L-Py, an open L-systems framework in Python

F. Boudon, T. Cokelaer, C. Pradal, C. Godin

Virtual Plants INRIA team,
Joint with CIRAD and INRA, Montpellier, France
Context: L-systems

- Introduced by A. Lindenmayer in 1968
  - Simulation of multi-cellular organisms
- Dynamic Systems with Dynamic Structures (Giavitto, 01)
- Well adapted for modeling plant growth. Widely used for FSPM
Definition

• L-systems consist of an alphabet $V$, an axiom $w$ and a set of productions $P$.

$$G = < V, w, P >$$

• Productions are applied to a string in parallel.

left context < predecessor > right context  →  successor

• Example

$V = \{F, A, +, -\}$

$w = A$

$P: A \rightarrow F[+A][-A]FA$

$F \rightarrow FF$
Existing Platforms

- **L-studio/Vlab** *(Prusinkiewicz et al., Univ Calgary, Canada)*
  - Cpfg : c-like
  - Lpfg : L-systems in C++

- **GroIMP** *(Kurt et al., Univ. Cottbus, Allemagne)*
  - XL : Java based L-systems language, extension to graph structure

- **Jpfg : Java based** *(Hanan et al., Univ. Queensland, Australie)*

- **LSystem** : very basic L-systems engine in python.
- **Plugins** Blender, Inkscape, Povray ...

- **Graphtal**
- **FractInt**
Motivations

• Mixing power of L-systems with the high level modeling language *Python*.

• Mixing power of L-systems with tools written/accessible in Python (*OpenAlea*, PlantGL, scipy...)

• *Easy to use* platform for beginner programmers such as students or biologists.

• Keep compatibility with others systems to provide complementary implementations to the community.
• L-Py Overview
• The rewriting system
• Geometrical Features
• An open system
• The Development Environment
• Cases of use
L-Py Architecture

- OpenAlea
- Python
- L-Py Kernel: C++/Python
- L-Py Visual Editor: Python
- PlantGL
- PyQt
- Qt
L-Py Development Environment
from random import random
MAX_AGE, dr = 10, 0.03  # constants

module Apex(age), Internode(length, radius)

Axiom: Apex(0)

derivation length: 5

production:
Apex(age) :
    if age < MAX_AGE:
        produce Internode(1+ random(), 0.3)
        /(137) [+ (30) Apex(age+1)] Apex(age+1)

Internode(l, r) --> Internode(l, r+dr)

interpretation:
Internode(l, r) --> _ (r) F(l)

end system
PlantGL Turtle

Compatible with $cpfg$ and $lpfg$ convention.

$F(3) + (60) F(2) - (40) F(1)$
• L-Py Overview
• The rewriting system
• Geometrical Features
• An open system
• The Development Environment
• Cases of use
L-systems features

- Parametric L-systems
- Contexts and new contexts sensitivity

\[
A(x) : \\
\text{if } x < 3: \\
\text{produce } B(2^x) \\
C(z) < B(y) << A(x) > D(w): \\
\]

(Prusinkiewicz et al., 90, 94, 96, 07)
L-systems features

- Parametric L-systems
- Contexts and new contexts sensitivity
- Stochastic L-systems

\[
\text{A}(x) : \\
\text{if } x < 3: \\
\text{produce } B(2^x) \\
\]

\[
\text{C}(z) < B(y) \iff \text{A}(x) > D(w): \\
\]

\[
\text{A}(x) : \\
\text{if } \text{random()} < \text{prob1}: \\
\text{produce } B(2^x) \\
\text{else : produce } C(0) \\
\]

(Prusinkiewicz et al., 90, 94, 96, 07)
L-systems features

- Parametric L-systems
- Contexts and new contexts sensitivity
- Stochastic L-systems
- Environmental Interaction
- Pruning
- Group of rules
- Forward and Backward application, ...

