the actual color, then for saturation is the purity of the color measured in percent from the cone center (0) to

the surface (100), and brightness is the brightness measured in percent from black (0) to white (100). Lookuptable color is a mechanism used to convert various input colors into a range of other colors. The L_{cut} (L-system cutting point) has the following characteristics:

- Number of entries in palette: determines the maximum number of colours that can appear on the screen simultaneously (a subset of the wider full palette, which must be understood as the number of colours that a given system is capable of producing or managing, for example a full RGB colour palette).
- Width of each entry in the palette determines the area of the number of colours on the palette.

C. IFS Transformation Affine

An affine IFS consists of a list of transformation matrices. Each matrices has six value points. The main calculation refers to the parallel formula (1) or (2).

$$(x,y) \rightarrow ((a*x) + (b*y) + e, (c*x) + (d*y) + f) \quad (1)$$

$$X_{new} = ax + by + c,\ Y_{new} = dx + ey + f \quad (2)$$

where $a,b,\ldots, f$ are the matrix point, x and y is the transformation point, and X and Y are the result points. This can be presented with the following code in Fig.4.

D. IFS Symmetric Transformation Affine

In order to get a symmetrical and more interesting image, the variables *a, b, c,* and *d* are replaced with a = cost, b = -int, c = sint and d = cost. This type of change rotates the plane through radians t counterclockwise (around origin), scale, shift or mirroring, which can be seen in Fig.5. This complex calculation requires a few additional variables defined in the soup.xsd schema. These variables consist of alpha, beta, gamma, lambda, and omega. Then these variables are put in the calculation of the affine or affine symmetry transformations. Complex IFS calculations are modifying the calculation of symmetry provided that it does not violate the IFS rules and can create varied calculations with better results as seen in Fig.6.

```
<xsd:complexType name="affineIfsChromosome">
  <xsd:complexContent>
   <xsd:extension base="chromosomeType">
    <xsd:sequence>
     <xsd:elementname="seed" yype="seedGene"/>
     <xsd:element name="affineTransformation"
type="affineTransformationGene" minOccurs="2"
maxOccurs="8"/></xsd:sequence></xsd:extension>
</xsd:complexContent></xsd:complexType>
```

Fig. 3 IFS code

```
protected void iterate() {
int tx;
if (numTransformations > 1) {
tx = matcher[rand.nextInt(maxMatcher)];
} else {
tx = 0;
}
//finding x,y (affline transformation)
double x2 = xw * mTa[tx] + yw * mTb[tx] + mTe[tx];
double y2 = xw * mTc[tx] + yw * mTd[tx] + mTf[tx];
xw = x2;
yw = y2;}
```

Fig. 4 Transformation code

```
//iteration method
protected void iterate() {
 super.iterate();
if (symmetry > 0) {
double angel = 2.0 * Math.PI * (double)
(rand.nextInt(symmetry)) / (double) symmetry;
double cosinus = Math.cos(angel);
double sinus = Math.sin(angel);
double x2 = cosinus * xw - sinus * yw;
double y2 = sinus * xw + cosinus * yw;
xw = x2;
if (Dn && rand.nextBoolean()) { yw = -y2;}
else {yw = y2;} } }
//affline symmetric
double zPowerN = xw * zReal - yw * zImag;
double p = lambda + alpha * zbar + beta * zPowerN;
x2 = p * xw + gamma * zReal - omega * yw;
y2 = p * yw - gamma * zImag + omega * xw;
xw = x2;
yw = y2;
```

