Generate Leaf Shapes using L-system and Genetic Algorithms

Yodthong Rodkaew[1], Suchada Siripant[2], Chidchanok Lursinsap[3] and Prabhas Chongstitvatana[4]

Department of Computer Engineering
Faculty of Engineering
Chulalongkorn University
Phayathai road, Bangkok 10330 Thailand
E-Mail: 43718130@student.chula.ac.th[1] ,
prabhas@chula.ac.th[4]

Advanced Virtual and Intelligence Computing
(AVIC) Research Center [2,3]
Department of Mathematics
Faculty of Science
Chulalongkorn University
Phayathai road, Bangkok 10330 Thailand.

Abstract: In this paper presents a method that combines with techniques L-system and Genetic Algorithms (GA) to search for a rewriting expression describing leaf shapes. L-system is used to construct a shape of leaf of a given rewriting expression, and GA is used to search an unknown rewriting expression's fitting parameters. Replacement of real value parameters to tag-function is introduced. The result shows both L-system and GA work together and produce an acceptable output.
Key words: Leaf Shapes, L-system, Genetic Algorithm

1. Introduction

In 1968, L-system [1] was introduced by a biologist, Aristid Lindenmayer, to create a realistic plant form by a context-free rewriting expression with conditional and stochastic rule selection concept. The computer graphical output from computer software that use L-system [2,3] looked like a real plant. However, there are some parts of plant that cannot be derived by a rewriting expression such as leafs or flowers. In the computer software [2,3] a predefined leaf and flower shape are used to compose a plant. The work [1, pp. 120-127] presented a predefined expression for leaf edge. We are interested in finding a rewriting expression for a leaf network (Fig.1), Is there has some expression that can create a leaf? The research [4] tried to construct a primary branch network with a given expression, the leaf shape is modified by changing parameters. No expression need to be changed. However, modification of the parameters by human is difficult because there are many parameters. In this paper, we propose to construct a primary branch and use Genetic Algorithm [5] to solve the problem of parameter fitting.

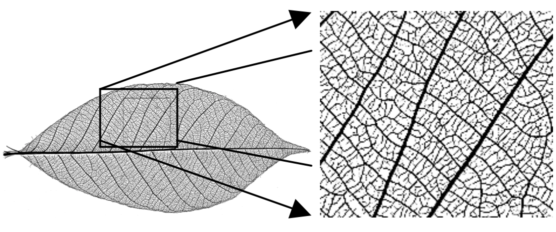


Figure 1: The leaf shape and leaf network.

The paper is organized as follows, Section 2 and 3 briefly introduces L-system and Genetic Algorithm. The experiment is presented in Section 4. Section 5 illustrates results. Finally, Section 6 contains a summary and comments on future work.

2. L-system

From [1], a parametric OL-system, which are context-free, operate on *parametic words,* which is a string consisted of *letters* and *parameters*, called *modules*. The letter as alphabet is denoted by V, and the set of parameters is the set of the real number $\Re$. A module with letter $A \in V$ and parameters $a_1,a_2,\ldots,a_n \in \Re$ is denoted by $A(a_1,a_2,\ldots,a_n)$. Every module belongs to the set $M = V \times \Re^*$, where $\Re^*$ is the set of finite sequences of parameters. The set of all string of modules are denoted by $M^* = (V \times \Re^*)^*$, and the set of all nonempty strings are denoted by $M^+ = (V \times \Re^*)^+$.

The real-valued *actual* parameters appearing in the words correspond with *formal* parameters which may occurs in the specification of L-system productions. Let Σ be a set of formal parameters, C (Σ) denotes a *logical expression* with parameter from Σ, and ε(Σ) is an *arithmetric expression* with parameter from Σ. The combination of formal parameters and numeric constants using the arithmetic operators {+,-,*, /, ^ (the exponentiation operator) } the relational operators {<,>,= } the logical operator { and, or, not } and parentheses { (,) }. The set of all correctly constructed logical and arithmetic expressions with parameters from Σ are noted C(Σ) and ε(Σ).

