Jari Perttunen and Risto Sievänen. 2005. Incorporating Lindenmayer systems for architectural development in a functional-structural tree model. Ecological Modelling, volume 181, number 4, pages 479-491.

Available online at www.sciencedirect.com
SCIENCE DIRECT®
Ecological Modelling 181 (2005) 479–491
ECOLOGICAL MODELLING
www.elsevier.com/locate/ecolmodel

Incorporating Lindenmayer systems for architectural development in a functional-structural tree model

Jari Perttunen*, Risto Sievänen

Vantaa Research Centre, Finnish Forest Research Institute, PL 18, 01301 Vantaa, Finland

Received 10 September 2003; received in revised form 6 May 2004; accepted 4 June 2004

Abstract

LIGNUM is a functional-structural tree model that represents coniferous and broad-leaved trees with modelling units corresponding to the real structure of trees. The units are tree segments, axes, branching points and buds. Metabolic processes are explicitly related to the structural units in which they take place.

This paper enhances the modelling capabilities of LIGNUM with the possibility to formally describe the architectural development of trees with Lindenmayer systems. This is achieved by presenting an algorithm to convert tree structures generated by Lindenmayer systems to the LIGNUM representation of trees with feedback of results of events or processes from LIGNUM to Lindenmayer system. We then give two example applications that model the development of Scots pine (*Pinus sylvestris* L.) and the dwarf shrub bearberry (*Arctostaphylos uva-ursi* L.). Finally we discuss our approach and its consequences for the future development of LIGNUM.

Keywords: Lindenmayer systems; Functional-structural tree models; Tree architecture

1. Introduction

An increasing number of models (see e.g. Annals of Forest Science. vol. 57, no. 5/6) try to depict the dynamics and growth of woody perennial plants by assessing the physiological processes in their three-dimensional arborescent form. Physiological processes involve, for example, photosynthesis of the foliage, respiration, flow of water or hormones and the allocation of nutrients. The structure of the tree and its changes over time are described with state variables representing different aggregated tree compartments or with detailed modelling components faithful to the botanical or morphological units of plants. Such models have been called functional-structural tree models (FSTM).

Kurth (1994b) has classified tree and forest models on the basis of whether the emphasis is on the structural traits or on the functioning of trees. Accordingly, there are principally two ways to construct a FSTM.

* Corresponding author. Tel.: +358 010 211 2328; fax: +358 010 211 2203.
E-mail address: jari.perttunen@metla.fi (J. Perttunen).
URL: http://www.metla.fi/pp/JPer/.

0304-3800/$ – see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.ecolmodel.2004.06.034

One can start from an architectural model (Jaeger et al., 1992; Kurth, 1994a) and add functional, i.e. physiological detail to it. The second approach is to begin with a process-based physiological model (Mäkelä and Hari, 1986; Landsberg, 1986; Mäkelä, 1997) and extend it with structural details. Physiological models are usually realized with the aid of differential or difference equations (Landsberg, 1986), whereas architectural models apply Lindenmayer systems (Kurth, 1999; Prusinkiewicz and Lindenmayer, 1990) or other formalisms.

Since the pioneering work by Honda (1971) methods have become available for treating plant architecture in models in an efficient way. Lindenmayer systems or L systems (Prusinkiewicz and Hanan, 1989), are the most widely used method to treat plant architecture although other formalisms also exist (e.g. de Reffye et al., 1997; Godin et al., 1999). L systems were invented by Aristid Lindenmayer (1968, 1971) and were initially meant to describe the development of multicellular organisms. L systems are string-rewriting systems and research on these systems is concerned with the question of what phenomena can be described with formal languages. The theory, tools and applications that utilize a L system framework for modelling plants and their environment have been developed by Prusinkiewicz (Prusinkiewicz and Hanan, 1989, 1992), Kurth (Kurth, 1994a) and other scientists. The theory and the progress made in L systems in modelling plants and trees up until the end of the 1990s is well documented in Prusinkiewicz and Lindenmayer (1990), Prusinkiewicz (1999) and Kurth (1999).

All process-based models (Landsberg, 1986) for tree and stand growth must subsume a notion for crown or canopy structure that matches the objectives of the modelling. A facile solution is to assume horizontally homogeneous layers of foliage used in mainly theoretical studies of tree and forest growth (Hari et al., 1982; Mäkelä, 1997). Models that captured the individual stem structure or partitioned the above ground part of the tree into even finer compartments of branches, shoots, etc. (Kellomäki and Strandman, 1995) expanded the scope of the process-based models for example to wood quality applications (Kellomäki et al., 1999; Mäkelä and Mäkinen, 2003). Though the traits of physiology has been taken into consideration in detail, the tree architecture has been treated in the same model varying in detail and the way it has been embedded in the program implementing the model. Hence, each architectural model is unique and there exist no straightforward way of comparing these models as regards of architecture.

The methods for plant architecture (Prusinkiewicz and Lindenmayer, 1990; de Reffye et al., 1997; Godin et al., 1999) offer means for treating tree architecture in a formal, and therefore comparable way. However, these formalisms have so far not matched the capabilities of general programming languages to deal with diverse programming tasks (e.g. modelling diffusion of substrates). In a number of model or modelling tools, the suitable combination of methods to deal effectively with both architecture and physiology has been addressed (Kurth, 1999; Eschenbach, 2000; Karwowski and Prusinkiewicz, 2003; Yan et al., 2004). The development of models, where the architecture and functioning interact, is of key importance for better understanding the structural dynamics of trees.

