L-System: Irregular Tree Growth Simulation

Amanda Showler and Joshua Trower
Faculty of Science
Ontario Tech University

April 12, 2023

Abstract

L-systems are used to model and simulate plant growth using an initial state and sequentially applied rewrite rules. This paper outlines an approach to the L-system simulation to account for the irregular shape of trees when generating multiple trees. The branch thickness is lowered for every level in the hierarchy away from the trunk. The angle of the branch, and the rewriting rule are pseudo-randomly generated. It was determined that random number generation for segment length did not improve the tree and it was removed. The simulation has the functionality to generate multiple trees sequentially so the variable shapes can be observed.

1 Introduction

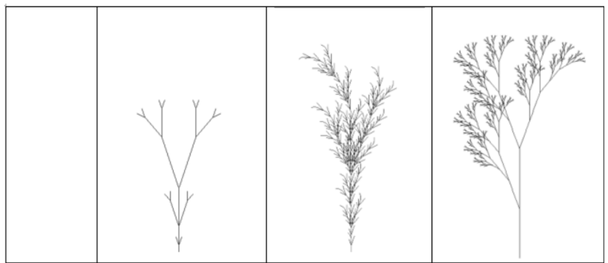
Lindenmayer systems, or L-systems, is a rule-based, parallel rewriting system. It provides a formalism to model the complex geometry of plant structures [9]. The branching structure of plants has fractal architecture. For this reason, L-systems are used to model the architecture by generating fractal patterns.

The formation of plant structures is a complex process that can be developed in parallel using rewriting rules. In sequence, these rules are used to replace each part of the current structure. They are defined by parameters including the axiom, angle, set of production rules, and the number of iterations. Figure 1 shows the plant structure modeled from a range of L-system parameters.

The L-system and the turtle graphics interpretation can be used to model tree architecture. The bracketed string is interpreted as a set of turtle commands to update the state of the end of each branch [18].

L-systems are used to model plants that range from algae [22] to trees [8], and roots [10]. For this paper, the goal is to simulate trees. Tree architecture is irregular and can be categorized as not entirely deterministic or random in nature [23]. Plant development is affected by environmental phenomena, including resources, gravity, and temperature [14].

To simulate multiple trees growing with variability, the features of the simulation include adding thickness to the branches, and four predefined rewrite rules randomly selected during each iteration. Our tree simulation incorporates branching asymmetry and the decreasing diameter of branches through iterations. This simulation utilizes L-systems, turtle graphics navigation rules, the Python



Axiom	X	F	X
Angle	20	20	20
Rules	X->FF[+FFX][-FFX] F->FF	F->[-FF]+[FFF]-FF[-F-F]	X->F[+X]F[-X]+X F->FF
Iterations	3	3	4

Figure 1: The models constructed dependant on L-system parameters.

language, and libraries Pygame and random to simulate randomly generated trees. The implementation uses parameters including axiom, angle, and the number of increments to randomize the simulated plant.

The code for the simulation is available at the following Github link:
https://github.com/Hollowhking/Treev2

2 Related Work

In 1968, Aristid Lindenmayer introduced a mathematical model for biological structures by using parallel rewriting systems [11]. This introduced the notion of bracketed L-systems to describe the branching structure of plants. Smith proposed L-system modeling for developing realistic plants in computer graphics and noted the relationship between fractals and L-systems [25]. Later, geometric modeling of L-systems was proposed for use in plant modeling by Prusinkiewics *et al.* [19].

2.1 Parallel Rewriting Systems

The parallel rewriting system is the substitution of parts of a simple initial structure using production rules. Since plants grow in parallel steps, parallel rewriting is imperative for the modeling of plant structures. L-systems are a special type of character-based rewrite system where the letters within the string are replaced in parallel [6].

Parallel rewriting systems can be written in the form of L-systems for modeling the growth of plants. For every module m within the axiom, a production rule is paired if the predecessor is equal to the module letter [12]. For each iteration, each module m is rewritten based on the successor.