A parametric OL-system is defined as an ordered quadruple G = (V,Σ,ω,P) where:

- V is the alphabet of the system.
- Σ is the set of formal parameters.
- $\omega \in (V \times \Re^{*})^{+}$ is a nonempty parametric word called the axiom.
- $P \subset (V \times \Sigma^{*}) \times C(\Sigma) \times (V \times \varepsilon(\Sigma))^{*}$ is a finite set of productions.

For an example, if the alphabets V in the system are { F, + } which may occur many times in a strings. Each letter is associated with a rewriting rule. The rule F → F+F means that letter F is to be replaced by F+F. The rewriting process starts from a distinguished string called the *axiom* or ω. Given the axiom string F, in the first derivation step, the string F is replaced by string F+F to be string F+F. In the second derivation step, the string F+F is replaced by string F+F+F+F.

2.1 Drawing mechanism in L-system

In the L-system the drawing is based on the *turtle graphics*. A state of *turtle* is defined as a triplet (x,y,α), the *Cartesian coordinates* (x,y) represented the turtle's position, and α is the direction of the turtle. Given a *step size* β and the *angle increment* δ, With β = 5.0 and δ = 90.0° (Fig. 2a). The symbol F means move forward a step, symbol - means turn left by an angle increment and symbol + means turn right by an angle increment. The string F+F+F+F draws a rectangle (Fig. 2b). The symbol [and] is a stack. Symbol [push the current (x,y,α), and symbol] pop a (x,y,α) from the stack and assigned to the current one (Fig. 2c).

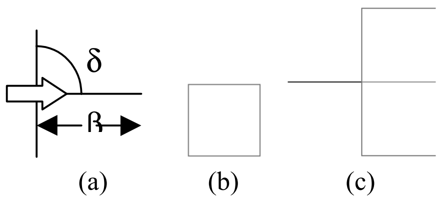


Figure 2: (a) The turtle. (b) The picture from a string F+F+F+F. (c) The picture from a string F [-F+F][+F-F]F.

2.2 The parametric words

One or more parameters can be associated with a symbol. A symbol F means move forward, then F (5) means move forward by 5 pixels.

F(α)	Move forward by α pixels
+(α)	Turn right by α angle
-(α)	Turn left by α angle

However, users can promote a new parametric rule by define it. Thus, the α is an *arithmetric expression* ε(Σ), the definition below is also valid in L-system:

ω : A → B(1)
P_1: B(a) → C(a,a+1)
P_2: C(a,b) → B(a)C(b,a+b)

The results after 1st derivation is: C(1,2)
The results after 2nd derivation is: B(1)C(2,3)
The results after 3rd derivation is: C(1,2)B(2)C(3,5)
The results after 4th derivation is: B(1)C(2,3)C(2,3) B(3)C(5,8)

2.3 The tag-function

The parametric words need a derivation, changing some parameters required calculation of a derived parameter. Since derivation of axiom consume CPU time, in this section, we introduced a *tag-function* to replace parameters. The tag-function can reduce CPU time because it reuses the calculation of parameters value. It is a marker of the function's position. User can change a function value without requiring regeneration of an axiom derivation.

The tag-function looked like tag in HTML, it begins with symbol < and end with >. Inside tag-function is the name of function such as <Fx> means this tag uses a Fx function. Users should assigned tag-function before symbol that should be parameters.

After derivation, all tag functions will be appended with a number that was assigned by our specific L-parser. This number indicates the position of the function that tag-function associated with. When drawing, the turtle will use both derived axiom and function to get a real value (Fig.3).

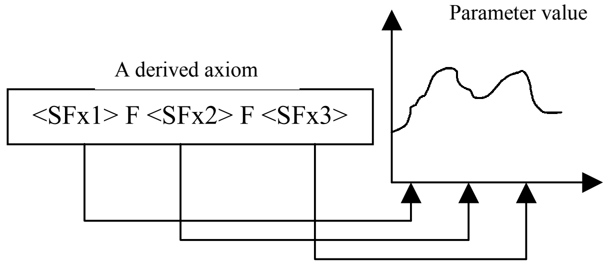


Figure 3: Convertion of tag-function to parameter value.

