

## RESEARCH

# Supporting Information: The use of plant models in deep learning: an application to leaf counting in rosette plants

Jordan Ubbens<sup>1</sup>, Mikolaj Cieslak<sup>2</sup>, Przemyslaw Prusinkiewicz<sup>2</sup> and Ian Stavness<sup>1\*</sup>

\*Correspondence:

ian.stavness@usask.ca

<sup>1</sup>University of Saskatchewan, 105 Administration Place, S7N 5C5 Saskatoon, Canada

Full list of author information is available at the end of the article

## Model implementation

Our Arabidopsis rosette model was specified in the L+C programming language [1, 2], which combines L-systems [3] and the C++ programming language. The model was executed using the *lpfg* simulator [1] included in the Virtual Laboratory software environment [4]. The source code for our model is given in Listing 1, and an *lpfg*-generated image of the model is shown in Figure S1.

Listing 1 L+C implementation of the Arabidopsis rosette model.

```

1  #include <lpfgall.h>
2
3  // Declaration of constants
4  const float n_max = 20.; // max. number of leaves
5  const float Δa = 0.1; // leaf age increment
6  const float t_min = 300.; // min. growing time
7  const float t_max = 820.; // max. growing time
8  const float d_age = 4.5; // min. drawing age
9
10 // Module definitions
11 module A(float); // an apex, with node number
12 module L(float,float); // a leaf, with node number and age
13
14 // Setup time of development and leaf counting
15 int s; // counts simulation steps
16 int s_max; // last simulation step
17 int N; // total number of leaves
18
19 Start: {
20     s = 0;
21     s_max = int(t_max * ran(1)) + t_min;
22 }
23 StartEach: {
24     N = 0;
25 }
26 EndEach: {
27     s = s + 1;
28     if (s == s_max) {
29         Stop();
30         Printf ("Leaf count: %d\n", final_num_leaves);
31     }
32 }

```

```

33
34 derivation length: 10000;
35
36 axiom: Right(nran(0,5.0)) RollL(nran(0,180)) A(0);
37
38 decomposition:
39
40 A(n) : {
41   if (n < nmax) {
42     produce SetWidth(10-2) F(10-3) // internode
43             SB L(n,0) EB // leaf
44             RollL(nran(137.5,2.5)) // phyllotaxis
45             A(n+1); // recreate apex
46   }
47 }
48
49 production:
50
51 L(n,a) : {
52   a = a + Δa;
53   // count leaf if above a certain age
54   if (a > dage)
55     N = N + 1;
56   produce L(n,a);
57 }
58
59 interpretation:
60 // declare extra module for drawing a leaf
61 module C(float,float,bool);
62
63 L(n,a): {
64   // do not visualize very small leaves
65   if (a ≤ dage)
66     produce;
67
68   // calculate the leaf's length
69   float ln = nran(1.,0.01) * flmax(n/nran(1.,0.05)) * fl(a);
70
71   // produce the leaf's geometry
72   produce Down(45 * fang(n)) // set inclination angle
73           CurrentContour(1) // specify leaf cross-section
74           SetWidth(ln * fw(0)) // set width at leaf base
75           StartGC C(0,0.65 * ln,0) EndGC // draw petiole
76           StartGC C(0,ln,1) EndGC; // draw leaf
77 }
78
79 C(x,l,q) : {
80   const float Δx = 0.01; // step size along midrib
81   if (x ≤ l) {
82     x = x + Δx;
83     float w = l;
84     if (q)
85       w = w * fw(x);
86     else
87       w = w * fw(0)/0.65;
88     nproduce F(l * Δx) SetWidth(w) Down(15 * Δx) C(x,l,q);
89   }
90   produce;
91 }

```

---

After declaration of constants (lines 4–8) and module definitions (lines 11–12), global variables are declared for the number of steps in the simulation and the total number of leaves in the rosette (lines 15–17). At the start of the simulation, the number of steps is initialized to zero and the maximum number of steps is computed as a discrete random uniform variable in the range  $[t_{min}, t_{max}]$  (lines 20–21). After each derivation step, the number of steps is incremented until the maximum is reached, which ends the simulation and prints the total leaf count (lines 27–30).

A single apex, A, is initiated in the axiom (line 36) after some randomization of the rosette's orientation. The first production generates a new internode and leaf up to a maximum number  $n_{max}$  by decomposing the module A (lines 40–45). Each internode is modeled as a cylinder (module F) with constant radius  $10^{-2}$  and length  $10^{-3}$ . Consecutive leaves are rotated around the main stem following a spiral phyllotaxis by an angle generated from a normal random distribution with mean  $137.5^\circ$  and standard deviation 2.5. The second production (lines 51–56) models a leaf's aging and increments the global leaf counting variable,  $N$ , if the leaf is above a certain age.