An L-system is a set formed by an alphabet of symbols, an axiom, and a set of production rules. Modules are the units that are used to build the plant structure. The set of production rules is represented in a bracketed string to describe the branching structure of a plant [3]. The axiom or predecessor is a string of modules to represent the initial state of the simulation. The '→' notation is used to separate the axiom and the set of production rules. The letters in the string denote modules.

The bracketed string has an alphabet of symbols to dictate the plant structure (e.g., 'X→FF[+FFX][-FFX]'). Every element is replaced according to the production rule. The number of iterations is given by the increment parameter. Figure 2 shows the changes through the first three iterations of an L-system model. Each iteration has a different amount of zoom to scale the models for easier viewing.

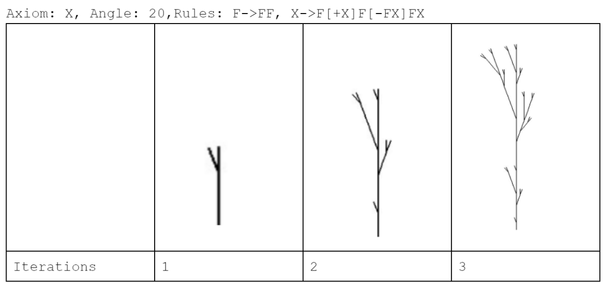


Figure 2: The first to third iteration of an L-System

2.2 Fractals

Fractals are spatial model systems where each part has the same pattern independent of scale. They use iterative algorithms and scaling rules to model structures where a pattern is recursively observed [15].

Fractals are built by recursively adding a pattern onto a starting object [13]. To represent a fractal by L-systems, the strings must be interpreted with geometric information by using a graphical interpretation [1]. The bracketed strings and L-systems were extended to computer graphical applications using the turtle graphics interpretation [17]. This extended the turtle commands by the set $\{[\ ,]\}$ to represent the branching structure of trees. Each set of brackets contains branches that are attached to the tree by the module defined one index left of the '['.

The state of a turtle is a tuple of coordinates (x_i, y_i, α), where x_i, y_i denote turtle position and α denotes turtle heading. For each step, the position in the bracketed string is incremented by one to get next command. The following symbols are used for turtle navigation [19]:

Symbol that controls turtle movement

F(d) Move forward step of length d. The state of the turtle is updated to $(x_{i+1}, y_{i+1}, \alpha)$ where $x_{i+1} = x_i + d\cos(\alpha)$ and $y_{i+1} = y_i + d\sin(\alpha)$. The line is drawn between the points (x_i, y_i) and (x_{i+1}, y_{i+1}).

Symbols that control turtle orientation

+(β) Rotate left by angle β. The state of the turtle is updated to $(x_i, y_i, \alpha + \beta)$.

-(β) Rotate right by angle β. The state of the turtle is updated to $(x_i, y_i, \alpha - \beta)$.

Symbols that model structures with branches

[Push the state of the turtle onto the stack.

] Pop the state of the turtle from the stack and replace the current state of the turtle with it.

2.3 Tree Structure

The architecture of trees is not entirely deterministic or random and it is likely the result of balancing tree stability and light-capturing efficiency [23]. As mentioned in Section 1, the irregular shape of trees is affected by resources, gravity, and temperature [14]. The resources that affect plant growth are light, water, and nutrients. The role of light in the shape of coleoptiles was studied using an asymmetrical light source and it resulted in the elongation of the shaded side so the coleoptile bends toward the light source [16]. This paper outlines how the shape of the plant is affected by the light resource. Trees have varying branch diameters between the parent and the set of sub-branches [7].

Pseudo-random numbers appear random but are generated deterministically. Random processes can be used to approximate phenomena [21]. Random L-systems are parametric L-systems that have a set of pseudo-random variables to introduce irregular patterns [20].

The irregular structure of the bonsai tree was modeled using an L-system process involving decomposition graphs to manipulate hierarchical parameters [4]. The approach to use more than one predefined generating rule was used in an L-system model that had growth direction defined by the user [5].

