Using the Language XL for Structural Analysis

Jan-Anton Dérer[1] Ole Kniemeyer[1] Reinhard Hemmerling[1] Gerhard Buck-Sorlin[2] Winfried Kurth[1]

[1]BTU Cottbus, Department of Computer Science, Chair for Practical Computer Science / Graphics Systems, P.O.Box 10 13 44, 03013 Cottbus, Germany

[2]Wageningen UR, Crop and Weed Ecology Group, Haarweg 333, 6709 RZ Wageningen, The Netherlands

Introduction

The analysis of plant structures is an important issue in functional-structural plant modelling, especially in the context of parameterization and validation. This holds equally for structures resulting from measurements of real plants and for modelled structures, i.e., the outcome of virtual plant simulations. Such analysis has to consider both the topology of the structure and the values of parameters of its constitutive entities like geometry-related parameters or the internal state.

This poster presents a beech model case study which uses the programming language XL within the open-source modelling environment GroIMP [1] for analysis purposes. Although XL has been designed as an extension of L-systems, i.e., for the implementation of functional-structural plant models, it turns out to be equally suitable for the analysis of plant structures. The advantage of this approach is that the same language can be used to model a virtual plant, to analyse the results, and to compare it with measured data.

Analysis With XL

The language XL provides graph query facilities which search for occurrences of patterns in the current structure that is represented as a graph. For example, the simple query

```
(* Shoot *)
```

finds all nodes of type `Shoot` in the current structure, and the expression

```
(* x:Shoot, (x.order == 0) *).length
```

combines the previous query with the condition `(x.order == 0)` so that it finds all shoots of order zero and then returns their length. The expression

```
((* p:Shoot [b:Shoot] *), b.length/p.length)
```

makes use of the traditional L-system notation which uses square brackets to enclose branches: it finds all pairs of parent shoots `p` and directly following branching shoots `b` and computes the ratio of their lengths.

The results of such expressions can be analysed with the library provided by GroIMP. Standard computations like `sum`, `mean`, `max` exist, but also facilities to draw a plot, to fit parameters of a function to given data, or to perform simple statistical analysis.

DTD Format

Measured tree data can be imported into GroIMP by means of the DTD format [3]. Each line describes a single growth unit. After the import, such a unit is represented within GroIMP as a node of class `DTDShoot`. The shoot parameters are node attributes, and the topology is represented by a tree graph.

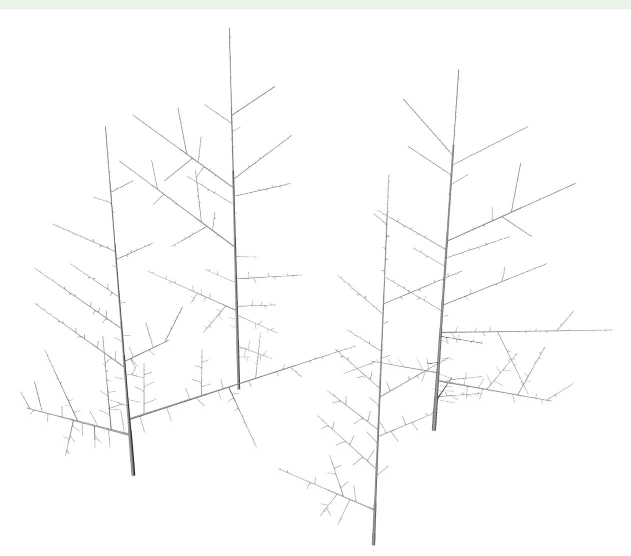

Figure 1: DTD beech trees within GroIMP

```
1   L185 ##   E9  Y3  Z2 + w90 D9.2
2   L141 #1   V  E9  Y3  Z1  D8.2
3   L96  #2   V  E7  Y3  Z1  D7.5
4   L128 #3   V  E4  Y1  D6.8
5   L221 #4   V  E8  Y2  Z1  D5.9
6   L197 #5   V  E8  Y2  D3.2
7   L233 #6   V  E8  B8  D2.2
8   Q3   #1   I3 -  W80
9   L27  #8.3 V  E2  D1.1
10  Q2   #9   I1 -  b3  W80
11  L19  #1   I2 +  E2  D2.0
12  L82  #11  V  E6  Y2  Z2  D1.8
...
```

Measurements

For this case study, we use data of young beech trees (*Fagus sylvatica* L.) which were measured in 1995 in the Solling (German midland mountains) [4]. Their age ranges from 7 to 14 years. The complete topology was recorded with annual shoots being the elementary units, and for each shoot its length, number of internodes, branching angle and some further parameters were measured. Figure 1 shows a visualization of the measured data of four beech trees within GroIMP.

