Methodical and technical aspects of functional-structural plant modelling

Dissertation
for the award of the degree
"Doctor rerum naturalium" (Dr. rer. nat.)
of the University of Göttingen

within the doctoral program Environmental Informatics (PEI)
of the Georg-August University School of Science (GAUSS)

submitted by
Michael Henke
from
Wilhelm-Pieck-Stadt Guben now Guben

Göttingen, 2017

Thesis Committee

Prof. Dr. Dr. h.c. Branislav Sloboda (retired)
Department Ecoinformatics, Biometrics and Forest Growth, University of Göttingen
Prof. Dr. Winfried Kurth
Department Ecoinformatics, Biometrics and Forest Growth, University of Göttingen

Members of the Examination Board

Reviewer: **Prof. Dr. Winfried Kurth**
Department Ecoinformatics, Biometrics and Forest Growth, University of Göttingen
Second Reviewer: **Prof. Dr. Gerhard H. Buck-Sorlin**
IRHS, INRA, AGROCAMPUS-Ouest, Université d'Angers, SFR 4207 QUASAV, 42 rue Georges Morel, 49071 Beaucouzé cedex, France

Further members of the Examination Board

Prof. Dr. Dieter Hogrefe
Telematics Group, University of Göttingen
Prof. Dr. Kerstin Wiegand
Department of Ecosystem Modelling, University of Göttingen
Prof. Dr. Dr. h.c. Branislav Sloboda (retired)
Department Ecoinformatics, Biometrics and Forest Growth, University of Göttingen
Prof. Dr. Stephan Huckemann
Institute for Mathematical Stochastics, University of Göttingen

Date of the oral examination: 2017-03-13

Acknowledgements

I would like to thank:

- Winfried Kurth - for being my main supervisor, and especially for his constructive criticism, his insistance, rigour and perseverance that helped me to always try to improve
- Branislav Sloboda - for his support and giving me the chance to become a researcher at Göttingen University
- Gerhard Buck-Sorlin - for his continuous support, trust and help, not only as a scientific supervisor; for inviting me to work in France for a brief period, and for helping to improve my knowledge in plant ecophysiology
- Wei - for her love and belief in me and for her patience when I turned the night into day
- Mum - for her continuous support and giving me the freedom to live my life in the way I wanted
- Katarina, Robert, Jan, Niki, Tim, Sebastian, Yongzhi und Ditdit - for being my friends and colleagues
- Ilona - for helping me whenever and wherever she could
- Reinhold - for his technical support
- Ole - for GroIMP and his support far beyond his time as a developer

Abstract

During the last decade, functional-structural plant models (FSPM) has become a more widely accepted paradigm and tool in life sciences. The increasing demand for FSPMs raised a number of new challenges, including the consideration of model reuse, model combination and comparison, and the enhancement of existing models. Current FSPMs are often developed intuitively, without a proper plan or standard. This necessitates the establishment of new methods of modelling. Functional-structural plant modelling is a highly complex process with many activities and sub-activities requiring skills and knowledge of different disciplines, each one of which still having a great potential for improvement. The aim of the present work is to provide support for the whole modelling workflow, by presenting efficient methods for data acquisition and techniques adapted from software engineering for FSPM (e.g., modularisation, prototyping, and model standard).

After a short introduction, the second chapter gives an extensive overview of former and current approaches of plant modelling in general. The chapter closes with a discussion of several aspects of the whole plant modelling workflow. In the third and fourth chapter of this thesis, two new methods for data acquisition for plant modelling are developed, introduced and tested. The concept of modularisation and providing independent model components is discussed in chapter five.
In the sixth chapter, a prototype for FSPM is introduced. Using techniques borrowed from software engineering that allow efficient model development an application of this model prototype is described in chapter seven.
The eighth chapter describes how advanced light modelling techniques can be used within FSPM thereby providing new application fields for FSPM.

Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

During the past decades, functional-structural plant models (FSPM) has become a more widely accepted paradigm and tool in the life sciences (e.g., Buck-Sorlin and Delaire 2013; Godin and Sinoquet 2005; Sievänen *et al.* 2014; Vos *et al.* 2010, 2007). This modelling paradigm is used, e.g., as a research tool, in teaching and as decision support tools in various disciplines, like biology, agronomy, agriculture, horticulture, landscaping, forestry, and ecology. 3-d models for the most important crops and trees have been proposed.

In this work methodical and technical aspects of FSPMs and the whole modelling workflow will be introduced and discussed, from methods for data acquisition and test environments to the development of a prototype for FSPM.

1.1. Motivation

A typical modelling workflow comprises the following sequence of steps: model conceptualisation, data acquisition, data processing, model parameterisation, model calibration, operation

phase, validation, use of the model as a hypothesis testing or decision-support tool, and finally model documentation (possibly considering user feedback). This sequence is not strict: e.g., some modelling exercises start with data acquisition and continue with model conceptualisation. Also, some of the steps have to be revisited several times, e.g., calibration and operation phase, until a satisfactory level of agreement between data and model output has been achieved. These steps can be further decomposed into several sub-activities. See Sec. 2.3.1 for more details. This alone contains a great potential for improvements of different types. Moreover, the increasing demand for FSPM is reason enough to investigate new methods for making the modelling process more efficient.

The planning or conceptual phase is a complex activity comprising several sub-activities that should not be underestimated with respect to their costs in time and intellectual investment. A conscientious preparation at this step of the project will always pay off later. Unfortunately, conducting a good planning requires experience that can only be acquired by accomplishing a number of modelling projects! This dilemma partly explains the relatively high failure rate of modelling projects, in other words when these projects do not deliver the expected result or, even worse, can not be finished in time. On the other hand, it also highlights the need for a guided modelling process and support in modelling, which can help the modeller to concentrate on more important things (in terms of modelling) by lightening the load of technical details (in terms of programming).

The increasing number of applications resulted in the development of a large variety of different models which reinforced the need for reusing models and of comparing and/or combining models. This, in turn, led to the necessity for standardisation of all aspects of the modelling process. In the past, there have been only a few attempts to providing guidelines for basic concepts in modelling and 'good modelling practice', see for example van Waveren *et al.* (1999) for an example in environmental sciences or Carson and Cobelli (2001) for one in physiology and medicine. Cournède *et al.* (2013) summarised three main steps from a mathematical point of view. Grimm and Railsback (2005) proposed the pattern-oriented modelling (POM) approach for designing, fitting and validating agent-based model (ABM)s.

1.1.1. Increasing Demand for Functional-Structural Plant Modelling

In botanical research as well as in agricultural and horticultural practice (growers, breeders, consultants, ...), the usefulness of FSPM is beginning to be recognised as a powerful tool for

diverse purposes (Buck-Sorlin and Delaire 2013; Fourcaud *et al.* 2008; Room *et al.* 1996; Vos *et al.* 2010, 2007). Given the wider portfolio of available models, it now seems timely to enter the next level in FSPM development, by introducing more efficient methods for model development. This would include the consideration of model reuse (by modularisation), combination and comparison, and the maintenance and enhancement of existing models. To facilitate this process, standards for design and communication would need to be defined and established. In this respect, software engineering and software project management can provide techniques that when adapted to plant modelling could bring great benefits.

1.1.2. Improvement in Quality and Quality Assurance

Each of the sub-activities within a modelling project (see Sec. 2.3.1) is challenging in its own right as it requires special knowledge and skills. Since the single steps are recurring, there is a great potential of introducing standards for accomplishment and process sequence. Doubtlessly, general modelling time will decrease with the knowledge and experience of the modeller but there are, on the other hand, some sub-activities that will become more time-consuming in principle. One of the more time-consuming sub-activities is data acquisition. Here, instead of increasing man power, which is forbidding regarding salary costs, new technical developments (automatisation) can be used to speed up measurements while at the same time increasing accuracy.

Typically, functional-structural models have been developed in a much more ad hoc way, i.e. unsystematically, using a combination of structure-determining rules and at best an arbitrary selection of physiological functions. However, there are no obvious reasons for this: Most physiological functions formalised in crop models could be used in the same general way in FSPM, and structures, such as plant organs, could be defined generally and then implemented for a crop species. In this respect, modularisation, object-oriented design, and component-based programming, are all measures to be taken from a software engineering point of view, to improve model quality. A first important step would be a 'library of FSPM modules'. Such a library could help distributing implementation work to experts in their field, which in turn would increase quality; would reduce the time used for model implementation for the user, and also would make models more comparable if the model to be compared would be based on the same sub-models.

1.1.3. Expert Knowledge Required

Models are often developed by Master and PhD students in the Plant Sciences but accomplishment of a whole modelling project is an interdisciplinary task that requires special knowledge, skills, and experiences one only can have after finishing some modelling projects. Developers of such models are often plant biologists who are keen to explore the impact of plant architecture (organ geometry and topology) on a limited range of physiological effects, e.g., the effect of leaf angle distribution on canopy radiation interception. These workers are often lacking experience in programming yet have a clear overview of the structure and scope of their model. Another group consisting of programmers and computer scientists who are interested in biological systems considers it as a challenge for the application of the rule-based paradigm. Thus, while plant biologists use an FSPM approach to study the effect of a static architecture on light interception and leaf photosynthesis, computer scientists study the way complex tree architectures could be created using a very limited set of production rules. Most physiological functions that are currently used in crop models could be used in the same general way in FSPM, and structures, such as plant organs, could be defined generally and then implemented for a crop species.

A model prototype like the one described in Sec. 6 contains all necessary elements and a basic set of structure-generating rules as well as modules to describe primary production and growth. It will give support to students and researchers with little programming experiences and can also be useful for people with rather limited knowledge of biology. A general FSPM can be an intuitive and versatile tool, usable for prototyping, teaching, as a research tool in production systems such as intercropping, as a decision support system, or to enhance our understanding of physiological processes taking place at different hierarchical levels (e.g., organ - individual - canopy). In addition, such an approach can help to establish a standard to make models more transparent and comparable, also for other researchers in the FSPM community.

1.2. Envisaged Objectives of this Thesis

The present thesis envisages to attain a number of objectives: one main aim is to achieve a better identification and formalisation of standardised processes for FSPM. As we will show the development of approaches for the rapid development of model prototypes can be one way towards this goal. This process often goes along with a more consequent modularisation of

models. As a consequence, access to modelling by virtue of FSPM can be granted to a larger community as experience from workshops and tutorials has shown.

Participants in modelling workshops and tutorials are often Master and PhD students in the Plant Sciences, with very varying experience in, and affinity for, plant modelling, especially FSPM. What is more, they often bring with them a specific problem pertaining to a specific crop species, plus disciplinary knowledge, and hope that they can take home a model tailor-made to resolve their problem. This expectation is, of course, illusory and potentially apt to create a high level of frustration or even disillusionment with respect to the power of modelling. During a number of workshop events in the past five years or so we have shown that a general-purpose FSPM as proposed in this thesis, can be an excellent starting-point for a more specific model to be created at a subsequent stage (See a list of projects related to the FSPM-P at Tab. 9.2.). In addition, the bases and principles of such a general model can be related to all participants, without the need to recur to very applied examples of specific crops (trees, cereals, ornamentals…), a practice which is known to lead to a segregation of participants into eager and bored individuals.

1.3. Structure of this Thesis

After the Introduction a general overview of the foundations, history and techniques of plant modelling is provided in Chapter 2. The main part of this thesis consists of six published or accepted research papers each one of which can be read independently from the others. In the first paper (Chapter 3), Henke *et al.* (2014a): 'Reconstructing leaf growth based on non-destructive digitising and low-parametric shape evolution for plant modelling over a growth cycle', an overview over the whole modelling workflow is presented: this workflow usually starts with data acquisition and preparation, continues with modelling and results in the integration of the developed model into a parameterisable module. Furthermore, a photometric methodology for non-destructive data acquisition was developed. The second paper (Chapter 4), Henke and Sloboda (2014): 'Semiautomatic tree ring segmentation using Active Contours and an optimised gradient operator', illustrates a new data acquisition approach involving techniques taken from image processing and photogrammetry. With the developed software, whole tree rings of digitalised stem discs can be extracted in an interactive processes. The third paper (Chapter 5), Henke *et al.* (2017): 'Realization and extension of the Xfrog approach for plant modelling in the graph-grammar based language XL', deals with a special concept for plant

modelling which is used in the Xfrog software (see Sec. A.17). During my diploma thesis it was reimplemented and combined with classical L-systems as new feature for the modelling software GroIMP. The modularised modelling concept analysed in this work already provided a first idea of how modularisation can be used within plant modelling. The fourth paper (Chapter 6), Henke *et al.* (2016): 'FSPM-P: Towards a general Functional-Structural Plant Model for efficient model development', describes a FSPM-Prototype that can be used for fast model development. Furthermore, it illustrates how principles known from software engineering can be applied to plant modelling to increase model quality. The fifth paper (Chapter 7), Henke *et al.* (2014b): 'Exploring root developmental plasticity to nitrogen with a three-dimensional architectural model', can be seen as application example of the FSPM-P, in which the FSPM-P was extended to develop a root model. With this model root system architecture was investigated under three types of plastic responses in combination with four distinct nutrient distribution scenarios including a completely random distribution, a layered distribution, a patch distribution, and a gradient distribution. The sixth and last paper (Chapter 8), Henke and Buck-Sorlin (accepted): 'Using a full spectral raytracer for the modelling of light microclimate in a functional-structural plant model', describes how advanced light modelling techniques can be used within FSPM thereby providing new application fields for FSPM. To illustrate the possibilities of a full spectral raytracer we presented two examples, visualisation of a dispersion effect, and a wavelength dependent photosynthesis model.
Chapter 9 summarises and concludes this thesis.

CHAPTER 2

Plant Modelling

The following Chapter provides a short overview of the foundations of plant modelling as well as its development and extensions during the past decades.

The book 'On Growth and Form' by the Scottish mathematical biologist D'Arcy Thompson (1917) was widely admired by biologists, anthropologists and architects amongst others and inspired thinkers like the biologists Huxley and Waddington or the mathematician Turing. He developed the 'theory of transformations' to show how forms of different species could be geometrically related to each other. The roots of plant modelling are going back to the early 1960s, when botanists and computer scientists started to produce first synthetic images of plant-like objects.

On the basis of cellular automata, first branching structures were produced by Ulam (1966). Cohen (1967) generated the first continuous growth models with realistic looking branching structures. More common and still of great impact are the string rewriting system (SRS)s developed by Lindenmayer in 1968 (subsequently called Lindenmayer systems or short, L-systems). His original mathematical formalism could be used to describe the development of linear and

branching structures at the cellular level. Independent of Lindenmayer, Honda (1971) introduced a first three-dimensional architecture model to produce tree structures.

By the beginning of the 1980s, the progress in computer and graphics performance had led to increased attention to this topic outside the scientific community. The term 'virtual plants' (Room *et al.* 1996) was used to describe the precursor of what should later be known as FSPM. Arguably, the modern animation and computer game industry would not be what it is now had it not been for computer-generated plant models. Recently, the simulation of physiological processes such as growth and development with the aid of computer graphical modelling has become quite popular as a method alongside experimentation in developmental plant biology. It also plays a significant role in systems biology and mathematical biology which has led to interdisciplinary applications in agronomy, horticulture or forestry, as reflected in numerous reviews, (e.g., Grieneisen and Scheres 2009; Jönsson and Krupinski 2010; Prusinkiewicz 2004).

Current data acquisition techniques, e.g., for tracking growth and branching in 3-d, measuring flows and concentrations of hormones or metabolites, techniques for scanning 3-d structural data, are yielding enormous amounts of data. Aksoy *et al.* (2015) used infrared stereo image sequences to track young leaves in an automatic and non-invasive manner. At the cellular level, Fernandez *et al.* (2010) developed an approach of multiangle image acquisition for three-dimensional reconstruction and segmentation of cells. New approaches to allow non-destructive and non-invasive estimation of root pressure using continuous measurements of sap flow and stem diameter variation were developed by de Swaef *et al.* (2013). These new techniques which are characterised by their enormous data output are not only challenging but also offer the possibility to do investigations at unprecedented temporal resolutions.

Today, several specialised software tools for plant modelling have been developed. For a short overview see Appendix A.

2.1. Methods of Plant Modelling

Depending on the scientific background and aim of the modeller several different approaches for plant modelling have been developed during the past decades. The following summary is not meant to be exhaustive but tries to keep a chronological order.

2.1.1. Reaction-Diffusion Systems

Turing (1952) postulated a one-dimensional ring of cells initially of identical length and configuration (concentration of chemical components). Using a specific set of chemical reactions and a set of reaction rate constants, he was able to demonstrate that statistical fluctuations in the concentration of certain cells would be sufficient to introduce a periodicity in the growth rate of the cells around the ring. With this work, he described how a natural pattern (a pattern of low and high concentrations) might spontaneously emerge out of an initially homogeneous, uniform state. The theory behind this is the so-called reaction-diffusion model. They are based on local chemical reactions in which the substances are transformed in space and time. Reaction-diffusion systems have served as basic models in theoretical biology and as prototype models for morphogenesis and pattern formation (Harrison 1994) and had a seminal influence on chaos theory (Gribbin 2005). Unfortunately, Turing was unable to finish his work on two-dimensional extensions of his model.

