A Look at the Visual Modeling of Plants Using L-Systems

Przemyslaw Prusinkiewicz

Department of Computer Science
2500 University Drive N.W.
Calgary, Alberta T2N 1N4
e-mail: pwp@cpsc.ucalgary.ca

Abstract. The kind invitation to present a paper at the German Conference on Bioinformatics prompted me to look back at the history of plant modeling using L-systems, relate the results obtained by my research group to the growing array of other contributions, and present an updated guide to the literature in the field.

1 Introduction

In 1968, Aristid Lindenmayer introduced a formalism for simulating the development of multicellular organisms, later named L-systems [59]. It was originally described in terms of linear or branching chains of finite automata, but its subsequent reformulation in terms of rewriting systems [60] proved more elegant. The close relationship between L-systems, abstract automata, and formal languages attracted the interest of computer scientists, who vigorously developed the mathematical theory of L-systems [40, 104, 106] (for an account of the early history of L-systems see also [67, 107]). This progress was followed by applications of L-systems to the modeling of plants, initiated by the development of the first simulation program based on L-systems called CELIA (an acronym for CEllular Linear Iterative Array simulator) by Baker, Herman, and Liu [3, 4, 39].

In 1984, Smith [112] introduced state-of-the-art computer graphics for visualizing a class of abstract branching structures discovered by Hogeweg and Hesper [43]. The beauty of Smith's images and the life-like appearance of his developmental simulations inspired me to design and implement my own simulation program, called `pfg` (an acronym for plant and fractal generator, C code listing included in [90]). The first results obtained using `pfg` were focused on the visualization of fractals and abstract branching structures [83, 84]. This work attracted the interest of Professor Lindenmayer and, along with my graduate student Jim Hanan, we collaborated on the application of L-systems to the realistic modeling of structures and processes found in real plants [90, 96]. The results obtained by 1990 (with crucial contributions by de Boer, Fowler, Fracchia, and Mercer) were collected in our book [95].

Many new results have been obtained since then. The purpose of the present paper is to survey current lines of research, and provide an updated guide to the literature on plant modeling using L-systems. For previous guides of a similar

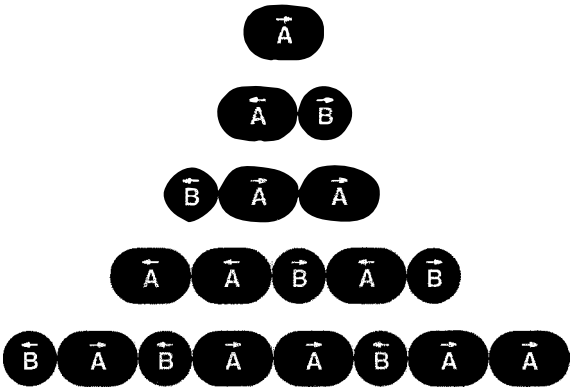

Fig. 1. Visualization of the L-system model of *Anabaena catenula*. The model captures the arrangement of shorter and longer cells in a vegetative segment of the filament.

nature see [68] and [90, Chapter 7]. General surveys of the modeling of plant architectures are presented in [16, 102]. For a recent tutorial introduction to visual modeling using L-systems see [86, 87].

2 What are L-systems?

In order to retrace the advancement of modeling techniques based on L-systems, let us first identify the main features of the original formalism. From a mathematical point of view, L-systems are parallel rewriting systems, operating on strings of symbols that may represent individual cells [59] or larger components (modules) of a growing organism [23]. One of the simplest biologically relevant examples of L-systems is a model of the filamentous blue-green bacteria *Anabaena catenula* [62, 63, 65]. The model describes the development of a so-called vegetative segment of *Anabaena* using rewriting rules (also called productions) operating on two types of cells, A and B. Each cell can have one of two different polarities indicated by superscript arrows: $\overrightarrow{A}$, $\overleftarrow{A}$, $\overrightarrow{B}$, and $\overleftarrow{B}$. In given time intervals, cells B elongate and change their state to A, while cells A divide, producing a cell A and a cell B. This process is characterized by the following rules:

$$\overrightarrow{A} \longrightarrow \overleftarrow{A}\overrightarrow{B} \qquad \overleftarrow{A} \longrightarrow \overleftarrow{B}\overrightarrow{A} \qquad \overrightarrow{B} \longrightarrow \overrightarrow{A} \qquad \overleftarrow{B} \longrightarrow \overleftarrow{A}$$

The L-system model integrates these local rules into a global description of a vegetative segment. The segment's development is simulated by an L-system derivation, in which productions are applied in parallel to all cells in the filament (Figure 1).

In spite of its simplicity, this model reveals several key features of the basic L-system formalism.

(a) The model is inherently *dynamic*: it describes the development of a structure over time. In the words of d'Arcy Thompson, the form is viewed as "an event in space-time, and not merely a configuration in space" [116].

(b) The model is *discrete* in three senses: "the state transformations are defined on discrete subunits (cells); each subunit may be present in one of a finite set of states; and the transformations are performed in discrete time steps" (quoted from [66]).

(c) The cells are arranged in a linear *filament*. The original formalism also makes it possible to describe branching structures [59].

(d) The model describes *topology*, in this case, the ordering of cells in the filament. There is no information regarding the geometry, that is the actual shape and size of the cells, and their position in space. In Figure 1, it was arbitrarily decided that the cells would be represented as circles and rectangles with rounded corners, arranged along a straight line.

(e) The topology of the organism changes as a result of cell division. No mechanism exists, however, to rearrange a set of existing cells. Consequently, L-systems are more suitable to model *plants* (in which cells are tightly cemented together) than animals (in which cells can move with respect to each other [114, page 2]).

(f) The model represents the organism as a *closed cybernetic system*, which controls its development autonomously, without interacting with the environment.

(g) The model describes *global development* of an organism in terms of *local rules*. In the *Anabaena* example, the state of each cell fully determines its fate in the next step. We say that these rules are context-free, and represent control of development by *lineage* [63]. The original L-system formalism also makes it possible to use context-sensitive rules, which capture interactions between adjacent elements of the developing structure.

Although these basic features are sufficient to create many models of linear and branching structures, advanced applications require extensions and modifications of L-systems. In the following sections we discuss three of them: the inclusion of continuous attributes, graphical interpretation of the models, and incorporation of external (environmental) influences on the development.

3 Continuous extension of L-systems

An essential component of the mathematical theory of L-systems is their discrete character. Nevertheless, in applications of L-systems to modeling and simulation, this can become a limiting factor. The postulate that each module may assume only a finite number of states was the first to be reconsidered. As early as 1972, the simulation program CELIA allowed for the association of numerical attributes and sets of attributes of different types with L-system symbols [4, 39]. The idea of "adding continuous components to L-systems" was subsequently discussed by Lindenmayer [61]. An analysis of the error related to the discretization

of continuous variables, such as the concentrations of substances in the cells, was given by Baker and Herman [4] (see also [40]).

