Semantic Death in Plant's Simulation Using Lindenmayer Systems

Erick Castellanos
ecastella@gdl.cinvestav.mx
Cinvestav Unidad Guadalajara
Jalisco, Mexico

Felix Ramos
framos@gdl.cinvestav.mx
Cinvestav Unidad Guadalajara
Jalisco, Mexico

Marco Ramos
marco.corchado@gmail.com
Universidad Autonoma del Estado de Mexico
Estado de Mexico, Mexico

***Abstract*—Plant's simulation through Lindenmayer Systems is a well know field, but most of the work in the area focus on the growth part of the developmental process. From an artificial life perspective, it is desired to have a simulation that includes all the stages of the cycle of life of a plant. That is the reason why this paper target the last stage and propose a strategy to include the concept of death through Lindenmayer systems. By using parametric and context-sensitive Lindenmayer systems in the modeling and simulation, the semantics of the mentioned concept can be captured and, thereby, with the proper interpretation, a graphic result, at a morphological level, can be displayed. A proof of concept that includes most of the concepts covered is also given.**

***Index Terms*—Plants simulation, Death of plants, Semantic death, Lindenmayer systems.**

I. Introduction

Since man became aware of him and understood that it was a being with life and not like several inanimate objects around him, the study of life began. In order to comprehend what was life, initial studies were conducted directly on living beings. However, interest for studying life from another perspective has emerged relatively recently. This new approach was called Artificial Life, a term coined in 1987 by Christopher Langton[1] (but it goes back even earlier[2]). This research area focuses on the study of artificial beings through simulation models.

As in the study of traditional life in which there are two major areas of interest, i.e. animal life and vegetal life, the same is true about the research of artificial life that can be divided or categorized into different areas of study. Thus, there is also a topic in the study of artificial plants in which models are created representing a plant species and these models contain features that are of interest mainly for botanists and biologists, and other researchers.

In particular, in the last few decades[3], there has been an interest in recreating the process of plant growth[4]. Being more specific, the morphological representation through formal systems of the mentioned process had acceptable success. However, as far as we know, the existent studies focus only on the birth, growth and reproduction of the plant, but there is a stage missing for the completion of the life cycle, i.e. death. Research on the simulation of this last stage is of great importance in order to have a complete model of an artificial plant.

In this paper the usage of Lindenmayer systems (L-systems) is proposed, with some of its extensions, to incorporate the concept of death into the simulation of the developmental process of a plant, in the context of its morphology, as well as incorporating some of the biological features that show real plants in their growth. The focus is primarily on the additions needed to the formal system, at a semantic level, to add the concept of death. Nevertheless, a visual representation is possible, in two or three dimensions, using the same techniques of other papers.

If the concept of death can be successfully incorporated in the modeling of plants, this could have a positive impact on biological virtual experimentation. As a side effect, the resulting techniques can be useful in other areas like serious games or architectural design.

In the following sections, a review of the main ideas that exists in the state of the art is given and that are relevant on the different topics that this research covers. Once that the existing methodologies are explained in section II, a detailed description of the approach taken in this paper is given in section III which will incorporated death in a novel way. There follows, in section II, some examples of models developed that contains the features mentioned with its description. Finally, in section V some conclusions about this work are given.

II. State of the Art

The relevant topics for this paper are, essentially, two: the formalism established by a biologist plant's cells development; and a mechanism to provide a graphical interpretation to the mentioned formalism. A summary of these topics follows.

A. Lindenmayer systems

Aristid Lindenmayer was a Hungarian biologist who sought to establish a formalism that would describe the process of development of some plant cells[5]. The resulting formalism, which is a formal language but including parallelism inherently, it was called Lindenmayer Systems in honor of its creator. This L-systems are formal languages but do not share the classical hierarchy described by Chomsky[6] because this systems are parallel by nature: instead of processing one symbol at a time, all the symbols that can be processed in the string are translated according to the given rules. For a formal analysis of L-systems [7] can be consulted. A diverse variety of L-systems have emerged but only four are of interest for this paper.

