

UC Davis

UC Davis Previously Published Works

Title

Design and Implementation of a Plant Morphology Modeling System Based on Evolutionary Strategy

Permalink

<https://escholarship.org/uc/item/70x1n6wb>

Authors

Ding, Weilong
Hu, Chen
Wu, Fuli
et al.

Publication Date

2014

DOI

10.1007/978-3-662-45283-7_19

Peer reviewed

Design and Implementation of a Plant Morphology Modeling System Based on Evolutionary Strategy

Weilong Ding^{1,2}, Chen Hu¹, Fuli Wu¹, Nelson Max²

¹College of Computer Science & Technology, Zhejiang University of Technology, Hangzhou 310023, China

²Department of Computer Science, University of California, Davis 95616, USA
{wlding, chenhu, fuliwu}@zjut.edu.cn, max@cs.ucdavis.edu

Abstract. In order to improve the efficiency of virtual plant modeling based on L-systems, a plant morphology modeling system using an evolutionary strategy is designed and implemented. First the axiom and the productions of an L-system are mapped to chromosomes, and then the initial population is designed. Genetic operators are designed to generate the individuals of a new population. The fitness function considers two aspects: the plant outline, and the internal branching. Using the fitness function for evolution, the L-system's production rules for the target plants can be obtained. Key implementation techniques, function modules, and interfaces of the system are briefly introduced. Finally, the validity of the system is tested in detailed experiments.

Key words: L-system, gene expression programming, genetic algorithm, plant morphology.

1 Introduction

Current research on virtual plants has involved the fields of computer science, applied mathematics, botany, biology and ecology. Virtual plant models can not only reproduce natural landscape, but also model the growth condition of plants under the influence of different physiological and ecological factors.

L-systems, which are effective approaches to model plant morphology [1,2], are widely used to create virtual plant models. With the continuous development of L-systems, some plant modeling packages which are based on them have begun to appear. A common characteristic of this kind of software is that the user must know how to set up the parameters of the axioms and the productions before modeling. To simulate a target plant, an L-system needs dozens of production rules. Most of the rules and parameters are artificially determined by trial and error. Because you don't know what shapes will be produced by using the parameters you set up, to get the

ideal target shape you need many iterations, so this manual method is blind and inefficient. Therefore, it is desirable to automatically obtain L-system rules and parameters to match the simulated target.

In this paper, we propose a plant morphology simulation algorithm based on an evolutionary strategy and develop a system based on the proposed algorithm. On the basis of gene expression programming (GEP), the system first automatically obtains the production rules of the L-system, and then simulates the morphological structure of a specific plant. The system established in this paper does not need the user to control the L-system directly; the user need only choose one plant type from the templates provided by the system. Through the interactions between the user and the computer, the system can generate various plant shapes. The method overcomes the blindness and inefficiency of manipulating the L-system directly.

2 Related work

To visually simulate plant morphology there are at present process-based methods, image-based methods, and reconstruction methods based on large-scale measurement data [3]. The process-based methods use a mathematical model to simulate plant growth or morphological characteristics, which include different types of L-systems, reference axis technology, dual-scale automaton model, fractal methods, iterated function systems and particle systems, etc. Although these models have their advantages, they all need to set up the rule parameters. The traditional methods of manual parameter acquisition are inefficient and difficult. Image-based modeling methods [4] directly construct 3D plant structure through the analysis and processing of one or more images of a plant. Compared with process-based or measurement-based methods, these methods are relatively simple, and the simulated plants they produce have a very strong sense of reality. However, these methods can only imitate a specific static plant; they are not suitable for simulating dynamic growth. Moreover, their modeling process requires a large amount of manual intervention. The advantage of the static structural model to simulate the shape of a plant [5] is its ability to accurately describe the morphology of specific plants, which can be used to study the quantitative and qualitative characteristic related to the plant spatial structure. However, these methods require lots of manual measurement data and a lot of time to process data.