3 Methodology

The goal of the simulation is to model the irregular growth of trees using pseudo-randomly generated parameters and production rules. The graphical interpretation of L-systems used is turtle graphics described in Section 2.2 where the state is determined using a starting state and a set of production rules. The functions to update rules for the state of the branch are the functions from turtle graphics, where d denotes segment length and α is the growth angle. The state of the end of the tree branch is stored in an array in the following form, ($posx$, $posy$). The state of the angle and thickness of the branch are stored in their own respective array.

The parameters will be defined by assignment, the user, or random number generation. The production rule, angle, and segment length will be pseudo-randomly generated using the Python library, random. The number of iterations and trees generated will be determined by the user. The remaining parameters will be assigned and calculated within the simulation code.

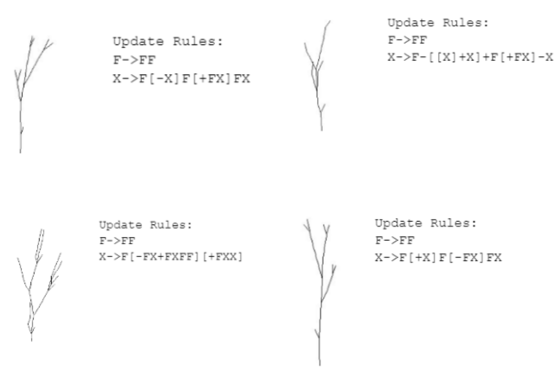


Figure 3: The four predefined production rules to 2 iterations

As a proxy for modeling irregular tree growth from environmental phenomena, the rewriting rule, segment length, and angle were randomly updated for each iteration. There are four predefined rewriting rules that are pseudo-randomly selected for each iteration. Figure 3 shows the model generated by each of the predefined rules. The rules were determined based on the level of similarity so the each tree is cohesive.

To simulate the tree(s) growth, a timer is set to 0.05 seconds for each branch. The equations to update the state of the end of the branch are outlined below using the variable names from the simulation code:

$$posx_{i+1} = posx_i + seg\_length \cdot \cos(grow\_angle) \tag{1}$$

$$posy_{i+1} = posy_i + seg\_length \cdot \cos(grow\_angle) \tag{2}$$

The thickness of branches can be scaled based on order to produce more realistic results. A tree will start at the maximum segment thickness which is calculated at the beginning of the simulation using Equation 3. The trunk of a tree is thicker than the subsequent branches. The thickness of branches is updated in conjunction with the state of the end of each branch using Equation 4. Figure 4 shows the varying thickness of a tree with three iterations and random production rule selection.

$$segment\_thickness = iterations * 5 \tag{3}$$

$$segment\_thickness = segment\_thickness - 1 \tag{4}$$

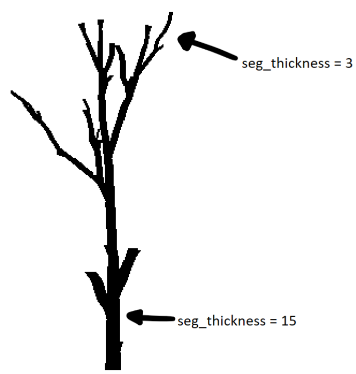


Figure 4: The varying thickness of branches from 3 to 15 for a tree with three iterations

4 Experiments and Results

The results of the randomly generated parameters for the model are compared to deterministic techniques. The simulation features command-line arguments for experimentation with the number of iterations and the number of trees generated. The first argument determines the number of iterations which can be either 2 or 3, where the height and complexity of the tree change. The second argument is the number of trees generated which ranges from 1 to 10. We observed as the number of trees generated increases, the chance of collision between them is higher.