1) Deterministic L-Systems: The most simple form of L-Systems are the deterministic ones[8]. Formally, a deterministic L-System is a triplet $G = \langle V, \omega, P \rangle$ where V is the alphabet of the system, $\omega \in V^+$ is the axiom and $P \subset V \times V^*$ is a finite set of production rules (or rewrite rules).

2) Parametric L-Systems: Parametric L-systems[9] are an extension to L-systems, with the possibility to use parameters in the rewrite rules. Formally: $G_\Sigma = \langle V, \Sigma, \omega, P \rangle$, where V is the alphabet of the system, Σ is a finite set of formal parameters, $\omega \in (V \times \mathbb{R}^*)^+$, is a nonempty parametric word called the axiom, and $P \subset (V \times \Sigma^*) \times \zeta(\Sigma) \times (V \times \xi(\Sigma))^*$ is a finite set of production rules. The sets of all valid logic and arithmetic constructions with parameter Σ are denoted $\zeta(\Sigma)$ and $\xi(\Sigma)$ respectively.

3) Stochastic L-Systems: Stochastic L-systems[10] feature randomness in the production rules, and thereby generate flexible systems for modeling non-deterministic processes. Formally, a stochastic L-systems is a tuple $G_\pi = \langle V, \omega, P, \pi \rangle$, where V is the alphabet of the system, $\omega \in V^+$ is the axiom, $P \subset V \times V^*$ is a finite set of production rules, and $\pi : P \to (0, 1]$ is a function called probability distribution. It is assumed that, for each symbol $a \in V$, the sum of the probabilities of the productions whose predecessor is a equals to 1.

4) Context-sensitive L-Systems: The formal definition of context-sensitive L-systems[8] is similar to the deterministic one with the exception that the production rules have the form $a_l < a > a_r \to \chi$, where a (known as strict predecessor) produce the word χ if, and only if, a is preceded by a_l and followed by a_r, such that a_l and a_r make the left and right context of a in this production. It is trivial to generalize the definition such that, instead of the context being comprised by a single letter, any of the to context is made by a word of length k y l, respectively.

It is important to note that such systems can coexist with deterministic L-systems (a production with one predecessor). In order to avoid conflicts between these two types of L-systems, a higher priority is given to the rules of context-sensitive ones, such that if two rules can be potentially applied to a symbol, the deterministic one will not be used.

B. Graphical representation of L-Systems

Associated with L-systems, it exists a mechanism to draw the words generated by the grammars[11]. This mechanism is commonly called “turtle interpretation”, which takes the idea from the Logo programming language[12], in which each word’s symbol (a letter) that is being interpreted graphically has a meaning for the turtle that takes it as an instruction. Typical instructions are: advance some distance to the front and draw, or not, a line under its way; turn certain degrees in a particular direction; do nothing; among others. There are, also, some symbols that do not have a graphical interpretation but are used as auxiliary commands for the interpreter, like push or pop the position in a stack. Under this scheme it is possible to generate basic figures, but it’s powerful enough to draw, at a structural level, a plant as shown in figure 1. Similarly, it is possible to extend the interpretation of the turtle to more complex structures or 3D environments as shown in figure 2.

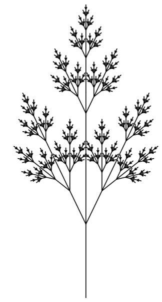


Fig. 1: Basic example of what can be achieved with L-Systems and the turtle interpretation.

Fig. 2: More complex examples of plant simulations by using formalisms.

III. Concept of death in plant simulation

Most of the approaches to plant simulation through L-systems focus on the growth process but it is equally important to be able to model the process of death in order to have a complete model of plant. In this section, the approach is to suggest a mechanism to capture the concept of death and, therefor, be able to create a visual representation of that process. For this purpose, a detailed description is given of the formal process that lead to the death semantics in L-systems.