Fig. 5 IFS symmetric code

C. Voronoi Calculation

Tingkat *headings* yang digunakan maksimum adalah 3 tingkat. Semua *headings* ditulis dalam huruf ukuran 11 pt. Setiap inisial kata dalam *heading* harus ditulis menggunakan huruf kapital kecuali kata-kata tertentu sebagaimana dalam bagian III-B.

```
//voronoi calculation
if (enterDepth()) {
while (more){
        double nearestDistance = Double.MAX_VALUE;
        int nearestKing = 0;
        int kx = numberOfKings;
        while (kx > 0) {
                --kx;
                double dist = kw[kx] * distance(kx);
                if (yard > 0.0) {
                        if (dist < yard)
                        dist = 0.0;
                        else
                        dist = dist - yard; }
                if (dist < nearestDistance) {
                        nearestDistance = dist;
                        nearestKing = kx; }}
                setPixel(intPalette[nearestKing]);
    nextLocation(); }
```

Fig. 7 IFS symmetric code

In a Voronoi diagram each point in 2D space is coloured depending on the distance to several control points. For the final image must count every pixel. In the Voronoi diagram, it is known as the closeness between points, for example on a canvas having two points P, it can be said that the two points apply good neighbourly characteristics if they share. A proximity graph is obtained by connecting each point to its neighbouring Voronoi. This graph is referred to as the Delaunay chart. At each point q in a plane, $C_m(q)$ becomes the largest circle centred on q that does not contain a point p on the inside. For each point $Pi \varepsilon P$, the closest point to P is the neighbour of the Voronoi. Therefore, the closest neighbour to the digraph is the sub graph of the Delaunay graph.

Starting with the enterDepth instruction (checking the depth) for each *Cm (q)*, the system will calculate which point is the closest before forming the graph. The process is repeated so that it forms a graph according to Voronoi's rules. If all sites points, free points, edges and vertices have been formed, the pixel palette processing is continued. Euclidean Minimum Spanning Tree (EMST) is a subgraph of the Delaunay chart. EMST is the minimum spanning tree of a series of n points in a plane, where the edge weight between each pair of points is the distance between two points.

D. L-System Calculation

In the calculation of the L-System is divided into three general parts, namely free context deterministic, free context deterministic, and context sensitive non deterministic (Fig. 8). Deterministic means a production rule, free context means that the production rule also refers to a symbol regardless of its neighbours, while context is sensitive on the contrary. Rule in L-System used (3).

$$G = (V, \omega, P) \quad (3)$$

where,
V = symbol set (contains replaceable elements)
Ω = start, axiom or initiator
P = the production rule

E. System Implementation

System implementation can be seen in Fig. 9. From Fig.9 we can see several implementations in the system to provide image evolution. The narration of each function in the system would be shown in Table I.

```
public void calculate(boolean paintOnScreen, StringdoExport) {
deferred = true;
LsysMashine.execute(chromosome, colorator, this);
deferred = false;
ready = true;}
```