The range of the segment length, d, and the growth angle, α can be adjusted to simulate trees with different shapes. The result of random and determined segment length for three iterations is shown in Figure 5. The result of Experiment 1 for the segment length randomly generated shows

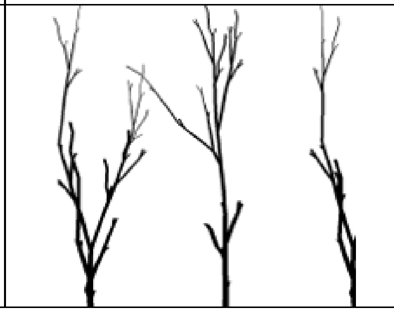
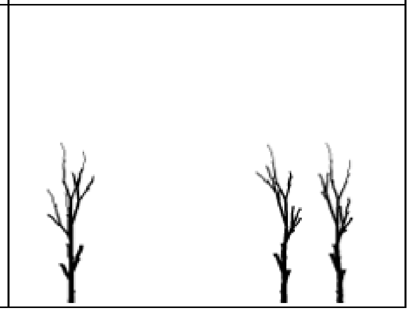
	Experiment 1 - Random Segment length	Experiment 2 - Static Segment length
Seg_length Assignment	`random.randint(15, 25)`	`10`
Simulation Result		

Figure 5: The comparison of random and deterministic segment length

elongated descendant branches. In experiment 2, the segment length was adjusted to be equal to 10 which caused the shape of the tree to have the expected branch length.

The simulation result produced trees that grew to the left as shown in Figure 6 Experiment 1. Since the result was observed in one of multiple trees, it was determined a random parameter was causing the problem. After further experimentation, it was discovered that the production rule "$X \rightarrow F - [[X] + X] + F[+FX] - X$" was causing the unrealistic angle shift to the left. The rule was commented out and the problem was resolved which is pictured in 6 Experiment 2.

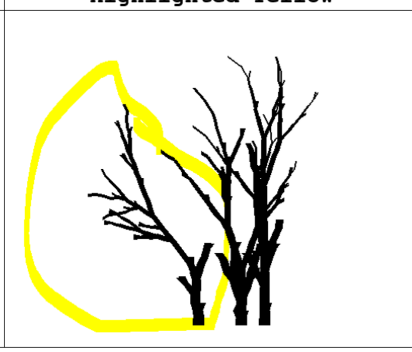
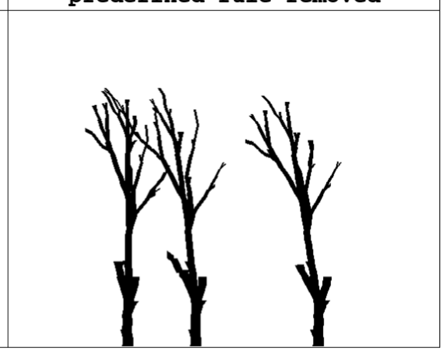
	Experiment 1 - Left Tree Highlighted Yellow	Experiment 2 - Second predefined rule removed
Simulation Result		

Figure 6: The comparison between random generated tree before and after second rule removed

The simulation was executed with random rule generation for each iteration to provide randomness to branch growth. The constant parameters were an axiom of 'X' and a growth angle of 20°. The changing parameters were the number of iterations the tree would 'grow', the base segment thickness as well as the number of trees. The location of the trees drawn was randomly distributed on $y = 0$ the figure below shows a base segment thickness of 10 for 3 iterations and a tree count of 2:

As mentioned in Section 3, the thickness of the tree can vary based on the number of descendants from the trunk. During experimentation, the thickness for three iterations showed no change in branch thickness for the third iteration which is shown in Figure 7 Experiment 1. The method used for calculating branch thickness for this figure was increasing the thickness by 2 for a new branch and decreasing thickness by 3 at the end of a branch. To improve the branch thickness, the method used for the growth angle where the value is stored in an array that is utilized as a stack was used. This improved the result of the thickness feature which is shown in Figure 7 Experiment 2.

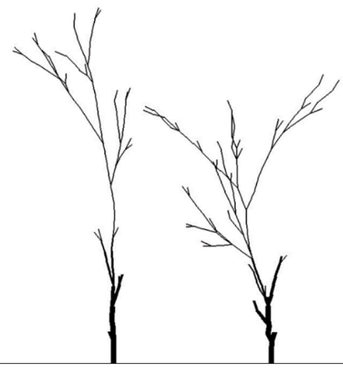
	Experiment 1 - Increase/Decrease Thickness	Experiment 2 - Stack Data Structure Thickness
Simulation Result		

Figure 7: The comparison of thickness assignment methods

The result of two trees generated from the random selection of a predefined production rule compared to two trees generated with one production rule is shown in Figure 8. The production rule used for Experiment 1 was the first rule in the set. The results show that a random selection of production rules can be used to introduce variability into the tree generation for applications where more than one tree will be generated.