The scholars of CIRAD studied a method to extract plant growth rules by using statistical analysis and pattern recognition, and applied it in a sub-system of the AMAP software, named AMAPmod [6]. Since the evolution of plant topological structure and the patterns or rules of geometry change are not known in advance and

also are very complex, fully and accurately extracting plant growth rules needs other intelligent technologies, instead of simply relying on pattern recognition.

In order to simulate a specific 2D plant structure, papers [7-9] studied the problem of how to extract the production rules and axiom for a simple L-system. Since the simulated target is a simple 2D plant form, the chromosome encoding and genetic operation forms were quite simple. How to automatically evaluate the fitness of an individual in the system is a key problem. Ochoa [7] designed a specific fitness function, considering the height, left-right proportion, leaf area, structural stability, and bifurcation points of simulated plants. Using the distance from the simulated plants to a specified two-dimensional plant structure, Bian *et al.* [8-9] designed a fitness function for two-dimensional simulation models. But evaluating the individual fitness of a three-dimensional simulated plant was not considered in their studies. By using an evolutionary algorithm, Hemberg and O'Reilly [10] generated the production-rule set of a Map L-system to simulate the growth of 3D digital surfaces. Venter and Hardy [11] proposed a 3D plant simulation model based on genetic programming. The model uses a "genotype" which is expressed by an L-system to describe a specific plant shape, and then forms different "phenotypes" through the graphical interpretation of the L-system. The goal of their study is to obtain a large number of different plant shapes, and to avoid or reduce the complexity of setting up the parameters for the axiom and the productions of an L-system. Since there is no explicit optimization target, their model did not use a fitness evaluation function, so the best individual has to be chosen by man-machine interactions according to the visual effects of individual plants. Obviously, the search efficiency of this manual method is very low.

3 Evolution of L-system rules based on GEP

3.1 Genetic encoding and initial population

An L-system is a string rewriting system. Its axiom and productions are composed of characters. Therefore, when the axiom and the productions of an L-system are mapped to chromosomes, symbols are mostly used to do the encoding for individual chromosomes [8, 9]. Some scholars integrated different encoding techniques in one chromosome. For example, Venter and Hardy [11] mixed the technique of symbol encoding with the technique of binary encoding. This hybrid encoding is convenient for the iteration of the L-system's string, and also uses the advantage of binary coded genes. Therefore, this paper also uses this hybrid chromosome coding method. Each individual is formed by chromosomes $\{c_0, c_1, \dots, c_{nc-1}\}$, where nc represents the number of chromosomes. Each chromosome is also composed of multiple genes $\{g_0, g_1, \dots,$

g_{ng-1} }, where ng represents the total number of genes in each chromosome. Chromosome c_0 corresponds to the axiom while chromosomes c_1 to c_{nc-1} correspond to the productions of an L-system. Each gene consists of a head and a tail, respectively expressing the real character corresponding to the character set of L-system and the parameters corresponding to the character. In addition, terminal symbols which are extra bits on the bit string defining the last parameter of the last gene are added into each chromosome. The size of chromosomes can be changed and the diversity of the individuals will be increased by using of the terminal symbols.

The individual chromosomes in the initial population are designed through the random combination of characters. According to the rules of L-systems, some combinations may be illegal. If the individuals with illegal style are a large proportion in the population, the search space will become enormous and the efficiency will become very low. In order to improve the effectiveness of the solution space, we add some predefined chromosomes into the initial population. The production rules of those chromosomes are determined through a manual and qualitative way according to the structural characteristics of the target plants, rather than in a completely random manner. The remaining individual genomes are generated through the control of gene g_0 in each chromosome and the number of genes which can generate branches.

3.2 Design of genetic operators

We first randomly select an individual and then do some genetic operations, such as crossover, mutation, and transposition, so as to generate the individuals of a new population. By improving the strategy to eliminate the bad individuals, the effects of bad characters can be reduced and the efficiency of the algorithm can be improved.