To generate a visualization of the rosette, we apply interpretation rules to the string generated by the L-system (starting from line 59). The first interpretation rule is applied to all leaf modules  $L(n, a)$ . The rule first ignores a leaf (produces nothing) if its age is below a threshold. Then, given the leaf's node number and age, the leaf's length (line 69) is calculated as described in the main text. The rule ends by starting the drawing of a generalized cylinder, module  $C(x, l, q)$  representing the leaf, which is separated into the petiole and lamina sections (lines 72–76).

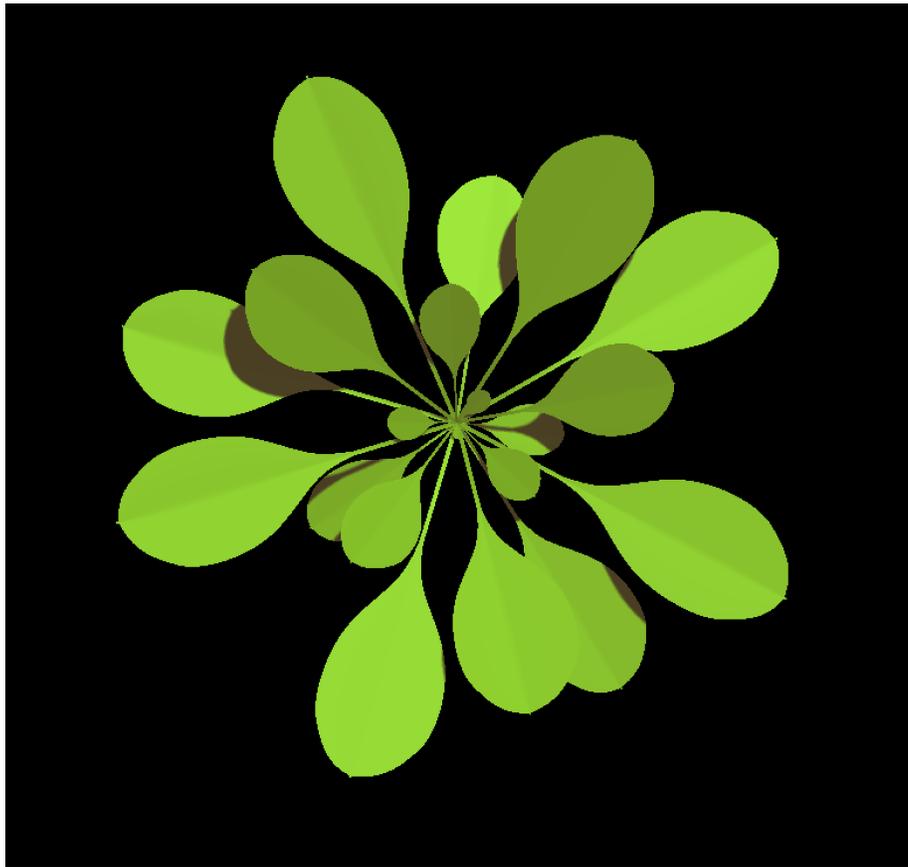
The interpretation rule for module  $C(x, l, q)$  (line 79) draws consecutive cross sections of the leaf along the midrib. This is effected by applying the rule recursively until the drawing is forced to stop (line 81) — when the variable  $x$  is greater than one. The width of the generalized cylinder is determined by its length  $l$  and its state variable  $q$ . For the lamina ( $q = 1$ ), the width is modified by a graphically defined function,  $f_w(x)$ , as the cylinder is being drawn, but, for the petiole ( $q = 0$ ), the width is constant (using the value at  $f_w(x = 0)$ ). Production of the  $F(l * \Delta x)$  and  $SetWidth(w)$  modules (line 88) draws a section of the generalized cylinder with the given length and width. The cylinder is also rotated downwards by  $(15 * \Delta x)^\circ$  after each section is drawn.

#### Author details

<sup>1</sup>University of Saskatchewan, 105 Administration Place, S7N 5C5 Saskatoon, Canada. <sup>2</sup>University of Calgary, 2500 University Dr NW, T2N 1N4 Calgary, Canada.

#### References

1. Karwowski, R., Prusinkiewicz, P.: Design and implementation of the L+C modeling language. *Electronic Notes in Theoretical Computer Science* **86**(2), 1–19 (2003). doi:10.1016/S1571-0661(04)80680-7
2. Prusinkiewicz, P., Karwowski, R., Lane, B.: The L+C plant-modelling language. In: Vos, J., Marcelis, L., De Visser, P., Struik, P., Evers, J. (eds.) *Functional–structural Plant Modelling in Crop Production*, pp. 27–42. Springer, Dordrecht (2007)
3. Prusinkiewicz, P., Lindenmayer, A.: *The Algorithmic Beauty of Plants*. Springer-Verlag, New York (1990). With Hanan J, Fracchia FD, Fowler D, de Boer MJM, and Mercer L.
4. Federl, P., Prusinkiewicz, P.: Virtual Laboratory: an interactive software environment for computer graphics. In: *Proceedings of Computer Graphics International*, pp. 93–100 (1999)



**Figure S1** An example rendering of the Arabidopsis rosette model with 20 leaves.