Related models were introduced by Gierer and Meinhardt (1972) under the name of activator-inhibitor and activator-substrate systems. Meinhardt (1982, 2009) extensively investigated such systems to simulate pigmentation patterns of shells of molluscs. In plant modelling, reaction-diffusion models play a minor role. They have been used, e.g., to explain the patterning of trichomes in leaves and hair cells in roots (Digiuni *et al.* 2008; Savage *et al.* 2008). Of a considerably greater impact were models of active transport (e.g., of the plant hormone auxin) which were developed to overcome the limitations of diffusion, which is very slow over long distances (Crick 1971). A spatially explicit reaction-diffusion system was used by Cartenì *et al.* (2014) to simulate spatial pattern of procambium, phloem and xylem, starting from a homogeneous group of undifferentiated cells.

2.1.2. Two-dimensional Growth Patterns

In 1961, Eden (1961) was one of the first to investigate two-dimensional growth patterns of cell populations. Starting from considering a single cell or a homogeneous population where the cells may repeatedly divide into daughter cells he looked at the structural properties of the resulting colony of cells and how various possible constraints affect the architecture.

2.1.3. Cellular Automata

The concept of cellular automata was originally developed in the 1940s by von Neumann (1966). Ulam (1966) used and extended them to come up with the first models of simple branching structures.

A cellular automaton is characterised as follows: It consists of a one-, two-, three- or higher-dimensional regular grid of cells equal in shape and size, where each cell can have one of a finite number of states, e.g., 'on' and 'off'. The initial state of all cells is defined by the user. A set of rules determines the new state of each cell in terms of the current state of the cell and the states of the cells in its neighbourhoods. The new state of the system is determined by application of the rules to all cells in parallel.

An 'elementary cellular automaton' is the simplest class of cellular automata. Its domain is a one-dimensional chain of cells (commonly called 'string'), were each cell only has one of two states ('on' or 'off' / '0' or '1'). The rule set applied to a cell depends on the values of its nearest neighbour. For a given cell there are only two neighbouring cells - a left and a right neighbours. Under the condition that each cell has only one of two states, there are $2{\times}2{\times}2 = 2^3 = 8$ possible binary states for the three cells. For each of these states, a new state of the middle cell needs to be defined when this rule can be applied. A set of rules of one elementary cellular automaton can be completely described by a table where for each possible binary configuration of the three cells the following state is defined. Since the following state again is a binary value, there are in total $2^8 = 256$ different elementary cellular automata, each one of which can be indexed by an 8-bit binary number whose decimal representation is known as the '*rule*' for the particular automaton (Wolfram 1983, 2002). The application of the rules to an initial state (generation zero) will generate the first generation which will usually be drawn in the second row. In the next step the produced generation will be the initial state for the next rule application, and so on.

On the top of Fig. 2.1 the rules for the elementary cellular automata '*rule 126*' are given, below the evolution it produces after 15 steps starting from a single black cell placed in the middle of the string is shown. In this diagram, the possible values of the three neighbouring cells are shown in the top row, and the resulting value the central cell takes in the next generation is shown below in the centre.

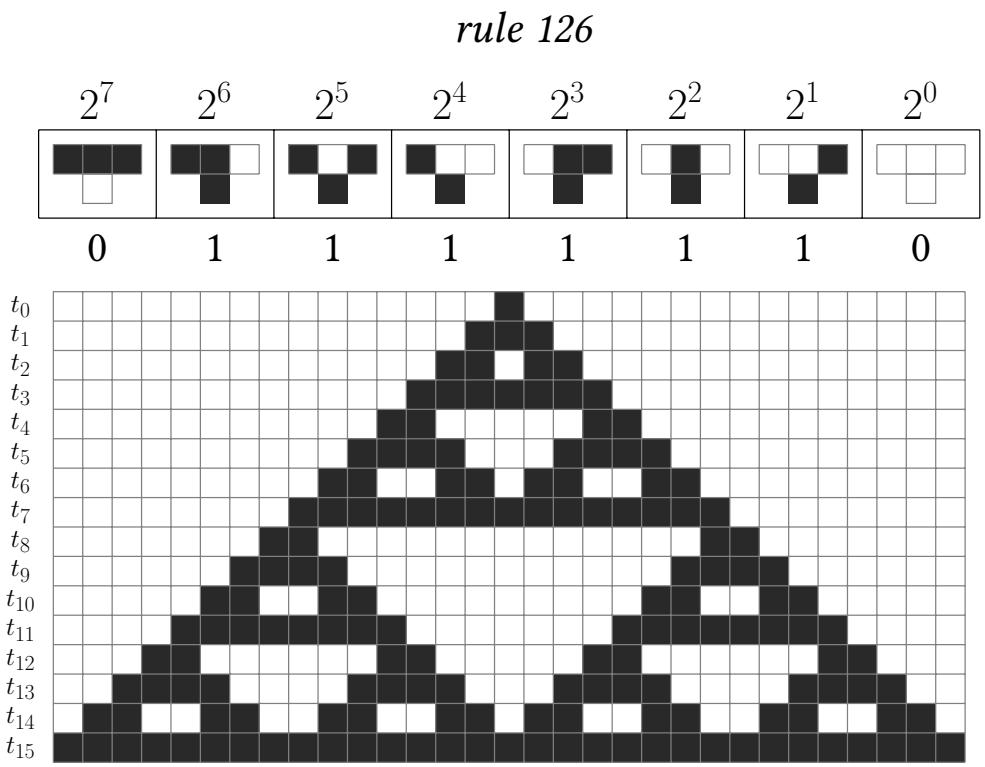


Figure 2.1.: Elementary cellular automaton '*rule 126*'; upper part shows the rules for this automaton, lower part the evolution of the first 15 steps.

Wolfram (1983) studied all 256 possibilities of such automata. In Wolfram (2002), he introduced four classes of automata according to the behaviour of these automata to examine their evolution starting with a random state. The so-called 'Wolfram classes' are given in Table 2.1.

Later, Wolfram (1984) worked on natural systems like snowflakes or mollusc shells and showed how the origins of their complexity could be investigated through cellular automata. They were analysed both as discrete dynamical systems and as information-processing systems.

The common shape of cells in two-dimensional cellular automata is quadratic. Depending on the shape different variants of neighbourhoods can be distinguished. The simplest case is the 'nearest neighbourhood', better known as von Neumann neighbourhood (Eq. 2.1), in which only the four cells directly adjacent to a given cell are taken into account. The Moore neighbourhood (Eq. 2.2), in addition to the von Neumann neighbourhood, considers the four 'corner cells'. For both the von Neumann and the Moore neighbourhood the range r defines

Table 2.1.: Wolfram classes of elementary cellular automata for a random initial state according to Wolfram (2002).

Class	Cellular automata which	Example rule
1	rapidly converge to an uniform state	0, 32, 160, 232
2	rapidly converge to a repetitive or stable state	4, 108, 218, 250
3	appear to remain in a random state	22, 30, 126, 150
4	form areas of repetitive or stable states, but also form structures that interact with each other in complex ways	110

the distance of the neighbouring cells to the given cell (x_0, y_0). Besides quadratic cells other divisions, e.g., triangular or hexagonal are possible.

$$N^N_{(x_0,y_0)} = \{(x, y) : |x - x_0| + |y - y_0| \leq r\} \tag{2.1}$$

$$N^M_{(x_0,y_0)} = \{(x, y) : |x - x_0| \leq r \wedge |y - y_0| \leq r\} \tag{2.2}$$

Ulam (1966) constructed simple two-dimensional branching structures. As an illustration, the first three generations of such a pattern and the initial situation with only one 'activated' cell placed in the middle are given in Fig. 2.2. Underlying this pattern is only one simple rule: 'Turn all neighbours of an 'activated' cell on, if there is only one neighbour turned on.'.

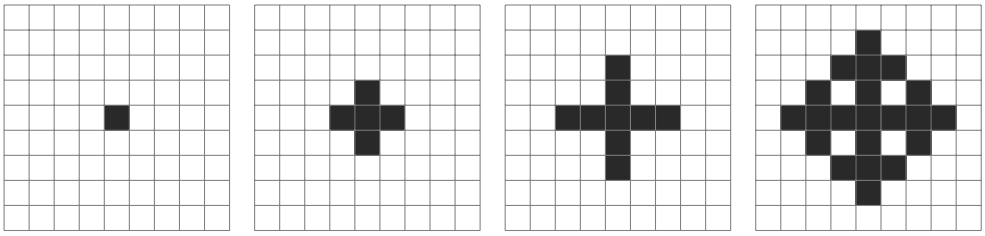


Figure 2.2.: Discrete branching structures following Ulam (1966). Initial state and first three iterations of a simple branching structure by Ulam (1966).

Based on an idea by von Neumann, who attempted to find a hypothetical machine that could build copies of itself, in 1970, Conway simplified his approach to what is known as 'Conway's

game of life', probably the best-known two-dimensional cellular automaton (see Gardner 1970, 1983). The game of life is a binary (two states) totalistic cellular automaton with a Moore neighbourhood of range $r = 1$ working on a potentially infinite grid with quadratic cells.

From a theoretical point of view, Conway's game of life is interesting because it has the power of a universal Turing machine. It has been shown that there exist universal cellular automata that are capable of simulating the behaviour of any other cellular automaton or Turing machine. Gács (2001) has proven the existence of fault-tolerant universal cellular automata, whose ability to simulate other cellular automata is not constrained by random perturbations provided that such perturbations are sufficiently sparse.

2.1.4. Procedural Models

Procedural models are parameterised algorithms that are usually designed for the simulation of a certain plant type or a single species, respectively. They could be seen as classical simulation programs ('stand alone programs'), see Fig. 2.40.

Cohen (1967) implemented a first procedural model to generate branching structures. He used three relatively simple rules implemented in Fortran (Backus 1998) to control growth and the type of branching in a structure:

- Growth takes place only at the tip of the branches.
- Strength and angle of growth are determined by the current state of the system.
- Branching is determined by a probabilistic measure that depends on the distance to the last branch and the current state of the system.

Characteristic for procedural modelling is that algorithms are applied to produce scenes and textures. In an iterative process, the above set of rules is applied to an initial state. At each discrete step, the rules are repeatedly applied to develop the final structure. The set of rules is typically implemented in the algorithm and configured by parameters.

The first three-dimensional procedural model was developed by Honda (1971) and Fisher and Honda (1977, 1979). The two botanists worked on the simulation of branching structures of trees and other plants (Fig. 2.3). Honda *et al.* (1981) used this approach to investigate local control mechanisms of branch interaction with a focus on preventing overlapping and intersection of branches in trees.

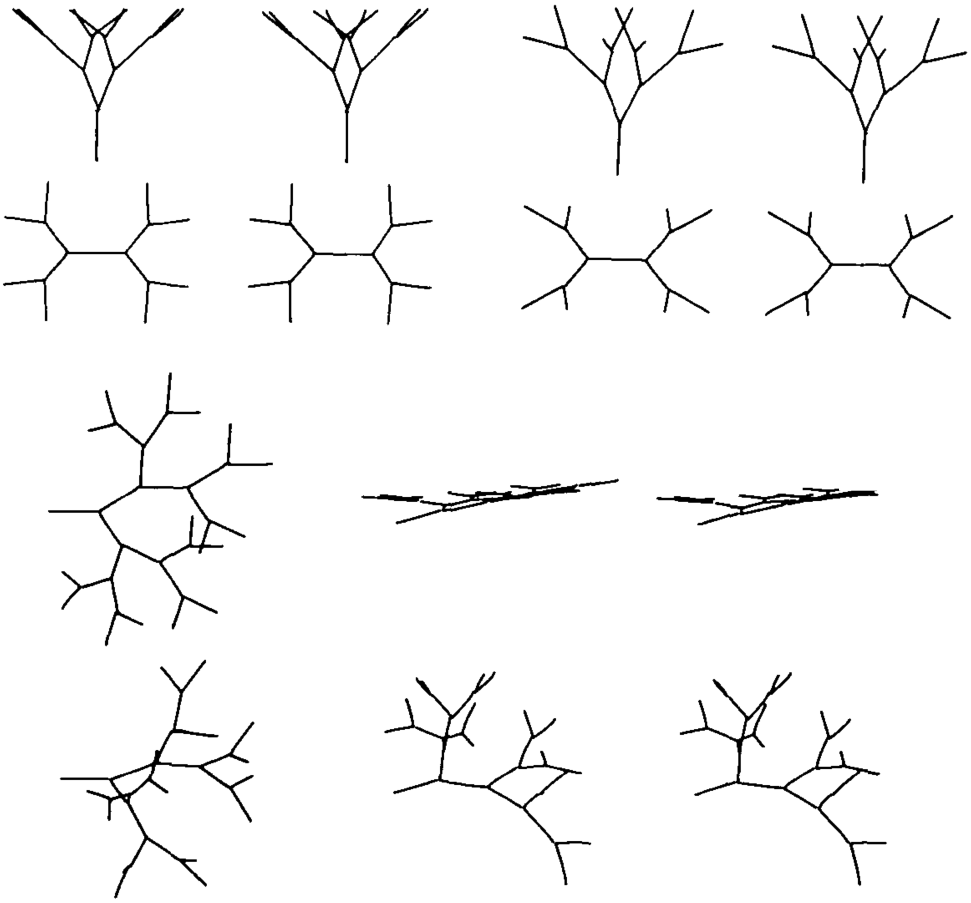


Figure 2.3.: Simulations of branching patterns by Honda *et al.* (1982).

At this point, the model itself is still relatively simple, based only on a few rules. Several simplifications are made, e.g., the internodes are straight line segments with no thickness; only binary branching occurs, while the branching intensity increases in discrete steps with each branching order; branching angles are constant. The produced 3-d skeletons are projected onto the viewing plane to obtain images.

Based on the work by Fisher and Honda (1977), Aono and Kunii (1984) extended the approach with respect to producing realistic branching patterns. They used bifurcation, i.e. binary

branching as control mechanism and introduced constant factors to decrease length and diameter of internodes with each increment in branching order. With the introduction of attractors and inhibitors, the influence of wind was simulated. At the same time, Borchert and Honda (1984) used an extended version of the approach to simulate geometry and development of the branch system for *Tabebuia rosea*. Kawaguchi (1982) obtained very impressive results using particular branching patters.

Honda and Hatta (2004) published a procedural model of a general branching system in which they included phyllotaxis and the effect of gravity.

2.1.4.1. Meristem-Oriented Plant Modelling

One of the first dynamical and morphological plant simulators based on meristem growth was introduced by Bell *et al.* (1979). Bell used three basic bud fates: break to form a new shoot, fall into dormancy and get reactivated, or die off. Similar to Bell, de Reffye *et al.* (1988) used a procedural approach to introduce a model which integrated botanical laws explaining plant growth and architecture. Based on his thesis (de Reffye 1979), in which the mathematical model was described and used to simulate the growth of coffee trees, de Reffye extended his approach to produce models for a great variety of plant and tree species. Another important aspect of his model was the integration of time which enabled the simulation of ageing, e.g., birth and death of leaves and branches.

As the meristem is a tissue that contains stem cells, it is the zone that produces new tissue within a bud. Therefore, de Reffye used a meristem-based approach of modelling, in which growth is simulated in discrete time steps. De Reffye *et al.* (1988) used a stochastic process with transition probabilities assigned to each bud to quantify the transition between the three different states, and this for different species. De Reffye *et al.* (1988) defined the following bud states:

- form a flower,
- rest (fall into dormancy),
- break and form one or several phytomers (internode, node, axillary meristem) plus a terminal bud,

- or die off (and thus disappear).

Geometric parameters, such as organ shape or branching angles, are calculated according to species-specific stochastic parameters. By applying these simple rules iteratively to an initial bud a sequence of nodes and internodes with leaves and branches can be generated, whereby the branching order influences growth speed in that structures of higher order grow more rapidly. At each step, internal parameters (order, age, dimension, position, etc.) are updated.

The simulation can be expressed in pseudo-code as follows (de Reffye *et al.* 1988):

```
for each clock signal do
  for each bud which is still alive do
    if bud doesn't die then
      if bud doesn't make a pause then
        create intenode {with position in space}
        create apical bud
        for each possible bud do
          if ramification then
            create axillary buds
        endfor
      endif
    endif
  endfor
endfor
```

For growth, de Reffye introduced the botanical term of the growth unit which is a sequence of internodes and nodes produced by the apical bud of the previous node within one growth cycle. He also used the term *order* of axis to describe the branching order of a structure as shown in Fig. 2.4.

Later, the approach of meristem-oriented modelling was included in AMAPstudio (see Sec. A.2.1 for details) as well as in the commercial software studio Bionatics (see www.bionatics.com).