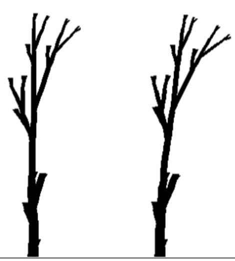
	Experiment 1 - One Production Rule	Experiment 2 - Random Selection of Production Rule
Simulation Result		

Figure 8: The comparison of production rule selection method

A background feature was added to add context to the trees generated by the simulation and it is shown in Figure 9. The background image is an original piece of artwork for the simulation titled 'A Cold Morning' [24]. The feature is toggled by the third command line argument, where 0 is no background and 1 has a background.

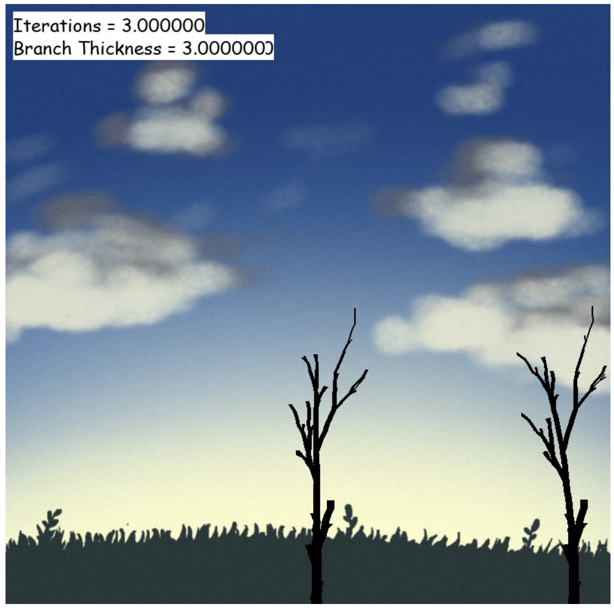


Figure 9: The simulation with the background feature

5 Conclusion

L-systems can be used in computer graphics to simulate plants that have complex structures. We utilized L-systems, turtle navigation, and Pygame to develop an asymmetrical plant growth simulation with varying branch descendent diameters.

This approach can be used to show the variable growth development of multiple trees. In Section 4, it was noted the randomly generated parameter, segment length, did not result in a more realistic tree structure and was removed from the simulation code. The best method to implement the thickness of the tree was determined to be using a stack data structure to store and assign thickness so it is uniform for each descendent level. The result of the randomly selected production rule showed the approach can be used when there are multiple trees to be generated with variability in shape.

The future areas of research are extending collision detection and response for the random production rules as tree collisions are apparent. The number of predefined production rules can be expanded to include more diverse trees. The thickness of the branches could be further extended to taper the branch so the diameter decreases as the branch length increases.

References

[1] Manuel Alfonseca and Alfonso Ortega. Representation of fractal curves by means of l systems. In *Proceedings of the Conference on Designing the Future*, APL '96, page 13–21, New York, NY, USA, 1996. Association for Computing Machinery.

[2] Manuel Alfonseca and Alfonso Ortega. Representation of fractal curves by means of l systems. *SIGAPL APL Quote Quad*, 26(4):13–21, jun 1996.

[3] Frederic Boudon, Christophe Pradal, Thomas Cokelaer, Przemyslaw Prusinkiewicz, and Christophe Godin. L-py: An l-system simulation framework for modeling plant architecture development based on a dynamic language. *Frontiers in Plant Science*, 3, 2012.

[4] Frédéric Boudon, Przemyslaw Prusinkiewicz, Pavol Federl, Christophe Godin, and Radoslaw Karwowski. Interactive design of bonsai tree models. *Comput. Graph. Forum*, 22:591–600, 09 2003.