As a basic genetic operation, crossover between the individuals of a population is used to exchange genes and generate new individuals. According to the difference of crossover positions on two chromosomes, there are some different kinds of gene exchange: one-point crossover, two-point crossover, three-point crossover, and genetic recombination. Genetic material is exchanged based on the unit of a gene, which avoids the separation of the head and the tail of a gene.

Transposition is the movement of a sequence of characters in a chromosome to other positions which are located in the same chromosome. It has two forms: insertion and gene conversion. The former means that a short character sequence is randomly selected in a chromosome, and then the selected character sequence is used to replace another character sequence in the same chromosome. The latter means that a gene in a chromosome is transferred to the head of chromosome, while the positions of the other genes are moved backward. This operation is only for a single complete gene, to ensure the integrity of the genetic information.

For the evolutionary mechanism, if we consider only the operations of crossover and transposition, the population will become too similar, and have premature convergence, so that the optimal solution cannot be obtained. In order to avoid premature convergence, the mutation operation of the individuals should be taken to make the offspring genes change according to a small probability after the operations of crossover and transposition. The individuals will produce new characters through these changes.

3.3 Fitness evaluation

A reasonable fitness function can guide the genome evolution, and avoid the loss of excellent individuals and premature convergence of the genetic algorithm. We need to judge whether an individual is similar to the target of plant to be simulated, in order to select a subset of the fittest individuals to produce the next generation. However, due to the complexity of structure of plants it is a challenging task to quantify the similarity between real plants and rebuilt 3D plant models, and then design an appropriate fitness function. In 1998, Ochoa [7] designed an evaluation function as a weighted mean of five elements of a virtual plant: plant height, proportion of left and right, leaf area, stability of the structure, and branching points. Bian et al. [8, 9] designed an individual fitness based on a distance function between the virtual plant and the 2D structure of the target plant. In this function, the number of organs, depth of branching and the distance between organs were considered. Zheng. [12] divided the structure of a tree into left tree, right tree, and middle tree, and then computed the difference of the number of child trees in each branching level using iteration. They took the total difference of branching as the difference of the two trees. The fitness evaluation function from the studies mentioned above are all for 2D simulation models, and can not evaluate the fitness of 3D simulated plants.

Unlike in the above papers, in this paper we first calculate the similarity between the simulated plants and the real plants, and we consider both the plant outline and the inner details of the plant structures. The three-dimensional convex hull is used to represent the plant outline. In order to evaluate the similarity between the outlines of the real plants and the rebuilt 3D plant models, we compute the 3D convex hulls and project them in different directions to get 2D projections, and then calculate the 3D similarity based on the similarities of these 2D projections. Based on this idea, we designed a coarse-grained fitness function, *Rfitness*, to effectively evaluate the similarity between the simulated plants and the target plants just from the shape of the plant. On the other hand, we propose an algorithm to evaluate the similarity of morphology between the simulated plants and the target plants according to the internal branching structure, which is described by using of geometric attributes of plant organs and topological connections between different organs. These attributes

and connections include the average branching angles on the stem, the ratios of cross-sectional area and diameters of the stem to the branches on main stem, the ratio of the plant's width to its height, and the average branching angles of 2nd level branches on 1st level branches. The similarities between these parameters can be calculated first, and then a similarity with a weighted average of these five aspects can be calculated. According to these calculated parameters, we designed a fine-grained fitness function, *Dfitness*, to evaluate the similarity of internal architectures between the simulated plants and the target plants.

Due to the complexity of the structure of plants, the method to calculate the similarity between different plants is also very complicated and needs a lot of words to explain clearly. Limited by the length of this paper, we will specifically explain the details of our method in another paper. Here we only present a brief formula combining these two kinds of fitness values to evaluate the fitness of an individual:

$$Fitness = \alpha * Rfitness + \beta * Dfitness \quad (0)$$

where α and β are empirical constants, which satisfy $\alpha + \beta = 1$. Combining these two fitness functions can help our system to find the optimal individuals, whose outline and internal topologies are both similar to the target plants, so as to generate a detailed simulation of a specific plant.