A. Semantic death

In L-systems, each symbol of the alphabet used represents (has a meaning) a part of the plant (e.g. roots, branches, leafs), or a hint to the graphical representation (e.g. turn, draw, color, position). Using the same approach, it is being proposed to add the concept of death through the association of this meaning to a symbol (or a set of symbols) of the alphabet.

B. Death in deterministic L-systems

For the case of deterministic systems, it is sufficient to prove that there is a set of rules that allows reaching the desired string. Therefore, it is proposed to use the following simple L-system that later will be modified to incorporate the concept of death:

Let $G = \langle V, \omega, P \rangle$ where $V = \{F, +, -, [,]\}$ is the alphabet, $\omega = F$ is the axiom, and $P = \{F \rightarrow F[+F]F[-F]F\}$ is the set of production rules. This L-system, after applying a graphic interpretation (turtle interpretation), generates a branching structure similar to a plant.

If this L-system G is modified as follows:

Let $G = \langle V, \omega, P \rangle$ where $V = \{F_1, F_2, ..., F_n, M, +, -, [,]\}, n \in \mathbb{N}$ is the alphabet, for a given n, $\omega = F$ is the axiom, and

$$P = \begin{cases} F_i \rightarrow F_{i+1}[+F_{i+1}]F_{i+1}[-F_{i+1}]F_{i+1}, 1 \leq i < n \\ F_n \rightarrow M\} \end{cases}$$

is the set of production rules.

This L-system will generate the same branching structure as the unmodified one, but the L-system will iterate only n times, after which all F symbols will start to translate to M symbols and will not evolve later on. The resulting string can be given the interpretation of a dead plant. However, as it can be easily seen, the modification done only had the effect of assigning a maximum age of life to the system, this because only deterministic rules are possible in this kind of systems. Nevertheless, the goal was achieved although the dead behavior is not similar to the real one. With other kinds of L-systems, as will be explained in the next sections, this situation will be remedied.

C. Death in parametric L-systems

For this kind of L-systems will be proceed in the same manner as the exercise above, but using the advantages that give us parameterization:

Let $G_\Sigma = \langle V, \Sigma, \omega, P \rangle$ where $V = \{F, +, -, [,]\}$ is the alphabet, $\Sigma = \{t, v\}$ is the set of formal parameters, $\omega = F(t, n), n \in \mathbb{N}$ for a given n, is the axiom, and $P = \{F(t, v) \rightarrow F(t, v)[+F(t, v)]F(t, v)[-F(t, v)]F(t, v)\}$ is the set of production rules. This L-system, after being graphically interpreted, creates the same branching structure as the deterministic one.

However, if this L-system G_Σ is modified as follows:

Let $G_\Sigma = \langle V, \Sigma, \omega, P \rangle$ where $V = \{F, M, +, -, [,]\}$ is the alphabet, $\Sigma = \{t, v\}$ is the set of formal parameters, $\omega = F(1, n), n \in \mathbb{N}$ for a given n, is the axiom, and

$$P = \begin{cases} F(t, v) : t < v \rightarrow F(t+1, v)[+F(t+1, v)]F(t+ \\ \qquad 1, v)[-F(t+1, v)]F(t+1, v) \\ F(t, v) : t \geq v \rightarrow M(t, v) \end{cases}$$

is the set of production rules.

The resulting L-system will generate the same branching structure as the unmodified one, but with the existence of the two parameters will help in bringing a semantic death to the system. On the one hand, the parameter t will help counting how many iterations have lived this particular node. On the other hand, the parameter v will help to assign a maximum lifetime to this kind of node. So that, when the node reaches the specified lifetime, the node becomes a death kind of node.