The LIGNUM model approaches FSTM from the physiological side; it is a single tree model (Perttunen et al., 1996) with fidelity to process-based modelling (see e.g. Nikinmaa, 1992; Sievänen, 1993; Mäkelä, 1997) but, instead of aggregated tree parts, it has a three-dimensional description of the above ground part of the tree. LIGNUM includes a detailed model of self-shading within a tree crown (Perttunen et al., 1998, 2001), from which the radiation regime for photosynthesis in different parts of the tree can be computed. If the photosynthates produced exceed the respiration costs then the net production is allocated to the growth of new parts. LIGNUM has been applied to both coniferous (Perttunen et al., 1996; Lo et al., 2001) and broad-leaved trees (Perttunen et al., 2001). The main focus in LIGNUM has been on the functional part of the model and less emphasis has been paid to structural development. The model does not include any formal method to define the architectural development of the tree structure.

We describe in the following how the LIGNUM model has been interfaced with L systems for specifying formally the architectural development of trees, thereby improving the applicability and ease of use of this FSTM. The goal is achieved by using the L language, which is an extension of L systems (Prusinkiewicz et al., 1999). Based on the definition of L, R. Karwowski created the original parser of L and has implemented the L + C language (Karwowski,

2002). He also included further improvements not present in L, most notably productions with multiple successors and the concept of new context (Karwowski and Prusinkiewicz, 2003) to allow fast linear time information transfer in the simulated plant.

We first present the use of L in LIGNUM based on the likeness of how LIGNUM and bracketed L systems represent branching structures of trees (Perttunen et al., 1996, 2001). Second, we implement an algorithm that translates the bracketed string of symbols in L to the LIGNUM representation of trees (Section 4). Third, we provide a communication mechanism between the L string and LIGNUM (Section 4.1), and give two example applications, one for Scots pine and the other for bearberry (*Arctostaphylos uva-ursi* L.). Finally, we discuss the consequences of our approach for the future development of LIGNUM in modelling trees and forest stands.

2. Structural units of LIGNUM

LIGNUM is intended as a generic model for both coniferous and broad-leaved trees. Different tree species can be simulated by implementing models of metabolism, structural dynamics of birth, growth and senescence, and by realizing branching rules for distinct tree architectures (Perttunen et al., 1996, 2001). Here we briefly present the main features of the model in order to understand how LIGNUM is adapted to use L systems.

LIGNUM represents the three-dimensional aboveground part of the tree with four structural units called tree segment (TS), bud (B), branching point (BP) and axis (A). A branching point is a set of axes. An axis is a sequence of tree segments, branching points and terminating bud. This captures the recursive structure of the tree crown.

The most important functioning unit is the tree segment, i.e. the section of woody material between two branching points. For conifers the needles are currently modeled as a cylindrical layer of foliage surrounding a tree segment (Fig. 1, left). For broad-leaved trees leaves are attached to recently formed tree segments and are modelled explicitly using a simple geometric form such as an ellipse to represent the leaf shape (Fig. 1, middle).

An axis is implemented as a list. Adopting the notation from our previous work Perttunen et al. (1996), let the left bracket (‘[’) denote the beginning of the list, the right bracket (‘]’) denote the end of the list, and the list elements are separated by commas. For example, the main axis of the model tree for a coniferous species in Figure 1, consisting of three tree segments, three branching points and the terminating bud, writes:

$$[TS_0, BP_1, TS_2, BP_3, TS_4, BP_5, B_6] \quad (1)$$

The indices denote the positions of the elements in the list. A branching point is implemented as a list of axes. The three branching points in the model tree contain two axes (i.e. branches) each. Thus, the main axis can be written:

$$[TS_0, [A, A], TS_2, [A, A], TS_4, [A, A], B_6] \quad (2)$$

Each of the four axes in the first two branching points consist of a tree segment, branching point and bud. The two axes in the third (last) branching point contain one bud each. When these are inserted into Eq. (2), the

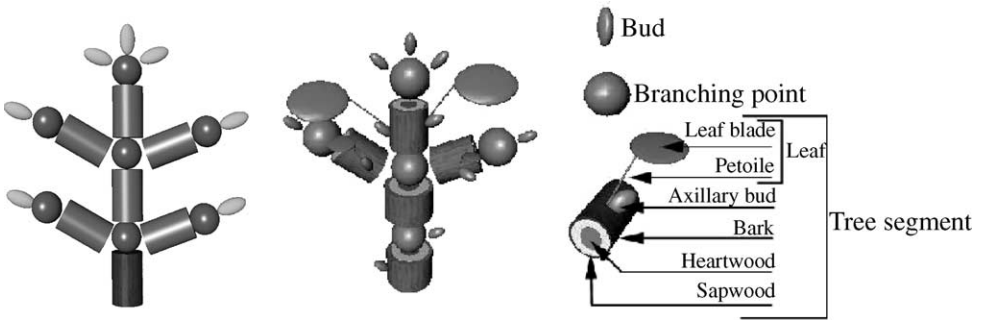

Fig. 1. Schematic presentation of a coniferous (left) and broad-leaved tree (middle) using structural units of LIGNUM. Also shown is the structure of a segment (right) for broad-leaved trees.

structure of the whole model tree can be expressed as (omitting the position numbering):

$$[TS, [[TS, [], B], [TS, [], B]], TS, [[TS, [], B], [TS, [], B]], TS, [[B], [B]], B] \quad (3)$$

The empty lists ([]) for branching points denote no forking off axes, thus maintaining the structural integrity of the model.

3. Introduction to the L language

Mathematically L systems[1] are parallel rewriting systems operating on strings, i.e. sequences of symbols. An L system is defined by an *alphabet* of symbols, a set of rewriting rules called *productions*, and an initial string called an *axiom*.