4 System implementation

4.1 System implementation environment and parameters setting

Based on the above methods, we implemented an experimental program on Windows XP, with Visual C++ 2005 as IDE. The program is implemented in C++ with the OpenGL library. The experimental machine has 4GB RAM, an Intel Pentium D 2.67GHz CPU and an Nvidia GeForce 7300GT graphic card. The flowchart of the system is given in Fig. 1.

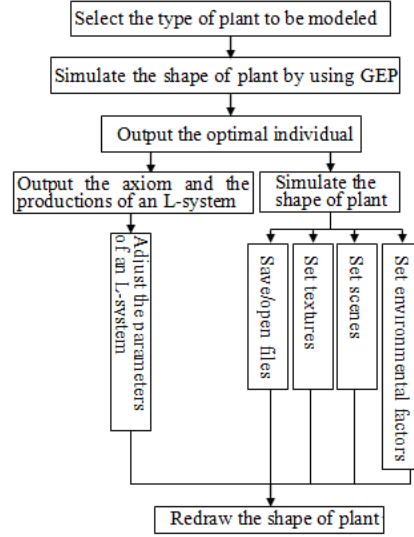


Fig. 1. The flowchart of the system

The number of individuals in the initial population (n_o), the number of chromosomes (nc), the number of genes in a chromosome (ng), the termination condition of evolution, the iteration number of an L-system, and the probabilities of crossover, transposition, and mutation are described in detail in Table 1.

Table 1. Parameters of the algorithm

initial population (n_o)	20
the number of chromosomes (nc)	4
the number of genes in a chromosome (ng)	6
the number of evolutionary generations	100
the iteration number	3
the probability of mutation (p_m)	0.02
the probability of Insert String (p_{is})	0.02
the probability of gene conversion (p_{gt})	0.02
the probability of one-point crossover(p_{1r})	0.2
the probability of two-point crossover (p_{2r})	0.2
the probability of three-point crossover(p_{gr})	0.2

4.2 Main functions and main modules of the system

The system can not only simulate different types of plant morphology, but also simulate the change of plant shapes under the influence of different light environments

and mineral resources. It provides a simple and practical plant morphological modeling software for the user. The system also provides various scenes and organ textures for the user to select. It can save the simulation results, as well as open saved plant morphology files for further modification. The system can be applied to visual simulation, virtual scene generation, and three-dimensional animation. It consists of seven modules:

(1) Preprocessing module

In this module, users can select a predefined plant type as a simulation target. The system can extract the features of the target plant's outline and its internal branching structure.

(2) Genetic algorithm module

In this module, the genetic algorithm is used to create various individuals by using different genetic manipulations, including individual choice, transposition, crossover and mutation, and calculation of the individual fitness. When the algorithm reaches the termination condition after these genetic manipulations, the system invokes the morphology output to present the shapes of the best individuals.

(3) Morphology output module

This module is used to draw the shapes of plant morphology according to the rule parameters of the optimal individuals.

(4) Parameter modification module

This module allows the user to modify the parameters of the L-system so as to create ideal plant shapes.

(5) Graphics rendering module

The module is used to render the scene. The user can select various scenes and plant organ textures.

(6) Environmental factors setting module

The module can provide interfaces for the users to set up different lighting and distributions of the mineral resources, and let them observe the change of plant morphology in response to different environmental conditions.

(7) File management module

The module provides functions to save the parameters for the evolved individuals, and to open and redraw saved plant morphologies.