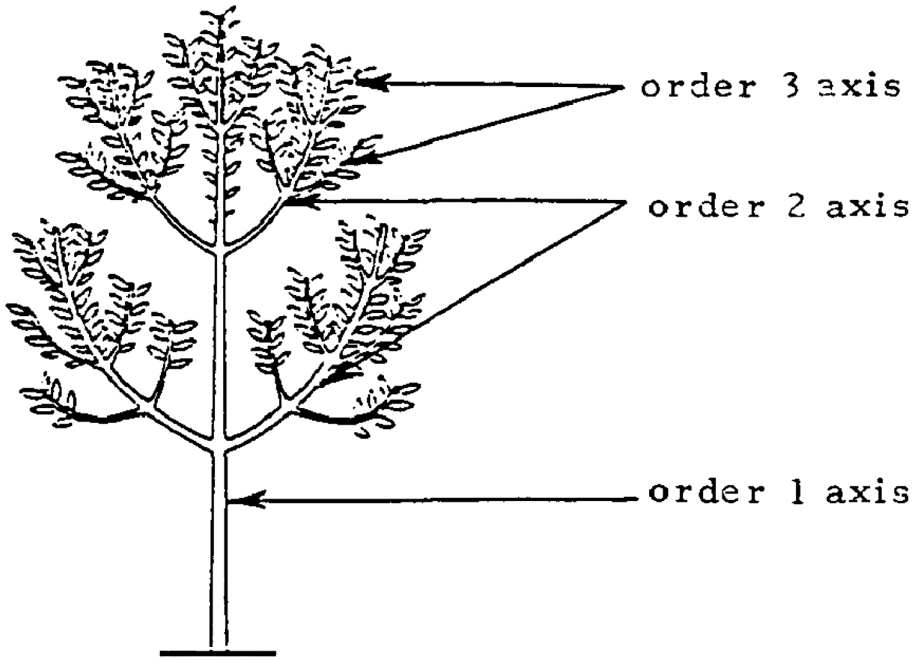


Figure 2.4.: Notation of order of an axis according to de Reffye *et al.* (1988).

2.1.5. Rewriting Systems

In mathematics, computer science, and logic several types of rewriting systems are known. Common to all of them is the method of replacing sub-terms of a formula with other terms. Typically, they consist of a set of objects, plus relations on how to transform those objects. Rewriting is a potentially non-deterministic process in which rules can be applied in different ways, or in which more than one rule is applicable.

Of particular importance for plant modelling are string rewriting systems and their extension, the so-called graph rewriting systems, which are both special cases of rewriting systems. The cellular automata described above can also be classified as rewriting systems.

2.1.5.1. String Rewriting Systems

The Norwegian mathematician Thue worked on an extension to logic which would allow to define mathematical theorems in a formal language with the intention to have a system to

prove and verify theorems in an automatic way in order to solve the word problem for finitely presented semigroups. Thue (1914) introduced a systematic treatment of SRSs which is now known as semi-Thue system, the historical name of SRS.

A SRS is typically defined by a finite alphabet, a binary relation between fixed strings over the alphabet, called rewriting rules, and an initial start word - a formal grammar. In an iterative way, the rules are applied first to the start word and in the next derivation to the derived string, and so on. During each derivation step, all matches of left-hand sides of the rules are searched in the current string and one is replaced by the corresponding right-hand side of the rule to obtain the next string. In the classical definition of SRSs strings are replaced sequentially. As a formalism, string rewriting systems are Turing complete.

2.1.5.2. Lindenmayer Systems (L-Systems)

The approach with probably the largest and most durable impact on plant modelling was developed by Aristid Lindenmayer (1925-1989), a biologist at the University of Utrecht. In 1968, he investigated growth patterns of cyanobacteria (blue-green algae), such as *Anabaena catenula* (Lindenmayer 1968). He aspired to obtain a formal description for the development of such simple filamentous structures. Later, extensions of this formalism in combination with graphical interpretations (cf. turtle geometry) were used to model the morphology of a variety of higher plants and complex branching structures. Several examples, applications and thorough explanations can be found in the book by Prusinkiewicz and Lindenmayer (1990) 'The Algorithmic Beauty of Plants', one of the standard books in computer graphics. Today, the approach is known as Lindenmayer systems or short, L-systems. With the success of this approach, several extensions and adoptions have been made during the past decades, to tackle new emerging challenges.

Smith (1984) introduced the term 'graftals' for this class of formal languages and their graphical interpretation. He argued that the generated models are often not fracta,l so the common parts of the fractal theory and plant theory are abstracted. Due to the recursive nature of the L-system rules, the graphical interpretation of L-systems shows a self-similarity and thereby, fractal-like forms (Ferraro *et al.* 2005; Prusinkiewicz 2004).

2.1.5.3. Formal Definition

L-systems are a special case of parallel string rewriting systems. An L-system is thus a formal grammar which can be defined by a triple $G = (\Sigma, \alpha, \Delta)$:

- Σ, a not empty, finite set of symbols called alphabet,
- $\alpha \in \Sigma^+$, an initial string called 'axiom',
- Δ, a set of production/derivation rules, i.e. $\Delta \subseteq \Sigma \times \Sigma^*$.

The application of a production rule to a string (word) at stage t to $t + 1$ is called rewriting or derivation step. New strings are derived from (or generated by) an iterative process starting from the initial state (axiom or α), repeated until a defined recursion depth is reached. The replacement follows these rules:

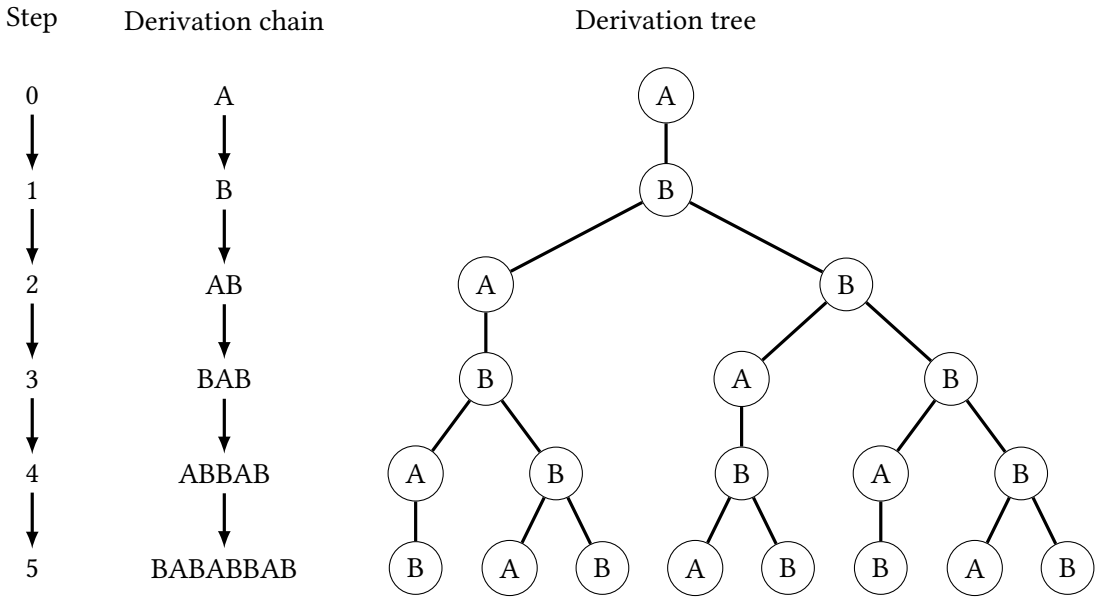
- Every symbol in the current string for which a matching left-hand side exists is replaced by the right-hand side of a rule.
- Rules are applied in parallel.
- Symbols to which no rules can be applied (i.e. for which no matching left-hand side exists), are taken over unchanged into the next string σ_{t+1}. These symbols are often denoted as constants and the identity production is assumed.

In this definition, only one individual symbol is allowed on the left-hand side, they are commonly known as **context-free** L-systems. Context-free L-systems are specified by either a prefix grammar, or a regular grammar. Furthermore, if there is exactly one production rule for each symbol defined, then the L-system is said to be **deterministic**, see Sec. 2.1.5.7. For deterministic context-free L-systems the term **D0L-system** was used by Prusinkiewicz and Lindenmayer (1990).

Example: Lindenmayer's original L-system for modelling the growth of algae.

Alphabet	:	$\sum = \{A, B\}$
Axiom	:	$\alpha = A$
Rules	:	$\Delta = \{A \rightarrow B,\ B \rightarrow AB\}$

The first five derivation steps are given below:



Interestingly, the length of each string in the sequence produced by this example follows the well-known Fibonacci series: 1 1 2 3 5 8 13 21 34 ...

2.1.5.4. Turtle Geometry

To obtain two- or three-dimensional visualisation of a string produced by an L-system, the string, or more precisely, some of its symbols need to be graphically interpreted. First work on this visualisation was done by Frijters and Lindenmayer (1974) and Hogeweg and Hesper (1974). Both used a two-stage process: 1) use L-systems to primarily determine the branching topology of the modelled plants, and 2) add the geometry in a post-processing phase, see Fig. 2.5. One commonly used approach doing this is the so-called turtle geometry, also referred as turtle graphics. Abelson and diSessa (1981) give an exhaustive introduction of 2-d turtle geometry, including turtle graphics on curved surfaces. The turtle can be seen as virtual pen upon a

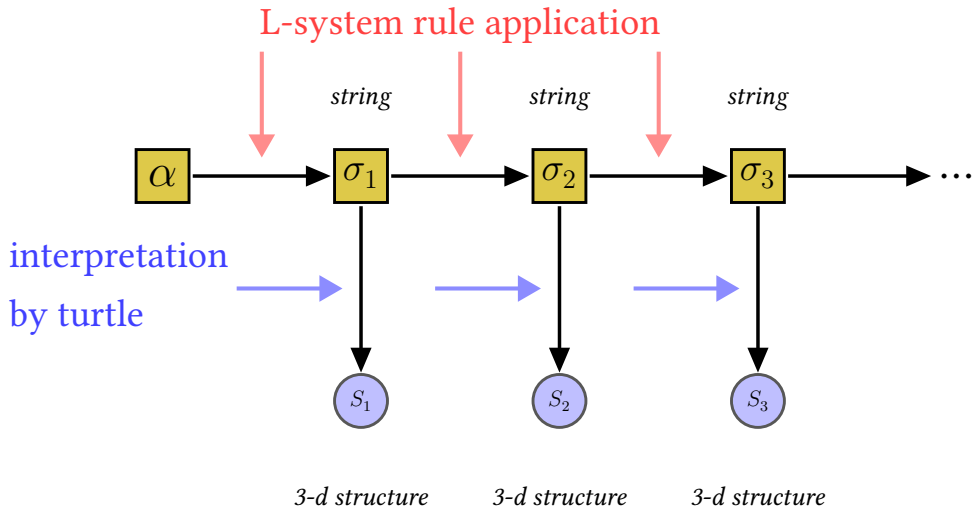
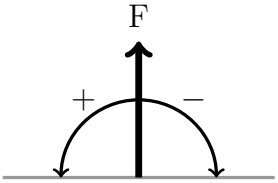


Figure 2.5.: Schematic representation or the functioning of a classical L-system using a two-stage process with turtle interpretation. The rules are applied to the start symbol α (axiom) and afterwards graphically interpreted by the turtle to obtain the final 3-d structures S_1, S_2, ...

Cartesian coordinate system which is controlled by commands. Each symbol of the alphabet will be interpreted according to a defined 'translation' as shown in Tab. 2.2. A turtle has three attributes: 1) a location, 2) an orientation, and 3) a pen, itself having attributes such as colour, width, etc. For a complete list of turtle commands refer to Prusinkiewicz and Lindenmayer (1990). The set of turtle commands becomes a subset of the alphabet of the L-system. Symbols that are not turtle commands will be ignored by the turtle during interpretation. In computer graphics, turtle geometry is classified as vector graphics (system).

Table 2.2.: Interpretation of turtle commands in 2-d (extract).

F	Move the turtle forward by one step of length d and draw a line.	
f	Move the turtle forward by one step of length d.	
+	Turn the turtle anticlockwise around its UP vector.	
-	Turn the turtle clockwise around its UP vector.	

The turtle can be described by a triple (x, y, α) in 2-d where x and y refer to the Cartesian coordinates - the position the plane - and α defines the angle/direction of movement. By adding the Z-axis and two additional angles the turtle geometry can be extended to the three-dimensional space.

Example: Koch curve, Koch (1906)

Alphabet	:	$\sum = \{F, +, -\}$
Axiom	:	$\alpha = F$
Rules	:	$\Delta = \{F \to F + F - - F + F\}$

The first four derivation steps are given below:

Step	Derivation chain	$\sharp\, F$	$\sharp\, +/-$
0	F	1	0
1	$F+F--F+F$	4	4
2	$F+F--F+F+F+F--F+F--F+F--F+F--F+F$	16	20
3	$F+F--F+F+F+F--F+F--F+F--F+F--F+F+$ $F+F--F+F+F+F--F+F--F+F--F+F--F+F--$ $F+F--F+F+F+F--F+F--F+F--F+F--F+F+$ $F+F--F+F+F+F--F+F--F+F--F+F--F+F$	64	84
4	...	4^n	$-4/3(1-4^n)$

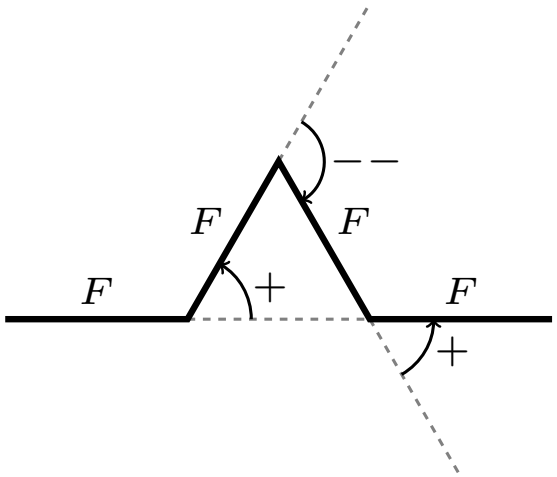


Figure 2.6.: Turtle interpretation of '$F + F - - F + F$'. When the base angle δ is set to 60 degrees it yields the first step of the Koch curve, (Koch 1906).

The (illustrated) interpretation of the first derivation step $F + F - - F + F$ is given in Fig. 2.6 for the base angle δ = 60 degrees. In words: draw a line of length d, turn the viewing direction about 60 degrees clockwise. Draw another line and turn the turtle twice about 60 degrees anti-clockwise. Draw one more line before the turtle is turned the last time clockwise, and the final

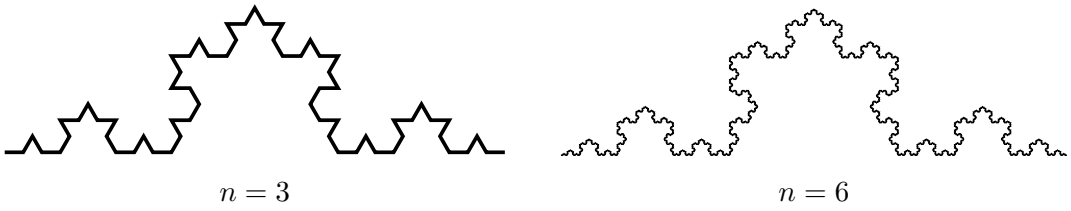


Figure 2.7.: Graphical interpretation of the 3rd and 6th derivation step of the Koch curve.

line is drawn. The graphical interpretations of the 3rd and 6th derivation step are given in Fig. 2.7.

By only changing the initial state from a single stroke to an isosceles triangle and keeping the rules and the base angle δ the same, the so-called 'Koch snow flake' (Fig. 2.8) can be obtained.

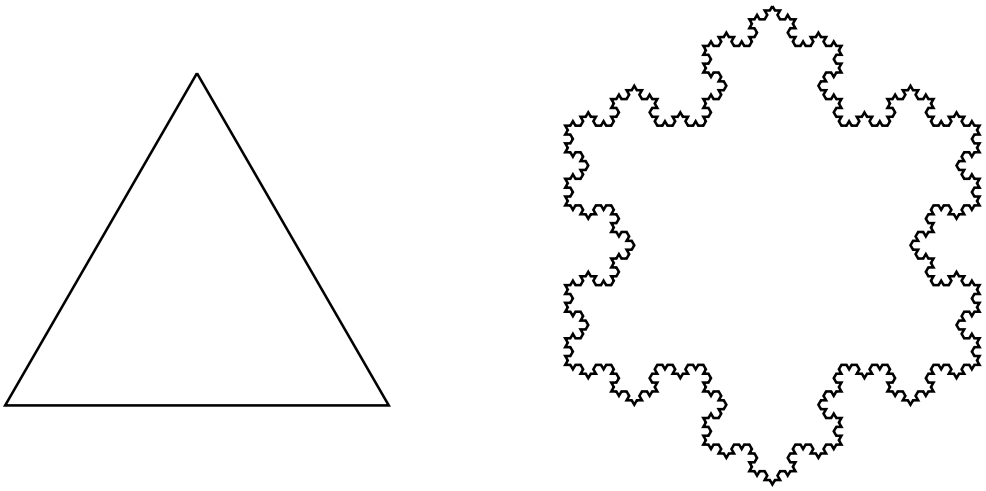


Figure 2.8.: The 'Koch Snow flake'; left, the initial triangle; right, the graphical interpretation of the 4th derivation step.

When it comes to the three-dimensional space the location of the turtle can be defined by a triple (x, y, z) where x, y and z refer to the Cartesian coordinates. To define the orientation of the turtle three angles are needed: one for tilting, one for inclining and one for turning, the so-called Euler angles or roll-pitch-yaw angles. In terms of turtle commands this refers to rotating around the left-, up-, and head-axis, respectively called RL, RU, and RH. In Fig. 2.9 these rotations are demonstrated with a box object.