In the plant modelling context the symbols of the alphabet in an L system represent the units (internodes, leaves, flowers, etc.) of the growing organism relevant to the modelling approach, and the string of symbols their topological ordering. To be able to generate the branching structures dominant in the plant world, the notion of strings with brackets (literally denoting the beginning and end of a branch with '[' and ']') or bracketed L systems was already present in the original formalism (Lindenmayer, 1968).

In parametric L systems (Prusinkiewicz and Hanan, 1990), symbols may take numerical arguments to depict properties of the units like size or cocentrations of substances. Symbols in parametric L systems are also called modules. In context-free L systems, the predecessor's context, i.e. modules on its left and right side, does not influence the production application. In context-sensitive L systems the preceding and following modules affect the rewriting.

The key concept in the L system formalism is the rewriting of modules or symbols (Prusinkiewicz and Hanan, 1989). The L language follows the same idea. A module in L has a name and can take any number of arguments of any type in C++ programming language (Stroustrup, 1997). A syntactically correct rule in L consists of a predecessor module, possibly with its context ending with colon, and a production defining successor string embraced within curly braces. A special module, *Start*, corresponds to the axiom. For example, define the alphabet of two modules $A()$ and $B()$, the axiom $A()$ and the set of following two rules:

$$\begin{aligned} &\text{Start} : \{A();\} \\ &A() : \{\text{produce } B();\} \\ &B() : \{\text{produce } A()B();\} \end{aligned} \quad (4)$$

Starting from the axiom $A()$, the first five strings produced by the L system in Eq. (4) are $A()$, $B()$, $A()B()$, $B()A()B()$ and $A()B()B()A()B()$. Note the parallel rewriting of modules.

To describe the geometry, and also for the graphical interpretation of the model structures, Prusinkiewicz (1986) introduced L system symbols as instructions controlling a LOGO-style turtle (Abelson and di Sesa, 1982) in three dimensions, hence the adopted name turtle graphics.

For branching structures of plants L predefines two modules $SB()$ and $EB()$ denoting the beginning and end of a branch, respectively. To control the movements of the turtle (the geometry engine), let us first define its orientation in space by three unit vectors $\vec{H}$, $\vec{L}$ and $\vec{U}$ denoting the turtle's heading, direction to the left and up at right angles to each other such that $\vec{U} = \vec{H} \times \vec{L}$. Then define module $F(s)$ so as to move the turtle forward along its heading step of length s, and three modules $Turn(\alpha)$, $Pitch(\alpha)$, $Roll(\alpha)$ to rotate the orientation of the turtle around $\vec{U}$, $\vec{L}$ and $\vec{H}$ by α.

4. Integrating the L language and LIGNUM

To model tree structures with L as in LIGNUM, let the module F denotes a tree segment and designate module B for a bud. Given the modules for turtle graphics and branching we can now describe the topology and geometry of the tree structures of LIGNUM (Eq. (3)) with rules of the L language. To accommodate L with LIGNUM we first need to construct an algorithm that translates strings of L into structural units of LIGNUM (Section 4.1). Second, as the metabolism implemented in LIGNUM will act on structural units (segments and buds most notably) and produce changes in their physiological states, in their dimensions and in their positions in space accordingly, this information

[1] We present all examples of L systems using language L notation (Karwowski, 2002) instead of the commonly used notation (Prusinkiewicz and Hanan, 1989).

must be transferred to turtle symbols and module B as this is necessary in rewriting (Section 4.2).

We present an algorithm that translates strings of L into structural units of LIGNUM with the aid of an example with Scots pine. We have programmed in L the architectural development of Scots pine in approximately the same way as it is in the LIGNUM simulations of Perttunen et al. (1996, 1998); the L program is shown in Appendix A. In the program the module, $B(A, L)$ represents a bud. In each iteration the apical bud ($A = 1$) moves forward by the length L and produces four new side branches. The branches ($A > 1$) are similar, but only two more side branches are created. Branching stops after the third order ($A > 3$), but the extension continues. Also, as A records branching it is updated as new branches are added. When the development starts from the main axis consisting of one segment, branching point and bud, the string after two iterations is

$$F\ []\ F\ [F[B][B]B]\ [F[B][B]B]\ [F[B][B]B]\ [F[B][B]B]\ F\ [B][B][B][B]\ B \quad (5)$$

where symbol [denotes $SB()$ and] denotes $EB()$, the modules for turtle rotations are not shown, and the arguments of the modules B and F are dropped.

4.1. From L to LIGNUM

We can now outline a straightforward algorithm to convert the string of symbols in L to structural units in LIGNUM using Eq. (5) as a specific example. The algorithm is based on first recognizing the main axis of the tree, then finding the other axes (branches and sub-branches), and finally grouping them together as branching points.

First, the symbol $F(s)$ in Eq. (5) is interpreted as a tree segment of length s. The symbol B corresponds to a bud. Each consecutive set of n branches between two F symbols will be a branching point with n axes. The string in Eq. (5) first becomes the main axis with three segments, three branching points and the terminating bud:

$$[TS, BP, TS, BP, TS, BP, B] \quad (6)$$

The last two branching points contain four axes each, the first branching point being empty:

$$[TS, [], TS, [A, A, A, A], TS, [A, A, A, A], B] \quad (7)$$

Finally, recursively constructing the axes we get:

$$[TS, [], TS, [TS, [[B], [B]], B], \ldots, [TS, [[B], [B]], B]], \quad TS, [[B], [B], [B], [B]], B] \quad (8)$$

The current status of the turtle that is updated according to turtle commands in the string defines the position and orientation of the tree compartments in LIGNUM.

