CROATICA CHEMICA ACTA
CCACAA **81** (2) 341–345 (2008)

ISSN-0011-1643
CCA-3250
Original Scientific Paper

Generation of the Figures of Some Fullerenes by Using L-Systems

Mehdi Vahidipour,[a,*] Hosein Sabaghian-Bidgoli,[a] and Gholamreza Vakili-Nezhaad[b]

[a]*Department of Computer Engineering, Faculty of Engineering, University of Kashan, Kashan 87317-51167, I. R. Iran*

[b]*Department of Chemical Engineering, Faculty of Engineering, University of Kashan, Kashan 87317-51167, I. R. Iran*

RECEIVED AUGUST 5, 2007; REVISED APRIL 16, 2008; ACCEPTED APRIL 24, 2008

In 1968, Aristid Lindenmayer introduced a biologically-motivated formalism for simulating the development of multi-cellular organisms, subsequently named L-systems. The applications include, on one hand, the modeling and visualization of plants at different levels of abstraction for a variety of purposes, and, on the other hand, geometric modeling of curves and surfaces. In this paper, we introduce L-system for generating NiceGraph drawing of fullerenes C_{20} and C_{60}, without information about the carbon atoms coordination. We chose L-systems because they can express drawing steps in a compact way and are parallel in nature. It will be a good trend for visualizing complex supramolecules.

Keywords
L-system
the figures of fullerenes
turtle interpretation

INTRODUCTION

A new method for generating figures of some fullerenes is presented and illustrated with examples. The idea is to generate a string of symbols using an L-system, and to interpret this string as a sequence of commands which control a »turtle«. Suitable generalizations of the notions of the L-system and of a turtle have been introduced in Ref. 1. The resulting mathematical model can be used to create a variety of (finite approximations of) fractal curves, ranging from Koch curves, classic space-filling curves, to relatively realistic-looking pictures of plants and trees. All these pictures are defined in a uniform and compact way.

A special kind of rewriting L-systems, *Map rewriting*, is used to simulate the development of single-layered cellular structures such as those found in fern gametophytes, animal embryos and plant epidermis. All structures considered are of microscopic dimensions and relatively undifferentiated, yet the presented methods may bring us closer to the modeling of more complex patterns, such as the venation of leaves. The process *Map rewriting* of cell division can be described by extension of string rewriting.[1,2]

A varity of algorithms are introduced to generate fullerenes and some of them are efficient. The algorithm in Ref. 3 uses the polyhedron Stone-Wales transformation and local random search to generate fullerenes at random. The search algorithm searches local minimum of specific energy function. In this work, generating realistic drawing of fullerenes was not the major goal.

In Refs. 4 and 5 based on the production of polygraph from monographs, the authors tried to formulate the fullerenes. The resulted formula produced graph that had to be given to graph drawing software to generate the NiceGraph geometrical drawing.

There are some algorithms for drawing molecular graphs in three-dimensional space. The article[6] compared them for their ability of plausible molecular geometric.

* Author to whom correspondence should be addressed. (E-mail: vahidipour@kashanu.ac.ir)

Here we introduce a novel application of the L-systems to generate the figures of some fullerenes. In the present work without any extra information and just with the L-system rewriting rules, the figures of some fullerenes were generated.

The L-system definitions and concepts are in the next section. After that in section "Generation of the Figures of Some Fullerenes by Using L-Systems", our idea about to use of L-system is presented. Figures of C_{60} and C_{20} that are generated by this L-system will be showed.

THE L-SYSTEMS

The central concept of L-systems is the rewriting that is implemented by some rules or productions. In general, rewriting is a technique for defining complex objects by successively replacing parts of a simple initial object. In 1968 a biologist, Aristid Lindenmayer, introduced a new type of string-rewriting mechanism, subsequently termed L-systems[7]. The method of applying productions is essentially different for Chomsky grammars[8] and L-systems. In Chomsky grammars productions are applied sequentially, whereas in L-systems they are applied in parallel. This means that all letters in a given word are simultaneously replaced according to productions. Parallel production applications have an essential impact on the formal properties of rewrite ng systems. For example, there are languages which can be generated by context-free L-systems (called 0L-systems) but not by context-free Chomsky grammars.[8,9]