Fig. 8 L-System calculation

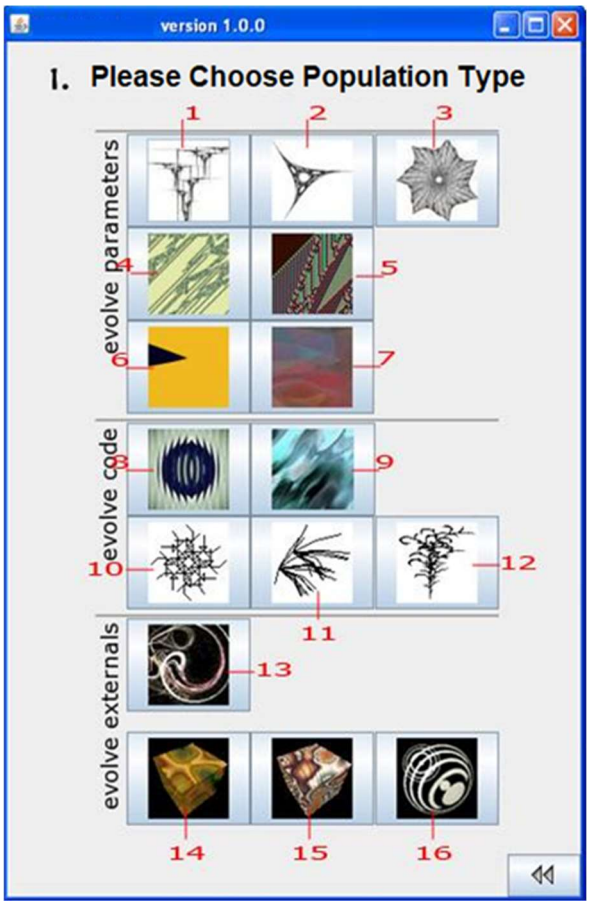


Fig. 9 System implementation overview

TABLE I
FUNCTION DESCRIPTION

Nu	Button Name	Function
1	Affine Iterated Function System (IFS) transformation button	The system will calculate the affine IFS transformation and use it as the basis for forming the image.
2	Symmetry IFS affine transformation keys	The system will perform the calculation of the symmetric and IFS affine transformation is used as the basis for the formation of the image.
3	IFS key symmetry complex calculations	The system will perform Symmetry and IFS calculations are used as the basis for image formation.
4	LCA totalistic rule button	The system will perform the totalistic rule LCA calculation and use it as the basis for forming the image.
5	LCA direct rule button	The system performs the LCA direct rule calculation and is used as the basis for generating the image.
6	Voronoi diagram button	The system will perform the Voronoi diagram calculation and used as the basis for forming the image.
7	Voronoi diagram button (transparent color)	The system will calculate the Voronoi diagram with a transparent color and is used as the basis for forming the image.
8	Scalar expression button	The system will perform the calculation of the scalar expression and is used as the basis for generating the image.
9	Expression button vector	The system will calculate the vector expression and use it as the basis for creating the image.
10	Deterministic keys and free context L-System	The system will perform deterministic and context-free calculations on the L-System rules and are used as the basis for forming the image.
11	Non-deterministic and context-free L-System button	The system will perform non-deterministic and context-free calculations on the L-System rules and is used as the basis for forming images.
12	Nondeterministic and context L-System keys	The system will perform non-deterministic and context-free calculations on the L-System rules and are used as the basis for image formation.
13	Fractal flame button	system will perform fractal calculation with the help of a flame (third party engine) and used as a basis for forming images.
14	Numbers 14,15,16 represent the pov-ray keys	the system will do pov-ray calculation (third party engine) with their respective rules and used as the basis for forming the image.

The population determination interface of this application is the second interface that will appear after the user selects the population type on the initial interface. In this interface there are 2 buttons, namely the new population button and the button that opens an existing population. When the user selects a new population button, a colour selection interface will appear. Conversely, when the user selects the button to open an existing population, the load existing population interface will appear. There is also a return button to the population / calculation type interface. The visualized interface is an interface for displaying images. This interface will appear after the user performs the process of selecting the type of calculation as a population, selecting a colour model, and selecting the number of images to be displayed. Fig.10 an example of the population results based on the IFS calculation type Affine Transform symmetry, LUT colour model, and 4x3 population size. From Fig.9, the number describes some of the function as below:

- Enlarge image button: this button functions to display the image enlargement interface.
- Chromosome edit button: this button functions to display the chromosome image editing interface.
- Image export button: this button functions to display the image export interface in the form of jpg, png, and svg.
- Image marker button as the first parent: this button functions to indicate the selected image as the first parent that will be used at the time of forming a

new generation when the mating process is carried out.

- Image marker button as another parent: this button functions to indicate the selected image as the first parent that will be used at the time of the formation of the new generation when the mating process is carried out.
- Image marker button for deletion: this button functions to delete old pictures and get new pictures.
- Marker button as best image: this button functions to indicate that the image is the best image and the image will not generate a new image generation when random generation is generated.
- Image generation by marriage button: this button functions to get a new image generation by merging the first parent and another parent. However, the two main images will not change.
- Clone image generation button: this button will be active when the user marks one of the images as the best image and functions to get an image similar to that image selection.
- Random image generation button: get random new image generation.
- Back button: this button serves to return to the previous population.
- Environment edit button: used to perform environmental value editing.
- Population duplication button: functions to duplicate the entire population including creating a new visualization interface.
- Save population button: this button functions to save population.
- Show log button: functions to display logs.
- Gen pool button: this button will display the gene structure of the image that has been previously saved.