The structural development of the tree described in L does not require the regeneration of a LIGNUM representation of trees (deleting the old tree and creating a new one) after each derivation. The algorithm assumes that an L system indeed generates an alternating sequence of tree segments, branching points and terminating buds, and the terminal buds only generate new structural units. The algorithm fails if this is not the case. Thus, after each derivation it is possible to match the existing string and LIGNUM representation and to insert the new structural units in the axis lists before the terminating buds whose positions and orientations will be updated.

4.2. From LIGNUM to L

Because the architectural development of a tree is defined with an L string, we can assume LIGNUM does not have to change the topology of the tree. Consequently the conversion from LIGNUM tree to L string is trivial.

However, as the physiological activities in LIGNUM (Perttunen et al., 1996) will change the dimensions and statuses of the tree segments and buds, it is necessary to be able to transfer information from LIGNUM to the interpretation of the L program. That is, the results of the physiological processes in LIGNUM must be able to change the parameter values of the modules F and B in the L string thus enabling interaction between the architectural part and the physiological part.

Since the parameter s in the module $F(s)$ is always interpreted as the length the turtle moves forward, and its meaning is the length of a segment in LIGNUM, the conversion algorithm can implicitly update the value of s using the length of the corresponding tree segment.

However, the parameters for module B are model specific. Thus explicitly given the (C++) type of the

parameters for module B, their assignment statements and their variable names (meaning) in LIGNUM, the conversion algorithm updates parameter values of module B using corresponding variable values from the associated bud. This way the results of computations in LIGNUM are passed to the L system. (Similarly, the values of the parameters of module B can be assigned to the variables of the corresponding bud in LIGNUM in Section 4.1.)

Technically, each L file is translated to C++ and the result of the translation is a C++ class providing an application programming interface (API) to the defined L system. Most notably, this API includes derivation of the string to model structural development, the conversion algorithm between L system and the model LIGNUM (Section 4.1) and vice versa. The translator also allows to embed C++ statements in the L file (Prusinkiewicz et al., 1999).

4.3. Two-way communication in the Scots pine simulation

Although in our Scots pine example (Appendix A) it is L in $B(A, L)$ that initially determine the length of a segment, the metabolic processes eventually resolve the amount of growth.

We interfaced the L language program of Appendix A with the functional part of LIGNUM that incorporates our previous work on the Scots pine model (Perttunen et al., 1996, 1998). The calculations in LIGNUM include the pairwise comparison of segments in order to compute the interception of solar radiation and the iterative allocation of net photosynthates to growth after respiration costs:

$$P - M = iW_n + iW_o + iW_r \tag{9}$$

where P and M are the photosynthetic production and the respiration of a tree, iW_n is the growth of new segments, iW_o is the secondary growth and iW_r is the root growth (cf. Perttunen et al., 1996, 1998, 2001). We omit some functions of LIGNUM, e.g. the number of secondary buds as a function of the foliage mass of the mother segment and let the L system determine branching.

The simulations show that the interaction of the L language program and the functioning part of LIGNUM markedly affects the outcome of the simulation (Fig. 2). Also the effect of two extreme cases of foliage mortality show that physiological characteristics have a great effect on the architecture of the trees. The simulation gives a much shorter and slimmer stem due to the smaller need for diameter growth for a pine

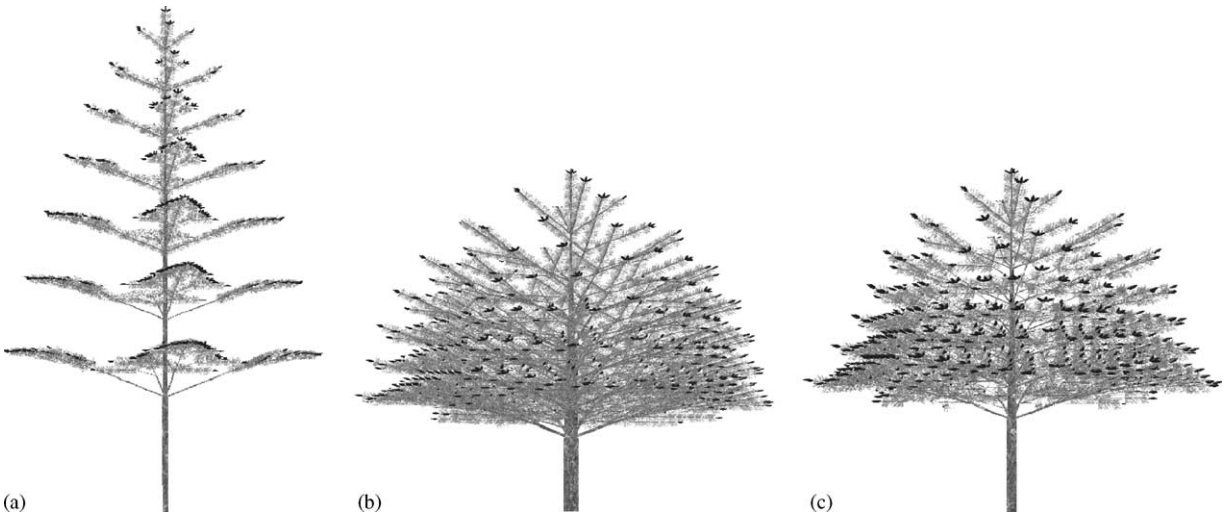

Fig. 2. The development of three pines after eight development steps when architectural development takes place according to the L program of Appendix A and metabolic functioning is as in Perttunen et al. (1996, 1998). Leftmost: development according to the L program only; middle and right: interaction of L language and LIGNUM depicting the effect of foliage mortality. Middle: foliage remains for 5 years. Length = 3.5 m, diameter at base = 10 cm. Right: foliage remains for 1 year. Length = 2.7 m, diameter at base 6 cm.

with short-living foliage in comparison to a pine with long-living foliage.