Impetus for further development of L-systems with parameters stemmed from the requirements of model visualization. Parameters were needed to specify the lengths of lines and the magnitudes of branching angles in the models (Section 4). Formal definitions of L-systems with parameters were given by Chien and Jürgensen [9, 10], and Hanan and myself [34, 91]. A simulation program `cpfg` (a continuous-parameter extension of `pfg`), implemented by Hanan and subsequently extended by James [50], and Hammel and Měch [89], is available over the Internet [82]. The use of parametric L-systems is the key advance in the modeling techniques presented in the book [95] over its predecessor [90].

The needs of model visualization, in particular for the animation of developmental processes, motivated another departure from the discrete characteristics of L-systems: the introduction of continuous time. An early attempt to specify continuous-time processes using L-systems was included in [95]. It was limited to context-free models, thus did not capture possible interactions between coexisting modules during development. This limitation was overcome in the next formalism, *differential L-systems*, introduced by Hammel, Mjolsness, and myself [88], and further explored by Hammel [30]. This is a combined discrete-continuous model of development, in which modules are created and cease to exist in discrete events captured by productions, but develop in a continuous fashion described by differential equations. Arguments to these equations may be provided by the neighboring modules, thus an exchange of information between modules can be expressed. To illustrate the formalism of differential L-systems, we created several animations of plant development [85].

One can contemplate whether the remaining discrete aspect of L-systems — partitioning the modeled system into discrete units — should also be relinquished in some applications. The resulting notion, which could be termed *partial differential L-systems* (Mjolsness, personal communication) would treat a developing organism as a continuous, possibly growing medium with a linear or branching topology. Such an approach was proposed by de Koster and Lindenmayer [13] as a possible model for a growing filament. Continuous media were also considered by Hammel and myself [30, 31] in an L-system restatement of reaction-diffusion models for pattern formation in sea shells. These models were originally formulated by Meinhardt and Klinger [72, 73] in terms of partial differential equations. A formal definition of partial differential L-systems remains an open problem.

4 Graphical interpretation of L-systems

The first algorithms for visualizing branching structures generated by L-systems were proposed in 1974 by Frijters and Lindenmayer [23] and Hogeweg and Hesper [43]. The geometric aspects of the modeled structures were defined using a set of drawing rules *external* to the L-systems under consideration, acting globally on all components of the modeled structure. This global definition made some structures impossible to specify. For example, the rule stating that branches should

be issued in alternating directions, first to the left, then to the right, did not allow modeling of structures with two consecutive branches oriented the same way. In 1979, Szilard and Quinton observed that L-systems could be applied to generate a variety of intricate geometric patterns if graphical interpretation was associated with specific symbols in the generated strings [115]. According to one technique, the L-system symbols represented lines (vectors) running in predefined directions: left, right, up and down. Thus, the strings defined the images according to the *chain coding* mechanism [19]. In another approach, directions were specified relative to the previous lines. Pursuing this latter route, I proposed [83] to consider L-system symbols as commands controlling a LOGO-style *turtle* [1]: move forward, turn to the left, and turn to the right. L-systems with turtle interpretation made it possible to generate many fractal curves. Moreover, saving and restoring the turtle's position on a pushdown stack allowed the creation of plant-like structures with branches.

Several extensions to turtle interpretation were introduced by Hanan, Hammel, Měch, and myself. They included an extension of turtle interpretation to three dimensions [84], the possibility of incorporating predefined surfaces to represent organs such as leaves and flower petals [33, 84], and the addition of numerical parameters needed to control quantitative attributes of model components [34, 91]. *Developmental surfaces* [34] (see also [89]), made it possible to simulate changes of organ shape in animations of plant development [85, 88]. Other methods for specifying the shape of plant organs included planar surfaces bound by sequences of turtle steps [33, 96], and implicit contours built around branching skeleton structures [32]. Generalized cylinders with various cross sections were recently incorporated into the framework of L-systems to model smoothly curving branches [89].

A further formalization of turtle interpretation was proposed by Kurth [55]. In particular, his work improved the method for manipulating the turtle's attributes using parametric L-systems.

The use of turtle interpretation is convenient in a biological context, because it makes it easy to express branching angles. However, absolute directions also play a significant role in plant development. In the words of Dawkins [12, page 128], "the world usually imposes a significant difference between up and down." For example, branches often show a tendency to grow upwards, and roots to grow downwards. Under the general term of *tropisms*, these phenomena have been captured by biasing turtle orientation in a predefined direction [95, 96]. Further research is needed, however, to fully integrate tropisms with turtle interpretation.

5 Incorporation of environmental factors

Plants modeled using the original formalism of L-systems were treated as closed cybernetic systems, developing without interaction with the environment. In reality, however, interaction with the environment plays a major role in the development of plant and plant communities, and cannot be neglected in practical models with predictive value. In the first step towards the inclusion of envi-

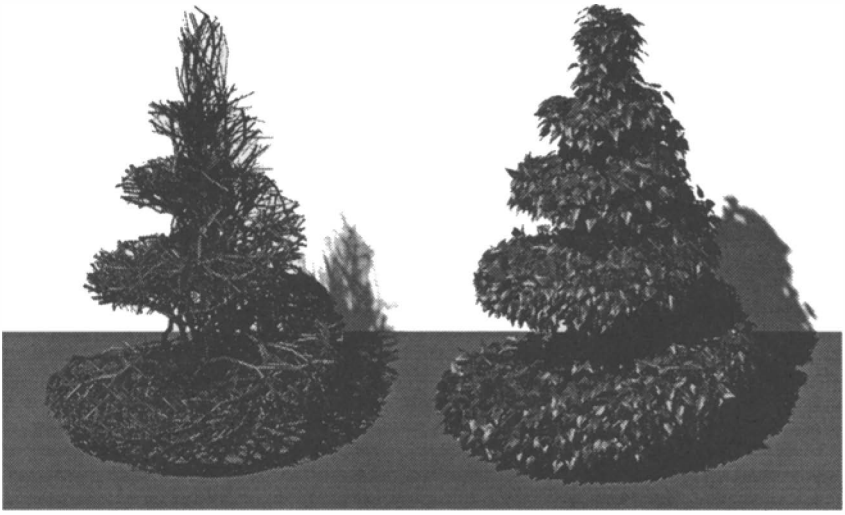

Fig. 2. An L-system model of plants affected by pruning. From [93].

ronmental factors, Rozenberg defined *table L-systems*, which allow changes to the production set from one derivation step to another [103] (see also [40, 106]). Table L-systems were applied, for example, to capture the switch from the production of leaves to the production of flowers by the apex of a flowering plant, due to a change in day length [20, 22, 23].

