From Modeling Foliage with L-systems to Digital Art

Glyn M. Rimmington
glyn.rimmington@wichita.edu

Mara Alagic
mara.alagic@wichita.edu

Wichita State University
1845 N Fairmount
Wichita, Kansas, 67260-0142 USA

Abstract

This paper provides many *bridges*: it connects graphical representations of plants' evolving properties with structural geometric representations; a visualization of plants' structure, from a simple branching formation to more complex plant architectures, and further to aesthetically pleasing, simulated, 3D plant-like representations. This is accomplished by use of L-systems in combination with Turtle Graphics, being displayed in a VRML viewer and then enhanced in terms of the background with Adobe Photoshop™. This process leads to many questions, such as: Can we take an artistic license and design these kinds of artificial plant forms? What creative value do corresponding pictures have? What creative value is in understanding the process of their design and related dynamic representation? Recognizing that 2D media does not do justice to the dynamics of the 3D-based processes, the authors are preparing an animated representation of the design process that will be incorporated in the Conference CD.

Introduction

Just as the path to understanding the nature of light and gravity by Einstein involved imagining the sensation of moving at the speed of light, or riding a photon, we can improve our understanding of the branching structure of plants by riding along on the embryonic plant tissue that is actively dividing, such as is found at the tip of stems and roots, the meristems. The average observer of plants and their structure rarely appreciates the geometric structure beyond van Hiele's [1] first level of geometric reasoning, that of visualization [2]. Further understanding at higher levels of van Hiele's geometric reasoning may be achieved through riding the graphical turtle of Abelson and diSessa [3] and adopting the viewpoint of a plant meristem. The meristem comprises undifferentiated cells at growing points on shoots and roots. Unlike in other parts of a plant, where differentiated cells will divide to produce only cells of the one type, meristematic cells can produce different types of cells and subsequently different types of plant tissue and organs, such as stems, leaves, roots, flowers, fruit and seeds. By riding on the tip of a meristem, we can trace the growth trajectory of individual internodes (stems) to nodes, or positions of further meristems, where the plant branches. This can be achieved by using L-systems combined with turtle geometry [3].

In this paper, we step through the process of riding the meristem(s) to produce plant-like graphical images for simple, unbranched plants through to more complex, highly branched structures using the combination of L-systems and turtle graphics. The recursive nature of L-systems allows the rules for achieving such branching patterns to be succinct. Often, less than two lines of instructions can produce images that are remarkably like real plants. On achieving higher levels of geometric reasoning [1], such as being able to articulate different types of branching (Level 2: Analysis), making and testing hypotheses about L-system rules and consequent changes to branching patterns (Level 3: Informal Deduction), constructing proofs using axioms, definitions and rules (Level 4: Formal Deduction), we can move beyond concrete examples

that can be found in nature to unexplored areas of plant structure using abstraction of L-systems and turtle graphics (Level 5: Rigor). The latter is where we can become creative as artists and use the mathematics of L-systems [4] and turtle graphics to make new plant-like patterns that don't exist in reality, but which might have an aesthetic appeal.

L-systems

The idea of L-systems emerged from analyses of the growth and branching patterns of filamentous algae by Aristid Lindenmayer [5], after whom they are named. Symbols in a string were mapped to different types of cells in the filament. On cell division, by a *parent* cell, the types of cells that result, *child* cells, depend on the type of parent cell that undergoes division. Sometimes, the type of child cells also depends on the type of cells surrounding the parent cell.

Parametric L-system grammars are parallel string rewriting systems and are formally defined as a 4-tuple:

> **G** = {*V*, *S*, ω, *P*}, where
> **V** (the *alphabet*) is a set of symbols containing elements that can be replaced (*variables*)
> **S** is a set of symbols containing elements that remain fixed (*constants*)
> **ω** (*start*, *axiom* or *initiator*) is a string of symbols from **V** defining the initial state of the system
> **P** is a set of *rules* or *productions* that define the way variables (predecessor) are replaced by combinations of constants and variables (successor). In a production rule the predecessor is replaced by the successor. In this language the symbol "=" represents "is replaced by."