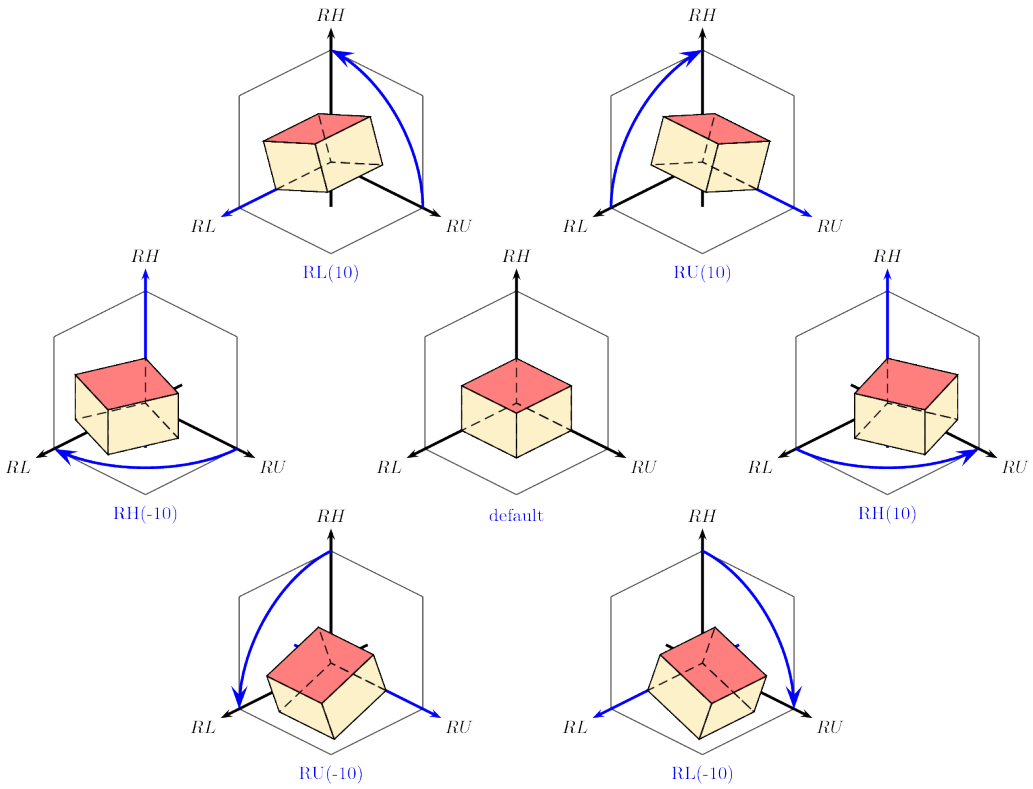


Figure 2.9.: Rotations of a box in 3-d around the origin. Starting with the default situation in the middle (z-axis matches the head-axis) rotations of 10 degrees are performed around each axis.

The turtle interpretation was later further extended by Prusinkiewicz (1987) to allow for incorporating predefined curved surfaces in the model.

2.1.5.5. Bracketed L-Systems

With the L-systems defined so far, it is only possible to create simple unbranched structures. An important extension that allows the representation of branched structures are bracketed L-systems. By introducing a stack, the current state of the turtle can be pushed on a stack and later be popped from the stack to make it the current turtle state again. Therefore, the alphabet needs to be extended by two symbols: an open square bracket '[' and a closing square bracket ']'. While '[' represent the pushing (storage) of the current turtle state onto the stack,

‘]’ induces the popping (recovery) of the first state from the stack and its use as the new state of the turtle.

Example: A simple 2-d branching structure.

$$
\begin{array}{lcl}
\text{Alphabet} & : & \sum = \{A, B, F, +, -, [,]\} \\
\textit{Axiom} & : & \alpha = A \\
\text{Rules} & : & \Delta = \{A \to F[-B][+B]A,\ B \to FB\}
\end{array}
$$

Derivation chain:

$$
\begin{array}{rcl}
A & \to & F[-B][+B]A \\
 & \to & F[-FB][+FB]F[-B][+B]A \\
 & \to & F[-FFB][+FFB]F[-FB][+FB]F[-B][+B]A \\
 & \to & \ldots
\end{array}
$$

One graphical interpretation of the symbols (cf. turtle geometry) and the derivation chain is given in Fig. 2.10.

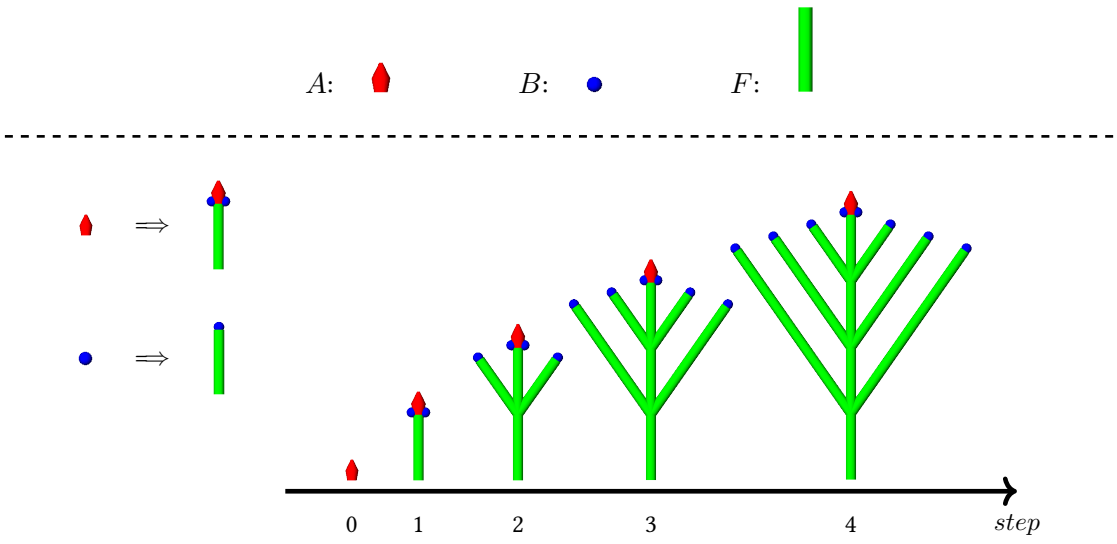


Figure 2.10.: Graphical interpretation of the symbols and the first four derivations steps of a simple branching structure.

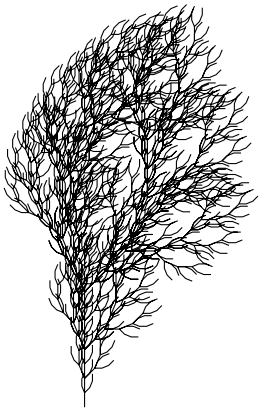
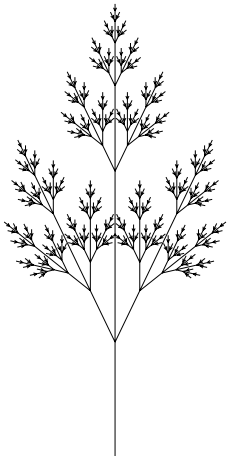
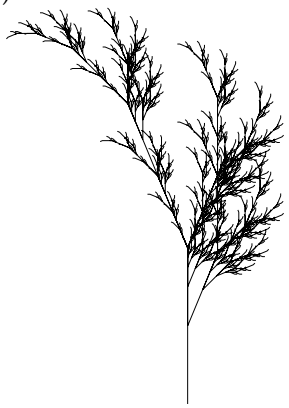
a)	b)	c)
		
n=4 $\delta = 22.5°$ $\alpha = F$ $F \Rightarrow FF - [-F + F + F] + [+F - F - F]$	n=7 $\delta = 25.7°$ $\alpha = X$ $X \Rightarrow F[+X][-X]FX$ $F \Rightarrow FF$	n=6 $\delta = 22.5°$ $\alpha = X$ $X \Rightarrow F - [[X] + X] + F[+FX] - X$ $F \Rightarrow FF$

Figure 2.11.: Examples of plant-like structures generated by bracketed L-systems (Prusinkiewicz and Lindenmayer 1990).

This small extension of the formalism already allows to generate very simple but plant-like structures (Fig. 2.11), which are, however, too regular to look realistic. To overcome this deficiency, further extensions are needed, e.g., context-sensitive L-systems, stochastic L-systems and parametric L-systems.

2.1.5.6. Context-Sensitive L-Systems

With the current definition of L-systems (Sec. 2.1.5.3) only one single symbol is allowed on the left-hand side of a rule. It is not possible to take any context into account for a rule to be applied. Therefore, such L-systems are called **context-free** L-systems.

To take a left and right context of a symbol into account, the former definition of context-free L-systems needs to be slightly changed with respect to the definition of Δ. The resulting L-systems are called **context-sensitive** L-systems.

A context-sensitive L-system is thus a formal grammar which can be defined by a triple $G = (\Sigma, \alpha, \Delta)$:

- Σ, a not empty, finite set of symbols called alphabet,
- $\alpha \in \Sigma^+$, an initial string called 'axiom',
- Δ, a set of production rules: $\Delta \subseteq \Sigma^+ \times \Sigma^*$.

Now the left-hand side of a rule consists of a left context, the symbol to be replaced, and a right context. A context-sensitive production rule, therefore, looks also for the symbols on the string appearing before and after the symbol that it is going to modify. For instance, the production rule:

$$a < b > a \Rightarrow c$$

will replace 'b' by 'c' if and only if 'b' is surrounded by two 'a' in the input string.

Similar to context-free L-systems, a symbol is not changed and taken over to the next derivation step if no rule can be applied. In case several rules can be applied to a symbol, the rule with the longest left-hand side is assumed to take precedence.

The Chomsky hierarchy of formal languages (Chomsky 1956) orders these formal languages according to their 'power'. In contrast to L-systems where the rules are applied in parallel and simultaneously replace all characters of a given word, in Chomsky grammars rules are applied sequentially. The languages produced by context-free and context-sensitive L-systems can be included in the Chomsky hierarchy (Fig. 2.12) but they will overlap with the Chomsky classes. For example, there are languages which can be generated by context-free L-systems but not by context-free Chomsky grammars (Prusinkiewicz and Lindenmayer 1990).

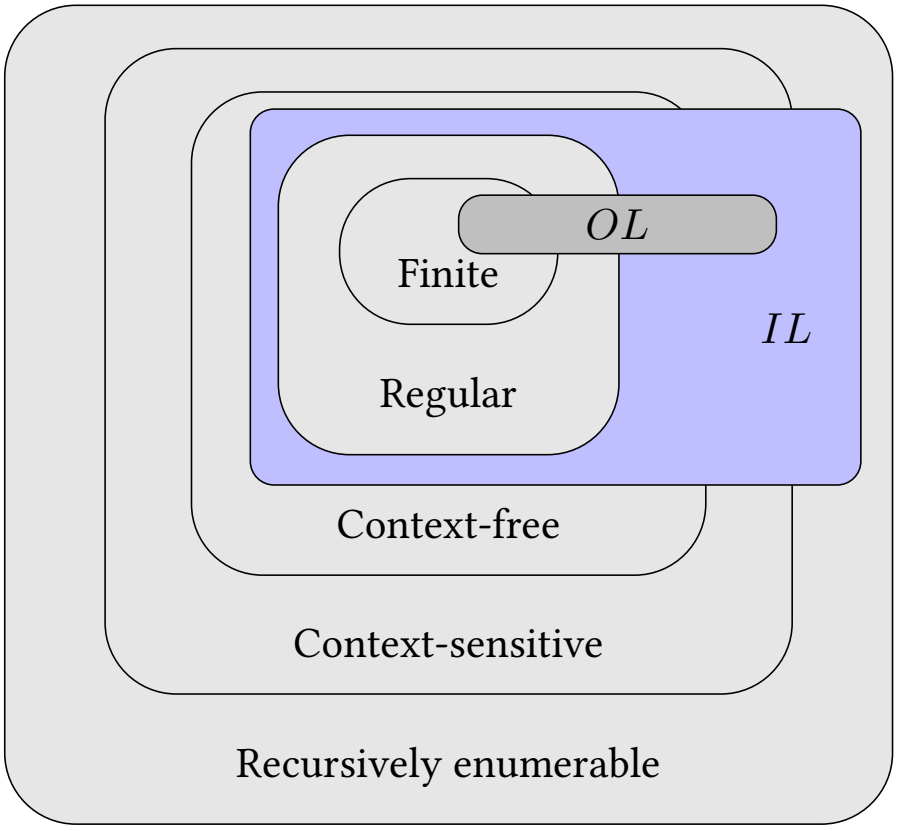


Figure 2.12.: Classification of the languages generated by context-free OL and context-sensitive IL L-systems into the Chomsky hierarchy of formal languages (Chomsky 1956) according to Prusinkiewicz and Lindenmayer (1990).

2.1.5.7. Stochastic and Deterministic L-Systems

An L-system is called deterministic, if for any symbol in the alphabet Σ exactly one rule is defined. Deterministic context-free L-systems are also known as D0L systems (Prusinkiewicz and Lindenmayer 1990). If, for the same symbol, several rules are defined and their application is associated with a certain probability function during each derivation step, the L-system is called **stochastic**. Consider the following two rules:

$$a, (0.6) \Rightarrow b$$

$$a, (0.4) \Rightarrow c$$

'a' will be replaced with 'b' with a probability of 60% and with 40% with 'c'. The summed probabilities applied to one symbol have to be less than or equal to 1 (100 percent). If only one probabilistic rule is defined, the symbol will be replaced only with the indicated probability.

With different random seeds used for the random number generator, a stochastic L-system will always produce different results, whereas the result will be identical if the same random seed is used. As a result, stochastic L-systems are one way to overcome the symmetry of common L-systems and consequently help to increase the level of realism of the produced structures.

2.1.5.8. Parametric L-Systems

With the preceding definitions and extensions of L-systems it is not possible to change the global parameter of an L-system, such as the base angle δ or the default length of a line d. This is an important limitation which dramatically reduces the number of drawable objects. Take, for instance, an isosceles triangle which has two sides of equal length. If the length of the equal sides is one the hypotenuse has to have the length of $\sqrt{2}$.

In **parametric** L-systems (Prusinkiewicz and Lindenmayer 1990) each symbol of the alphabet Σ can have a list of parameters attached. Such a parameterised symbol is often called module. Within a module definition, the parameters can be used to manipulate, e.g., the global constants of the turtle.

Example: Parameterised version of the simple 2-d branching structure example from page 25.

$$
\begin{array}{lcl}
\text{Alphabet} & : & \sum = \{A, B, F, +, -, [,]\} \\
\textit{Axiom} & : & \alpha = A \\
\text{Rules} & : & \Delta = \{A \rightarrow F[-B][+B]A,\ B(x) \rightarrow F(x)B(0.6 * x)\}
\end{array}
$$

Here a parameter x is attached to B and F. At each derivation step where the second rule is applied to a B the x is transferred to F and to B. While for the new B the x is multiplied by 0.6 and therefore reduced by 40%, it is used to set the length of a stroke within F. The resulting graphical interpretations of the first four derivation steps are given in Fig. 2.13.

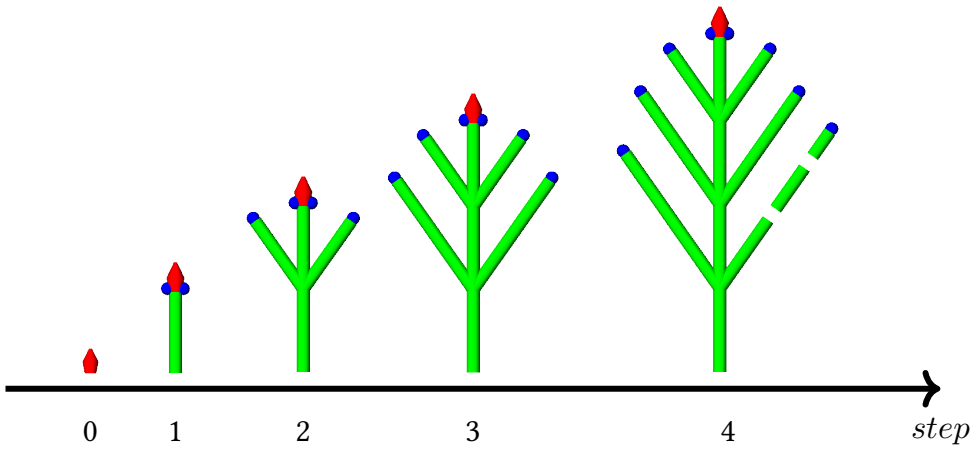


Figure 2.13.: Graphical interpretation of the first four derivations steps of a simple parametric L-system. At the first right branch of the fourth step the individual branch segments are separated and moved slightly to highlight the successive reduction in length of the segments.

By combining parametric L-systems with conditions, parameters can also be used within the condition of a rule.

$$a(x), (x < 10) \Rightarrow b$$

In this rule a is only replaced as long as its parameter x is smaller than 10. In a FSPM such a parameter could be interpreted as a concentration of a hormone or of any other threshold value, e.g., age or temperature.