Table L-systems can only capture the impact of global environmental characteristics on plant development. Many phenomena depend, however, on local aspects of the environment. James, Měch, and myself introduced *environmentally-sensitive L-systems* to capture situations where the environment affects the plant, but the reciprocal influence of the plant on the environment can be ignored [93]. This formalism was illustrated using examples of plant responses to pruning (Figure 2). A related approach was applied by Fournier to model the effect of the local temperature of organs on maize development [17].

Plants may also interact with the environment in a feedback loop that includes information flow to and from the environment. Examples include competition for space between individual plants (ramets) in a clonal plant, competition for light between branches of a tree (where the upper branches change the amount of light available to the lower branches), and competition between roots for water in the soil. To express such phenomena, Měch and myself introduced the formalism of *open L-systems* [75] (Figure 3). It extends the L-system alphabet with *communication symbols*, which can exchange parameter values with the environment. Thus, a model of a developmental process consists of two components: a plant model expressed using an L-system, and a program simulating the relevant aspects of the environment.

A different organization of the modeling software was proposed by Kurth [55,

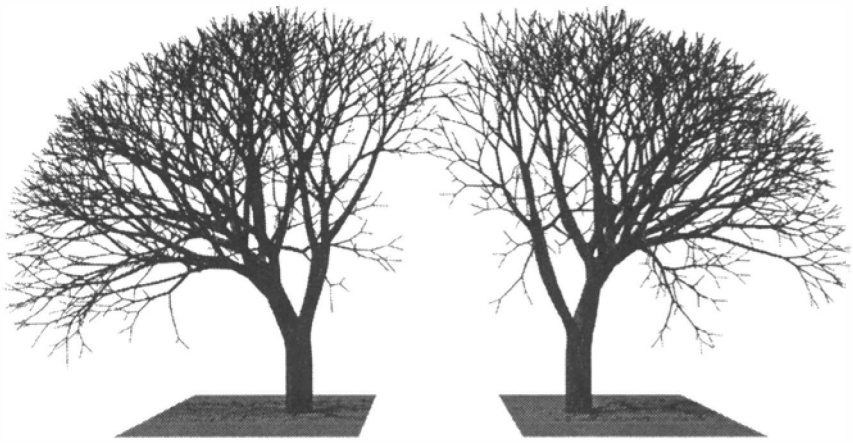

Fig. 3. A model of deciduous trees competing for light. The trees have been and moved apart after they grew to reveal adaptation of crown geometry to the presence of the neighbor tree. Leaves are not shown to expose the branching structures. From [75].

57], who incorporated predefined functions that return environmental information directly into the simulation program. This approach requires the simulator to be recompiled each time a new environmental function is added. A technique for incorporating environmental factors into L-systems has also been presented by Vaario, Ogata, and Shimohara [119]. An interesting aspect of this latter work is the merging of two fundamental models of morphogenesis: L-systems and diffusion-limited aggregation models [120]. All of these extensions require the location of different modules in 3D space to be known; thus, they have been formulated for L-systems with turtle interpretation, rather than L-systems in general.

A large amount of fundamental work on modeling developmental processes in an environmental context has been carried out outside the L-system framework, especially by Blaise [7] and Kaandorp [52]. Given the environmentally-sensitive and open L-system extensions, a link between their work and L-systems could now be established (see Section 9).

6 L-systems as programming languages

As Chomsky grammars are the foundation for many common programming languages, the formalism of L-systems is the basis on which programming languages for the modeling and simulation of plants have been and are being built. These languages offer the users of simulation programs the capability of expressing models easily, without the burden of constructing them "from scratch" in a general-purpose language, and without the limitations of predefined, "hard-coded" models, where only numerical parameter values can be easily changed.

This advantage is particularly relevant to computer-assisted biological research, where a convenient mechanism for specifying, modifying, and experimenting with all aspects of the models is highly important.

To outline the gradual evolution of L-systems from theoretical concept to programming languages, let us first consider the simulation programs `pfg` and `cpfg` (Sections 1 and 3). The first version of these programs adopted a straightforward syntax, mimicking the mathematical notation for production specification [83, 90]. Modeling experience indicated that further constructs were needed to increase the expressive power and the flexibility of the language. The introduction of numerically-valued parameters (Section 3) was the first major improvement. This concept was extended with local variables (limited to individual productions) and global variables (shared between productions) [34, 92]. Further extensions included the incorporation of standard programming constructs such as predefined mathematical functions [50], arrays, and flow control statements [89]. Sub-L-systems [34], based on the idea of subroutines, made it possible to partition complex models into a hierarchy of components, which can be defined independently.

Several extensions have been inspired, or can be related, to the notions of L-system theory. Programming constructs for stochastic L-systems [84, 95] closely follow their formal definitions [15, 77, 121]. The "cut symbol", introduced to simulate the shedding of organs such as leaves, petals, or entire branches [34, 95], has its counterpart in L-systems with fragmentation [105, 108]. The idea of fragmentation is also related to the modeling of collections of objects, such as clonal plants that become separated during vegetative propagation [87]. L-systems with homomorphisms [89] make use of the homomorphic transformations of generated strings [76, 106] to separate the logic of the models from the details of their graphical interpretation. This separation makes complex models clearer and better structured [97].

The link between L-system theory and practical programming constructs is rarely straightforward. Many theoretical notions have been introduced as mechanisms for defining new classes of languages on the basis of L-systems. This motivation, rooted in formal language theory, often leads to results that do not meet the needs of biological modeling. For example, theoretical concepts that have been established for non-bracketed, context-free L-systems without parameters must be generalized to become useful in programming languages.

Extensions analogous to those outlined above using the example of `pfg` and `cpfg` can also be found in other implementations of languages based on L-systems. A variant of the parameter-passing mechanism for parametric L-systems was proposed by Borovikov [8] and implemented in the commercial program World Builder [2]. It makes it possible to handle modules with large numbers of parameters in a concise manner. Constructs borrowed from standard programming languages have been incorporated in the modeling systems ELSYS by Goel and Rozehnal [27], and GROGRA by Kurth [55]. Kurth has also implemented homomorphic transformations of strings (under the name of two-phase growth grammars). An idea similar to fragmentation has been applied to the animation

of multiple interacting objects by Noser and Thalmann [78, 79]. One of their examples, particularly interesting from a biological perspective, is the simulation of butterflies flying in a flower field.