The productions, or rules, are applied iteratively, beginning with the axiom string, but a given symbol from V can appear in both successors and in predecessors in a circular fashion, so that the resultant behavior will be recursive. This leads to the pattern of self-similarity, so often observed in real plants [4].

Parametric L-systems and Turtle Graphics

A significant breakthrough came in 1986, when L-systems were combined with Turtle Graphics [6]. In the latter, a set of symbols determine the behavior of a Turtle in 3-space. For example, F, commands the Turtle to move forward (in the direction it is facing) a defined distance and draw a line along that path. Other commands determine changes in direction in 3-space in much the same way a pilot flies an aircraft: by yawing about a z-axis (+/-), rolling about an x-axis (>/<) or pitching about a y-axis (&/^).

A further enhancement to the symbol set was the addition of "parameters". For almost any instruction, a parameter can be specified. For example F(.8) instructs the turtle to move forward and draw a line (or cylinder) eight tenths of the default length for F. >(45) instructs the turtle to roll 45°, c(7) instructs the turtle to use color number 7, when drawing.

By using a combination of Turtle and non-Turtle symbols in L-systems, complex patterns can be produced with a succinct set of instructions. For example,

```
recursion 4
angle 30
axiom FA
rule A = +FA
rule F = FF
```

after 4 iterations, will produce the string,

FFFFFFFFFFFFFFF+FFFFFFFF+FFFF+FF+FA

This process, while iterative, is referred to as recursive because patterns are repeated within patterns. In other words the resulting string exhibits self similarity at different scales. This makes more sense, when the symbols are translated into a graphical path by the Turtle.

Some additional symbols in the Turtle Graphics language allow the drawing of branches or polygonal surfaces. The symbols "[" and "]" are used to allow branching. The symbol "[" instructs the Turtle to remember where it is and what direction it is facing and the symbol "]" instructs to the Turtle to return to that position and face in that direction after carrying out instructions enclosed by these symbols.

The symbols "{" and "}" enclose symbols that result in the painting of a surface bounded by a polygon. The polygon is defined by the symbols "g" and ".". The symbol "g" is similar to an "F" in that the Turtle is instructed to move forward and draw an edge. The symbol "." instructs the Turtle to remember a vertex. The symbol "}" closes the polygon, by causing the Turtle to resume the position of the first vertex. The edges and vertices form the polygon.

By combining all these symbols, a variety of plant-like structures can be produced. The examples presented in this paper use the lparser software [7, 8] with the generated plants being displayed in a VRML viewer (available from the same site) and are then enhanced in terms of the background with Adobe Photoshop™.

Branching Patterns in Plants

The simplest plant structure is an unbranched plant in which only the apical meristem is active. This results in a structure that may resemble a palm tree. The axiom for such a structure could comprise a stem (S) and a whorl (W) of leaves (L) that appear at the apex. The eight leaves are positioned in the whorl by rolling about the stem axis in increments of 45° using the > instruction.

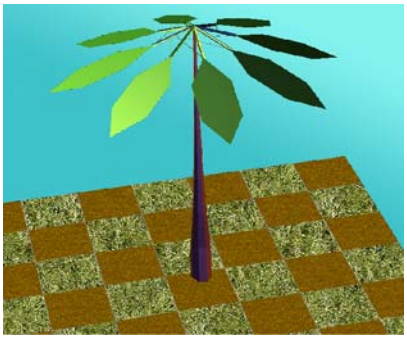

Figure 1: *Unbranched plant.*

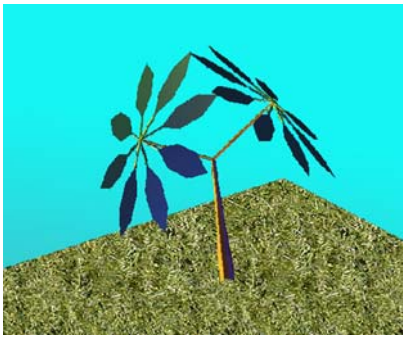

Figure 2: *Plant with apical branching.*

For Figure 1 there are four instructions that precede the axiom and rules. The "recursion" instruction dictates the number of times the rules are iterated. The "switch_yz" instruction swaps the y and z axes so the y axis projects out from the scene toward the person viewing it, while the z axis runs up and down on the plane of the scene. The "shape" instruction tells the turtle to draw cylinders. Finally, the "thickness" instruction specifies the diameter of the cylinders as a percentage of the unit length resulting from "F". The substring "&(-90)>(90)-(90)" orients the turtle to be facing up on the y axis. The next instruction