2.1.5.9. Contour Tracing and Interpretation

With the definition and extensions of L-systems so far only simple lines can be drawn. There is no possibility to have surfaces or complex geometric objects. Even single plant organs like leaves need to be constructed with L-system commands, which could be a challenging and time-consuming task when objects turn to be complex like fragile leaflets or petals. Moreover, to simulate organ development, it is necessary to have a mechanism at hand for changing the shape as well as the size of surfaces during rule application.

Prusinkiewicz and Lindenmayer (1990) introduced, therefore, an extension called **contour tracing** that allows defining whole plant organs within one step. The contours of such an object are defined by a closed planar polygon that has no influence on the derivation. Only the

Table 2.4.: Interpretation of turtle commands used for contour tracing using a tree structure as a framework.

{	Start a new polygon. If there is already a polygon which was started before, it will be put on a stack of polygons and a new, empty polygon will be started.
.	Add the current position of the turtle to the polygon.
G	Is handled in the same way as a F, only that the end point is not added to the polygon.
}	Close the polygon and draw it. If there are further polygons on the stack pop them from the stack and turn them into the new current polygon.

interpreter - the turtle geometry - has to be extended to draw filled polygons. Table 2.4 gives an overview of the new symbols introduced by Prusinkiewicz and Lindenmayer (1990) used for contour tracing.

The following L-system by Prusinkiewicz and Lindenmayer (1990) illustrates this technique at the example of the development of a cordate leaf. A visual interpretation is given in Fig. 2.14. Due to the definition of the rules the resulting leaf shape is symmetric. It is initialised with A and B, representing the left- and right-hand side of the leaf. The two rules for A and B produce the supporting structure, starting at the origin of the leaf, while the last rule is increasing the length of the lines. The endpoints of the lines are then used as coordinates for the polygon.

$$
\begin{aligned}
\alpha &= [A][B] \\
\Delta &= \{ \\
&\quad A \rightarrow [+A\{.].C.\}\ , \\
&\quad B \rightarrow [-B\{.].C.\}\ , \\
&\quad C \rightarrow GC \\
&\quad \}
\end{aligned}
$$

Based on this approach, Kurth (1994a) extended the way how to apply rules (until now only **generative rules**) and introduced the so-called **interpretative rules**. They were mainly considered as tools to directly draw objects and therefore can be seen as a preliminary step towards

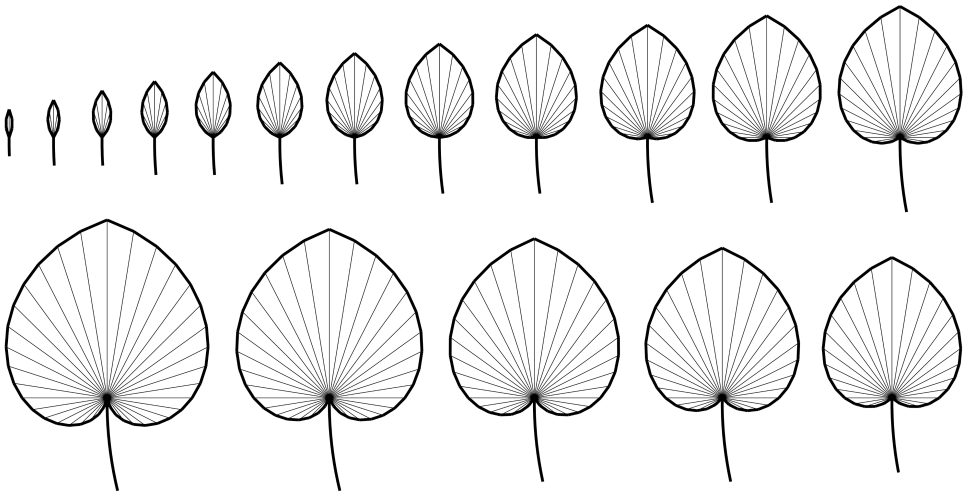


Figure 2.14.: Developmental sequence of a cordate leaf generated using an L-system with contour tracing (Prusinkiewicz and Lindenmayer 1990). The polygons are not filled to allow the supporting structure to be seen.

turtle interpretation. With the help of polygons, this technique allows predefining structures without a complex derivation process. Kurth (1994a) used this technique to model a beech twig where the leaf and bud object are to be drawn by interpretative rules. A graphical interpretation is given in Fig. 2.15. A similar concept by Prusinkiewicz *et al.* (2000a) is called **interpretation rules**. Fournier and Andrieu (1998) used the approach of interpretation under the term 'homomorphism' to specify geometric aspects of a maize model.

2.1.5.10. Limitations of L-Systems

From the theoretical point of view, L-systems are a powerful formalism. With all the extensions described above, they can be used to produce realistic simulations of a large number of macroscopic plant structures. The simplicity of the formalism surely is attractive. However, they have drawbacks when it comes to complex, multi-level plant models including functional aspects and/or when genetic control is required. With increasing complexity, models tend to loose transparency and become difficult to handle.

Another problem has its origin in the linear (data) structure that allows no cycles. In L-systems, substitution rules are designed to generate growing structures, whereas the structures in networks are often fixed. Therefore, network structures like gene regulatory networks, metabolic

Figure 2.15.: Model of a beech twig using interpretation rules; a) objects used for interpretation, b) at step 5 in winter, c) at step 6 in summer with the buds grown out into leaves; from Kurth (1999).

networks, or cellular automata cannot be described. The string nature of the data structure not only restricts information flow between plant organs of the same individual but also that between different individuals. This also limits possibilities to include information about the global or local context, which is essential for modelling interactions. One approach to overcome this problem is to switch from pure strings to graphs: graph rewriting system.

There are further problems related to the graphical interpretation of L-systems, see Prusinkiewicz (1986). Commonly only basic primitive objects without any further properties are provided. More complex 3-d shapes like triangulations are not provided, which is critical when the shapes of complex plant organs, e.g., flowers, fruits or seeds have to be generated. Later extensions were made to provide a greater variety in available objects (e.g., Kniemeyer 2008; Pradal *et al.* 2009; Prusinkiewicz *et al.* 2000a).

Apart from the technical drawbacks, L-systems have a more fundamental problem in handling that lies in the nature of the two-stage process of modelling, Fig. 2.5. First of all, the rules of an L-system are applied to a string and in a second step the resulting string has to be translated into 3-d-structure, the latter being the actual object of modelling. The generation of the final

structure is purely theoretical. There is no alternative way to estimate the final structure just by the set of rules than 'running' the L-system and applying the rules until the desired stage is reached and the final structure is generated. The reverse problem is known as 'inference problem'. The inference problem for L-systems is shortly defined as follows: Is it possible to reconstruct the L-system that generated a given structure derived from a sequence of an L-system? See Feliciangeli and Herman (1973) for a detailed definition. This problem turns out to be hard. Therefore, during modelling, the principle of trial and error prevails, which makes it complicated to find an L-system of a given structure like a plant. Small changes can lead to totally different structures. The set of rules containing all the conditions and parameters, therefore, tends to become extensive and confusing. As a consequence, the generation of an L-system can be done (in a good way) only manually, in an iterative process where an experienced modeller is evaluating each step. This turns model creation into a very time-consuming process. Therefore, fast modelling environments are needed to see the outcome of the changes to the rules and parameters.

2.1.5.11. Sensitive Growth Grammars

Kurth (1994a) introduced sensitive growth grammars as an extended variant of L-systems to overcome the problem that 'the pure L-systems formalism cannot cope properly with the representation of the great variety of plant architecture and growth behaviour.' Therefore, some mechanisms to handle global sensitivity were included. Global sensitivity here is defined as a dependence of growth on overall influences, outside the plant. These could be the environment, in the form of temperature or light conditions or the influence of a tropism, e.g., gravitropism.

As a consequence, the formalism of stochastic, sensitive growth grammars was developed and implemented in the software GROGRA (Kurth (1994c), Sec. A.8), a software tool for the 3-d plant modelling and analysing the generated structures.

2.1.5.12. Graph Rewriting Systems

A large number of extensions to L-systems have been developed to provide solutions for partial problems. The transition from string-based L-systems to graph rewriting systems provides an elegant way to overcome some of the main problems mentioned in Sec. 2.1.5.10.

Graph rewriting systems are using graphs as a data structure. In the context of formal languages, 'graph transformation systems' or 'graph grammars' is often used as synonyms for 'graph rewriting systems'. Formally, they operate on attributed graphs which include the use of arbitrary edge types, i.e. considerably extending the notion of successor edges known from 1-d strings. Instead of simple string replacement rules, graph rewriting rules are used as transformation rules. Such rules are constructed similar to the rules commonly used in L-systems. They have a left-hand side L, a searching pattern, that will be searched on the host graph, and a replacement graph on the right-hand side R that will replace all matching sub-graphs:

$$L \Longrightarrow R$$

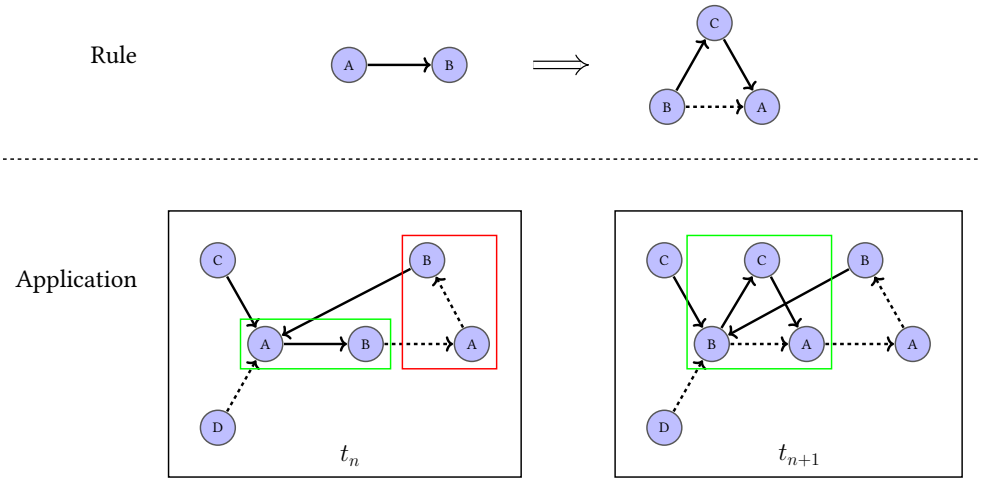


Figure 2.16.: An example of a graph replacement rule application: On the upper part of the figure, a simple rule is defined where two related nodes are replaced by a sub-graph consisting of three nodes. In the application example below, the search pattern can be found only once (green rectangle) in the initial graph at step t_n. The relations on the red rectangle do not fit the search pattern since the relation edge connecting the two nodes A and B is no longer matching. The resulting graph after rule application at step t_{n+1} shows the new situation with the replaced sub-graph highlighted in green.

An example of such a graph replacement rule is given in Fig. 2.16. The rule is searching for an A followed by a B. The two nodes are linked by a so-called 'successor' relation which is visualised by a solid line in this notation. If this pattern is found in a graph, it will be replaced by the right-hand side of the rule. In the graph of the sample application, a relation between A and B can be found twice but only once with the required successor relation (green box) while the box highlighted in red contains a branching relation. Therefore, the rule can be applied

only to the first match. All incoming and outgoing edges of the original sub-graph will be updated in a second step after the nodes are replaced.

Figure 2.17 illustrates a graph replacement rule where no nodes are replaced. Instead, the nodes are reconnected - an operation that is not possible with string based L-systems. Since common L-systems are a subset of graph rewriting systems they can be simulated with graph rewriting systems.

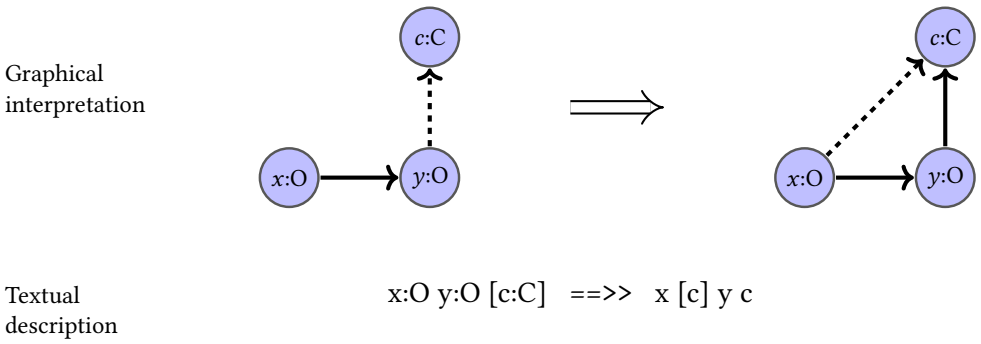


Figure 2.17.: An example of a graph replacement rule illustrating a special case where nodes are not replaced by other nodes but instead differently reconnected with each other. The former branching edge from node y to node c was deleted while a new branching relation from x to c was established. Furthermore, nodes y and c were connected by a successor edge.

In a next step the two-stage process of rule application and graphical interpretation by turtle geometry, see Fig. 2.5, was simplified by replacing the symbols of the graph by 'real' objects of the underlying programming language. These can be simple geometric primitive objects like boxes or spheres but also complex meshes made of any arbitrary triangulation. The underlying graph structure still needs to be traversed as a scene graph to generate the final 3-d structure (to obtain spatial information by applying standard transformations: rotation, reflection, shear, scale, and translation) but the nodes themselves do not need to be additionally interpreted and therefore can be taken over unchanged, Fig. 2.18.

One special type of graph rewriting systems are relational growth grammars (RGG) (Kniemeyer *et al.* 2004; Kurth *et al.* 2004). This formalism incorporates rule-based, procedural and object-oriented concepts. The first programming language that implements the concept of RGG is called extended L-system language (XL) (Kniemeyer 2004, 2008) and was made available for plant modelling as part of the modelling platform **Gro**wth-grammar related **I**nteractive **M**odelling **P**latform (GroIMP) (Kniemeyer *et al.* 2007). XL is implemented as a superset of

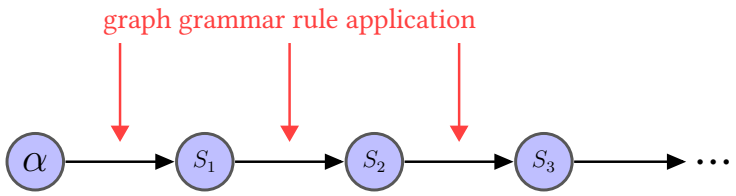


Figure 2.18.: Principle of a relational growth grammar (Kurth *et al.* 2004). The graph replacement rules operate directly on a host graph that contains nodes representing geometric objects. The rules are applied to the initial graph α (axiom) and directly generate the graphs representing the final 3-d structures S_1, S_2, ..., thereby makes rendering the string interpretation step unnecessary.

the well known Java programming language. It is a multi-paradigm language that combines the object-oriented and imperative paradigms from the Java language with the rule-based paradigm from L-systems (Fig. 2.19), additionally it offers the possibility for chemical programming. Doubtlessly, the most important extension for plant modelling are the rule-based mechanisms to cover graph grammars in an elegant way.

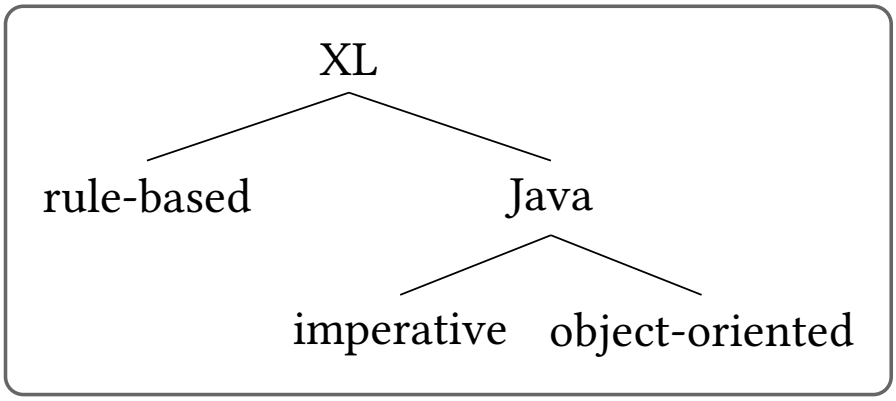


Figure 2.19.: XL as superset of the well known Java programming language extended by an implementation for the rule-based programming paradigm.

The RGG graphs used in the GroIMP software are edge-labelled directed graphs in which the nodes are objects in the sense of object-oriented programming of the underlying programming language XL (Kniemeyer 2004, 2008). To define the relations between two nodes, XL supports several edge types such as successor and branching edges. Beside the default edge

types, additional edge types can be defined by the user. Special refinement edges are used to decomposition relations as used in the multiscale tree graph (MTG) format (Ong *et al.* 2014). Simple strings as used in L-systems are a special case, making L-systems a subset of RGGs.