Unfortunately, in spite of the practical importance of programming languages based on L-systems, they have not yet been extensively studied from the perspective of programming language theory. A notable exception is the work by Oritz, Pinter, and Pinter [80, 81], where L-systems are considered in the context of programming massively parallel computers such as the Connection Machine.

Plant modeling can also be regarded in the general framework of simulation theory. This point of view was first adopted by Hogeweg, who considered L-system derivations as discrete-event simulations, and used SIMULA to implement the models [41, 42]. Pursuing a similar approach, Hammel [30] applied a combined discrete-continuous simulation extension of SIMULA called DISCO [38] to implement differential L-systems (Section 3). Both implementations made it possible to relinquish the assumption of a strictly synchronous operation of L-systems, which may be unrealistic from a biological perspective. In addition, they took advantage of the object-oriented programming environment provided by SIMULA to conveniently express the models. Object-oriented extensions of L-systems have also been proposed using the framework of C++ by Borovikov [2, 8] and Guzy [29].

The development of programming languages and environments based on L-systems is an active research domain. As the understanding of modeling using L-systems grows, we may expect that new, more systematically designed languages will emerge. They will take full advantage of L-system theory, and combine useful constructs found at present in separate languages and programs.

7 Applications to plant modeling

In general, plant models can be divided into *mechanistic* (causal) and *empirical* (descriptive). The purpose of mechanistic models is to gain an understanding of plant development in terms of the interactions between the component modules and processes. Thus, "mechanistic modeling follows the traditional reductionist method that has been so very successful in the physical sciences" [117]. In contrast, empirical models reproduce the morphology of the described plants without reenacting the control mechanisms. We will discuss the mechanistic models first.

L-systems were introduced as a formalism for modeling and simulating the development of simple multicellular organisms, such as filamentous bacteria and algae [59]. In this spirit, Tunbridge and Jones recently applied a context-sensitive parametric L-system to model the development of fungus *Aspergillus nidulans* [118]. In 1974, Frijters and Lindenmayer proposed L-systems for modeling the structures found in higher plants, in particular compound inflorescences [20, 21, 22, 23]. From a biological perspective, the main purpose of their studies was to present plausible explanations of flowering sequences and differences in relative branch sizes in the studied plants. Frijters and Linden-

mayer observed [24] that simple control of development by lineage, expressed by non-parametric context-free L-systems, could not capture basipetal flowering sequences (with the flowering zone progressing from the top of the plant downwards) and acrotonic patterns of branch development (with the largest branches situated near the top of the plant). A formal analysis supporting this observation was presented later by Lück, Lück, and Bakkali [69], and Kari and myself [94]. Overcoming this limitation, Janssen and Lindenmayer showed [51, 64] that acrotonic flowering patterns and basipetal flowering sequences *could* be reproduced assuming control of development by hormones that flow through the developing structure and trigger developmental events. This work formed the basis for subsequent realistic modeling and visualization of herbaceous plants by Lindenmayer, Hanan, and myself [90, 95, 96].

The ease of describing interactive control mechanisms using context-sensitive L-systems is one of the most appealing features of the L-system formalism. In addition to the work cited above, examples include models of trees affected by pruning, in which signals initiate the development of dormant lateral buds after the apices of the main branches have been removed (Section 5). In a less typical application, inspired by Room (personal communication), a signal represents an insect that feeds on a plant [86, 87]. The incorporation of parameters makes it possible to quantify concentrations of substances flowing in a growing structure, such as water, minerals, or products of photosynthesis. Consequently, L-systems can be applied to express the class of physiologically-based *resource-allocation* models. A simple example of a plant model including the flow and partitioning of resources between the shoot and the root is presented in [86, 87]. The flow of resources (products of photosynthesis) is also a part of the models of trees competing for light, proposed by Měch and myself [75].

Apart from the work carried out using `cpfg`, resource allocation models have been implemented by Kurth using the program GROGRA [54]. In this case, the available resources (carbon compounds) are computed using special-purpose extensions incorporated into a basically context-free simulation language, rather than generic mechanisms of information transfer provided by context-sensitive productions.

L-systems have also been used to construct descriptive developmental models. A significant body of work devoted to modeling algae has been carried out by Corbit and Garbary [11, 25], and Morelli, Schneider, Walde, and Akstin [74, 109, 110]. The latter group suggested an interesting if hypothetical link between branching architectures, their L-system models, and the genetic makeup of the studied species. It is exemplified by the following statement: "If the character strings necessary to code our graphical images corresponded in some way to the information represented in the genetic code of *Dipterosiphonia* species, then only minor genetic changes would be necessary to account for speciation where branching pattern was the main defining species characteristic." [110].

In contrast to the relatively simple models of algae, empirical models of higher plants rely on large amounts of quantitative data and statistical analysis of plant morphology. Examples include models of the Japanese cypress scale leaves [77],

green ash shoots [98], young green ash trees [99], Norway spruce trees [58], cotton plants [100], bean plants [35], maize shoots [17] and maize root systems [111]. A novel use of L-systems has been proposed by Battjes and Bachmann [6], who related parameter values in the L-system models to genetic variation between modeled plants (four species of *Microseris*, a herbaceous plant in the aster family).

The large amount of observational data needed to construct models raises a number of practical problems. What features of plant morphology should be measured? What devices should be used to perform the measurements? How should the acquired data be represented in a database for easy access, processing, and incorporation into the final models? An overview of these problems in the context of L-system modeling has been presented by Remphrey and myself [99]; a detailed case study of empirical model construction is given in our joint paper with Davidson and Hammel [98]. The process of data acquisition has been presented in detail by Hanan and Room [37], and emphasized in our tutorial paper on the applications of L-systems to plant modeling [101]. The underlying digitizing software is available through the Internet [36]. Theoretical aspects of model construction according to quantitative data have been described by Godin *et al.* [26]. Although this paper is not presented in the framework of L-systems, the results can be easily adapted.

8 L-systems and evolution

On an abstract level, the L-system productions can be viewed as "genes" that control plant development. Consequently, L-system models can be subject to a cyclic process of artificial evolution, in which changes to the rules are introduced, the resulting models are evaluated, and the L-systems producing the best models (for a given criterion) are selected for the next iteration of changes and evaluation. An early pursuit of this concept was presented by MacKenzie and myself [70, 71], in a work extending unpublished results by Smith [113]. The key idea was to apply genetic algorithms (for example, see [28]) to introduce variations in the class of L-systems being explored. Experiments included several selection mechanisms, such as the fractal dimension and the amount of light captured by the resulting structures. Recently, the concept of evolving L-system models has been extensively studied by Jacob [44]–[49]. Evolving models of abstract structures, motivated by plants, were also proposed by Kim [53]. Their postulated relationship to L-system models requires further analysis.