4.4. Two-way communication in the bearberry simulation

Salemaa and Sievänen (2002) have studied the growth habits, axillary bud activation and branching architecture of the horizontally spreading clonal shrub bearberry (*A. uva-ursi* L.) growing in South Western Finland in varying pollution, nutrient and light levels. They have designed an L system model to study the qualitative features of the branching patterns of bearberry by simulation. Here their bearberry model growing in a sandpit is realized in such a way that a collision detection algorithm can detect if a clonal branch blockades the growth space of an active bud. This also serves as an example of a stochastic L system (Prusinkiewicz and Lindenmayer, 1990) and about how to implement global sensitivity (Kurth, 1994a) in LIGNUM.

In their L system model for bearberry (Salemaa and Sievänen, 2002), the plant grows horizontally and consists of annual growth units bearing lateral buds and one apical bud. The buds are divided into dominant (D), subdominant (SD) and nondominant (ND) types (Remphrey et al., 1983). A living bud produces a shoot of its own type. Axillary buds have a time delay before they release and produce a shoot.

For the implementation in L, we define module $B(T, S, C)$, where T denotes the bud type (D, SD or ND), S its status, i.e. the time delay left before release, and C is collision. The module definition for B determines the lengths of the shoots produced, branching angles for lateral shoots forking off alternately to the left or to the right, and the number of axillary buds produced. In general, type D shoots are longer than SD and ND shoots, and in the sandpit the colonizing plant shows intensive lateral branching due to favorable light conditions. An outline of the L system is given in Appendix B. For details see Salemaa and Sievänen (2002).

The collision detection algorithm of LIGNUM simply examines a given opening angle symmetrical to both sides of the growth direction of a bud and checks whether it is free of other shoots and buds within a given distance. More precisely, define $\vec{P}_1$ as the position of the bud and $\vec{D}$ as its growth direction. Define $\vec{P}_2$ as the position of the potential obstacle. Then $\vec{P}_3 = \vec{P}_2 - \vec{P}_1$ is the direction from the bud to the obstacle. Given the opening angle α and the distance l, the obstacle hinders the growth of the bud if $\cos(\alpha/2) \leq \vec{D} \cdot \vec{P}_3/|\vec{D}||\vec{P}_3|$ (as the cosine increases when the angle decreases) and $|\vec{P}_3| < l$.

Investigation of whether a bud collides with another plant compartment is implemented as a pairwise comparison of structural units in LIGNUM, i.e. not by rewriting the rules in L language. The value of parameter C in the module B is updated (Section 4.2) using the result of collision detection calculations in LIGNUM. The results of the development of the bearberry model after 15 iterations with three different collision models are presented in Fig. 3. As one would expect, the bearberry models show less lateral branching with increasing opening angle for collision detection.

Fig. 3. Realization of the bearberry model of Salemaa and Sievänen (2002) with the connection of L language program of Appendix B and LIGNUM. Collision detection is accomplished by LIGNUM. Three simulations with 15 iterations using different collision detection parameters for active buds. From left to right: the opening angle and the distance to obstructing branch are 35°/30 cm, 45°/30 cm and 65°/30 cm, respectively.

5. Discussion

This study presents a formal way to model the architectural development of trees using L systems through the language L (Prusinkiewicz et al., 1999) in the functional-structural tree model LIGNUM implemented using a general purpose programming language. Essentially the use of L is based on the similarity of how bracketed L systems and LIGNUM represent the branching structure of trees. Similar conversions between modelling frameworks and tools have been reported, for example, by Ferraro et al. (2002) and Dzierzon et al. (2003). In fact, already Kurth (1994b) reported convergence between tree models produced by AMAP and the same models expressed in L systems. Also, a large body of mathematical forms known as fractals, used to depict plant structures, have been described with L systems (Kurth, 1999).

The L systems provide the sort of good scientific abstraction needed in plant modelling (cf. Regev and Shapiro, 2002). An L system captures the relevant properties of the phenomena in its set of symbols highlighting only the essential characteristics of the model. It is computable to support qualitative and quantitative reasoning of the model properties. It is extensible, new symbols can capture additional features of the model if required and it is understandable, the formal notation allows the sharing and comparison of scientific knowledge. For example, part of an L system model in itself can appear in a publication as a model description, unlike models implemented with general programming languages.

The theoretical advancements that have since been implemented in tools based on L system formalism have been motivated by the desire to find out what phenomena in plant modelling can be formally described and simulated. The range of circumstances where L system formalism is applicable is quite extensive (Prusinkiewicz, 1999). Kurth (1999) has demonstrated this by realizing a number of published architectural tree models including LIGNUM Scots pine (Perttunen et al., 1996) using the L system based tool GROGRA (Kurth, 1994a). The simulation of plant communities has been reported by Deussen et al. (1998), Kurth (1999) and with multiset L systems developed by Lane and Prusinkiewicz (2002). Inevitably one has to ask why not reimplement LIGNUM with L+C language (Karwowski, 2002; Karwowski and Prusinkiewicz, 2003) and make L systems the basis of its future development?

Plants are not closed systems since interaction with their environment has an important function in their development. Modelling of such phenomena includes, for example, computation of the light regime in plant communities and competition for growing space (cf. example on bearberry, Section 4.4). To model such phenomena, Měch (1997) and Měch and Prusinkiewicz (1996) have extended L system formalism with communication symbols that can pass parameter values between the plant model in the L system and a separate program (in general purpose language) simulating, for example, the relevant characteristics of its environment. Kurth (1994a) has implemented a set of predefined functions to return environmental information to the L system in the GROGRA program. Once implemented, these separate programs or predefined functions are easily used and reused.