"c(12)" specifies a color that is close to that of the stem of a plant. In the first rule, the "!" symbol reduces the diameter of the stem by 10%, so the sequence F!F!F!F!F!F results in a tapering stem.

```
recursion 3  #iterate the rewriting 3 times

switch_yz 1  #switch the y and z axes
shape 1   #use cylinders for F
thickness 50  #diameter is 50% of length

#orient the turtle, set the color and start with S (stem) and W (whorl)
axiom &(-90)>(90)-(90)c(12)SW

# draw the stem with ! causing it to taper by 10% per F
rule S = F!F!F!F!F!F

# establish the whorl with 8 arms
rule W = L>(45)L>(45)L>(45)L>(45)L>(45)L>(45)L>(45)L

#draw a leaf with two surfaces using an elongated hexagon
rule L = [&(110)c(4)F[{.+(30)g(.866).-(30)g.-(30)g(.866).-(120) g(.866).-(30)g.-(30)g
        (.866) g.}]> (180)c(4) {.+(30)g(.866).-(30)g.-(30)g(.866).-(120)g(.866).-(30)g.-
        (30)g (.866)g.}]
```

The symbols for drawing a polygon, {. g. g.} mentioned earlier, are used to create the leaf. Each polygon is visible on one side only. So the turtle has to be rolled 180° to create the second surface. Instructions to draw polygons or branches or any structure that is repeated are enclosed in [ ] symbols as "branches" in an abstract sense, to isolate those instructions from their context. Instructions for a leaf "L" are enclosed in [ ] symbols. Similarly, the instructions to create two surfaces of the leaf are also enclosed in [ ] symbols.

In Figure 2, we see an example of apical branching, which results in two branches and whorls of leaves.

```
recursion 3
switch_yz 1
shape 1
thickness 50

# at the top of the stem create two branches on opposite sides
axiom &(-90)>(90)-(90)c(12)S[BW]>(180)[BW]

rule S = F!F!F!F!F

# incline the branches at 60 degrees
rule B = &(60)F!F!F

rule W = L>(45)L>(45)L>(45)L>(45)L>(45)L>(45)L>(45)L

rule L = [&(100)c(4)F[{.+(30)g(.866).-(30)g.-(30)g(.866).-(120)g (.866).-(30)g.-
        (30)g(.866)g.}]> (180)c(4){.+(30)g(.866).-(30)g.-(30)g(.866).-(120)g(.866).-(30)g.-
        (30)g(.866)g.}]
```

Many plant species have lateral meristems that can progress to being lateral branches. In nature it is possible to find a variety of lateral branching patterns, including opposite and alternate as well as phyllotactic. In the next example (Figure 3), we present a tree with phyllotactic branching. As meristems produce branches higher on the trunk, they are separated around the vertical axis by the "golden" angle of 137.5°.