9 Concluding remarks

Applications of L-systems to programming, simulation, and visualization extend beyond the modeling of plants. They also include generation of fractals, tilings, and other geometric patterns, graphical modeling and animation of objects other than plants, and extensions of L-systems for the modeling of cellular layers and volumetric structures. These topics have not been included in the present survey.

Advancements in the modeling of plants using L-systems have been largely motivated by the desire to expand the range of phenomena that can be formally described, simulated, and studied. Now that this range is quite extensive, questions regarding the relationship between L-systems and other models of plant architecture emerge. It appears that the use of a special-purpose modeling language is the most distinctive feature of the L-system-based approach. It makes it easy to specify models as an input to general-purpose simulation programs or as a part of a model description in publications. In contrast, models expressed in general-purpose programming languages may require multi-page program listings. The essence of the models, however, is often similar in spite of different software implementations. Indeed, the modeling power of L-systems has been repetitively evaluated by reimplementing various models constructed originally within different frameworks [75, 87, 95]. In this context, Françon [18] and Kurth [56, 57] observed an interesting convergence between models expressed using L-systems and the large body of models developed at the Atelier de Modélisation de l'Architecture des Plantes AMAP, CIRAD, Montpellier, France (for a recent description of the work at AMAP see [5, 14]). A further comparison of different approaches to the modeling of plant architecture would be an interesting research project in itself.

Acknowledgements

Many of the papers included in this review were sent to me directly by the authors, or brought to my attention by my colleagues. I would like to express my gratitude for this input, without which important references may have been missed. I would also like to thank Jim Hanan, Hugh McEvoy, and Lynn Mercer for helpful comments on the manuscript. This work has been supported in part by research and equipment grants from the Natural Sciences and Engineering Research Council of Canada, and by the Killam Resident Fellowship held at the University of Calgary in the Fall of 1996. This support is gratefully acknowledged.

References

1. H. Abelson and A. A. diSessa. *Turtle geometry.* M.I.T. Press, Cambridge, 1982.
2. Forcade & Associates, editor. *AnimatTek's World Builder: User guide, tutorials, and reference.* Digimation, St. Rose, 1996.
3. R. Baker and G. T. Herman. CELIA — a cellular linear iterative array simulator. In *Proceedings of the Fourth Conference on Applications of Simulation* (9–11 December 1970), pages 64–73, 1970.
4. R. Baker and G. T. Herman. Simulation of organisms using a developmental model, parts I and II. *International Journal of Bio-Medical Computing*, 3:201–215 and 251–267, 1972.
5. D. Barthélémy, F. Blaise, T. Fourcaud, and E. Nicolini. Modélisation et simulation de l'architecture des arbres: Bilan et perspectives. *Revue Forestière Française*, XLVII:71–96, 1995.

6. J. Battjes and K. Bachmann. Computer modeling of quantitative morphological changes in *Microseris*. Manuscript, Institute of Plant Genetics and Crop Plant Research. Gatersleben, Germany, 1996.
7. F. Blaise. *Simulation du parallélisme dans la croissance des plantes et applications*. PhD thesis, Université Louis Pasteur, Strasbourg, July 1991.
8. I. A. Borovikov. L-systems with inheritance: an object-oriented extension of L-systems. *ACM SIGPLAN Notices*, 30(5):43–60, 1995.
9. T. W. Chien. Graphical interpretation of the biological development modelled by VDOL-systems. Master's thesis, University of Western Ontario, 1989.
10. T. W. Chien and H. Jürgensen. Parameterized L systems for modelling: Potential and limitations. In G. Rozenberg and A. Salomaa, editors, *Lindenmayer systems: Impacts on theoretical computer science, computer graphics, and developmental biology*, pages 213–229. Springer-Verlag, Berlin, 1992.
11. J. D. Corbit and D. J. Garbary. Computer simulation of the morphology and development of several species of seaweed using Lindenmayer systems. *Computers and Graphics*, 17(1):85–88, 1993.
12. R. Dawkins. *Climbing Mount Improbable*. W. W. Norton and Co., New York, 1996.
13. C. G. de Koster and A. Lindenmayer. Discrete and continuous models for heterocyst differentiation in growing filaments of blue-green bacteria. *Acta Biotheoretica*, 36:249–273, 1987.
14. P. de Reffye, F. Houllier, F. Blaise, D. Barthélémy, J. Dauzat, and D. Auclair. A model simulating above- and below-ground tree architecture with agroforestry applications. *Agroforestry Systems*, 30:175–197, 1995.
15. P. Eichhorst and W. J. Savitch. Growth functions of stochastic Lindenmayer systems. *Information and Control*, 45:217–228, 1980.
16. J. B. Fisher. How predictive are computer simulations of tree architecture. *International Journal of Plant Sciences*, 153 (Suppl.):137–146, 1992.
17. C. Fournier. Introduction des réponses écophysiologiques à la température dans un modèle de plante à la base de L-Systèmes. Master's thesis, Institut National Agronomique Paris-Grignon, 1995.
18. J. Françon. Sur la modélisation de l'architecture et du développement des végétaux. In C. Edelin, editor, *L'Arbre. Biologie et Développement*. Naturalia Monspeliensia, 1991. N^{o} hors série.
19. H. Freeman. On encoding arbitrary geometric configurations. *IRE Trans. Electronic. Computers*, 10:260–268, 1961.
20. D. Frijters. An automata-theoretical model of the vegetative and flowering development of *Hieracium murorum* L. *Biological Cybernetics*, 24:1–13, 1976.
21. D. Frijters. Mechanisms of developmental integration of *Aster novae-angliae* L. and *Hieracium murorum* L. *Annals of Botany*, 42:561–575, 1978.
22. D. Frijters. Principles of simulation of inflorescence development. *Annals of Botany*, 42:549–560, 1978.
23. D. Frijters and A. Lindenmayer. A model for the growth and flowering of *Aster novae-angliae* on the basis of table (1,0)L-systems. In G. Rozenberg and A. Salomaa, editors, *L Systems*, Lecture Notes in Computer Science 15, pages 24–52. Springer-Verlag, Berlin, 1974.
24. D. Frijters and A. Lindenmayer. Developmental descriptions of branching patterns with paracladial relationships. In A. Lindenmayer and G. Rozenberg, editors, *Automata, languages, development*, pages 57–73. North-Holland, Amsterdam, 1976.