The functionalities enhancing L system languages make it possible to realize parts of the simulations using general purpose programming languages (e.g. unit to unit interactions) that would be difficult to implement with the parallel rewriting semantics. The design of L and L + C allows embedding of C++, thus making such constructs unnecessary. But note that the time spent in these environmental models is the time spent outside the L system formalism with some general purpose programming language. Hence, complicated architectural tree and plant models, such as FSTMs, implemented using L systems inevitably employ both an L system and general purpose language parts. LIGNUM combined with L language also contains those parts, although in different proportions than models realized with L system tools. Therefore, as the quest goes on to find optimal formalisms and modelling paradigms to implement complicated plant and tree models, our current solution is to mix L systems, general purpose programming languages and programming libraries implementing submodels such as radiation climate and soil properties.

A further challenge to tree and plant modelling is to implement source–sink relationships in architectural tree and plant models. Local production and consumption of resources, which are affected by the environment and status of particular structural units, control the

growth of the three-dimensional structure. Source sink phenomena have been modelled by considering unit to unit interactions (Balandier et al., 2000), accumulating information along the pathways from root tip to shoot tip (de Reffye et al., 1997) or solving partial differential equations (Deleuze and Houllier, 1997; Palovaara, 2003). How the intensive calculations required by the sink-source approach are best implemented in the three-dimensional plant structure is still unknown. Recent advancement in the L systems (Karwowski, 2002; Karwowski and Prusinkiewicz, 2003) allowing fast information transfer in the plant may open up new possibilities. A hybrid approach utilizing both L systems and general purpose languages or other means (e.g. solvers of differential equations) offers one alternative.

FSTMs can be applied in the cases when the heterogeneous environment is an important factor of the studied phenomena. Coates et al. (2003) suggest that linking empirical studies to models is the best way to provide insight and better understanding of the implications of the silvicultural strategies and the importance of structure in forest stands. For example de Chantal et al. (2003) have studied the early development, size and morphology, of Scots pine and Norway spruce in an experimental gap-edge environment with asymmetric distribution of radiation. Such experimental work to understand tree regeneration might benefit if the LIGNUM model was applied to describe the study plots, the gap-edge zones, size of the gaps, the spatial distribution of seedlings, and then based on the radiation climate simulate the sapling development.

Acknowledgments

We thank Przemyslaw Prusinkiewicz for giving impetus for this work and inviting to University of Calgary. The original specification of L was jointly developed by Prusinkiewicz et al. (1999). Radoslaw Karwowski wrote the original parser of L and has further implemented the L+C language. We thank for the possibility of using the original L parser. J.P. was supported by research grants 72569 and 50708 from the Academy of Finland. We thank two anonymous referees for their valuable comments.

Appendix A. L system mimicking pine growth

The L system mimicking pine growth starts with one segment and one bud. The main axis, ($A = 1$), creates one segment and four branches forking off. From then on side branches ($A > 1$) create one segment and two additional branches. Ramification stops after third order branches. Bending of the branches is modeled by pitching the second order branches ($A = 2$) down in the *Bend* module.

```
open Pine;
const double PI = 3.1415926535897932384;
module F(double);
module B(int,double);
module Pitch(double);
module Roll(double);
module Turn(double);
module Bend(double);

Start:{produce F(0.30)SB()EB()B(1,1.0);}
B(A,l):
{
   if (A==1)
   produce F(l) SB() Pitch(PI/4.0)  B(A+1,l*0.6) EB()
                SB() Roll(PI/2.0) Pitch(PI/4.0) B(A+1,l*0.6) EB()
                SB() Roll(PI) Pitch(PI/4.0)  B(A+1,l*0.6) EB()
                SB() Roll(3.0*PI/2.0) Pitch(PI/4.0)  B(A+1,l*0.6) EB()
           B(A,l*0.9);
   else if (A==2)
   produce  Bend(0.3) F(l) SB() Turn(PI/4.0)  B(A+1,l*0.4) EB()
                           SB() Turn(-PI/4.0) B(A+1,l*0.4) EB()
            Bend(-0.2)B(A,l*0.6);
   else if (A==3)
   produce F(l) SB() Turn(PI/4.0)  B(A+1,l*0.3) EB()
                SB() Turn(-PI/4.0) B(A+1,l*0.3) EB()
           B(A,l*0.4);
   else
   produce F(l) SB() EB() B(A,l);
}
Bend(s):
{
   produce Pitch(s);
}
close Pine;
```