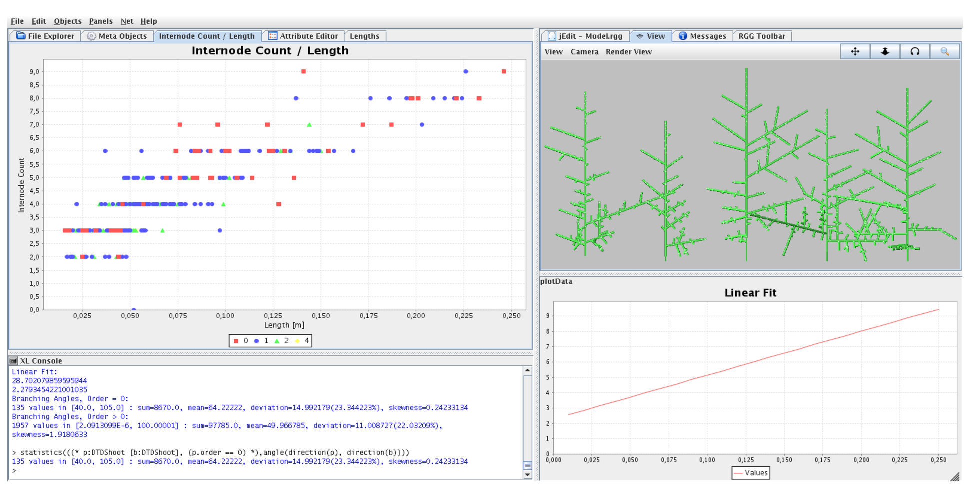

Figure 2: Result of analysis within GroIMP

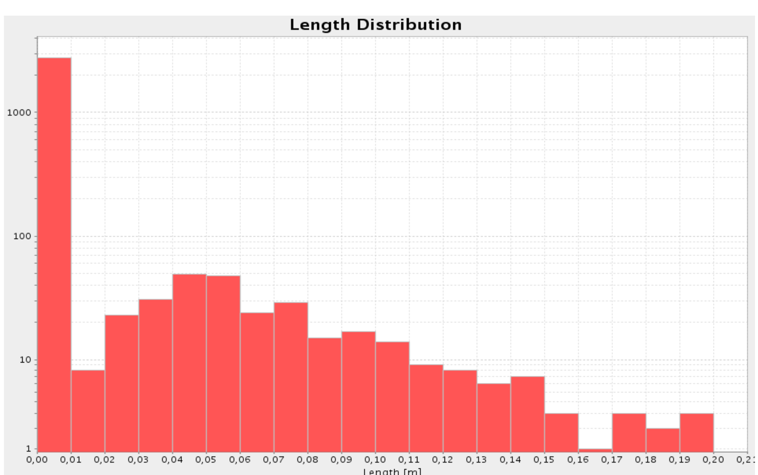

Figure 3: Distribution of shoot lengths

Analysed Properties

For this case study, we extract two simple properties out of the data. At first, the relation between the length of a shoot and the number of its internodes is studied. For each shoot `a` which is not a short shoot (longer than 8 mm), the pair `(a.length, a.internodeCount)` is written to a `table`, where we use different rows of the table for different orders:

```
((* a:DTDShoot, (a.length > 0.008) *),
table.getRow(a.order).add(a.length,a.internodeCount));
```

This table can be exported to various formats like CSV or XLS if one wants to use external tools to further analyse the data. But it can also be plotted within GroIMP by

```
chart(table, SCATTER_PLOT);
```