The net effect obtained is similar to that of deterministic L-systems. However, the construction of the system was much simpler since it was not needed to specify a given finite quantity of symbols for the alphabet. It was just enough to add the M symbol associated with the death concept and use a strategy through the parameters to be able to reach it.

D. Death in stochastic L-systems

Will continue with the same exercise as before, but now applied to stochastic L-systems. As before, we will start with a simple L-system:

Let $G_\pi = \langle V, \omega, P, \pi \rangle$ where $V = \{F, +, -, [,]\}$ is the alphabet, $\omega = F$ is the axiom, and $P = \{F \xrightarrow{1} F[+F]F[-F]F\}$ is the set of production rules. The graphical interpretation of this system is the same branching structure as the deterministic one.

However, if this L-system G_π is modified as follows:

Let $G_\pi = \langle V, \omega, P, \pi \rangle$ where $V = \{F, M, +, -, [,]\}$ is the alphabet, $\omega = F$ is the axiom, and

$$P = \begin{cases} F \xrightarrow{0.90} F[+F]F[-F]F \\ F \xrightarrow{0.10} M \end{cases}$$

is the set of production rules.

With the given modification, each symbol of the structure will now have a chance of 90% to reproduce normally, and a chance of 10% to become a dead node. This strategy, even if it manages to incorporate the concept of death, bring a problem with it since, randomly, exists simultaneously alive and dead nodes in different positions of the branching structure, resulting in an inconsistent graphical (and biological) interpretation. This situation will be solved with the following kind of L-systems.

E. Death in context-sensitive L-systems

Following the same line of thought, this section will start describing a simple context-sensitive L-system.

Let $G = \langle V, \omega, P \rangle$ where $V = \{F, N\}$ is the alphabet, $\omega = NFFFF$ is the axiom, and $P = \{N < F \rightarrow N, N \rightarrow F\}$ is the set of production rules. This L-system has the behavior of moving the N symbol to the end of the string. (By the way, this mechanism may serve as a simulation of passing biochemical signals in the plant.)

But the same system with a slight variation, is sufficient to incorporate the concept of death:

Let $G = \langle V, \omega, P \rangle$ where $V = \{M, F\}$ is the alphabet, $\omega = MFFFF$ is the axiom, and $P = \{M < F \rightarrow M\}$ is the set of production rules. The only difference is the change of the alphabet, to follow our convention, and the elimination of a production rule. With these changes, now the L-system proposed can be interpreted by saying that once a dead node exists in the system, in the next iteration will cause the node that follows it also to become a dead node. For example, from a biological perspective, if the root is dead then there will be a lack of passage of nutrients that will result in dead of the upper nodes.

IV. Proof of concept

In this section, a more elaborated example will be given to serve as a proof of concept of the proposal mentioned above. For this purpose, a mix of the L-systems described will be used. After that, a different example is given in order to show one possible visual interpretation. Finally, some other potential uses to the mechanism described are mentioned.

A. Symbolic example

Although a formal definition for mixed L-systems will not be given, intuitively one can think of them as the union of the different types presented so far. The only consideration to take into account is the assignment of priorities to avoid ambiguity in the application of rewrite rules.

To illustrate a mixed L-system, and including the concept of death in a more elaborate way, the following system is presented:

Let $G_\Omega = \langle V, \Sigma, \omega, P\rangle$ where $V = \{S, N, M\}$ is the alphabet, $\Sigma = \{t, v\}$ is the set of formal parameters, $\omega = S$ is the axiom, and the set P of production rules contains the following elements:

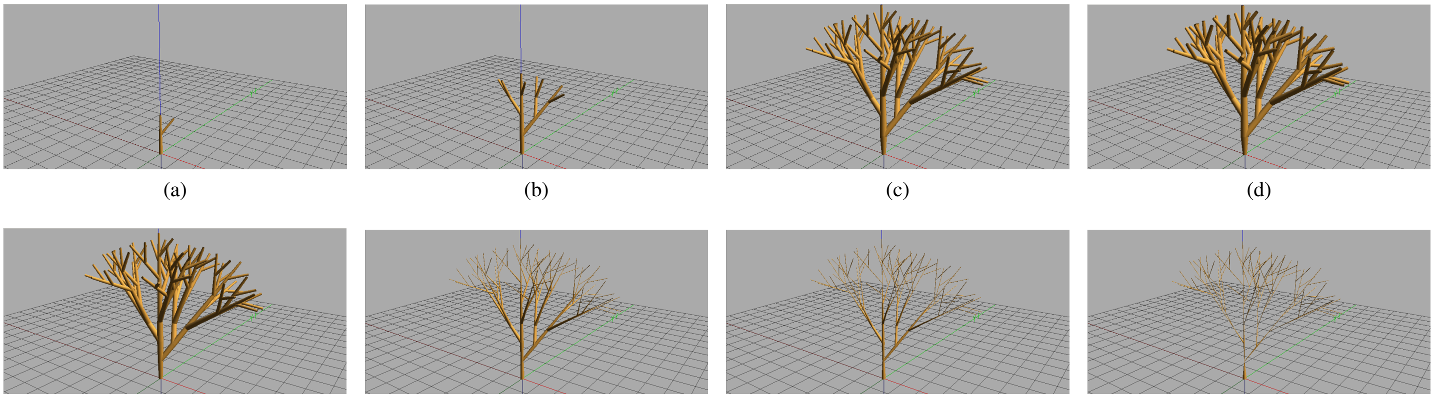
$$
\begin{aligned}
p_0 &: S \rightarrow N(1,1),\\
p_1 &: N(v,t) \xrightarrow{70\%} N(v+1,t+1),\\
p_2 &: N(v,t) \xrightarrow{15\%} N(v+1,t+1)[+N(1,1)],\\
p_3 &: N(v,t) \xrightarrow{15\%} N(v+1,t+1)[-N(1,1)],\\
p_4 &: N(v,t) : v = (70\% \cdot V\_MAX) \rightarrow N(v+1,t+1)N(1,1),\\
p_5 &: N(v,t) : v \geq V\_MAX \rightarrow N(v,t+1),\\
p_6 &: N(v,t) : t > T\_MAX \rightarrow M(v,t+1),\\
p_7 &: M(v,t) < N(v,t) \rightarrow M(v,t),\\
p_8 &: M(v,t) \rightarrow M(v-1,t+1),\\
p_{9.1} &: M(v,t) : v \leq 0 \rightarrow M(0,t+1),\\
p_{9.2} &: M(v,t) : v \leq 0 \rightarrow \epsilon,
\end{aligned}
$$

In the described L-system, the alphabet's symbols have the following meaning: the symbol S has the meaning of being the seed of a plant, the symbol N is any node that conforms a part of plant, and the symbol M is a dead node. The parameters have the same meaning as in section III-C. The axiom is composed of a single seed. The three constants used in the production rules are: V_MAX is the maximum vitality that a node can have, a value of 100 will be desired so it can be interpreted as a percentage; T_MAX is the maximum lifetime that any node of the plant can achieve, this value must be greater than V_MAX to have a consistent system, two or three times V_MAX would be suitable for simulation; and, finally, the value of 70% is an arbitrary limit that establish the point where a node can reproduce and create a new node. The vitality concept mentioned is a measure of the healthiness of a node. Finally, the explanation of the importance of each of the rewrite rules of this L-system is described below:

- Rule p_0 is the birth moment of the plant. It is in this step that a seed becomes a live plant (or a node of that plant). The node starts with a vitality of one and, since it is the first iteration of existence, its lifetime is also one.
- Rule p_1 to p_3 are stochastic rules that suggest that most of the times (70% of the times) the node will simply increase its vitality as well as its lifetime. However, there exists a 30% chance that the node will create a ramification (15% chance to left, and 15% to the right).
- Rule p_4 is a parametric way to establish that, once a node have reach 70% of the maximum vitality, it is time to reproduce and create a new node with vitality and lifetime equal to one, and that new node will be a child of the current node.
- Rule p_5 is there to avoid increasing node's vitality above the maximum possible. However, node's lifetime keeps increasing with each iteration.
- Rule p_6 is the one that kills a node once it has reach the maximum lifetime established for the nodes. From this iteration on, this node can be interpreted as being in a decomposition process.
- Rule p_7 indicates that once the node that precedes it is in a dead state, then this node must also die. The semantic reasons for this may vary but, for example, it could be mentioned the null transmission of nutrients from the parent as a possible reason.
- Rule p_8 handles the diminution of the vitality parameter of the node. This value allows an interpretation of the rottenness of this node.
- Finally, rule $p_{9.1}$ is responsible for maintaining the vitality parameter in zero (avoiding its decrement) for a given node. It is important to note that the dead node will remain in the generated string. Alternatively, rule $p_{9.2}$ can be used which, instead of leaving the dead node in the string, the symbol will be substituted with the empty symbol ϵ (that is equivalent to removing the symbol from the string). These last two rules cannot coexist simultaneously in a grammar applied to the same symbol. It must be chosen one approach or another. Or apply for different node types, for example, you keep the dead nodes for trunk parts (these do not disappear), and disappear the nodes associated with leaves (these indeed decompose relatively soon).

B. Simple graphical example

For sake of completion, in figure 3 an example is given to graphically illustrate how a simple branching structure dies. In image 3a a plant is born, passing from a seed to a small set of live nodes representing the trunk and just one branch. From image 3b to 3d, the plant grows to its maximum: the oldest nodes have reached their maximum lifetime. From image 3e to 3g the plant is in its decomposition process: each live node becomes a dead node and the vitality parameter of each one is continually decreasing. Finally, in image 3h all the nodes



(a) (b) (c) (d) (e) (f) (g) (h)

Fig. 3: Proof of concept for a plant that dies.

have zero vitality meaning a complete death. In the example, it was decided to let the branches' nodes to remain instead of disappearing from the image. The vitality parameter was used for determining the thickness of each of the segments of the plant: a thick segment is healthy segment, and a thin one is not healthy. This is a naive interpretation but it's trivial to also use colors or modify other aspects of the structure.

The image was created with OpenAlea platform[13] and the PlantGL[14] and L-Py[15] modules.

C. Other possibilities

As it was shown in section III-C, by using parametric L-systems and a value that express the lifetime of a node, a rule can be triggered that converts a living node to a dead one. The same can be done to express other biological concepts and that will imply the plant's dead (directly or indirectly).

For example:

- If it is proposed to have a variable that represents water availability in the root of a plant, then the rule $p_{water} : R(v, W) : W < 0 \rightarrow R(v-1, W)$ will give the meaning of root R not being hydrated ($W < 0$, water not available) and therefore diminishing its vitality (and eventually dying).
- If it is proposed to have a variable that represent sunlight, then the rule $p_{leaves} : L(v, S) : S < 0 \rightarrow L(v-1, S)$ will imply that a leave L is not receiving sunlight ($S < 0$) with its associated negative effect.

Other possibilities include considering soil quantity for the roots, temperature for each node, nutrients available in the environment and so on.

V. Conclusion

The purpose of this paper was to provide a mechanism to include the concept of death in the simulation of plants through L-systems. With this achievement, plants' cycle of life is complete. In section III, a systematic process was used to show that it was possible to have the semantic meaning of dead in four different kinds of L-systems. Even though there could be inconsistencies, from a biological perspective, in the developmental process of the simulation of a plant, those issues could be tackled by using a mix of the different kind of L-systems. The key point for the successfully inclusion of death in this kind of simulations is to appoint one, or more, of the symbols of the alphabet with the meaning of death, and therefor have a set of auxiliary production rules that create the transition from a living node to a dead one in a convenient step so that it is congruent in a biological context. Also, a simple graphical interpretation of a dying plant was shown.