Formal definitions describing D0L-systems and their operation are given below. Interested reader may refer to the works of Herman and Rozenberg[8] and Roaenberg and Saloma.[10] Let Σ denotes an alphabet, Σ^* the set of all words over Σ, and Σ^+ the set of all nonempty words over Σ. A 0L-system is an ordered triplet $G = \langle \Sigma, \omega, \rho \rangle$ where Σ is the alphabet of the system, $\omega \in \Sigma^*$ is a nonempty word called the axiom and $\rho \subset \Sigma \times \Sigma^*$ is a finite set of productions. We write $\alpha \to X$, if a pair (α, X) is a production. The letter α stands for the predecessor and the word X denotes the production successor. A 0L-system is deterministic (D0L-systems) iff for each $\alpha \in \Sigma$ there is exactly one $X \in \Sigma^*$ such that $\alpha \to X$.

Let $\mu = \alpha_1\alpha_2...\alpha_m$ be an arbitrary word over Σ. The word $V = X_1X_2...X_m \in \Sigma^*$ is directly derived from (or generated by) μ and write $\mu \Rightarrow V$, iff $\alpha_i \to X_i$ for all $i = 1,2,...,m$. A word V is generated by G in a derivation of length n if there exists a sequence of words $\mu_0,\mu_1,...\mu_n$ such that $\mu_0 = \omega$ and $\mu_n = V$ as well as $\mu_0 \Rightarrow \mu_1 \Rightarrow ... \Rightarrow \mu_n$.

In the next stage we introduce the mapping procedure of the string to the picture. A (graphic) interpretation function $I : \Sigma^* \to R^3$ is mapping the set of strings over the alphabet Σ into the set of pictures. There are some functions that can be used[11–13] but the Prusinkiewicz focused on an interpretation based on a LOGO-style turtle[14]

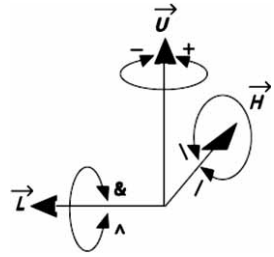


Figure 1. Controlling turtle in three dimensions.

and presented more examples of fractals and plant-like structures modeled using L-systems.[15,16] The sate of turtle is the basic idea of turtle interpretation. The state is defined as Position and orientation. The turtle's position is represented by a triplet (x, y, z) in the Cartesian coordinates. The current orientation of the turtle in space is represented by three perpendicular unit vectors. These vectors $\vec{H}, \vec{L}, \vec{U}$ indicate the turtle's heading, the left direction, and the up direction.[14] These vectors satisfy the equation $\vec{H} \times \vec{L} = \vec{U}$. See Figure 1.

There are some symbols and commands in Table I that control turtle position and orientation in space. String with brackets is introduced to delimit a branch. When the turtle finds the '[' in string, the current state of the turtle is pushed into a stack. In the case of finding ']', the current state of the turtle is popped from the stack.

The edge rewriting and node rewriting are two modes of operation for L-systems with turtle interpretation, using terminology borrowed from graph grammars.[2,17,18] In the case of edge rewriting, productions substitute figures for polygon edges, while in node rewriting, productions operate on polygon vertices. Both approaches rely on capturing the recursive structure of figures and relating it to a tiling of a plane.[1]

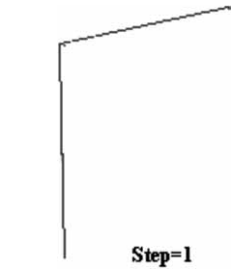
GENERATION OF THE FIGURES OF SOME FULLERENES BY USING L-SYSTEMS

As mentioned above, the rewriting rules should be discovered to obtain the self-similarities in shape. As the first step we found the form of L-system productions for drawing hypercube. Since, there is self-similarity on nodes of hypercube; the node rewriting system has been used.

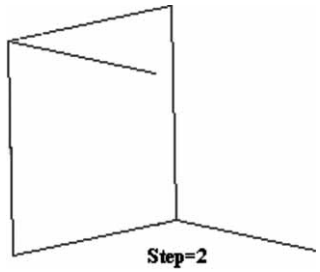
For an example the node rewriting L-systems for generating a simple cube is described. Suppose a node in a cube, three edges are joined to it. If we obtain L-systems that draw two edges connected to one node, the third edge can be produced by rewriting the L-system productions. Usage of stack is needed to draw a point with two branches. The following simple L-system is used to draw a cube.