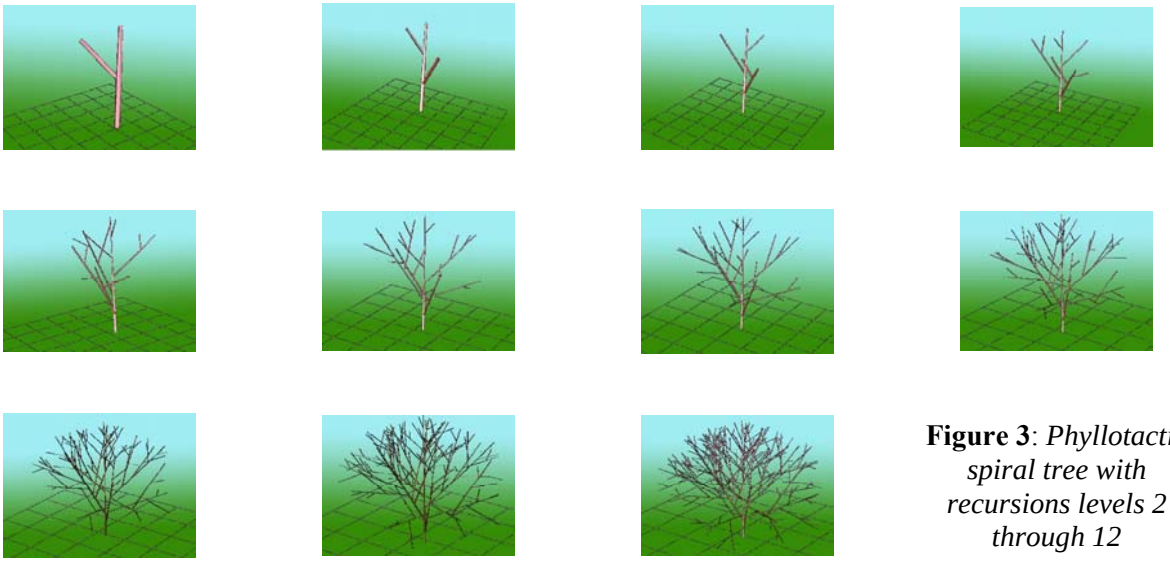

Figure 3: *Phyllotactic spiral tree with recursions levels 2 through 12*

```
recursion 12

switch_yz 1
shape 1
thickness 10

axiom &(-90)>(90)-(90)c(12)A

rule A = FBA
rule B = !>(137.5)[&(45)FA]
```

In Figure 3 the axiom is A and on each iteration A is replaced by a length of trunk or branch (F), a branch (B) and A. In turn, B produces a branch with an inclination angle of 45 and the branches are separated by rolling about the vertical axis by 137.5. The same pattern, which occurs on the main trunk, also occurs on every branch. In this model there is no restriction on the level of branching. The development of this tree can be seen in Figure 3 in which each frame represents the result for between 2 and 12 levels of recursion.

In a process similar to that used for the phyllotactic tree, it is possible to generate a Dandelion-like inflorescence. Figure 4 illustrates that process. The initial structure contains 6 axes, which subsequently produce four branches and each of those a further four branches. After just six iterations, the structure takes on an appearance like a Dandelion inflorescence. Changing lighting effects in VRML environment allows for more or less resemblance with a white dandelion from nature (Figure 6).

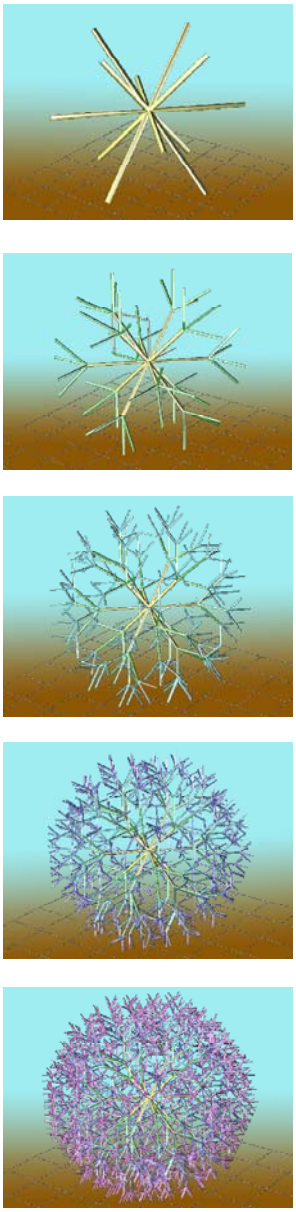

Figure 4: *Generating L-dandelion*

```
recursion 6

# switch_yz 1
angle 45
shape 1
thickness 5

axiom &(90)+(90)[Q][-(90)Q][-(180)Q][-(270)Q]

rule Q = ["(.7)c-P]["(.7)c^P]["(.7)c+P]["(.7)c&P]

rule P = F["(.7)c-P]["(.7)c^P]["(.7)c+P]["(.7)c&P]
```