25. D. J. Garbary and J. D. Corbit. Lindenmayer-systems as models of red algal morphology and development. *Progress in Phycological Research*, 8:143–177, 1992.
26. C. Godin, Y. Guédon, E. Costes, and Y. Caraglio. Measuring and analysing plants with the *AMAP* software. In M. T. Michalewicz, editor, *Plants to ecosystems. Advances in computational life sciences I.* CSIRO Publishing, Melbourne, 1997. To appear.
27. N. Goel and I. Rozehnal. A high-level language for L-systems and its applications. In G. Rozenberg and A. Salomaa, editors, *Lindenmayer systems: Impacts on theoretical computer science, computer graphics, and developmental biology*, pages 321–251. Springer-Verlag, Berlin, 1992.
28. D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley, 1989.
29. M. R. Guzy. A morphological-mechanistic plant model formalized in an object-oriented parametric L-system. Manuscript, USDA-ARS Salinity Laboratory, Riverside, 1995.
30. M. Hammel. *Differential L-systems and their application to the simulation and visualization of plant development.* PhD thesis, University of Calgary, June 1996.
31. M. Hammel and P. Prusinkiewicz. Visualization of developmental processes by extrusion in space-time. In *Proceedings of Graphics Interface '96*, pages 246–258, 1996.
32. M. S. Hammel, P. Prusinkiewicz, and B. Wyvill. Modelling compound leaves using implicit contours. In T. L. Kunii, editor, *Visual computing – Integrating computer graphics with computer vision*, pages 199–212. Springer Verlag, Tokyo, 1992.
33. J. S. Hanan. PLANTWORKS: A software system for realistic plant modelling. Master's thesis, University of Regina, November 1988.
34. J. S. Hanan. *Parametric L-systems and their application to the modelling and visualization of plants.* PhD thesis, University of Regina, June 1992.
35. J. S. Hanan. Virtual plants — Integrating architectural and physiological plant models. In P. Binning, H. Bridgman, and B. Williams, editors, *Proceedings of ModSim 95*, volume 1, pages 44–50, Perth, 1995. The Modelling and Simulation Society of Australia.
36. J. S. Hanan and P. M. Room. Virtual plants. A hypertext document and digitizing software distribution, Cooperative Research Centre for Tropical Pest Management, Brisbane, Austalia, 1996. Available at: `http://www.ctpm.uq.edu.au/Programs/IPI/ipivp.html`.
37. J. S. Hanan and P. M. Room. Practical aspects of virtual plant research. In M. T. Michalewicz, editor, *Plants to ecosystems. Advances in computational life sciences I.* CSIRO Publishing, Melbourne, 1997. To appear.
38. K. Helsgaun. DISCO — A SIMULA-based language for combined continuous and discrete simulation. *Simulation*, 35(1):1–12, 1980.
39. G. T. Herman and W. H. Liu. The daughter of CELIA, the French flag, and the firing squad. *Simulation*, 21:33–41, 1973.
40. G. T. Herman and G. Rozenberg. *Developmental systems and languages.* North-Holland, Amsterdam, 1975.
41. P. Hogeweg. Simulating the growth of cellular forms. *Simulation*, pages 90–96, September 1978.
42. P. Hogeweg. Locally synchronized developmental systems: Conceptual advantages of discrete event formalism. *International Journal of General Systems*, 6:57–73, 1980.

43. P. Hogeweg and B. Hesper. A model study on biomorphological description. *Pattern Recognition*, 6:165–179, 1974.
44. C. Jacob. Genetic L-system programming. In *Parallel Problem Solving from Nature PPSN III*, Lecture Notes in Computer Science 866, pages 334–343, Berlin, 1994. Springer-Verlag.
45. C. Jacob. Genetic L-system programming: Breeding and evolving artificial flowers with *Mathematica*. In *Proceeding of the First International Mathematica Symposium '95*, pages 215–222, Southampton, UK, 1995. Computational Mechanics Publications.
46. C. Jacob. MathEvolvica*: Simulierte Evolution von Entwicklungsprogrammen der Natur*. PhD thesis, University of Erlangen, May 1995.
47. C. Jacob. Evolution programs evolved. In H.-M. Voigt, W. Ebeling, I. Rechenberg, and H.-P. Schwefel, editors, *Parallel Problem Solving from Nature PPSN-IV*, pages 42–51, Berlin, 1996. Springer-Verlag.
48. C. Jacob. Evolving evolution programs: Genetic programming and L-systems. In *Proceedings of the First Annual Conference on Genetic Programming*, pages 107–115, Cambridge, 1996. M.I.T. Press.
49. C. Jacob. *Principia Evolvica, Simulierte Evolution mit Mathematica.* dpunkt-Verlag, Heidelberg, 1997. To appear.
50. M. James, J. Hanan, and P. Prusinkiewicz. CPFG version 2.0 user's manual. Manuscript, Department of Computer Science, University of Calgary, 1993, 50 pages.
51. J. M. Janssen and A. Lindenmayer. Models for the control of branch positions and flowering sequences of capitula in *Mycelis muralis* (L.) Dumont (Compositae). *New Phytologist*, 105:191–220, 1987.
52. J. Kaandorp. *Fractal modelling: Growth and form in biology.* Springer-Verlag, Berlin, 1994.
53. J. Kim. *Untersuchungen zur Evolution von morphologischer und taxonomischer Diversität und Komplexität anhand von Computermodellen.* PhD thesis, University of Cologne, January 1996.
54. W. Kurth. Some new formalisms for modelling the interactions between plant architecture, competition, and carbon allocation. Fourth workshop on individual-based structural and functional models in ecology, Wallenfels, September 25–28, 1996. To appear in *Bayreuther Forum für Ökologie*. 49 pages.
55. W. Kurth. *Growth grammar interpreter GROGRA 2.4: A software tool for the 3-dimensional interpretation of stochastic, sensitive growth grammars in the context of plant modeling. Introduction and reference manual.* Forschungszentrum Waldökosysteme der Universität Göttingen, Göttingen, 1994.
56. W. Kurth. Morphological models of plant growth: Possibilities and ecological relevance. *Ecological Modelling*, 75/76:299–308, 1994.
57. W. Kurth. Stochastic sensitive growth grammars: A basis for morphological models of tree growth. *Naturalia Monspeliensia*, 1996. In press.
58. W. Kurth and D. Lanwert. Biometrische Grundlagen für ein dynamisches Architekturmodell der Fichte (*Picea abies* (L.) Karst.). *Allgemeine Forst und Jagdzeitung*, 166:177–184, 9/10 1995.
59. A. Lindenmayer. Mathematical models for cellular interaction in development, Parts I and II. *Journal of Theoretical Biology*, 18:280–315, 1968.
60. A. Lindenmayer. Developmental systems without cellular interaction, their languages and grammars. *Journal of Theoretical Biology*, 30:455–484, 1971.