Some result from the implementation of the methods is shown in Fig.11.

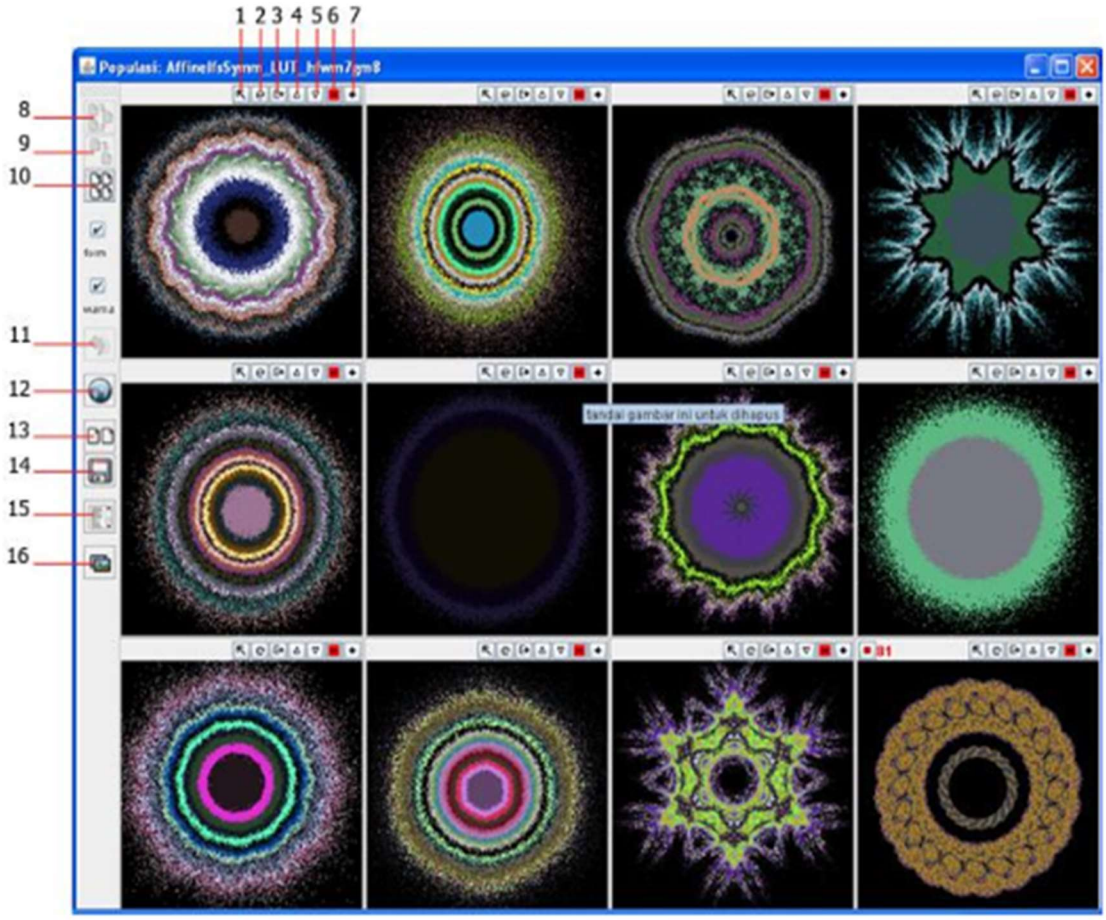


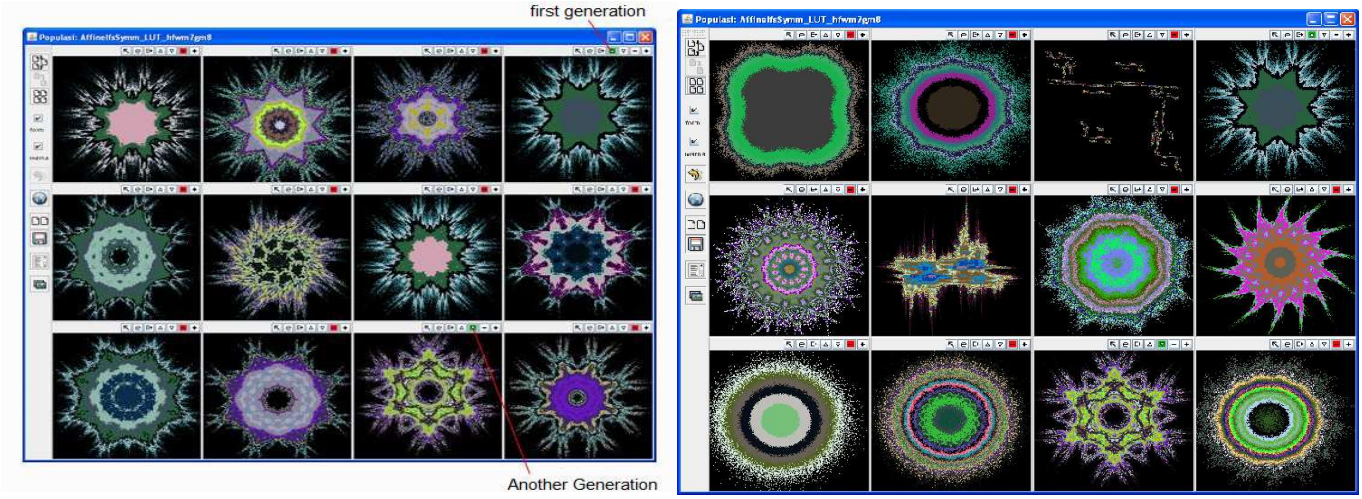
Fig. 10 Image transformation result

Fig.11 (a) showed an example of the interface of the visualization results after the morphogenesis process the (1, 4) and (3, 3) that were marked as green light on the system. The visualization result interface were visualised as the red light in the interface. Meanwhile, Figure 6(b) showed the random transformation of the image (1, 4) and (3, 3).