The meaning of tag-function used in this paper is described below:

Table 1: Tag-function

Tag - Function	Meanings	Output Range
<Fx*N*>	$\gamma = \text{Fx}(N/N_{max})$	0.0 - 1.0
<Lx*N*>	Turn left by $\text{Lx}(N/N_{max})$ angle	0.0 – 90.0
<Rx*N*>	Turn right by $\text{Rx}(N/N_{max})$ angle	0.0 – 90.0
<SFx*M*>	$\gamma = \text{SFx}(M/M_{max})$	0.0 – 1.0
<SLx*M,N*>	$\sigma = M/M_{max}$ $\theta = (\sigma\times2) \times \text{LL}_1\text{x}(N/N_{max}) + (1.0\text{-}\sigma\times2) \times \text{LL}_2\text{x}(N/N_{max})$; if $(0.0 \le \sigma < 0.5)$ $\theta = ((\sigma\text{-}0.5)\times2) \times \text{LL}_2\text{x}(N/N_{max}) + (1.0\text{-}(\sigma\text{-}0.5)\times2) \times \text{LL}_3\text{x}(N/N_{max})$; if $(0.5 \le \sigma \le 1.0)$ Turn left by θ angle	0.0 – 90.0
<SRx*M,N*>	Same as <SLx*M,N*>, Turn right by θ angle	0.0 – 90.0

The N and M in tag-function are added by L-parser which N is increased by the number of derivation, M is increased by the number of tag-function. For each function has their own N or M. The symbol γ is denoted to a multiple value of step size β. A symbol F will move forward by step size ($\gamma \times \beta$) pixels.

The function's output range from table is the output from function (not value of N/N_{max}) for an example, function $\text{Lx}(N/N_{max})$ can produce a value 50.0, N value must in rannge 0.0 - N_{max} , also M, and the value of N/N_{max} and M/M_{max} should be in range 0.0 - 1.0. The tag-function <SLx> and <SRx> is calculated by using tag-function LL_1, LL_2, LL_3 .

2.4 Experiment's rule definition

In this paper, we will use the definition for an experiment:

ω : A → <LT>M!N
P_1: M → [BBBBBBBBBBB]
P_2: N → [CCCCCCCCCCC]
P_3: B → <FF>[L]
P_4: C → <FF>[R]
P_5: L → <LT><Lx><SFx>J
P_6: R → <RT><Rx><SFx>K
P_7: J → <SRx><Fx><Fx>J
P_8: K → <SLx><Fx><Fx>K

Where $\beta = 15.0$, $\delta = 90.0°$ and using 8^{th} derivative.

The definition derived from the idea of skeleton (Fig.4a), with an adjustment of function, the skeleton can be transformed (Fig.4b-d). Figure 4e shows function adjustment by hand.

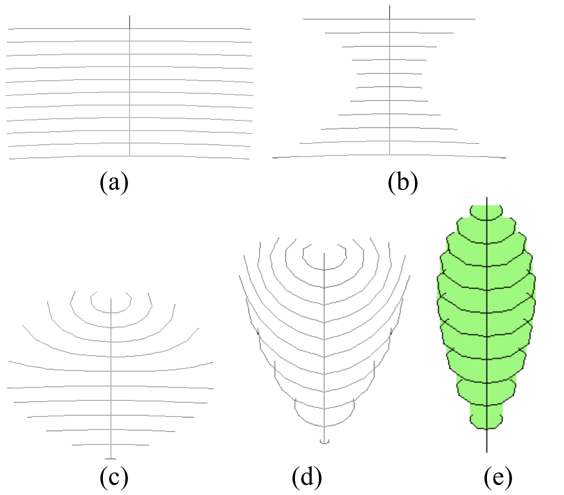


Figure 4: The picture create from rules: (a) The skeleton (b) (c) (d) Various transformation shape by adjusting function. (e) an adjusted function by hand.

3. Genetic Algorithms

The Genetic Algorithms [5] or called GA, is based on an inspiration from natural selection. GA was developed by John Holland, his colleagues, and his student at the University of Michigan. It is a *robust* algorithm use for search and optimization of solutions. Each solution called *individual.* A group of individual called *population*. GA evaluates each individual to measure *fitness*. An individual who has high fitness can produce their children or *offspring* for the next *generation*.