61. A. Lindenmayer. Adding continuous components to L-systems. In G. Rozenberg and A. Salomaa, editors, *L Systems*, Lecture Notes in Computer Science 15, pages 53–68. Springer-Verlag, Berlin, 1974.
62. A. Lindenmayer. Algorithms for plant morphogenesis. In R. Sattler, editor, *Theoretical plant morphology*, pages 37–81. Leiden University Press, The Hague, 1978.
63. A. Lindenmayer. Developmental algorithms: Lineage versus interactive control mechanisms. In S. Subtelny and P. B. Green, editors, *Developmental order: Its origin and regulation*, pages 219–245. Alan R. Liss, New York, 1982.
64. A. Lindenmayer. Positional and temporal control mechanisms in inflorescence development. In P. W. Barlow and D. J. Carr, editors, *Positional controls in plant development*. University Press, Cambridge, 1984.
65. A. Lindenmayer. Models for multicellular development: Characterization, inference and complexity of L-systems. In A. Kelemenová and J. Kelemen, editors, *Trends, techniques and problems in theoretical computer science*, Lecture Notes in Computer Science 281, pages 138–168. Springer-Verlag, Berlin, 1987.
66. A. Lindenmayer and H. Jürgensen. Grammars of development: Discrete-state models for growth, differentiation and gene expression in modular organisms. In G. Rozenberg and A. Salomaa, editors, *Lindenmayer systems: Impacts on theoretical computer science, computer graphics, and developmental biology*, pages 3–21. Springer-Verlag, Berlin, 1992.
67. A. Lindenmayer, J. Kelemen, and A. Kelemenová. Answers by A. Lindenmayer to questions of J. Kelemen and A. Kelemenová. *Bulletin of the European Association for Theoretical Computer Science*, 23:185–198, June 1984.
68. A. Lindenmayer and P. Prusinkiewicz. An annotated bibliography of plant modeling and growth simulation. In C. G. Langton, editor, *Artificial Life*, pages 625–643. Addison-Wesley, Redwood City, 1988.
69. J. Lück, H. B. Lück, and M. Bakkali. A comprehensive model for acrotonic, mesotonic, and basitonic branching in plants. *Acta Biotheoretica*, 38:257–288, 1990.
70. C. MacKenzie. Artificial evolution of generative models in computer graphics. Master's thesis, University of Calgary, September 1993.
71. C. MacKenzie and P. Prusinkiewicz. Artificial evolution of plant forms. Proceedings of the Fifth Annual Western Computer Graphics Symposium held in Vernon, BC, March 28–30, 1993, 9 pp.
72. H. Meinhardt. *The algorithmic beauty of sea shells*. Springer-Verlag, Berlin, 1995.
73. H. Meinhardt and M. Klinger. Pattern formation by coupled oscillations: The pigmentation patterns on the shells of molluscs. *Lecture Notes in Biomathematics*, 71:185–198, 1987.
74. R. A. Morelli, R. E. Walde, E. Akstin, and C. W. Schneider. L-system representation of speciation in the red algal genus Dipterosiphonia (Ceramiales, Rhodomelaceae). *The Journal of Theoretic Biology*, 149:453–465, 1991.
75. R. Měch and P. Prusinkiewicz. Visual models of plants interacting with their environment. Proceedings of SIGGRAPH '96 (New Orleans, Louisiana, August 4–9, 1996) ACM SIGGRAPH, New York, 1996, pp. 397–410.
76. M. Nielsen, G. Rozenberg, A. Salomaa, and S. Skyum. Nonterminals, homomorphisms and codings in different varitions of OL-systems. Part I. Deterministic systems. *Acta Informatica*, 4:87–106, 1974.
77. T. Nishida. K0L-systems simulating almost but not exactly the same development — the case of Japanese cypress. *Memoirs Fac. Sci., Kyoto University, Ser. Bio*, 8:97–122, 1980.

78. H. Noser and D. Thalmann. Simulating life of virtual plants, fishes and butterflies. In N. Magnenat Thalmann and D. Thalmann, editors, *Artificial Life and Virtual Reality*. J. Wiley & Sons, Chichester, 1994. Chapter 10.
79. H. Noser and D. Thalmann. The animation of autonomous actors based on production rules. In *Proceedings of Computer Animation '96*, pages 47–57, Los Alamitos, 1996. IEEE Computer Society Press.
80. L. F. Oritz, R. Y. Pinter, and S. S. Pinter. An array language for data parallelism: Definition, compilation, and applications. *The Journal of Supercomputing*, 5:7–29, 1991.
81. R. Y. Pinter and S. S. Pinter. Efficient breadth-first expansion on the Connection Machine, or: Parallel processing of L-systems. Technical Report YALEU/DCS-/TR-719, Department of Computer Science, Yale University, July 1989.
82. P. Prusinkiewicz (project leader). Virtual plant laboratory. A hypertext document and software distribution, Department of Computer Science, University of Calgary, 1996. Available at:
`http://www.cpsc.ucalgary.ca/projects/bmv/vlab/index.html`.
83. P. Prusinkiewicz. Graphical applications of L-systems. In *Proceedings of Graphics Interface '86 — Vision Interface '86*, pages 247–253, 1986.
84. P. Prusinkiewicz. Applications of L-systems to computer imagery. In H. Ehrig, M. Nagl, A. Rosenfeld, and G. Rozenberg, editors, *Graph grammars and their application to computer science; Third International Workshop*, pages 534–548. Springer-Verlag, Berlin, 1987. Lecture Notes in Computer Science 291.
85. P. Prusinkiewicz and M. Hammel. Visual models of morphogenesis: A guided tour. Hypertext document, Department of Computer Science, University of Calgary, 1994. Available at:
`http://www.cpsc.ucalgary.ca/projects/bmv/vmm/intro.html`.
86. P. Prusinkiewicz, M. Hammel, J. Hanan, and R. Měch. L-systems: from the theory to visual models of plants. In M. T. Michalewicz, editor, *Plants to ecosystems. Advances in computational life sciences I.* CSIRO Publishing, Melbourne, 1997. To appear.
87. P. Prusinkiewicz, M. Hammel, J. Hanan, and R. Měch. Visual models of plant development. In G. Rozenberg and A. Salomaa, editors, *Handbook of formal languages*. Springer-Verlag, Berlin, 1997. To appear.
88. P. Prusinkiewicz, M. Hammel, and E. Mjolsness. Animation of plant development. Proceedings of SIGGRAPH 93 (Anaheim, California, August 1–6, 1993). ACM SIGGRAPH, New York, 1993, pp. 351–360.
89. P. Prusinkiewicz, M. Hammel, R. Měch, and J. Hanan. L-systems: from the theory to visual models of plants. In D. Saupe and J. Hart, editors, *SIGGRAPH 1996 Course Notes on Fractal Models for Image Synthesis, Compression, and Analysis*, pages 113 – 185. ACM SIGGRAPH, 1996.
90. P. Prusinkiewicz and J. Hanan. *Lindenmayer systems, fractals, and plants*, volume 79 of *Lecture Notes in Biomathematics*. Springer-Verlag, Berlin, 1989 (second printing 1992).
91. P. Prusinkiewicz and J. Hanan. Visualization of botanical structures and processes using parametric L-systems. In D. Thalmann, editor, *Scientific visualization and graphics simulation*, pages 183–201. J. Wiley & Sons, Chichester, 1990.
92. P. Prusinkiewicz and J. Hanan. L-systems: From formalism to programming languages. In G. Rozenberg and A. Salomaa, editors, *Lindenmayer systems: Impacts on theoretical computer science, computer graphics, and developmental biology*, pages 193–211. Springer-Verlag, Berlin, 1992.