Axiom: A

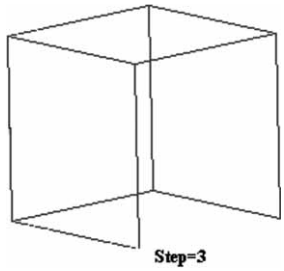
Rule: $A \to A\,[\&(-90)FA]\,[+(90)FA]$ (1)



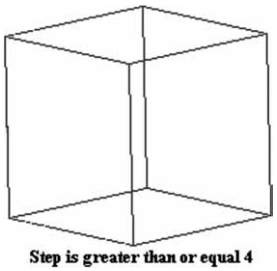
[&(–90)F]
[+(90)F]



[&(–90)F]
[+(90)F]
[&(–90)F
[&(–90)F]
[+(90)F]]
[+(90)F
[&(–90)F]
[+(90)F]]



[&(–90)F]
[+(90)F]
[&(–90)F
[&(–90)F]
[+(90)F]]
[+(90)F
[&(–90)F]
[+(90)F]]
[&(–90)F
[&(–90)F]
[+(90)F]
[&(90)F …



[&(–90)F]
[+(90)F]
[&(–90)F
[&(–90)F]
[+(90)F]]
[+(90)F
[&(–90)F]
[+(90)F]]
[&(–90)F
[&(–90)F]
[+(90)F]
[&(90)F
[&(–90)F]
[+(90)F]]…

Figure 2. The subsequent steps of cube generation and related strings.

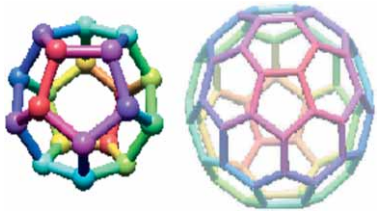


Figure 3. The figures of C_{20} and C_{60} fullerenes.

Using & (–90) to draw the first branch, the orientation of turtle head ($\vec{H}$) is rotated by axis ($\vec{L}$). After drawing an edge with F command, the symbol 'A' is put at the end of it for drawing other part of cube figure. Since before drawing the first branch the start point had been pushed in stack. Now, to draw a second branch of node, a pop from stack is occurred.

Using +90 to draw the second branch, the orientation of turtle head ($\vec{H}$) is rotated by axis ($\vec{U}$). The edge F and rewriting symbol 'A' are produced in the same way. At the end of forth step the shape of cube is generated. By more rewriting of L-system production rules the same cube is produced. The new edge is rewritten on earlier edges. Figure 2 shows the subsequent steps of the cube generation. The obtained strings by following L-System rules are displayed in Figure 2 too. As mentioned above, the turtle interpreter translates these strings to figures as shown in Figure 2.

The pentagon is a good start point to find the self--similarity if you concentrate on figures of C_{20} and C_{60} (Figure 3). Each pentagon has 5 branches except for its edges. Each branch shows a chemical bond and connects two pentagons to one another. These pentagons are lo-

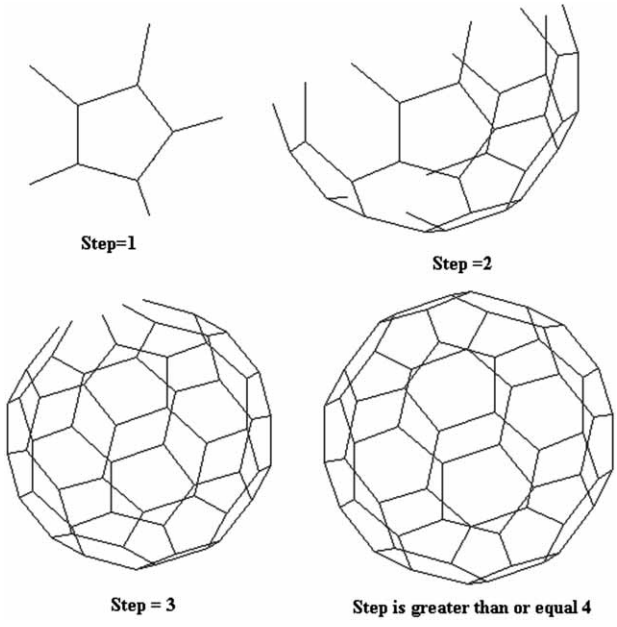


Figure 4. The subsequent steps of C_{60} generation.