The result for five beech trees is shown in Fig. 2. A linear relation $n(l) = p_0 l + p_1$ between length l and number n seems to be suitable. The library of GroIMP can be used to compute the optimal values for p_0, p_1 (namely $0.287\,\mathrm{cm}^{-1}$ and 2.28, respectively) and plot the resulting function, which is also shown in Fig. 2:

```
double[] p = {25, 2};
DoubleToDouble n = double x => double p[0]*x + p[1];
fitParameters(n, table, p, precision);
plot(n, 0.01 * (1:25));
```

As a second property, we consider the branching angle. The expression

```
((* p:DTDShoot [b:DTDShoot], (p.order == 0) *),
    angle(direction(p), direction(b)))
```

computes the branching angle between all parent shoots `p` of order zero and their branching shoots `b`. If we apply the built-in `statistics` function to this expression, we obtain a statistics of the branching angles including mean value and deviation: $64^\circ \pm 15^\circ$ for order zero, $50^\circ \pm 11^\circ$ for higher orders.

Beech Model

Figure 4: Beech individuals with different light conditions after ten years

We developed a beech FSPM based on carbon production, transport and distribution which contains the results of the previously described analysis of experimental data [2]. Figure 4 shows the 3D visualization of three individuals at the age of ten years. The model is a true FSPM, but we had not enough experimental data to fully parametrize it on the basis of such data.

The outcome of the model can be analysed with XL expressions just like measured data. For example, we may study the distribution of annual shoot lengths (which is an emergent property of the FSPM) and compare it with measured data:

```
table << (* Shoot *).length;
chart(table, HISTOGRAM | Y_LOG);
```

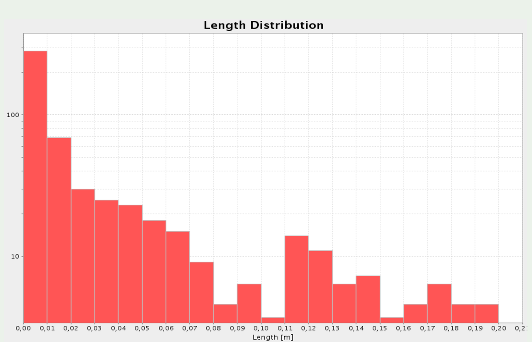

Figure 5: Distribution of shoot lengths of simulated tree

The result in Fig. 5 differs in shape from the corresponding diagram for measurements in Fig. 3. This could be the starting point for refinements of the model.

Conclusions

We have shown a case study which uses the language XL to analyse experimental data, to implement a beech FSPM and to analyse the outcome of the latter. The presented work has to be understood as a "proof of concept": although the model itself is not meant to be realistic in the sense that it could be used to precisely predict beech growth, this case study has shown that the language XL, combined with the environment GroIMP, is in principle suitable for such a purpose.

[1] O. Kniemeyer.
Design and Implementation of a Graph Grammar Based Language for Functional-Structural Plant Modelling.
PhD thesis, BTU Cottbus, 2007.
(forthcoming, see http://www.grogra.de).

[2] O. Kniemeyer, R. Hemmerling, G. Buck-Sorlin, and W. Kurth.
The rule-based language XL and the modelling environment GroIMP, illustrated with simulated tree competition.
In *Proceedings of FSPM07*, 2007.

[3] W. Kurth and G. Anzola Jürgenson.
Triebwachstum und Verzweigung junger Fichten in Abhängigkeit von den beiden Einflußgrößen Beschattung und Wuchsdichte: Datenaufbereitung und -analyse mit GROGRA.
In D. Pelz, editor, *Deutscher Verband Forstlicher Forschungsanstalten, Sektion Forstliche Biometrie und Informatik, 10. Tagung Freiburg i. Br. 1997*, pages 89–108. Ljubljana, Biotechnische Fakultät, 1997.

[4] M. Steilmann.
Morphologische Untersuchungen zur Modellierung des Wachstums in Abhängigkeit von den Licht- und Konkurrenzverhältnissen von Jungbuchen.
Diploma thesis, University of Göttingen, 1996.