Appendix B. Fragmentary L system for bearberry growth

Fragmentary L system for bearberry growth. The *Start* module creates the initial plant. The *B* module, line 15, checks for collision. Lines 18–40 determine branching and growth. The pattern of ramification is based on field data and implemented in the uniform random variables $r1$ and $r2 \in [0, 1]$. $r1$ initializes the branching to the left or right. Branching and growth depends on the bud type, its status and the value of $r2$. The counter on line 41 eventually activates dormant buds ($s > 0$). Bud types: D = dominant, N = nondominant and S = subdominant.

```
1.open Bearberry;
2.const PI = 3.1415926535897932384;
3.module B(double type,double status,double collision);
4.derivation length: 15;
5.Start:{produce F(0.1) SB() EB() B(D,0.0,0.0);}
6.B(T,s,C):
7.{
 8.  double g = 0.0;
 9.  double r1 = ran();
10.  double r2 = ran();
11.  if (r1 < 0.5)
12.     g = -5*PI/180;
13.  else
14.     g =  5*PI/180;
15.  if (C == 1.0){
16.    produce B(T,s,C);
17.  }
18.  else if (T == D && s == 0.0){
19.    if (r2 < 0.26)
20.      produce Turn(g)F(0.6)SB()Turn( 30*PI/180)B(N,2,C)EB()
21.              Turn(g)F(0.1)SB()Turn(-30*PI/180)B(N,1,C)EB()
22.              Turn(g)F(0.1)SB()Turn( 30*PI/180)B(S,1,C)EB()
23.              Turn(g)F(0.1)SB()Turn(-30*PI/180)B(S,0,C)EB()
24.              Turn(g)F(0.1)SB()EB()B(D,0,C);
25.    else if (r2 <= 0.52)
26.      produce Turn(g)F(0.6)SB()Turn(-30*PI/180)B(N,2,C)EB()
27.              Turn(g)F(0.1)SB()Turn( 30*PI/180)B(N,1,C)EB()
26.              Turn(g)F(0.1)SB()Turn(-30*PI/180)B(S,1,C)EB()
```

```
28.              Turn(g)F(0.1)SB()Turn( 30*PI/180)B(S,0,C)EB()
29.              Turn(g)F(0.1)SB()EB()B(D,0,C);
30       ..................................................
32.  }
33.  else if (T == S && s == 0.0){
33.    if (r2 < 0.037)
34.      produce Turn(g)F(0.48)SB()Turn(-30*PI/180)B(S,1,C)EB()
35.              Turn(g)F(0.08)SB()EB()B(S,0,C);
36.       ..................................................
37.  }
38.  else if (T == N && s == 0.0){
39.       ..................................................
40.  else{
41.    produce B(T,max(s-1,0),C);
42.  }
43.}
44.close Bearberry;
```

References

Abelson, H., di Sesa, A.A., 1982. Turtle Geometry. MIT Press, Cambridge.

Balandier, P., Lacointe, A., LeRoux, X., Sinoquet, H., Cruiziat, P., LeDizes, S., 2000. SIMWAL: a structural-functional model simulating single walnut tree growth in response to climate and pruning. Ann. For. Sci. 57, 571–585.

Coates, K.D., Canham, C.D., Beaudet, M., Sachs, D.L., Messier, C., 2003. Use of spatially explicit individual-tree model (SORTIE/BC) to explore the implications of patchiness in structurally complex forests. For. Ecol. Manage. 186.

de Chantal, M., Leinonen, K., Kuuluvainen, T., Cescatti, A., 2003. Early response of *Pinus sylvestris* and *Picea abies* seedlings to an experimental canopy gap in a boreal spruce forest. For. Ecol. Manage. 176, 321–336.

de Reffye, P., Fourcaud, T., Blaise, F., Barthélémy, D., Houllier, F., 1997. A functional model of tree growth and tree architecture. Silva Fennica 31 (3), 297–311.

Deleuze, C., Houllier, F., 1997. A transport model for tree ring width. Silva Fennica 31, 239–250.

Deussen, O., Hanrahan, P., Lintermann, B., Měch, R., Pharr, M., Prusinkiewicz, P., 1998. Realistic modeling and rendering of plant ecosystems. In: Cunningham, S., Bransford, W., Cohen, M.F. (Eds.), Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques. ACM Press, New York, pp. 275–286.

Dzierzon, H., Perttunen, J., Kurth, W., Sievänen, R., Sloboda, B., 2003. Enhanced possibilities for analyzing tree structure as provided by an interface between different modelling systems. Silva Fennica 37 (1), 31–44.

Eschenbach, C., 2000. The effect of light acclimation of single leaves on whole tree growth and competition—an application of the tree growth model ALMIS. Ann. For. Sci. 57 (5/6), 599–609.

Ferraro, P., Godin, C., Prusinkiewicz, P., 2002. L-systems and MTGs: integrating simulation and formal analysis of architectural plant models. In: Krzysztof, A., Meuth, H. (Eds.), Proceedings of 16th European Simulation Multiconference. Modelling and Simulation 2002. SCS Europe, Erlangen, Germany, ESM 2002, 3–5 June 2002, Darmstadt, Germany, 2002, pp. 418–422.

Godin, C., Costes, E. Sinoquet, H., 1999. A method for describing plant architecture which integrates topology and geometry, Ann. Bot. 84 (3), 343–357.

Hari, P., Kellomäki, S., Mäkelä, A., Ilonen, P., Kanninen, M., Korpilahti, E., Nygren, M., 1982. Dynamics of early development of tree stand. Acta Forestalia Fennica 177.

Honda, H., 1971. Description of the form of trees by the parameters of the tree-like body: effects of the branching angle and branching length on the shape of the tree-like body. J. Theor. Biol. 31, 331–338.

Jaeger, M., de Reffye, Ph., 1992. Basic concepts of computer simulation of plant growth. J. Biosci. 17, 275–291.

Karwowski, R., 2002. Improving the Process of Plant Modeling: The L + C Modeling Language. University of Calgary.

Karwowski, R., Prusinkiewicz, P., 2003. Design and implementation of the L + C modelling language. Electron. Notes Theoret. Comput. Sci. 86 (2) 19 pp.

Kellomäki S., Ikonen, V.-P., Peltola, H., Kolström, T., 1999. Modelling the structural growth of Scots pine with implications for wood quality, Ecol. Model. 122, 117–134.

Kellomäki, S., Strandman, H., 1995. A model for structural growth of young scots pine crowns based on light interception by shoots. Ecol. Model. 80, 237–250.