Acknowledgment

This research was supported in part by the Consejo Nacional de Ciencia y Tecnologia (CONACYT), grant number 219074, and Centro de Investigacion y Estudios Avanzados (Cinvestav).

References

[1] C. G. Langton, "Studying artificial life with cellular automata," *Physica D: Nonlinear Phenomena*, vol. 22, no. 13, pp. 120 – 149, 1986, proceedings of the Fifth Annual International Conference. [Online]. Available: http://www.sciencedirect.com/science/article/pii/016727898690237X

[2] J. Riskin, "The Defecating Duck, or, the Ambiguous Origins of Artificial Life," *Critical Inquiry*, vol. 29, pp. 599–633, 2003.

[3] P. Prusinkiewicz and A. Lindenmayer, *The algorithmic beauty of plants*. New York, NY, USA: Springer-Verlag New York, Inc., 1996.

[4] P. Prusinkiewicz, A. Lindenmayer, and J. Hanan, "Development models of herbaceous plants for computer imagery purposes." in *SIGGRAPH*, R. J. Beach, Ed. ACM, 1988, pp. 141–150.

[5] A. Lindenmayer, "Mathematical models for cellular interaction in development: Parts i and ii." *Journal of Theoretical Biology*, vol. 18, 1968.

[6] N. Chomsky, "Three models for the description of language," *Information Theory, IRE Transactions on*, vol. 2, no. 3, pp. 113–124, 1956.

[7] G. Rozenberg and A. Salomaa, *Mathematical Theory of L Systems*. Orlando, FL, USA: Academic Press, Inc., 1980.

[8] A. Lindenmayer and G. Rozenberg, "Developmental systems and languages," in *Proceedings of the Fourth Annual ACM Symposium on Theory of Computing*, ser. STOC '72. New York, NY, USA: ACM, 1972, pp. 214–221. [Online]. Available: http://doi.acm.org/10.1145/800152.804917

[9] J. Hanan, "Parametric l-systems and their application to the modelling and visualization of plants," Ph.D. dissertation, University of Regina, 1992.

[10] T. Yokomori, "Stochastic characterizations of {E0L} languages," *Information and Control*, vol. 45, no. 1, pp. 26 – 33, 1980. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0019995880908372

[11] P. Prusinkiewicz, M. Hammel, J. Hanan, and R. Mech, "L-systems: from the theory to visual models of plants," in *Proceedings of the 2nd CSIRO Symposium on Computational Challenges in Life Sciences*, vol. 3. Citeseer, 1996, pp. 1–32.

[12] H. Abelson and A. A. diSessa, *Turtle geometry*. M.I.T. Press, Cambridge, 1982.

[13] C. Pradal, S. Dufour-Kowalski, F. Boudon, C. Fournier, and C. Godin, "Openalea: A visual programming and component-based software platform for plant modeling," *Functional Plant Biology*, vol. 35, no. 9 & 10, pp. 751–760, 2008. [Online]. Available: http://www-sop.inria.fr/virtualplants/Publications/2008/PDBFG08a

[14] C. Pradal, F. Boudon, C. Nouguier, J. Chopard, and C. Godin, "PlantGL : a Python-based geometric library for 3D plant modelling at different scales," INRIA, Research Report RR-6367, 2007. [Online]. Available: http://hal.inria.fr/inria-00191126

[15] F. Boudon, C. Pradal, T. Cokelaer, P. Prusinkiewicz, and C. Godin, "L-py: an l-system simulation framework for modeling plant development based on a dynamic language," *Frontiers in technical advances in plant science*, 2012. [Online]. Available: http://www-sop.inria.fr/virtualplants/Publications/2012/BPCPG12