93. P. Prusinkiewicz, M. James, and R. Měch. Synthetic topiary. Proceedings of SIGGRAPH '94 (Orlando, Florida, July 24–29, 1994). ACM SIGGRAPH, New York, 1994, pp. 351–358.
94. P. Prusinkiewicz and L. Kari. Subapical bracketed L-systems. In J. Cuny, H. Ehrig, G. Engels, and G. Rozenberg, editors, *Graph grammars and their application to computer science; Fifth International Workshop*, Lecture Notes in Computer Science 1073, pages 550–564. Springer-Verlag, Berlin, 1996.
95. P. Prusinkiewicz and A. Lindenmayer. *The algorithmic beauty of plants.* Springer-Verlag, New York, 1990 (second printing 1996). With J. S. Hanan, F. D. Fracchia, D. R. Fowler, M. J. M. de Boer, and L. Mercer.
96. P. Prusinkiewicz, A. Lindenmayer, and J. Hanan. Developmental models of herbaceous plants for computer imagery purposes. Proceedings of SIGGRAPH '88 (Atlanta, Georgia, August 1–5, 1988). In *Computer Graphics* 22, 4 (August 1988), pages 141–150, ACM SIGGRAPH, New York, 1988.
97. P. Prusinkiewicz and R. Měch. Application of L-systems with homomorphism to graphical modeling. Manuscript, Department of Computer Science, University of Calgary, 1996.
98. P. Prusinkiewicz, W. Remphrey, C. Davidson, and M. Hammel. Modeling the architecture of expanding *Fraxinus pennsylvanica* shoots using L-systems. *Canadian Journal of Botany*, 72:701–714, 1994.
99. W. R. Remphrey and P. Prusinkiewicz. Quantification and modelling of tree architecture. In M. T. Michalewicz, editor, *Plants to ecosystems. Advances in computational life sciences I.* CSIRO Publishing, Melbourne, 1997. To appear.
100. P. M. Room and J. S. Hanan. Virtual cotton: a new tool for research, management and training. In G.A. Constable and N.W. Forrester, editors, *Proceedings of the World Cotton Research Conference 1: Challenging the Future*, pages 40–44. CSIRO, New York, 1995.
101. P. M. Room, J. S. Hanan, and P. Prusinkiewicz. Virtual plants: new perspectives for ecologists, pathologists, and agricultural scientists. *Trends in Plant Science*, 1(1):33–38, 1996.
102. P. M. Room, L. Maillette, and J. Hanan. Module and metamer dynamics and virtual plants. *Advances in Ecological Research*, 25:105–157, 1994.
103. G. Rozenberg. T0L systems and languages. *Information and Control*, 23:357–381, 1973.
104. G. Rozenberg, M. Penttonen, and A. Salomaa. Bibliography of L systems. *Theoretical Computer Science*, 5:339–354, 1977.
105. G. Rozenberg, K. Ruohonen, and A. Salomaa. Developmental systems with fragmentation. *International Journal of Computer Mathematics*, 5:177–191, 1976.
106. G. Rozenberg and A. Salomaa. *The mathematical theory of L systems.* Academic Press, New York, 1980.
107. G. Rozenberg and A. Salomaa. When L was young. In G. Rozenberg and A. Salomaa, editors, *The book of L*, pages 383–392. Springer-Verlag, Berlin, 1986.
108. K. Ruohonen. Developmental systems with interaction and fragmentation. *Information and Control*, 28:91–112, 1975.
109. C. W. Schneider and R. E. Walde. L-system computer simulations of branching divergence in some dorsiventral members of the tribe Polysiphonieae (Rhodomelaceae, Rhodophyta). *Phycologia*, 31(6):581–590, 1992.
110. C. W. Schneider, R. E. Walde, and R. A Morelli. L-systems computer models generating distichous from spiral organization in the Dasyaceae (Ceramiales, Rhodophyta). *European Journal of Phycology*, 29:165–170, 1994.

111. S. Shibusawa. Modelling the branching growth fractal pattern of the maize root system. *Plant and Soil*, 165:339–347, 1994.
112. A. R. Smith. Plants, fractals, and formal languages. Proceedings of SIGGRAPH '84 (Minneapolis, Minnesota, July 22–27, 1984). In *Computer Graphics*, 18, 3 (July 1984), pages 1–10, ACM SIGGRAPH, New York, 1984.
113. R. Smith. Evolution of L-systems. Manuscript, Department of Computer Science, University of Regina, 1989.
114. T. A. Steeves and I. M. Sussex. *Patterns in plant development.* Cambridge University Press, Cambridge, 1989.
115. A. L. Szilard and R. E. Quinton. An interpretation for DOL systems by computer graphics. *The Science Terrapin*, 4:8–13, 1979.
116. d'Arcy Thompson. *On Growth and Form.* University Press, Cambridge, 1952.
117. J. H. M. Thornley and I. R. Johnson. *Plant and crop modeling: A mathematical approach to plant and crop physiology.* Oxford University Press, New York, 1990.
118. A. Tunbridge and H. Jones. An L-systems approach to the modelling of fungal growth. *The Journal of Visualization and Computer Animation*, 6:91–107, 1995.
119. J. Vaario, N. Ogata, and K. Shimohara. Synthesis of environment directed and genetic growth. To appear in the proceedings of the Artificial Life V conference, held in Nara, Japan, May 16–18, 1996. Included in *ALIFE V oral presentations* (preliminary version of the proceedings), pp. 207–214.
120. T. A. Witten and L. M. Sander. Diffusion-limited aggregation, a kinetic critical phenomenon. *Physical Review Letters*, 47(19):1400–1403, 1981.
121. T. Yokomori. Stochastic characterizations of EOL languages. *Information and Control*, 45:26–33, 1980.