In the first generation, GA randomly generates a population with a specific parameters: a number of individual, a length of individual etc. and evaluates them. The second generation, GA randomly selects an individual with probability according to its fitness to produce offspring and modifies them by *genetic operators* (reproduction, crossover, and mutation). This process is repeat until the terminating condition is reached such as an individual can solve the problem or reach the specific number of generation.

3.1 Individual

An individual in this experiment is denoted to the points of β-*Spline* function [6]. A tag-function is consisted of 4 points for each β-Spline. There are 7 tag-functions in this experiment, therefore, an individual contains 28 floating-point variables (Fig.5).

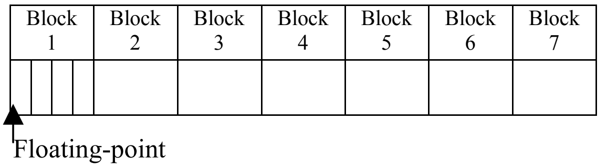
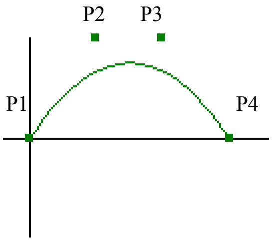


Figure 5: An individual.



Block			
P1	P2	P3	P4
0.0	1.0	1.0	0.0

Figure 6: The β-Spline constructed from the block .

The block number in an individual is matched to the spline number. The spline No.1 is denoted <Fx>, No.2 is denoted to <Lx>, No.3 is denoted to <Rx>, No.4 is denoted to <SFx>, No.5-7 are denoted to <LL1> <LL2> <LL3> respectively.

3.2 Genetic Operator

In this paper, we used *reproduction*, *crossover*, and *mutation*. The reproduction is the duplication from parent to their child. With the crossover operator, 2 individuals is to be selected as 2 parents and make a random point to split each individual in to 2 parts, and recombine them. This method generates 2 offstrings. The last one is mutation, it produces an offspring by randomly change four values in the parent.

3.3 Fitness Function

The comparison from the output from L-system and the target picture is the fitness function. In this paper, we are interested in an outline of leaf. The fitness function is calculated by this formula:

$$Fitness = 100.0 - \left(\sum_{i=0}^{width} \left| x_i^{target} - x_i^{source} \right| + \sum_{i=0}^{height} \left| y_i^{target} - y_i^{source} \right| \right)$$

4. The Experiment

An individual as set of parameters is evaluated by L-system which produces a picture as an output. We matched the target picture and the picture from L-system and measure the fitness of an individual. The fitness depends on the difference between two pictures using the function in section 3.3.

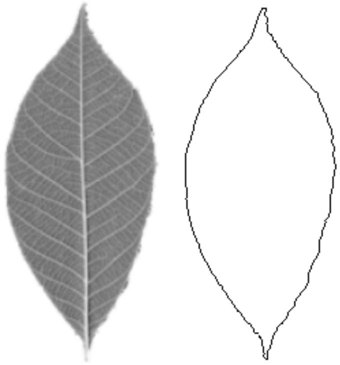
4.1 Genetic Parameters

The genetic parameters are as follow:

Number of Individual	200
Reproduction	20%
Crossover	40%
Mutation	40%
Number of Generation	200

4.2 Target

The target picture is taken from the outline of a real leaf. A sample of the target is shown below:



(a) (b)

Figure 7: (a) The real leaf and (b) its outline.

4.3 Results

Our system generates the output satisfactority. Initially the match good but gradually becomes better. Figure 8 shows the fiteest individual of the first generation. Figure 9 shows the result of the final generation. The output matches closely to the target. It is better than the one produced by manually adjusting parameters by human.

5. Conclusion

In this paper, we presented a method that combines two techniques L-system and Genetic Algorithms (GA) to search for a rewriting expression describing leaf shape. Replacement of real value parameter by tag-functions has been introduced. The result from GA is better than function that is adjusted by human. The leaf shape from a rewriting expression looked like a leaf. However, the real leaf is not symmetric. Our result can be improved by modify a rewriting expression. For the future work, we will try to construct a primary branch, same as a real leaf.