A 'GroIMP graph' starts always with a node of type *Node* followed by a branch edge to a node of type *RGGRoot*. During model initialisation, the axiom is connected by a successor edge to this *RGGRoot* node. For visualisation in the 3-d view special node types (extensions of type *ShadedNull*) are required to build a so-called scene graph - a general data structure commonly used to represent vector-based graphics in modern computer graphics applications. This scene graph, more precisely a tree structure, which can be reached from *RGGRoot* by using only successor and branching edges, is traversed to draw the geometrical representation of each node. With this approach, it is possible to have a single representation for the complete model information including structure (specified by arbitrary relationships), geometry and internal states. In GroIMP, the whole 'GroIMP graph' with all other node and edge types can be visualised in a separate 2-d view, a hierarchical or flat object inspector or can be exported as text.

2.1.6. Particle Systems

Encouraged and sponsored through film industry, Reeves *et al.* (1983, 1985) had a different motivation when they developed their approach for plant modelling. Botanical correctness took a backseat in their approach since it was their aim to find a fast method to produce relatively realistic looking representations of vegetation such as forests or meadows. Figure 2.20 is such an example of grasses generated by particle systems for Lucasfilm in 1983.

Particle systems use a very large number of very small objects, so-called particles, to simulate certain kinds of 'fuzzy' phenomena. Typically, a particle system is controlled by what is referred to as an emitter which acts as the source of the particles. A stochastic process is used to specify the initial location of the particles in 3-d space what determines where they will move to. A further set of parameters controls the overall behaviour of the model. Typical parameters are: initial position and direction, initial and maximal number of particles, number of new particles generated at each step (emitting speed), lifetime or colour of a particle. Initially, particles are just logical objects to which graphic features must be assigned so that they become visible. During one iteration step, typically, two distinct stages can be distinguished, the parameter update/simulation stage and the rendering stage.

Figure 2.20.: The 'White.sand' image by Alvy Ray Smith, created at Lucasfilm in April, 1983. The grasses are generated by particle systems. Picture taken from: http://www.alvyray.com/Art/WhiteSands.htm

When particle systems are used for plant modelling, plants are discretised into strings of particles where each particle grows depending on local information. Branches are produced recursively and further attached to the trunk.

Arvo and Kirk (1988) included mechanisms for particle-environment interactions. Such external conditions can be for instance light exposure, gravity or wind forces. When for each particle an additional internal state parameter is used to define the role of it within the plant, different plant organ types, e.g., a meristem, leaf or internode can be distinguished.

Today, particle systems are a widely used tool in physics engines and computer graphics, especially in computer animations. They can be created and modified by common 3-d modelling and rendering packages. They are commonly used to simulate chaotic systems like fire, fog, smoke, water or snow. When particles are rendered over their whole lifetime at once, the produced tracks can be used to simulate hair and grass.

Since the rendering process becomes problematic with increasing number of particles newer GPU-based implementations are used to overcome this drawback. They are used, e.g., for real-time rendering of large particle system (Kolb *et al.* 2004) or to visualise steady 3-d flow fields (Kruger *et al.* 2005).

Under the restriction that plants can be discretised into strings of symbols using only local information, an L-system can be translated into a particle system by representing each symbol with a particle. On the other hand, the behaviour of particles can be reproduced with rules in an L-system where the particles are replaced by symbols. In this sense, they are interchangeable.

2.1.7. Space Colonisation

Based on the work by Honda (1971) and Ulam (1962), Runions *et al.* (2007) promoted an alternative modelling approach in which the competition for space plays the dominant role in determining the form of trees and shrubs. While in L-systems branches are built by a recursive (replacing) process, space colonisation uses a procedural method in which new elements are added iteratively to the structure formed in previous steps. The proximity of points and the availability of free space is controlling the position of new points. Runions (2008) used a small number of parameters to control his algorithm. Figure 2.21 illustrates the impact of two of his parameters: the number of attraction points N and the kill distance d_k.

2.1.8. A Fractal Tree Model

During the early 1980s, computer graphics was still a young discipline within computer science. Against this background, Oppenheimer (1986) introduced the creation of natural objects using fractals. Influenced by the work of Mandelbrot (1977, 1982), Oppenheimer developed a recursive algorithm to simulate tree-like structures.

On the one hand, such recursive procedures provide an easy way to simulate the self-similarity often encountered in trees; however, on the other hand, a strict self-similarity is problematic when realistically looking results are the goal. Oppenheimer inserted random parameters to his algorithm in order to reduce symmetry and similarity and increase realism, respectively.

a) $N = 1500, d_k = 2D$ b) $N = 375, d_k = 20D$ c) $N = 375, d_k = 40D$

Figure 2.21.: Example of space colonisation by Runions (2008). Impact of the number of attraction points, N, and the kill distance, d_k, on the tree form. The kill distance is expressed as a multiple of D, the distance between adjacent nodes of the tree skeletons.

In his work, Oppenheimer focused on the fast generation of fractal trees. He used an interactive editing system that allowed to edit the tree parameters, (both geometric and topological) via graphical sliders for near real-time rendering.

2.1.9. Tree Modelling Using Strands

Five hundred years ago, Leonardo da Vinci (1452-1519) observed a nearly universal growth pattern: a particular relationship between the size of a tree's trunk and of its branches. His observation about 'The relative thickness of the branches to the trunk' can be found in the collection of his notes 'The Notebook of Leonardo da Vinci' ('The Codex Arundel') (da Vinci 1508), under remark 393 to 396 (Fig. 2.23).

Remark 394 says that 'All the branches of a tree at every stage of its height when put together are equal in thickness to the trunk [below them].' More precisely, the combined cross-sectional

Figure 2.22.: Fractal tree structure with snow effect by Williams from Oppenheimer (1986). The scene was rendered twice: the first time with a light source on the side, and a second time with a snow source above the tree.

areas of a tree's daughter branches d_1 and d_2 are equal to the cross-sectional area of the mother branch. Thus

$$d^2 = d_1^2 + d_2^2 \tag{2.3}$$

which later turned out to be an excellent estimate for most botanical trees, based on a large number of empirical investigations. That the nearly universal character of this relationship holds was shown by Murray (1927) and his generalisation of Eq. 2.3.

$$d^\Delta = d_1^\Delta + d_2^\Delta \tag{2.4}$$

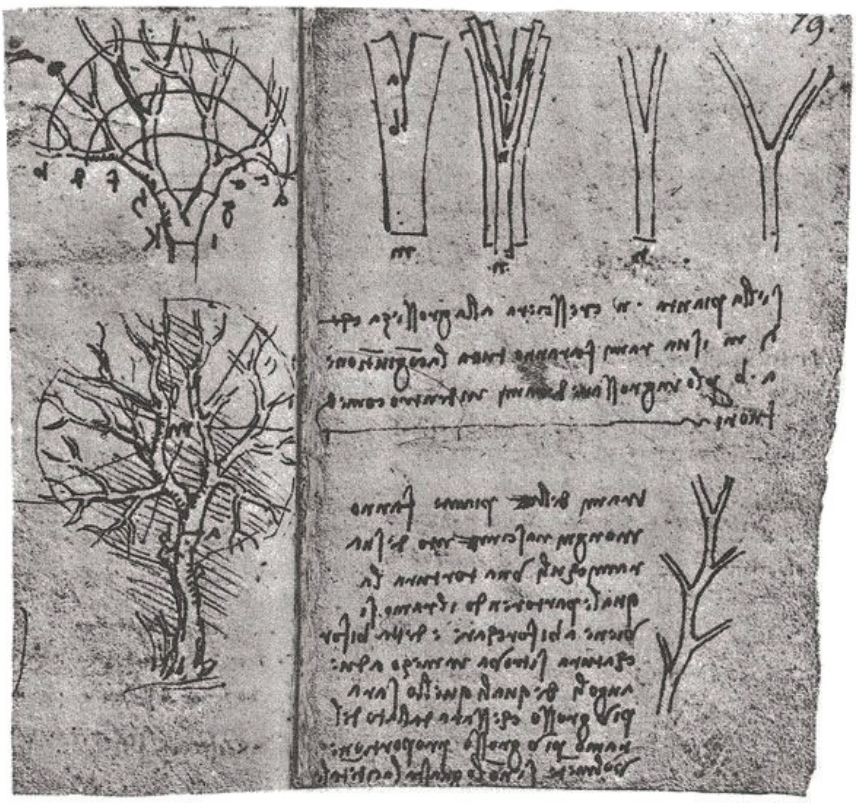


Figure 2.23.: A sketch of a tree from Leonardo da Vinci's Notebooks, PL. XXVII, No. 1, by da Vinci (1508), written in his characteristic left-handed mirror-writing.

Equation 2.4 holds for example, for the bronchial tree of the lungs, where $\Delta = 3$ applies. The value of Δ for arteries is ≈ 2.7 while for river systems $\Delta = 2$. For real trees, the exponent is not always equal to 2 but rather varies between 1.8 and 2.3 depending on the geometry of the tree species (Krieger 2011). But the general equation is still pretty close and holds for almost all trees. To illustrate this, one imagines that each leaf is connected with the root through a strand, Fig. 2.24. The sum of the strands in a branch determines its volume and diameter. In other words, if a tree's branches were folded upward and squeezed together, a cylinder of the same thickness from top to bottom would be obtained.

Murray (1926, 1927) found a relationship between the thickness of two child branches and the branching angle. If a branch bifurcates into two branches with equal diameter, then the branches will form equal angles with the line of direction of the parent branch. However, the smaller the angle to the parent branch, the thicker the new branch, Fig. 2.25.

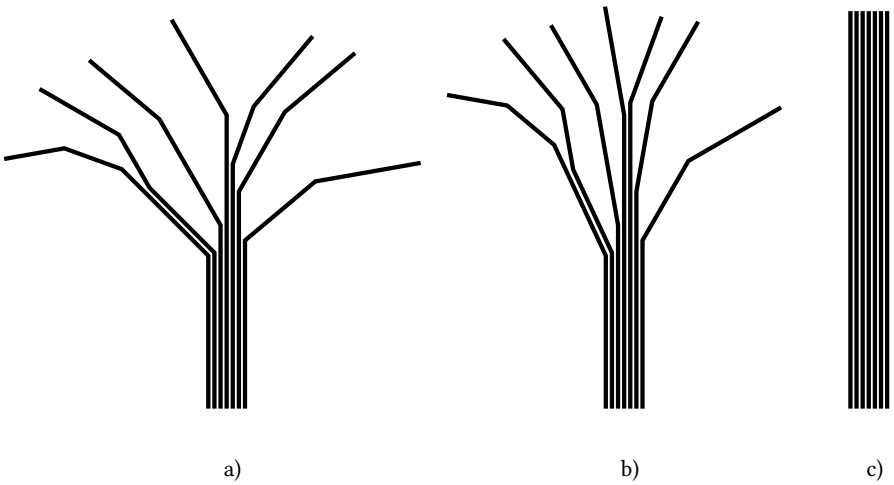


Figure 2.24.: Visualisation of da Vinci's observation that the total thickness of the branches of a tree at a particular height is equal to the thickness of the trunk. An initial bundle of strings starting from the roots is divided at every fork (branching point) until single strings remain. The respective cross-sectional areas of a branch follow the number of strings it is built of. When branches are successively bent upwards (without changing the length of branches) from sub-figure a) to c) a cylinder can be obtained.

While da Vinci had no explanation for the observation he made, several hypotheses has been proposed since then. One common explanation involves hydraulic considerations - how a tree pumps water from its roots to the leaves; the idea being that the tree needs the same total vein diameter from top to bottom to properly irrigate the leaves (Hallé *et al.* 1978). Eloy (2011) investigated wind-induced stresses in trees as a possible explanation while Minamino and Tateno (2014) analysed different biomechanical models to explain the lack of agreement with da Vinci's rule. However, a few modelling methods like the 'pipe model' are based on this observation.

Shinozaki *et al.* (1964) used da Vinci's rule for their pipe model in which each branch was made up of a bundle of pipes (Fig. 2.24). In this model a pipe serves both as a vascular passage for water and solutes, and as a mechanical support for the branching structure. Holton (1994) used an extended version of the pipe model to simulate branching structures, that are influenced by gravitropic and phototropic effects.

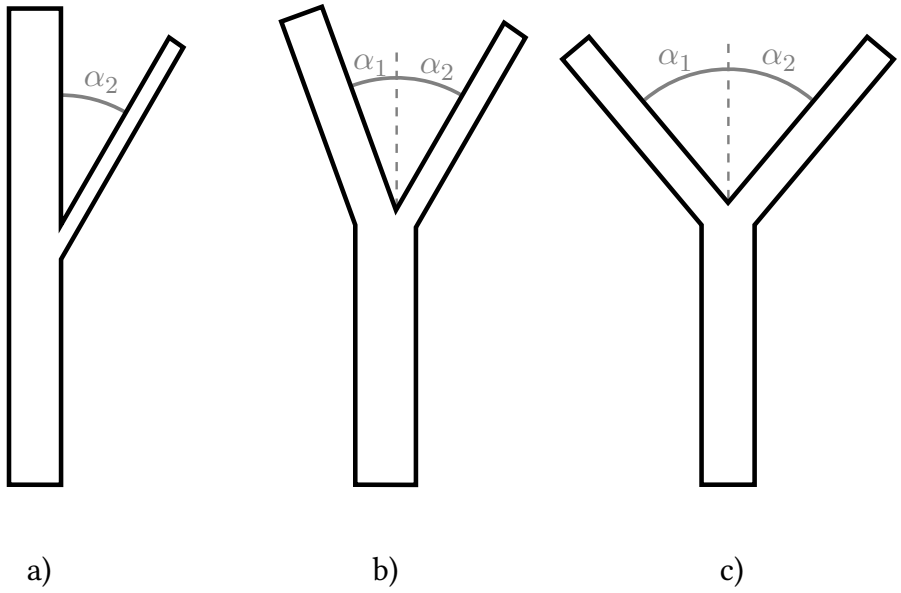


Figure 2.25.: The relation between branching angle and branch diameter. According to Murray (1927), a direct relation can be observed between the thickness of a branch and the branching angle. With increasing angle between the two branches and equalisation of angle α_1 and α_2 from sub-figure a) to c) the thicknesses of the two branches become equal.

Further relations in trees were investigated, e.g., by Huber (1928). The latter empirically affirmed that the total surface of the leaves of a branch proportionally relates to both the volume within the outlines of the branch and the diameter of the branch, see Eq. 2.5 and 2.6.

$$\frac{branch\ leaf\ surface}{branch\ volume} = constant \tag{2.5}$$

$$\frac{branch\ leaf\ surface}{branch\ cross\ section} = constant \tag{2.6}$$

Equation 2.7 states that the wind resistance of a tree is approximately proportional to the branch and leaf surface and thus to the third power of its height, while the resistance of a branch is proportional to the square of its diameter.

$$\frac{tree\ height^3}{trunk\ area^2} = constant \tag{2.7}$$

For each branch it follows:

$$\frac{diameter\ of\ space\ occupied\ by\ a\ branch^3}{branch\ diameter^2} = constant \tag{2.8}$$

On the basis of these architectural relations, it was possible to develop a further, hydraulic ratio. For example, the leaf to sapwood area ratio (A_L/A_S) is normally defined as the total projected area of leaves divided by the cross sectional area of sapwood supplying water to those leaves. The inverse of this ratio, i.e. the sapwood to leaf ratio (A_S/A_L), is also used in the literature, and often referred to as the Huber value (Huber 1928). When the hydraulic conductivity of sapwood K_S is also taken into account, the leaf specific conductivitics (LSC), which is calculated as

$$LSC = \frac{K_S}{A_L/A_S} \tag{2.9}$$

can be obtained, Zimmermann (1978).

2.1.10. Iterated Function Systems

By repeatedly applying a function to an initial set of points interesting results can be obtained. In mathematics, such a procedure is called iterated function systems (IFS). They were conceived in their present form by Hutchinson (1981) and popularised by the book 'Fractals Everywhere' by Barnsley (1988). IFSs can be used to generate self-similar fractals of any number of dimensions, but are commonly computed and drawn either in 2-d or 3-d. By the repeated application of a function (hence 'function system') to an initial state the result of it is being transformed at each step. The obtained point set, the so-called attractor set or short attractor forms a fractal, the actual result.

Normally, the function is required to be linear, or more generally an affine transformation and hence represented by a matrix. However, theoretically, this restriction is not necessary. Other functions like non-linear functions including projective transformations and Möbius transformations can be used as well.