F. System Testing

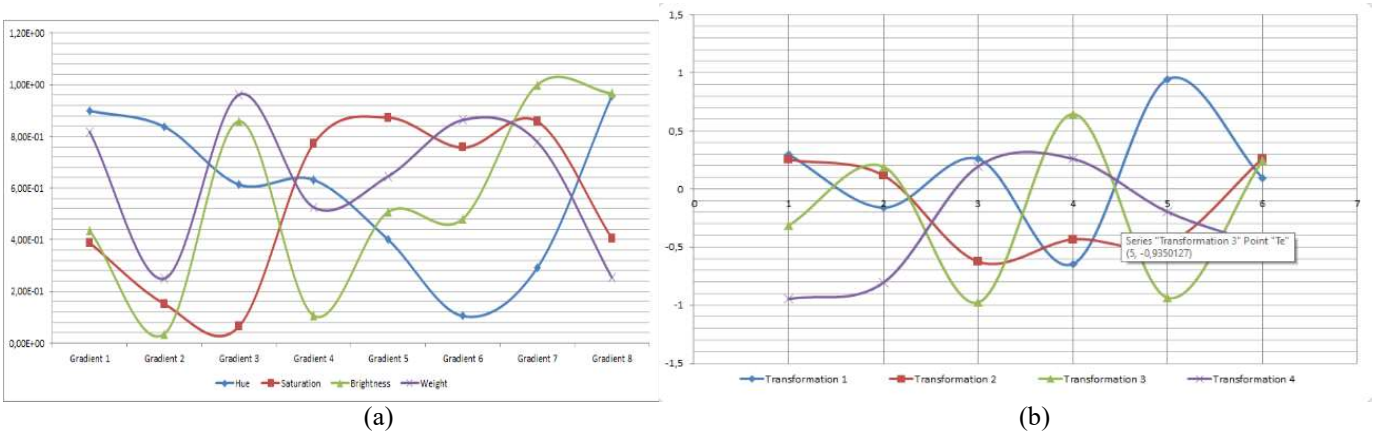
This software test was carried out to determine the image formed from generating randomly obtained value variables through genetic algorithms and these values are used in the formation of image patterns using calculation methods consisting of LCA (loss-contracting algorithm), iterated function systems, Voronoi diagrams, and L-System. Here we show the graph in Fig.12 to see the performance of the algorithm to perform image transformation based on affine value of the image.