Kurth, W., 1994a. 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 Universitet Göttingen, Göttingen.

Kurth, W., 1994b. Morphological models of plant growth: possibilities and ecological relevance. Ecol. Model. 75/76, 299–308.

Kurth, W., 1999. Die Simulation der Baumarchitektur mit Wachstumsgrammatiken. Wissenschaftlicher Verlag, Berlin, 327 pp.

Landsberg, J.J., 1986. Physiological Ecology of Forest Production. Academic Press, London.

Lane, B., Prusinkiewicz, P., 2002. Generating spatial distributions for multilevel models of plant communities. In: Proceedings of Graphics Interface, Calgary, Alta., May 27–29, pp. 69–80.

Lindenmayer, A., 1968. Mathematical models for cellular interaction in development, parts I and II. J. Theor. Biol. 18, 280–315.

Lindenmayer, A., 1971. Developmental systems without cellular interaction their languages and grammar. J. Theor. Biol. 30, 455–484.

Lo, E., Zhang, M.W., Lechowicz, M., Messier, C., Nikinmaa, E., Perttunen, J., 2001. Adaptation of the LIGNUM model for simulations of growth and light response in Jack pine. For. Ecol. Manage. 150 (3), 279–291.

Mäkelä, A., 1997. Carbon balance model of growth and self-pruning in trees based on structural relationships. For. Sci. 43 (1), 7–24.

Mäkelä, A., Hari, P., 1986. Stand growth model based on carbon uptake and allocation in individual trees. Ecol. Model. 33, 315–331.

Mäkelä, A., Mäkinen, H., 2003. Generating 3D sawlogs with a process-based growth model. For. Ecol. Manage. 184, 337–354.

Měch, R., 1997. Modeling and Simulation of the Interactions of Plants with Their Environment Using L-systems and Their Extension. University of Calgary.

Měch, R., Prusinkiewicz, P., 1996. Visual models of plants interacting their environment. In: Proceedings of SIGGRAPH '96, New Orleans, LA, 4–9 August 1996. ACM SIGGRAPH, New York, pp. 397–410.

Nikinmaa, E., 1992. Analysis of the growth of Scots pine; matching structure with function. No. 235 in Acta Forestalia Fenica. The society of Forestry in Finland - the finnish Forest Research Institute, 68 pp.

Palovaara, L., 2003. Reaktio-kuljetusyhtälö puumaisessa rakenteessa. Reaction-transport equation in tree-like structure. Master's thesis, Teknillinen korkeakoulu. Helsinki University of Technology, Espoo.

Perttunen, J., Nikinmaa, E., Lechowicz, M.J., Sievänen, R., Messier, C., 2001. Application of the functional-structural tree model LIGNUM to sugar maple saplings (*Acer saccharum* Marsh) growing in forest gaps. Ann. Bot. 88 (3), 471–481.

Perttunen, J., Sievänen, R., Nikinmaa, E., 1998. LIGNUM: a model combining the structure and the functioning of trees. Ecol. Model. 108 (1–3), 189–198.

Perttunen, J., Sievänen, R., Nikinmaa, E., Salminen, H., Saarenmaa, H., Väkevä, J., 1996. LIGNUM: a tree model based on simple structural units. Ann. Bot. 77 (1), 87–98.

Prusinkiewicz, P., 1986. Graphical applications of L-systems. In: Proceedings of Graphics Interface '86—Vison Interface '86, pp. 247–253.

Prusinkiewicz, P., 1999. A look at the visual modeling of plants using L-systems. Agronomie 19, 211–224.

Prusinkiewicz, P., Hanan, J., 1989. Lindenmayer systems fractals and plants. In: No. 79 in Lecture Notes in Biomathematics. Springer–Verlag, Berlin.

Prusinkiewicz, P., Hanan, J., 1990. Visualization of botanical structures and processes using parametric L-systems. In: Thalmann, D. (Ed.), Scientific Visualization and Graphics Simulation. J. Wiley & Sons, pp. 183–201.

Prusinkiewicz, P., Hanan, J., 1992. Lindenmayer systems: Impacts on theoretical computer science, computer graphics and developmental biology. In: Rozenberg, G., Salomaa, A. (Eds.), L-systems: From Formalism to Programming Languages. Springer–Verlag, pp. 193–211.

Prusinkiewicz, P., Karwowski, R., Perttunen, J., Sievänen, R., 1999. Specification of L–a plant modeling language based on L-systems, version 0.5. Department of Computer Science, University of Calgary, Internal report.

Prusinkiewicz, P., Lindenmayer, A., 1990. The Algorithmic Beauty of Plants. Springer-Verlag, New York.

Regev, A., Shapiro, E., 2002. Cells as computation. Nature 419, 343.

Remphrey, W.R., Neal, B.R., Steeves, T.A., 1983. The morphology and growth of *Arctosaphylos uva-ursi* (bearberry): an architectural model simulating colonizing plant. Can. J. Bot. 2451–2458.

Salemaa, M., Sievänen, R., 2002. The effect of apical dominance on the branching architecture of *Arctostaphylos uva-ursi* in four contrasting environments. Flora 197 (1189), 1–14.

Sievänen, R., 1993. A process-based model for the dimensional growth of even-aged stands. Scand. J. For. Res. 8 (1), 28–48.

Stroustrup, B., 1997. The C++ Programming Language, 3rd ed. Addison-Wesley, Reading, MA.

Yan, H.-P., Kang, M.Z., de Reffye, P., Dingkuhn, M., 2004. A dynamic, architectural plant model simulating resource-dependent growth. Ann. Bot. 93, 591–602.