(Prusinkiewicz et al., 90, 94, 96, 07)
• L-Py Overview
• The rewriting system
• Geometrical Features
• An open system
• The Development Environment
• Cases of use
Geometrical Features

- Use of any PlantGL primitive with `@g`

```python
from openalea.plantgl.all import *

module Crown(heigth, radius)
Axiom: Apex(0)
derivation length: ...
production:
...

interpretation:
Crown(h,r) :
    p = compute_params(h,r)
    produce @g(AsymmetricHull(*p))
endlsystem
```
Branch Geometry

- Geometrical embedding of a branch may be complex

```python
length = 10
Axiom: Branch(length)
derivation length: 1
production:

interpretation:
Branch(l) :
    for i in xrange(l):
        nproduce f(0.1) F(1)
endlsystem
```
Branch Geometry

- Use of tropism to change orientation of branch

length = 10

**Axiom:** Elasticity(0.02) Branch(length)

derivation length: 1

production:

```
interpretation:
Branch(l) :
    for i in xrange(l):
        nproduce f(0.1) F(1)
endlsystem
```
Branch Geometry

• Use of predefined geometrical embedding

\[
\text{length} = 10 \\
\textbf{Axiom: } \text{SetGuide}(\text{path, length}) \quad \text{Branch(length)} \\
\text{derivation length: } 1 \\
\text{production:}
\]

\[
\text{interpretation:} \\
\text{Branch(l)} : \\
\quad \text{for i in xrange(l):} \\
\qquad \text{nproduce } f(0.1) \ F(1) \\
\text{endlsystem}
\]
Example on a simple tree structure

Simple recursive structure
Example on a simple tree structure

Using tropism
Example on a simple tree structure

Using a predefined branch path
Interpolating profiles

axisSet = [axis1, …, axisN]
times = [0, t1, …, 1.0]
interpol = ProfileInterpolation(axisSet, times, degree = 3)

Axiom: BG(0) Branch(length)
production:
BG(t) --> BG(t+dt)

interpretation:
BG(t) --> SetGuide(interpol(t), length)
endSystem
• L-Py Overview
• The rewriting system
• Geometrical Features
• An open system
• The Development Environment
• Cases of use
An open system

- L-Py defines accessible data structures:

```python
>>> Lsystem, Lstring, ...
```

- Compatibility with MTG

```python
def EndEach(lstring, geometries):
    g = lstring2mtg(lstring, geometries)
    node = pm['vplants.fractalysis']['diffuseInterception']
    f = function(node)
    g = lstring2mtg(lstring, geometries)
    newg = f(g)
    lstring = mtg2lstring(newg)
    return lstring
```

- Invoking OpenAlea dataflow
Compatibility with OpenAlea

- Lpy can be controlled from VisuAlea
- Post processing with other modules
• L-Py Overview
• The rewriting system
• Geometrical Features
• An open system
• The Development Environment
• Cases of use
Graphical parameters

- Direct use of objects into the code
- Continuous modeling
Debugging

```
101 derivation_length: 60
102 production:
103 # produces a flower apex in state 1 initially
104 if r == 0:
105   produce I(0)/(phyllangle)(T_F)
106 elif t < ops : produce I(0)[ml(0)]/(phyllangle)A(r,t-1)
107   else: produce A(r,t-1)
108
109 # internode
110 I(t):
111   produce I(t+dt)
112
113 # axillary leaf
114 AL(t):
115   produce AL(t+dt)
```

Debugger

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>32.0</td>
<td>Float</td>
</tr>
</tbody>
</table>

Line 115, Column 0 (3133)
Profiling
• L-Py Overview
• The rewriting system
• Geometrical Features
• An open system
• The Development Environment
• Cases of use
Performances

• Comparison with Lpfg on MAAppleT project

(Costes et al., MAAppleT, FPB08)
Pedagogic test

- One year project with high school students
- Model a virtual scrubland from South of France
  - study on field of plant architecture and landscape distribution.
  - Learning L-systems and modeling concepts

Collaboration with E. Farcot, Y. Caraglio and M. Beziz and Pompidou high school
Final Rendering

Final rendering with blender and special FX: J. Chopard
Conclusion

• Mature open source project
• Downloadable through OpenAlea platform
• Easy to use software, well adapted for pedagogic use

• Work in progress with P. Prusinkiewicz:
  – Influence of the language on L-systems.