6. References

[1] Prusinkiewicz P. and Lindenmayer A., *The Algorithmic Beauty of Plants.*, Springer-Verlag, 1996.

[2] Cynthia F. Kurtz and Paul D. Fernhout, *PlantStudio.*, Computer Software, http://www.kurtz-fernhout.com/PlantStudio/.
[3] Chuai-aree S., *PlantVR.*, Computer Software, http://www.geocities.com/csomporn_2000/plantvr/plantvr.htm.
[4] Lursinsap C., Sophatsathit P., and Siripant S., *Simulation of Leaf Growth Based On A Rewriting System: A Unified Leaf Model.*, Advanced Virtual and Intelligence Computing (AVIC) Research Center, Chulalongkorn University., Technical Report No. 01.01.2000.
[5] Goldberg D.E., *Genetic Algorithms in Search, Optimization, and Machine learning.*, Addison-Wesley, Redwood City, 1989.
[6] Watt A., Watt M., *Advanced Animation and Rendering Techniques.*, Addison-Wesley, New York, pp.90-91.

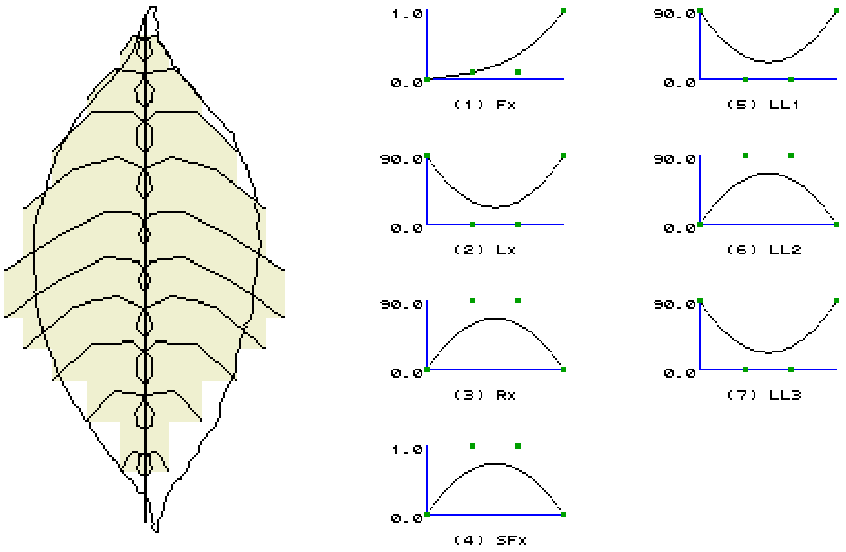


Figure 8: The best individual in the 1st generation.

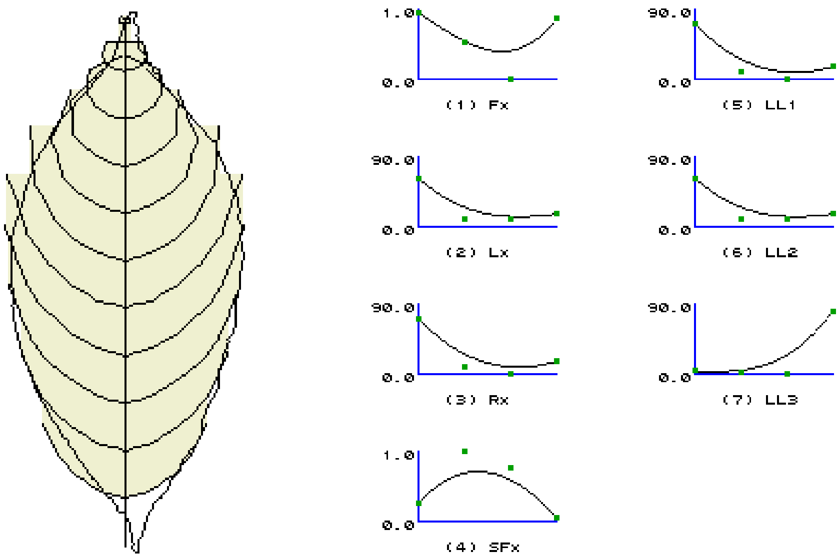


Figure 9: The best individual in the 200th generation.