TABLE I. The symbols and commands to control turtle in space

$F(\sigma)$	Move forward a step of length $\sigma > 0$. The position of the turtle changes to $(x', y' z')$, where $x' = x + \sigma_{\vec{H}x}$, $y' = y + \sigma_{\vec{H}y}$ and $z' = z + \sigma_{\vec{H}z}$. A line segment is drawn between points (x, y, z) and (x', y', z').
$f(\sigma)$	Move forward a step of length $\sigma > 0$ without drawing a line.
+ (α)	Rotate around $\vec{U}$ by an angle of α degrees. If α is positive, the turtle is turned to the left and if α is negative, the turn is to the right. If α is negative, the symbol '–' can be used also.
& (α)	Rotate around $\vec{L}$ by an angle of α degrees. If α is positive, the turtle is pitched down and if α is negative, the turtle is pitched up. The pitch up rotation can be represented by ^(α).
/ (α)	Rotate around $\vec{H}$ by an angle of α degrees. If α is positive, the turtle is rolled to the right and if α is negative, it is rolled to the left. The roll left can be represented by '\' symbol.
|	Turn around $\vec{U}$ by an angle 180.
[	Push the current state of the turtle onto stack. The information saved on the stack contains the turtle's position and orientation, and possibly other attributes such as the color and width of lines being drawn.
]	Pop a state from the stack and make it the current state of the turtle. No line is drawn, although in general the position of the turtle changes.

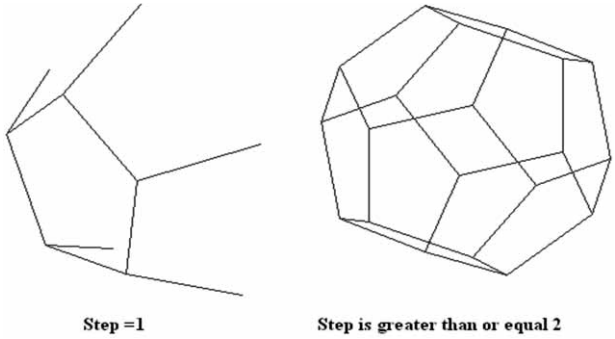


Figure 5. The subsequent steps of C_{20} generation.

cated in different planes in relation to each other. This means that the turtle must rotate by ($\vec{L}$) for drawing pentagons. The amount of pitch rotation varies in different fullerenes. On the other hand, for drawing edges of a pentagon the turtle should rotate by ($\vec{U}$) in a plane. The amount of rotation is depends on pentagon angles.

At first, consider a pentagon with 5 branches to draw C_{60}. In the end of each branch put the symbol 'A' to draw another pentagon. The production rule replaces A with pentagon rules that repeat this operation. Choosing 31.7175 for pitch rotation value, the following L-system rules can produce the C_{60} shape:

Axiom: A

Rule: A → |& (31.7175)[^ (31.7175) F]
+ (–126) F [+(54) ^ (31.7175) FA]
+ (–72) F [+(54) ^ (31.7175) FA]
+ (–72) F [+(54) ^ (31.7175) FA]
+ (–72) F [+(54) ^ (31.7175) FA]
+ (–72) F (2)

Figure 4 shows the subsequent steps of C_{60} generation. After 4 steps, figure C_{60} is generated and by more rule rewriting the same figure is obtained.

Choosing 58.283 for pitch rotation of turtle leads us to obtain the shape of C_{20}. Figure 5 shows figures that are generated by L-system rules in subsequent steps.

CONCLUSION

We introduced L-system for generating plausible drawing of fullerenes. L-system has been chosen because it can express drawing steps in a compact way and is parallel in nature. It is a good trend for visualizing of complex supramolecules. Discovering the self-similarities is essential for obtaining the rewriting rules in fullerenes. Here this was done for C_{20} and C_{60}. Using L-systems to generate figures of other nanostructures can be subject of the further works.