From Fig.12 we can see that the affine values were changed over time due to the image transformation over the time. In another hand, the Hue, Saturation, Brightness, and Weight value were showing the changes over time to see evolution of the images. The calculation of the changes over time showed that 99.5% of the values of affine were changes over time, and 99.29% of the image's attributes were also change over time from 25 images tested. Furthermore, this research was also taken from the transformation of image properties as seen in Fig.12. From the figure we can see that the properties of image were changes over the transformation process which shown the working phase of the algorithm on changing the shape of the fractal. From Fig.12, we can see that the implementation of the algorithm was working properly to share the image from merging process of 2 images with morphogenesis phase.



(a) (b)

Fig. 11 (a) Result for morphogenesis process (b) Result for random transformation



(a) (b)

Fig. 12 (a) Affine transformation value (b) Properties transformation

IV. CONCLUSION

Genetic algorithms can be applied in making software in the field of digital art in the process of making image generation. The use of LCA calculation formula rules, iterated function systems, Voronoi diagrams, and L-Systems can be used in the formation of image patterns. To obtain images, it can be done by merging the two images and the next generation of images to resemble the structural arrangement of the two images. An unlimited number of images are generated based on the type of calculation/population used. The resulting images are generated randomly based on the selection of the calculation type, color, and size used. The resulting image can be exported to various formats, such as: jpg, png, svg, and postscript in one local drive.

REFERENCES

[1] W. Wang, G. Zhang, L. Yang, and W. Wang, "Research on garment pattern design based on fractal graphics," *Eurasip J. Image Video Process.*, vol. 2019, no. 1, pp. 59–78, 2019.

[2] L. Hadimani and N. Mittal, "Development of a computer vision system to estimate the colour indices of Kinnow mandarins," *J. Food Sci. Technol.*, vol. 56, no. 4, pp. 2305–2311, 2019.

[3] P. M. Mather and M. Koch, "Classification," in *Computer Processing of Remotely-Sensed Images*, John Wiley & Sons, Ltd, 2011, pp. 229–284.

[4] K. Lai, L. Bo, X. Ren, and D. Fox, "A large-scale hierarchical multi-view RGB-D object dataset," *Proc. - IEEE Int. Conf. Robot. Autom.*, pp. 1817–1824, 2011.

[5] M. Han, X. Zhu, and W. Yao, "Remote sensing image classification based on neural network ensemble algorithm," *Neurocomputing*, vol. 78, no. 1, pp. 133–138, 2012.

[6] S. G. Mallat, "A Theory for Multiresolution Signal Decomposition: The Wavelet Representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 11, no. 7, pp. 674–693, 1989.

[7] B. B. Chaudhuri and N. Sarkar, "Texture Segmentation Using Fractal Dimension," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 17, no. 1, pp. 72–77, 1995.

[8] C. Voiron-Canicio, "A spatio-morphological modelling for spread predicting," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 5072 LNCS, no. PART 1, pp. 210–220, 2008.