4.3 The interface of the system

The interface of the system includes the unit for selecting the target plant type, the image display unit, the parameter modification unit, the environmental factor setting unit, and the file operations unit. For example, suppose a user wants to simulate a sympodial branching tree, whose shape is shown in Fig. 2. Aiming at this target, he first needs to select a module which describes the features of sympodial branching

from the predefined library of plant shapes. Then the system will generate a 3D plant morphology similar to the target plant and gives its parameters to the user. The user can modify these L-system parameters and the environmental factors of the scene to create different plant shapes. The system interface and simulated results are shown in Fig. 3. Through setting different environment factors, the simulated result will show different morphologies. When the user selects the effect of side light, the created plant shape will change accordingly under the influence of the light source position, as shown in Fig. 4. We modify the topological parameters of the branches according to their positions on the tree. If the light source is on the left side of a tree, the length and the branching angles of branches on the left side will be increased within a certain range, while those on the right side will be decreased. But if the light is on the right side, the situation is just the opposite. The user can also choose an uneven distribution of resources to observe the distribution and growth conditions of individuals in a region. From these examples, we can see that the system can realistically simulate plant morphologies with different branched structures.



Fig. 2. Target tree

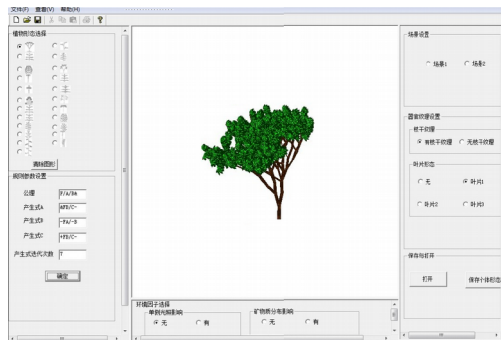


Fig. 3. System interface and simulated result



a) Left-side light b) Light from above c) Right-side light

Fig. 4. The effects of directional light on plant morphology

5 Discussion

Most current works that focus on the simulation of plant shape based on genetic algorithms do not have a specific plant shape to be simulated. The purpose of evolution is just to obtain a lot of different plant forms, since there is no explicit target form. What they did was just to reduce the complexity in the process of setting parameters of the axiom and the productions of an L-system. So they did not design any evaluation function to judge the individual fitness. The superior individuals which form a new population were chosen manually in every iteration step [11]. In terms of search efficiency, there are limitations in such selection methods, which are based on human visual impressions [13]. A reasonable fitness function can evaluate the merits of individuals automatically and also improve the efficiency of searching. Therefore, Christian Jacob [14], Ochoa [7], and Bian [8, 9] designed fitness functions to guide the evolution. But those fitness functions were only for the 2D simulation of plants, and did not consider 3D plant structure. In contrast to their fitness functions, we not only consider the plant outline, but also consider the inner details of the plant structures, such as branching angles and diameters of the different organs. We designed a coarse-grained fitness function and a fine-grained fitness function, to evaluate respective aspects. Therefore, our method is suitable for dealing with 3D plants model and can obtain the best individuals easily and more accurately.

Our plant modeling methods also have some shortcomings. 1) In order to improve the effectiveness of the solution space, the method only designed twenty three kinds of templates, and these are only for woody plants. Thus, our method can only simulate limited types of plant shape and is not suitable for simulating the shapes of the herbaceous plants. 2) It can only generate plant models which are similar to the target plants. Because plant structure is very complex, it is very difficult to design an ideal fitness function. The currently fitness function uses the similarity between real plants and the 3D reconstructed model. Complex 3D plant structures cause the similarity measure between plants individuals to also be complicated. Hence, we need to develop a simple and quick way to compare the similarity between real plants and 3D models. Then we can design a reasonable and efficient fitness function.

The authenticity of the visual effects should be reflected in four aspects: shape similarity, topology similarity, geometric similarity, and geometric detail realism. The plant shapes generated in this paper are similar to the simulated target in external morphology. But in the geometric details, the simulation results of our system is not visually satisfactory, since we neither add textures to leaves and bark, nor add the effects of shadows and other details. Thus we will do future work to compensate for these weaknesses. Nevertheless, our system can automatically seek out the plant shapes which are similar to the target plants through an evolutionary algorithm. So the

system can avoid the cumbersomeness and low efficiency of manually determining the axiom and production rules of an L-system.