For illustration purposes, an affine transformation $f : \mathbb{R}^2 \to \mathbb{R}^2$ is given, describing a shift with direction $b = (b_1, b_2)$ of a linear transformation $A = \left(\begin{smallmatrix} \alpha_{11} & \alpha_{12} \\ \alpha_{21} & \alpha_{22} \end{smallmatrix}\right)$:

$$f(x) = Ax + b = \begin{pmatrix} \alpha_{11} & \alpha_{12} \\ \alpha_{21} & \alpha_{22} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} \tag{2.10}$$

By successive application of the transformation above to a single point $x \in \mathbb{R}$ the following sequence is built:

	f	$f \circ f$	$\cdots$	$f \circ \cdots \circ f$
x	$Ax + b$	$A(Ax + b) + b$	$\cdots$	$A^n x + (A^{n-1} + \cdots + A)b + b$

Taking the point $x = (1.5, 1.5)$ and an affine transformation given by $A = {}^1/_5\left(\begin{smallmatrix} 1 & -5 \\ 4 & 1.5 \end{smallmatrix}\right)$ and $b = (-0.1, 0.13)$ for example, the graph of Fig. 2.26 shows the behaviour of the obtained sequence which was numerically generated using the following MATLAB® code:

```
x=[1.5; 1.5]
A=1/5*[1,-5; 4,1.5]
b=[-0.1; 0.13]
X = x

for i=1:55
    x = A*x+b
    X=[X x]
end
plot(X(1,:),X(2,:), 'b-')
```

A more complex example is the Sierpiński triangle (Fig. 2.27), Sierpiński (1915). It can be constructed by several methods including 'removing triangles', 'shrinking and duplication', 'cellular automata' (Sec. 2.1.3), 'Pascal's triangle' and the 'chaos game'. In mathematics, the term chaos game refers to a method of creating fractals using a polygon. In an iterative process, starting with an initial random point inside the polygon, new points are created at a fraction of the distance between the previous point and one of the vertices of the polygon. At the end of this process, depending on the used fraction, a fractal shape can be produced.

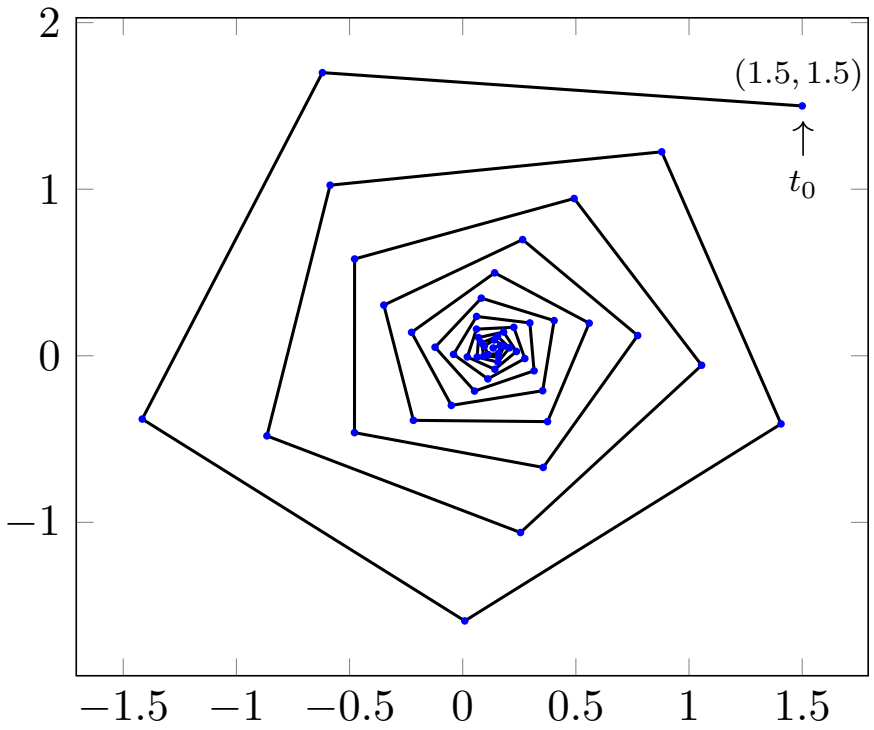


Figure 2.26.: Result of the simple IFS generated by the affine transformation defined above after 55 iterations. Starting at time t_0 at point $x = (1.5, 1.5)$, each consecutive point computed in the iteration gets closer to the attractor or fixed point of the system. The system finally converges to $(0.135, 0.047)$.

In the case of the Sierpiński triangle the vertices of the polygon form an equilateral triangle. Using the affine transformation of Tab. 2.5, the Sierpiński triangle given in Fig. 2.27 was created. Figure 2.28 illustrates this process for the first six iterations.

Beside the standard example of the Sierpiński triangle, the Barnsley fern (Fig. 2.29) is probability the second best-known example of an IFS, Barnsley and Demko (1985) and Barnsley (1988). Later in his work, Barnsley used IFSs and fractals to model a diverse range of phenomena in science and technology, but most specifically plant structures.

For modern FSPMs, IFSs are of no importance. The reasons for that are numerous and comprehensible. One important drawback is that in a chaos game points are created all over the attractor in a random order. No relation between neighbouring points/objects can be identified, which, however, is essential for nearly all physiological processes. No information about

Table 2.5.: Definition of the affine transformation used to generate the Sierpiński triangle given in Fig. 2.27. For each function w_i the probability of application p is set equal to one third.

w_i	a	b	c	d	e	f	p
1	0.5	0	0	0.5	0	0	0.333
2	0.5	0	0	0.5	0.25	0.5	0.333
3	0.5	0	0	0.5	0.5	0	0.333

the structure is included. The code of an IFS might be compact but due to the nature of an affine transformation the result is hardly predictable. Like in a chaotic system, slight changes applied to the transformations can cause huge changes in the results. Furthermore, it is very complicated to find the right set of functions that can produce an attractor to a given 3-d structure like a plant. One problem here is that plants are not that self-similar and fractal as the common results of an IFS.

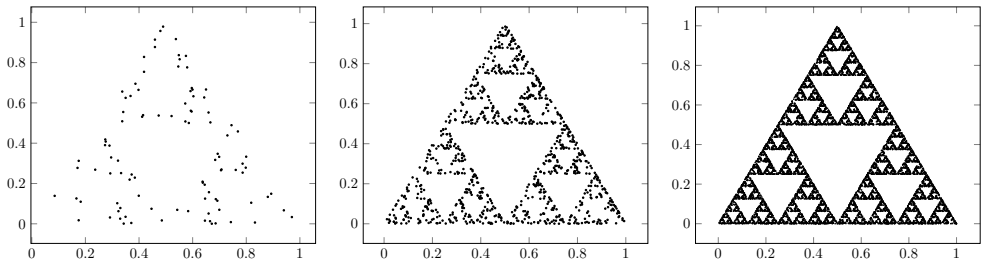


Figure 2.27.: The Sierpiński triangle at different iterations from $n = 100$ to $n = 10000$.

The problem of finding an IFS whose attractor is 'close', relative to the Hausdorff distance (distance between two subsets of a metric space), to a given set is solved approximately by the Collage Theorem (Barnsley 1988). According to Barnsley, the theorem implies that one must endeavour to find a set of transformations - contraction mappings on a suitable set within which the given set lies - such that the union, or collage, of the images of the given set under transformation is near to the given set. This principle is typically used in fractal compression.

In order to incorporate generation of new organs into IFS, two approaches can be distinguished. The first one, called recurrent iterated function systems, was introduced by Barnsley *et al.* (1989), which, later, found one application in image compression (Yun *et al.* 2008). The second approach is related to the notion of hierarchical iterated function systems (Reuter 1987).

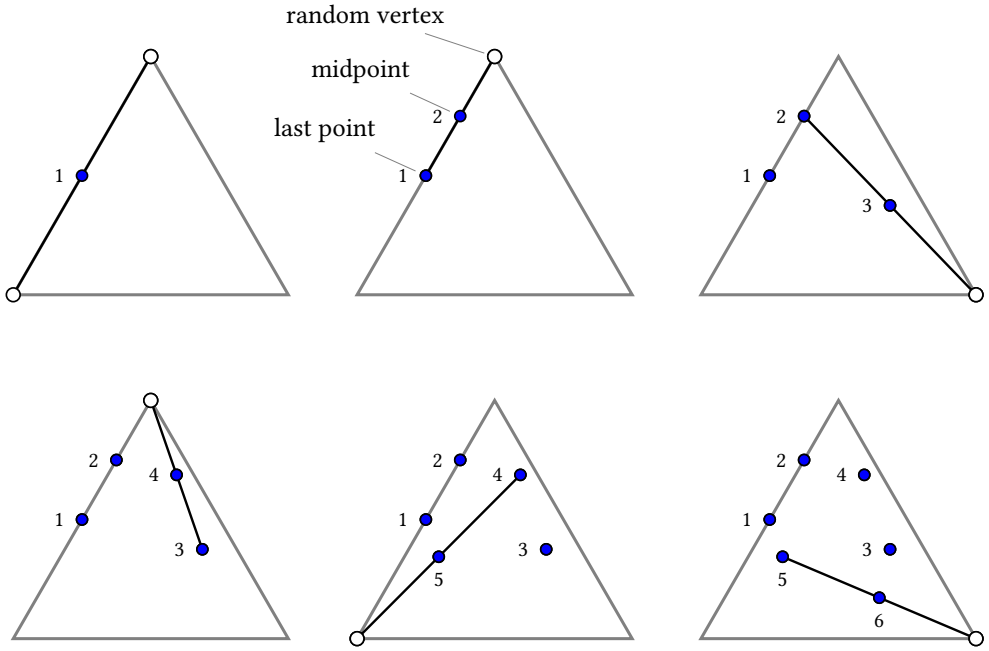


Figure 2.28.: First six steps of an iterated function system to produce the Sierpiński triangle. Beginning by selecting one point of the vertices of an equilateral triangle and picking one of the other two vertices at random a new point is plotted midway between the first two. Then the new point is used as 'last point', again one of the vertices is picked at random and a new point plotted. The Sierpiński triangle will be the result (attractor) of repeated application of the same operation.

2.1.11. Voxel Space Automata

Greene (1989) discretised the 3-d space into identical cubes (volume elements or voxels) and called it voxel space. In this voxel space, objects are represented as a collection of voxel records. An initial geometric object is used to generate the final structure according to rules based on simple relationships like intersection, proximity, and occlusion. One advantage of this approach is that illumination can be efficiently estimated for each plant organ - which was, together with other methods available at that time a computationally intensive task. Discretisation into a voxel space also simplifies the modelling of environmental aspects. Intersection testing and measuring the relative proximity of objects may be determined by testing each voxel that it intersects to see if it is already occupied.

Figure 2.29.: The Barnsley fern generated with a chaos game after $n = 150k$ iterations.

2.1.12. Solid Modelling

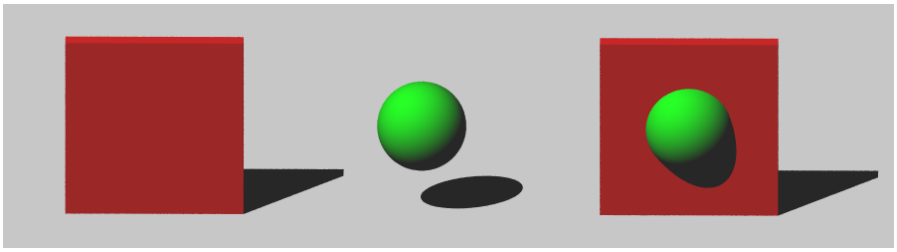
In contrast to the methods described before, where the shape is described implicitly by an algorithm, in solid modelling three-dimensional solids are defined by the combination of (primitive) three-dimensional solids. A distinction is made between direct representation schemes that describe the volume itself and indirect schemes in which the description of edges and surfaces occurs. The most frequently used direct representation scheme is called constructive solid geometry (CSG), while boundary representation is the most common way of surface description.

The main application of these techniques lies in computer-aided design (CAD) with a focus on effective representation and manipulation of three-dimensional geometry in a fashion that is consistent with the physical behaviour.

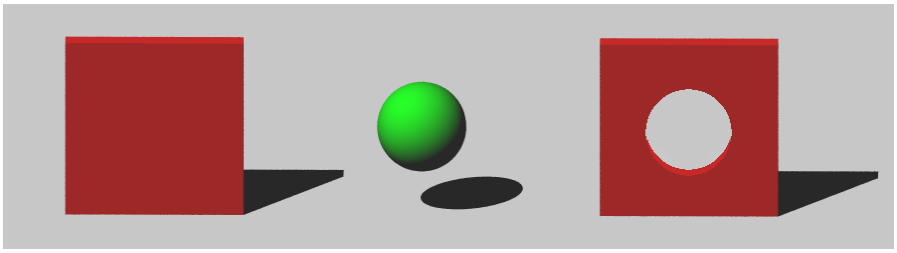
2.1.12.1. Constructive Solid Geometry

Constructive solid geometry (CSG) represents direct representation schemata that use geometric primitives like boxes, spheres, cylinders and combine them with boolean operations on sets: intersection, difference and union. CSG is part of many CAD systems since it allows to describe objects in a natural way. Using some primitives and a few operations (Fig. 2.30) can

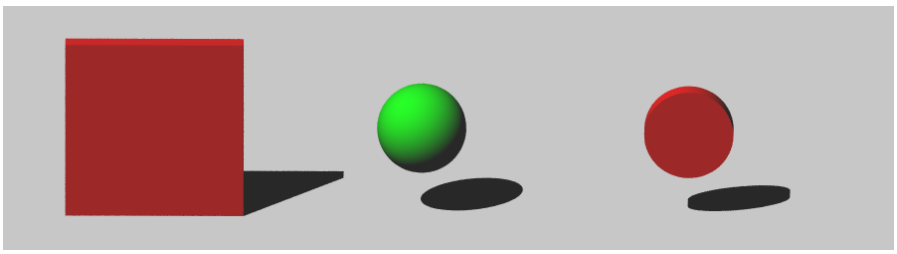
produce quite complex objects. Such a CSG shape can be described by a sequence of operations applied to primitives and visualised as (binary) parse/syntax tree (Fig. 2.31). A closer look at the syntax tree reveals that the construction is not unique which can be problematic for some applications.



a) Union '⋃': Merger of two objects into one



b) Difference '⟍': Subtraction of one object from another



c) Intersection '⋂': Portion common to both objects

Figure 2.30.: Allowed CSG-operations, which are typically boolean operations on sets: union, difference and intersection.

It can be easily shown that the obtained structure is 'solid' or water-tight if all of the primitive shapes are water-tight, which is of special interest for some manufacturing or engineering applications. Furthermore, CSG shapes make it easy to decide if an arbitrary point is inside or outside a given shape. This is a desirable quality, e.g., for ray tracing; however, a renderer can not process CSG shapes directly, they first have to be converted to their boundary representation which can be a challenging task.

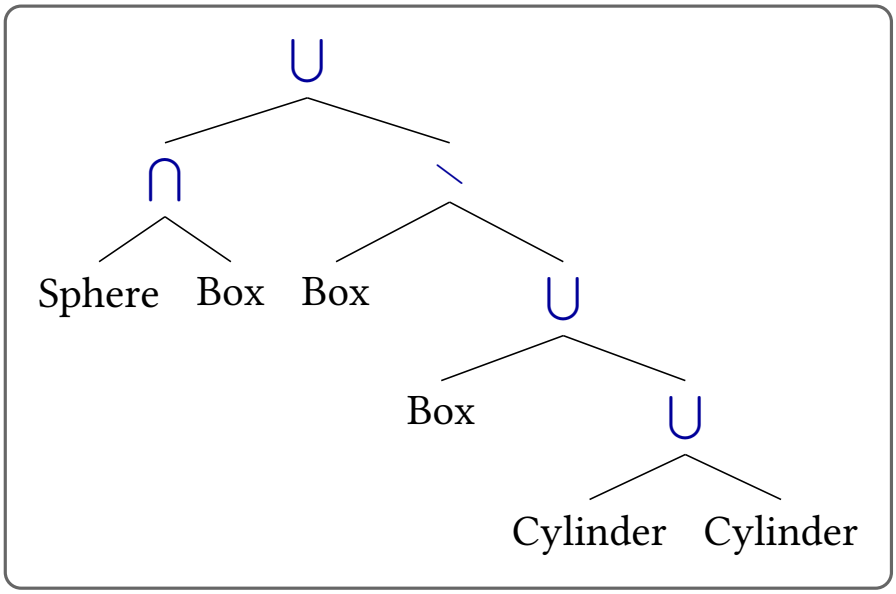


Figure 2.31.: Visualisation of a syntax tree of a CSG shape. The final structure is the result of the application of boolean set operations (union '⋃', difference '⧵' and intersection '⋂') to basic geometric primitive objects.

Gervautz and Traxler (1996) extended the former description by allowing cyclic CSG graphs, and by permitting arbitrarily refined descriptions of objects.

2.1.12.2. Boundary Representation

Boundary representation designates the most common indirect method for representing shapes. It is using limits to represent a solid as a collection of connected surface elements, the boundary between solid and non-solid. Polygon meshes are the usual way to describe such shapes.

Objects represented by boundary representation are usually composed of so-called free-form surfaces which can be controlled by reference points. One widespread approach of free-form surfaces are non-uniform rational B-Spline (NURBS). For reasons of efficiency NURBSs are usually converted into polygon or triangle meshes (surface triangulation) before being visualised.

The technique of boundary representation has its advantages when it comes to rendering but the large memory use and the problematic test if the formed object is solid are significant drawbacks. By using only Euler-operations which keep the Euler-characteristic when applied, the solidity can be partially guaranteed.

Today, NURBS and polygon meshes are frequently used in FSPMs to model leaf structures and fruits.

2.1.13. Image-Based Modelling

Kang and Quan (2009) advocate a system based on sets of 2-d images to reconstruct the 3-d structure of plants (Quan *et al.* 2006) and trees (Tan *et al.* 2007). This method of digitising has the advantage that the resulting model inherits the realistic shape and complexity of a real object. While for plants leaves are modelled directly from images, for trees, leaves can be approximated due to the small size and large number. In a similar way, hidden or small branches can be approximated by using fitted L-systems (Shlyakhter *et al.* 2001) or simple rules in voxel space (Sakaguchi and Ohya 1999).