[9] É. Grossiord, B. Naegel, H. Talbot, N. Passat, and L. Najman, "Shape-based analysis on component-graphs for multivalued image processing," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9082. pp. 446–457, 2015.

[10] A. Lindenmayer, "Mathematical models for cellular interactions in development I. Filaments with one-sided inputs," *J. Theor. Biol.*, vol. 18, no. 3, pp. 280–299, 1968.

[11] W. Warchalowski and M. J. Krawczyk, "Line graphs for fractals," *Commun. Nonlinear Sci. Numer. Simul.*, vol. 44, pp. 506–512, 2017.

[12] B. R. Bielefeldt, E. Akleman, G. W. Reich, P. S. Beran, and D. J. Hartl, "L-System-Generated Mechanism Topology Optimization Using Graph-Based Interpretation," *J. Mech. Robot.*, vol. 11, no. 2, pp. 26–37, 2019.

[13] R. L. Ogniewicz and O. Kübler, "Hierarchic voronoi skeletons," *Pattern Recognit.*, vol. 28, no. 3, pp. 343–359, 1995.

[14] H. Edelsbrunner and R. Seidel, "Voronoi diagrams and arrangements," *Discrete Comput. Geom.*, vol. 1, no. 1, pp. 25–44, 1986.

[15] D. Leitner, S. Klepsch, G. Bodner, and A. Schnepf, "A dynamic root system growth model based on L-Systems," *Plant Soil*, vol. 332, no. 1, pp. 177–192, 2010.

[16] P. J. Mavares Ferrer, "Visualization of the Chaos Game for non-hyperbolic iterated function system," *Rev. ODIGOS*, vol. 1, no. 2, pp. 9–20, 2020.

[17] L. L. Zhang, T. C. Chang, and Y. M. Mao, "Sensing and controlling with Markov process for locally independent fractal image," *Sensors Mater.*, vol. 32, no. 6, 2020.

[18] S. M. A. Aghamirmohammadali, R. Bozorgmehry Boozarjomehry, and M. Abdekhodaie, "Modelling of retinal vasculature based on genetically tuned parametric L-system," *R. Soc. Open Sci.*, vol. 5, no. 5, pp. 47–58, 2018.

[19] B. R. Bielefeldt, G. W. Reich, P. S. Beran, and D. J. Hartl, "Development and validation of a genetic L-System programming framework for topology optimization of multifunctional structures," *Comput. Struct.*, vol. 218, pp. 152–169, 2019.

[20] S. Aksoy, K. Koperski, C. Tusk, G. Marchisio, and J. C. Tilton, "Learning bayesian classifiers for scene classification with a visual grammar," *IEEE Trans. Geosci. Remote Sens.*, vol. 43, no. 3, pp. 581–589, 2005.

[21] K. Sheeba and M. Abdul Rahiman, "Gradient based fractal image compression using Cayley table," *Meas. J. Int. Meas. Confed.*, vol. 140, pp. 126–132, 2019.

[22] B. Bruegge and A. H. Dutoit, *Object-oriented Software Engineering: Using UML, Patterns and Java*. Prentice Hall, 2004.

[23] M. Hussain, D. Chen, A. Cheng, H. Wei, and D. Stanley, "Change detection from remotely sensed images: From pixel-based to object-based approaches," *ISPRS J. Photogramm. Remote Sens.*, vol. 80, pp. 91–106, 2013.

[24] A. Vatresia, J. P. Sadler, R. R. Rais, and E. Imandeka, "The development of mobile application for conservation activity and wildlife in Indonesia," in *Proceeding - 2016 International Conference on Computer, Control,*

Informatics and its Applications: Recent Progress in Computer, Control, and Informatics for Data Science, IC3INA 2016, 2017, vol. 2016-Septe, pp. 203–208.

[25] S. Bennett and R. McRobb, SteveFarmer, *Object-Oriented System Analysis and Design*. Mc Graw Hill, 2005.