6 Conclusion

In this article, a new system for simulating plant morphology based on evolutionary strategy is designed and implemented. The system does not need the user to manipulate the L-system directly, so it has the advantages of simple operation, strong interactivity, and wide range of applications, in contrast to traditional L-system-based plant simulation software.

In future work, we will study an optimized strategy for generating the initial population and further refine the design of the fitness function to more precisely simulate the target plants. We will also simulate the dynamic interaction between different environment factors and plant morphologies. Moreover, we also need to find suitable methods to simulate the shapes of herbaceous plants based on an evolutionary strategy.

Acknowledgements

This work was supported by the Natural Science Foundations of Zhejiang Province (LY14C130015). This is a post-peer-review, pre-copyedit version of an article published in [Communications in Computer and Information Science]. Please see https://link.springer.com/chapter/10.1007/978-3-662-45283-7_19 for the final published version.

References

1. Prusinkiewicz P, Lindenmayer A: The Algorithmic Beauty of Plants. Springer Verlag, New York (1990)
2. Frédéric Boudon, Christophe Pradal, Thomas Cokelaer, et al: An L-System Simulation Framework for Modeling Plant Architecture Development Based on a Dynamic Language. *Frontiers in Plant Science*, 3:76(2012)
3. Weilong Ding, Zhijun Cheng, Mingzhi Chen: Interactive Modeling Plant Architectures Based on Template Library. *Proceedings of the 2009 International Conference on Modeling, Simulation and Visualization Methods*. Las Vegas, pp.121-126 (2009)
4. Ping Tan, Tian Fang, Jianxiong Xiao, et al: Single Image Tree Modelling. *ACM Trans. Graph*, 27(5): Article 108(2008)

5. Bangyou Zheng, Lijuan Shi, Yuntao Ma, et al: The study of rice canopy field in-situ 3D digital and virtual layer. *Scientia Agricultura sinica*, 42(4):1181--1189(2009)
6. Christophe Godin, Yann Guédon, Evelyne Costes, et al: Measuring and analyzing plants with the AMAPmod software. In: Michalewicz M T, ed. *Plants to Ecosystems Advances in Computational Life Sciences*. Melbourne: CSIRO, Australia, pp.53--84(1997)
7. Gabriela Ochoa: On Genetic Algorithms and Lindenmayer Systems. *Parallel Problem Solving From Nature (PPSN V)*, Lecture Notes in Computer Science 1498, Springer-Verlag, Berlin, pp.335--344(1998)
8. Runqiang Bian, Phoebe Chen, Kevin Burrage, et al: Derivation of L-system models from measurements of biological branching structures using genetic algorithms. *Lecture Notes in Computer Science* 2358, 514--524(2002)
9. Runqiang Bian, Hanan, Jim, Norishige Chiba: Statistical data directed evolution of L-system models for botanical trees. *Proceedings of the 4th International Workshop on Functional-Structural Plant Models*, 253--256(2004)
10. Martin Hemberg, Una-May O'Reilly: Integrating generative growth and evolutionary computation for form exploration. *Genetic Programming and Evolvable Machines*, 8(2):163--186(2007)
11. Johannes Venter, Alexandre Hardy: Generating plants with gene expression programming. *Proceedings of Afrigraph 2007*, pp.159--167(2007)
12. Meirong Zheng, Zhenfeng He: Construction of virtual plants based on genetic algorithm with dual-scale automata. *Journal of Fuzhou University (Natural Science)*, 38(2):208--212(2010)
13. Jon McCormack. *Aesthetic Evolution of L-Systems Revisited*. *EvoWorkshops 2004*, LNCS 3005, pp. 477--488(2004)
14. Christian Jacob: Genetic L-System Programming. *PPSN III - Parallel Problem Solving from Nature*, International Conference on Evolutionary Computation, Lecture Notes in Computer Science 866, Springer-Verlag, Berlin, pp. 334--343(1994)