REFERENCES

1. P. Prusinkiewicz and A. Lindenmayer (with J. S. Hanan, F. D. Fracchia, D. R. Fowler, M. J. M. de Boer, and L. Mercer), *The Algorithmic Beauty of Plants*, Springer-Verlag, New York, 1990.
2. A. Lindenmayer, *An Introduction to Parallel Map Generating Systems*, in: H. Ehrig, M. Nagl, A. Rosenfeld, and G. Rozenberg (Eds.), *Graph Grammars and their Application to Computer Science,* Third International Workshop, Lecture Notes in Computer Science 291, Springer-Verlag, Berlin, 1987, pp. 27–40.
3. B. Plestenjak, T. Pisanski, and A. Graovac, *J. Chem. Inf. Comp. Sci*. **36** (1996) 825–828.
4. T. Pisanski, A. Zitnik, A. Graovac, and A. Baumgartner, *J. Chem. Inf. Comp. Sci.* **34** (1994) 1090–1093.

5. D. Babić, A. Gravoac, B. Mohar, and T. Pisanski, *Discrete Appl. Math*. **15** (1986) 11–24.
6. M. Kaufman, T. Pisanski, D. Lukman, B. Borštnik, and A. Graovac, *Chem. Phys. Lett*. **259** (1996) 420–424.
7. A. Lindenmayer, *J. Theor. Biol.* **18** (1968) 280–315.
8. G. T. Herman and G. Rozenberg. *Developmental Systems and Languages,* North-Holland, Amsterdam, 1975.
9. A. Salomaa, *Formal Languages,* Academic Press, New York, 1973.
10. G. Rozenberg and A. Salomaa, *The Mathematical Theory of L-systems,* Academic Press, New York, 1980.
11. A. L. Szilard and R. E. Quinton, *The Science Terrapin*, **4** (1979) 8–13.
12. B. B. Mandelbrot, *The Fractal Geometry of Nature*, W. H. Freeman, San Francisco, 1982.
13. R. Siromoney and K. G. Subramanian, *Space-filling Curves and Infinite Graphs*, in: H. Ehrig, M. Nagl, and G. Rozenberg (Eds.), *Graph Grammars and their Application to Computer Science;* Second International Workshop, Lecture Notes in Computer Science 153, Springer-Verlag, Berlin, 1983, pp. 380–391.
14. H. Abelson and A. A. diSessa, *Turtle Geometry*, M. I. T. Press, Cambridge, 1982.
15. P. Prusinkiewicz, *Graphical Applications of L-systems*, in: *Proceedings of Graphics Interface '86 — Vision Interface '86*, CIPS, 1986, pp. 247–253.
16. P. Prusinkiewicz, *Applications of L-systems to Computer Imagery*, in: H. Ehrig, M. Nagl, A. Rosenfeld, and G. Rozenberg, (Eds.), *Graph Grammars and their Application to Computer Science; Third International Workshop*, Lecture Notes in Computer Science 291, Springer-Verlag, Berlin, 1987, pp. 534–548.
17. A. Habel and H.-J. Kreowski, *On Context-free Graph Languages Generated by Edge Replacement*, in: H. Ehrig, M. Nagl, and G. Rozenberg (Eds.), *Graph Grammars and their Application to Computer Science, Second International Workshop*, Lecture Notes in Computer Science 153, Springer-Verlag, Berlin, 1983, pp. 143–158.
18. A. Habel and H.-J. Kreowski, *May We Introduce to You: Hyperedge Replacement*, in: H. Ehrig, M. Nagl, G. Rozenberg, and A. Posenfeld (Eds.), *Graph Grammars and their Application to Computer Science,* Third International Workshop, Lecture Notes in Computer Science 291, Springer-Verlag, Berlin, 1987, pp. 15–26.

SAŽETAK

Korištenje L-sustava u dobijanju crteža nekih fullerena

Mehdi Vahidipour, Hosein Sabaghian-Bidgoli i Gholamreza Vakili-Nezhaad

Aristid Lindenmayer je 1968. god. uveo biološki motiviran formalizam, tzv. L-sustave, za simulaciju razvoja višestaničnih organizama. Primjene idu od modeliranja i vizualizacije biljaka na raznim razinama apstrakcije i za razne potrebe pa do geometrijskog modeliranja krivulja i površina. U ovom radu se, bez pozivanja na koordinate ugljikovih atoma, L-sustav koristi za dobivanje NiceGraph crteža fullerena C_{20} i C_{60}. L-sustav izražava korake u risanju na kompaktan način i paralelan je po prirodi. Očekuje se da bi bio prikladan i u vizualizaciji kompleksnih supramolekula.