When the tree with phyllotactic branching, shown in Figure 3 has leaves added (Figure 5) it takes on the appearance of a real plant. The sequence of T producing U, which produces W, which produces FT introduces a delay so that leaves tend to be located toward the periphery of the structure. The symbol t(-.5) introduces the effect of gravity on curvature of branches, while the symbol ~(60) allows random variation in yaw, roll and pitch. This improves the match between this model and real plants.

```
recursion 12

axiom &(-90)>(90)-(90)c(12)TA

rule A = FBA
rule T = U
rule U = W
rule W = FT

rule B = !>(137.5)t(-.5)[&(45)F[c(4)~(60)F[L]A]

rule L = [&(100)c(4)F[{.+(30)g(.866).-(30)g.-(30)g(.866).-
         (120) g(.866).-(30)g.-(30)g(.866)g.}]>(180)c(4)
         {.+(30) g(.866).-(30)g.-(30)g(.866).-(120)g
         (.866).-(30)g.-(30)g(.866)g.}]
```

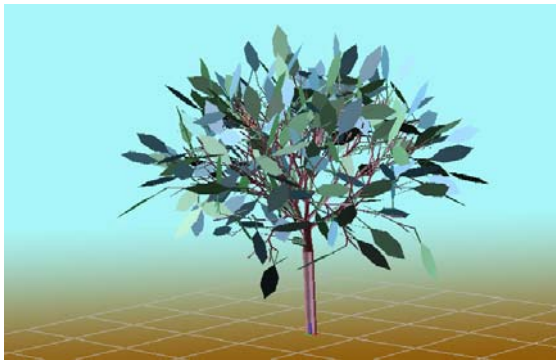

Figure 5: *Tree with phyllotactic branching and ovate leaves.*

Artistic License with Plant Architecture

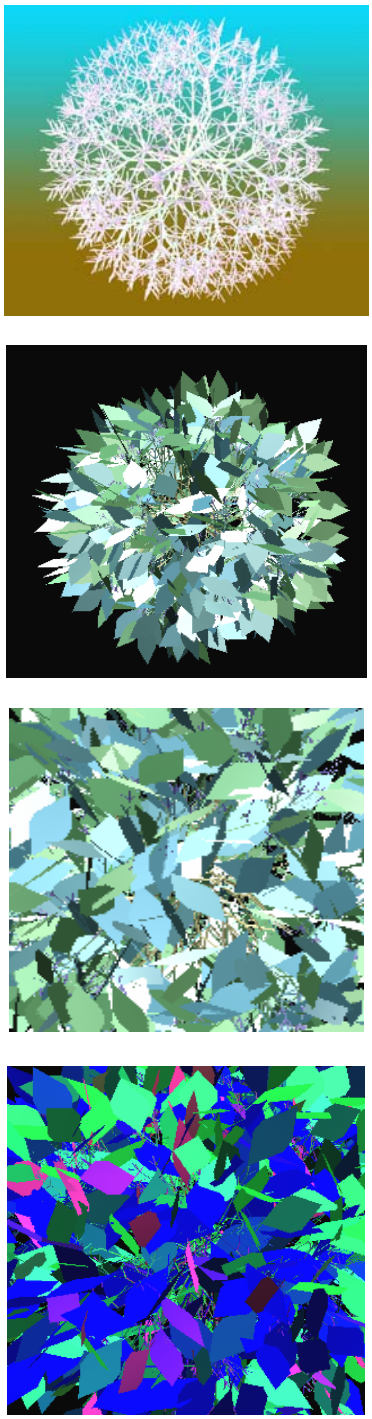

Figure 6-9: *From a dandelion to a "bushy" structure, and to abstract digital "watercolors".*

```
recursion 6
angle 45
shape 1
thickness 5

axiom &(90)+(90)[Q][-(90)Q][-(180)Q][-(270)Q]

rule Q = ["(.7)c-P]["(.7)c^P]["(.7)c+P]["(.7)c&P]

rule P = F["(.7)c-P]["(.7)c^P]["(.7)c+P]["(.7)c&P]
```