[5] Takashi Ijiri, Shigeru Owada, and Takeo Igarashi. The sketch l-system: Global control of tree modeling using free-form strokes. In *Lecture notes in computer science*, Lecture Notes in Computer Science, pages 138–146, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[6] Christian Jacob. Evolving evolution programs: Genetic programming and l-systems. pages 107–115, 07 1996.

[7] KDS Jinasena and DUJ Sonnadara. Computer simulation of tree development with random variations and probabilistic growth of branches. 2013.

[8] Winfried Kurth, Branislav Sloboda, et al. Growth grammars simulating trees–an extension of l-systems incorporating local variables and sensitivity. 1997.

[9] Daniel Leitner, Sabine Klepsch, Astrid Knieß, and Andrea Schnepf. The algorithmic beauty of plant roots – an l-system model for dynamic root growth simulation. *Mathematical and Computer Modelling of Dynamical Systems*, 16(6):575–587, 2010.

[10] Daniel Leitner and Andrea Schnepf. Root growth simulation using l-systems. In *Proceedings of ALGORITMY*, pages 313–320. Citeseer, 2009.

[11] Aristid Lindenmayer. Mathematical models for cellular interactions in development i. filaments with one-sided inputs. *Journal of Theoretical Biology*, 18(3):280–299, 1968.

[12] Markus Lipp, Peter Wonka, and Michael Wimmer. Parallel generation of multiple l-systems. *Computers Graphics*, 34(5):585–593, 2010. CAD/GRAPHICS 2009 Extended papers from the 2009 Sketch-Based Interfaces and Modeling Conference Vision, Modeling Visualization.

[13] Benoit B Mandelbrot and Benoit B Mandelbrot. *The fractal geometry of nature*, volume 1. WH freeman New York, 1982.

[14] Radomir Mech. *Modeling and simulation of the interaction of plants with the environment using L-systems and their extensions*. University of Calgary, 1997.

[15] E. Perfect and B.D. Kay. Applications of fractals in soil and tillage research: a review. *Soil and Tillage Research*, 36(1):1–20, 1995.

[16] Barbara Gillespie Pickard and Kenneth V Thimann. Transport and distribution of auxin during tropistic response. ii. the lateral migration of auxin in phototropism of coleoptiles. *Plant Physiology*, 39(3):341, 1964.

[17] Przemyslaw Prusinkiewicz. Applications of l-systems to computer imagery. In *Graph-Grammars and Their Application to Computer Science*, Lecture Notes in Computer Science, pages 534–548. Springer Berlin Heidelberg, Berlin, Heidelberg, 1987.

[18] Przemyslaw Prusinkiewicz. Modeling plant growth and development. *Current Opinion in Plant Biology*, 7(1):79–83, 2004.

[19] Przemyslaw Prusinkiewicz, Mark Hammel, Jim Hanan, and Radomir Mech. L-systems: from the theory to visual models of plants. In *Proceedings of the 2nd CSIRO Symposium on Computational Challenges in Life Sciences*, volume 3, pages 1–32. Citeseer, 1996.

[20] R. Quiros, J. Lluch, M. Chover, and R. Vivó. Geometric substitution using random l-systems. *Computers Graphics*, 20(5):713–721, 1996.

[21] Faisal Qureshi. *Random Processes Lecture Notes*. Ontario Tech University, 2023.

[22] C.W. Schneider, R.E. Walde, and R.A. Morelli. L-systems computer models generating distichous from spiral organisation in the dasyaceae (ceramiales, rhodophyta). *European Journal of Phycology*, 29(3):165–170, 1994.

[23] Dominik Seidel, Peter Annighöfer, Melissa Stiers, Clara Delphine Zemp, Katharina Burkardt, Martin Ehbrecht, Katharina Willim, Holger Kreft, Dirk Hölscher, and Christian Ammer. How a measure of tree structural complexity relates to architectural benefit-to-cost ratio, light availability, and growth of trees. *Ecology and evolution*, 9(12):7134–7142, 2019.

[24] Erika Showler. *A Cold Morning*. Digital Art, 2023.

[25] Alvy Ray Smith. Plants, fractals, and formal languages. In *Proceedings of the 11th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '84, page 1–10, New York, NY, USA, 1984. Association for Computing Machinery.

[]