"Adding" leaves to dandelion structure in Figure 6 produces a spherical artificial bush (Figure 7), details of which appear not to have any organized structure.

```
recursion 6
no_wait 1

angle 30
shape 1
thickness 5

axiom &(90)+(90)[Q][-(90)Q][-(180)Q][-(270)Q]
        [^(90)Q][&(90)Q]

rule Q = ["(.7)c-P]["(.7)c^P]["(.7)c+P]["(.7)c&P]

rule P = A["(.7)c-P]["(.7)c^P]["(.7)c+P]["(.7)c&P]

rule A = FBA

rule B = !>(137.5)t(-.5)[&(45)F[c(4)~(60)
        F[{.+(30)g(.866).-(30)g.-(30)g (.866).-
        (120) g(.866).-(30)g.-(30)g(.866)g.}]>
        (180)c {.+(30) g(.866).-(30)g.-(30)
        g(.866).-(120) g(.866).-(30)g.- (30)
        g(.866) g.}]A]
```

Topiary-like artistic bush (Figure 7) can be observed in 3D environment which allows an observer to examine it from different angles and experience its three-dimensional features, depth and light effects. Figures 8 and 9 are authors' depictions of details from the Figure 7, created through play with avatar and lighting features of the VRML environment.

Reflections: Aesthetics of Process and Output of L-systems

The creativity of process and outputs of L-systems may be viewed from multiple perspectives. The first may be as a plant enthusiast, who admires the form, patterns, growth and morphogenesis of real plants that are emulated by the output from certain L-systems. The second is as someone simply admiring a creation because they find it pleasing in itself. In this instance, the output may not fall within the constraints that define a plant. A third perspective is that of the mathematician, who finds beauty in the elegance of a succinct formulation, which results in a pleasing form or morphological sequence. In the case of Prusinkiewicz and Lindenmayer [4], it is the *algorithm* or process of creation of an interesting object and its basis in an elegant piece of "code" that could be regarded as aesthetically pleasing. It has mathematical beauty.

Mathematical beauty is inherent in L-systems and their application in a number of ways. The first is that the piece of L-system code is short (e.g. an axiom and two production rules that produce the Dandelion-like inflorescence in Figure 4). There is also beauty in the small number of steps needed to produce an aesthetically pleasing object. For example only 6 recursions are needed to produce an object like a dandelion-like inflorescence (Figure 4). Another dimension of the beauty of L-systems is that the results are surprising, especially in terms of the complexity of the output. The insights that L-systems provide, such as the self-similarity or repetition of patterns at different levels of detail, are another form of beauty. After seeing examples of L-systems, the realization that an author can easily create new forms, provides another form of beauty: the beauty of generalizability. For example, after seeing Figures 5 and 6 and their L-systems code, a hybrid form can be produced as in Figure 7 by adding "leaves" to the Dandelion-like inflorescence. Further, once the hybrid is generated, its inner form can be explored to artistic effect, generating new and exciting images by zooming in or changing the spectral properties of a three-dimensional lighting environment. Just as you may stand and admire a painting in a gallery for different angles the VRML (Virtual Reality Modeling Language) allows you to move in three dimensions relative to the object—panning, tilting and zooming—as well as changing the light sources.

References

[1] Van Hiele, P., *Structure and insight: A theory of mathematics education*. 1986, Orlando, FL: Academic Press, Inc.

[2] Alagic, M., *Fostering understanding of mathematics visualization*, in *Bridges for teachers, teachers for bridges: 2004 workshop book*, M. Alagic and R. Sarhangi, Editors. 2004, Academx Publishing Service: Bel Air, MD. p. 1-17.

[3] Abelson, H. and A. diSessa, *Turtle Geometry: The Computer as a Medium for Exploring Mathematics*. 1980, Cambridge (MA), London: The MIT Press

[4] Prusinkiewicz, P. and A. Lindenmayer, *The Algorithmic Beauty of Plants (The Virtual Laboratory)*. 1990: Springer-Verlag.

[5] Lindenmayer, A., *Mathematical models of cellular interaction in development. Parts I and II.* Journal of Theoretical Biology, 1968. **18**: p. 280-315.

[6] Prusinkiewicz, P. *Graphical applications of L-systems*. in *Graphics Interface '86 / Vision Interface '86*. 1986: CIPS.

[7] Lapre, L. *New Lparser*. http://home.wanadoo.nl/laurens.lapre/lparser.html [Cited December 1, 2006].

[8] Operating instructions for Lparser. http://education.wichita.edu/alagic/BTTB/BTTB.htm [Cited April 18, 2007].