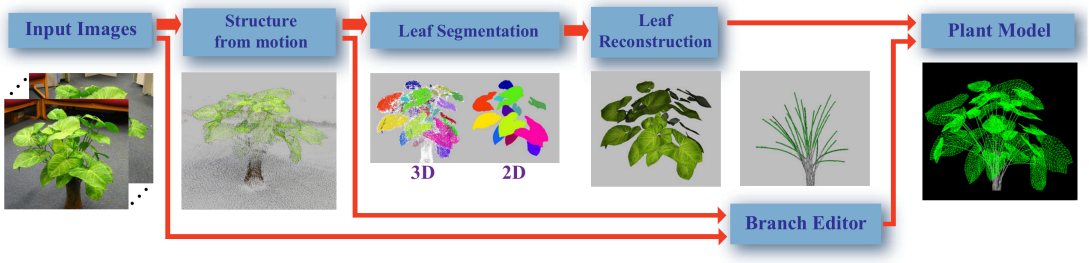


Figure 2.32.: Overview of image-based plant modelling approach. Redrawn from Quan *et al.* (2006) (modified).

The interactive reconstruction process (Fig. 2.32) of the image-based approach by Quan *et al.* (2006) starts with a sequence of images taken from different angles. These images are then used to calculate a point cloud of the structure. In the next step plant organs, especially leaves, are identified using segmentation algorithms. Branches and twigs are also obtained by segmentation of the point cloud but can be interactively manipulated in an editor. The final 3-d plant structure is a combination of results of the previous steps.

2.2. Classification of Modelling Approaches

Based on individual backgrounds, preferences, and interests of the scientists and their teams several plant modelling 'schools' can be identified. The technical developments and new fields of research had also a huge impact on the development of new methods during the past decades. The classification of plant modelling approaches can be done according to different criteria depending on, e.g., the focus on the method used to obtain or describe the structure (formalism / formal language used), or the research background.

2.2.1. The Triangle of Plant Models

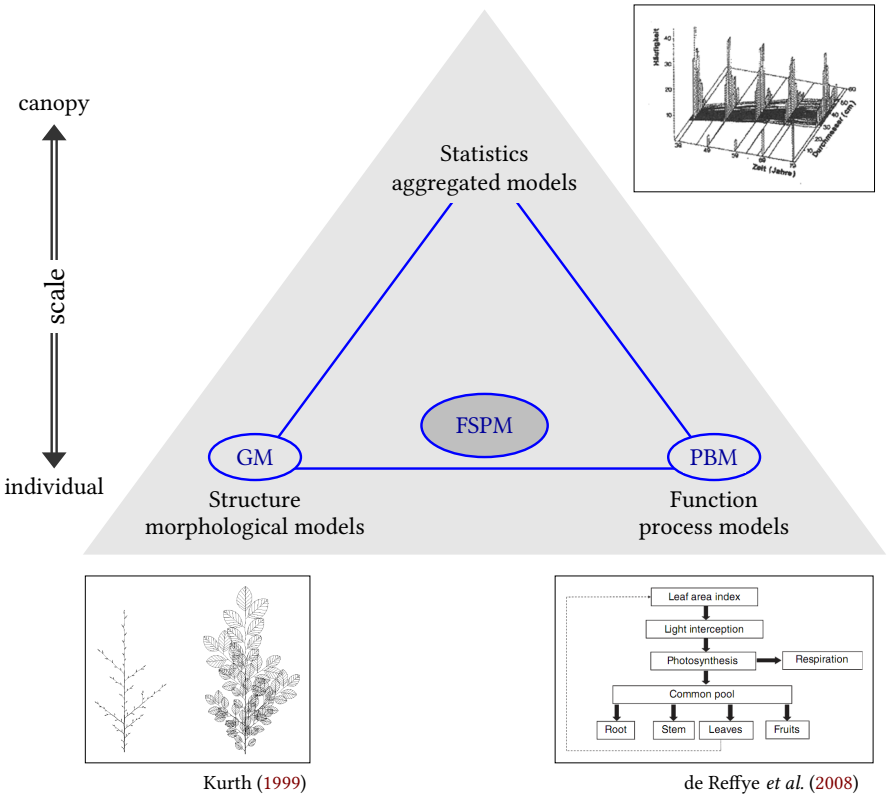
According to the triangle of plant models, a systematisation of modelling approaches by Kurth (1994b), plant models can be classified into three main classes: aggregated models, morphological models, and process models. FSPM combine morphology with physiology and therefore form their own class located between structural and functional models at the hierarchical level of the plant individual.

Another common classification distinguishes modelling approaches depending on the methods, aims, and processes employed. The three main classes are referred to as geometrical models (GM) (Sec. 2.2.1.1), process-based models (PBM) (Sec. 2.2.1.2), and functional-structural plant models (FSPM) (Sec. 2.2.1.3).

Since GMs can be referred to as morphological models and PBMs are process models, both classifications can be combined to an extended triangle of plant models (Fig. 2.33).

2.2.1.1. Geometrical Modelling

Geometrical models (GM), also known as morphological or structural models, are only concerned with the pure 3-d representation of plants and their development. Plant development is often defined by L-systems (cf. Sec. 2.1.5.2) or turtle geometry in the static case. The main focus lies on the visual output of the model. While the overall plant size may be increased by rule application, the organ sizes are fixed and typically based on empirical observations on real plants. The same applies to morphological data like branching angles or phyllotaxy.



Kurth (1999) de Reffye *et al.* (2008)

Figure 2.33.: Extended model triangle of plant models based on Kurth (1994b). FSPMs can be located at the level of the plant individual, halfway between structural models and process oriented models.

Specialised tools, like the Xfrog (Deussen and Strothotte (2000), Sec. A.17) software, have been developed, that are able to produce nice 3-d mockups widely used in computer animated films, landscape planning, and games.

2.2.1.2. Process-Based Modelling

Process-based models (PBM) or process-based crop models are models based on primary production of biomass and centred around a generic implementation of photosynthesis. The majority of process-based crop models are designed to predict yield from the simulation of biomass production on a per-square-meter basis, under field or greenhouse conditions. They simulate plant functioning according to endogenous plant properties and environmental conditions without any 3-d representation or spatial information (Fig. 2.34). Plants are considered only at

a simplistic level of organ compartments - typically leaf, stem, fruit, and root compartment - which renders fitting of models easy but poses at the same time a potential source for errors, since a very large number of plant architectures could correspond to a plant fitted with a PBM (de Reffye *et al.* 2008). The calculation of photosynthesis is a typical case for such simplified models, which are purely concentrating on function, at the expense of structure. E.g., the 'big leaf' model (Thornley *et al.* 1992) represents the leaves of the whole canopy as one single (big) leaf.

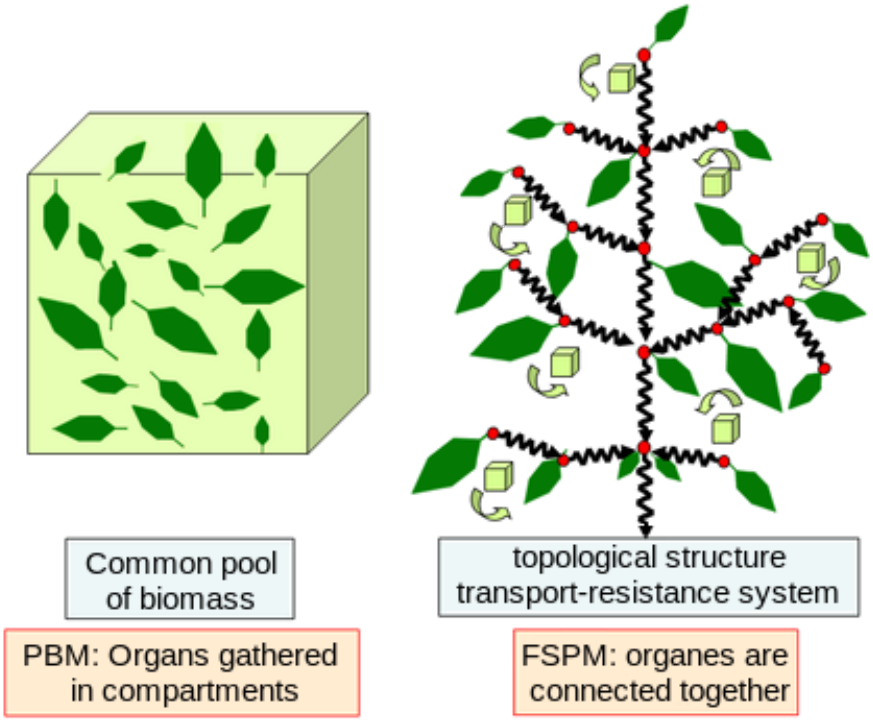


Figure 2.34.: Organ source and sink principle in process-based models (PBM) and functional-structural plant models (FSPM). The organ compartments are usually limited to organ types in PBM, competing for a common biomass pool, while in FSPM, each organ is individualised. The assimilates produced by local sources are transported in the direction of sinks according to their sink strength. (Drawing: de Reffye, CIRAD, http://greenlab.cirad.fr/GLUVED/html/P1_Prelim/Model/Model_FSPM_001.html)

Most crop growth models have been based on a selection of general, primary processes describing (process-based) the mechanisms of primary production. A typical flowchart of a PBM for plant modelling is given in Fig. 2.35. They have been developed to enhance understanding

of the basic processes of crop growth and development. Generally, in these models factors determining potential, attainable and actual crop growth are distinguished, allowing the use of these models for a variety of crop species, given the availability of a standard set of crop parameters.

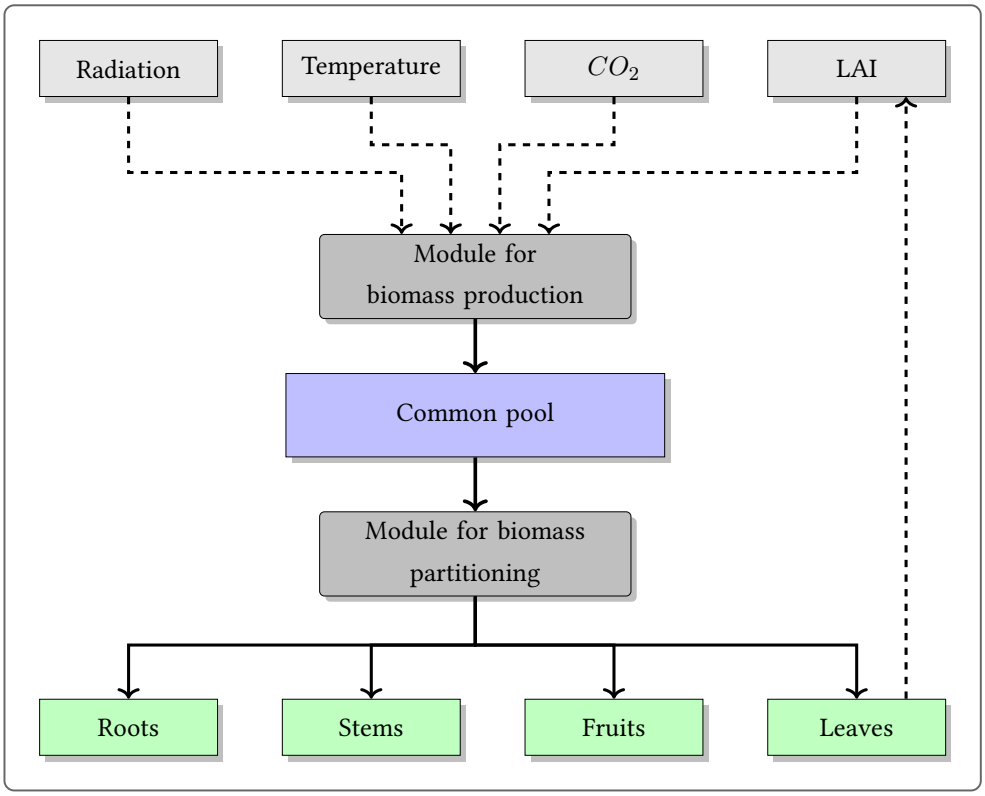


Figure 2.35.: Simplified workflow of a PBM for plant modelling. The 'Module for biomass production' covers processes like light interception, photosynthesis and respiration.

A number of the drawbacks of PBMs have their origin in the total neglect or oversimplification of 3-d plant architecture and its plasticity, such as the interactions between growth and development. E.g., an accurate calculation of light interception is a very important input factor for realistic plant modelling. Further simplifications are, e.g., the missing direct interaction between sources and sinks, the common pool of biomass, and the computation of both biomass production (dry matter) and biomass partitioning by single equations/models.

The leading school of PBM was founded in Wageningen UR and dates back to 1958, when de Wit implemented first crop models for agriculture, modelling linkage between transpiration and growth (de Wit 1958). A first model for simulating photosynthesis in plant communities

was introduced by Duncan *et al.* (1967). Jones *et al.* (1974) introduced a first nitrogen balance for a cotton growth model.

Crop modelling is based on the assumption that the only ingredients necessary for yield prediction are organ number, biomass production and its partitioning among organs. Some of the successors of the school of 'de Wit' at Wageningen UR have acknowledged the importance of 3-d structure and FSPM for PBM, which is reflected in an increasing number of publications (Fourcaud *et al.* 2008; Vos *et al.* 2010, 2007).

2.2.1.3. Functional-Structural Plant Modelling

'FSPM, refers to a paradigm for the description of a plant by creating a (usually object-oriented) computer model of its structure and selected physiological and physical processes, at different hierarchical levels: organ, plant individual, canopy (a stand of plants), and in which the processes are modulated by the local environment. Structure comprises the explicit topology (connection between organs) and geometry (orientation, inclination, and shape) of the organs and the plant. At the individual level, this is also referred to as plant architecture...'

Buck-Sorlin (2013)

FSPMs are integrated models as they combine plant function with structure. They are typically organised in a modular way, where plant structure is described regarding basic units, e.g., organs, compartments, growth units, or metamers. The models are often object-oriented, i.e. each plant organ is mapped/linked to a specific organ class in the model. Processes like transport take place between instances of these organ classes that are distributed in 3-d, while other processes like transpiration or photosynthesis take place only locally, e.g., in the leaf organs. A set of rules is used to describe the morphological development, with submodels for metabolic processes driving plant growth. Usually, the actual 3-d shape is directly linked to this organ class for rapid visualisation. Modern modelling languages like L-Py (Sec. A.13.1) or XL (Sec. A.9.1) support such mechanisms directly.

FSPMs are highly interdisciplinary and involve a wide range of disciplines of the natural sciences, including botany, plant physiology, plant anatomy, plant morphology, mathematics, computer science, cellular biology, ecophysiology, ecology and agronomy.

2.2.2. Research Background

The origins and influences of plant modelling are quite diverse. Researchers with various scientific backgrounds and objectives developed their approaches and provided input to the topic. Figure 2.36 illustrates the main sources and influencing factors for plant modelling classified by the predominating research backgrounds.

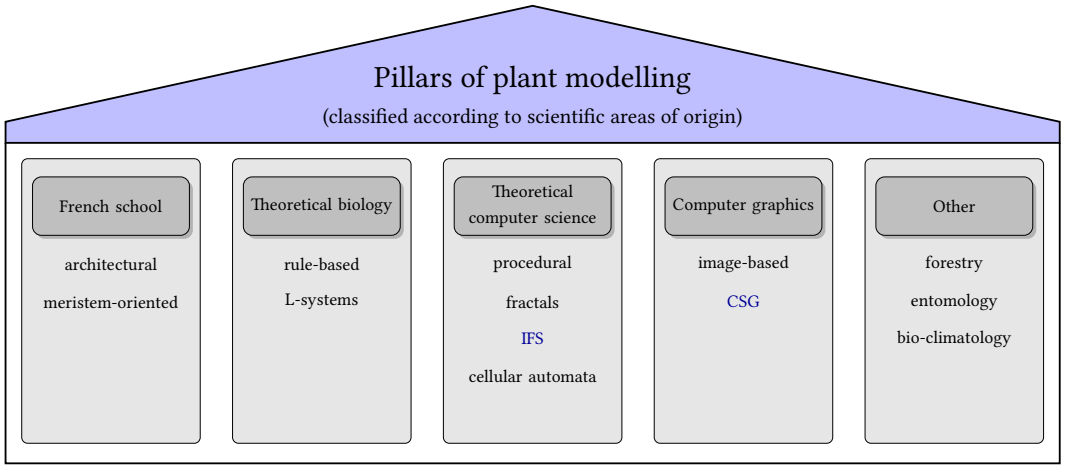


Figure 2.36.: Pillars of plant modelling classified by research background (non exhaustive).

2.2.2.1. French School

The so-called French school is mainly based on the work by Francis Hallé, a botanist who was professor of Botany at Montpellier university. Hallé investigated the diversity of crown architectures of tropical trees in the rainforest (Hallé 1971; Hallé and Oldeman 1970). In Hallé *et al.* (1978), he identified 23 different architectural patterns (Fig. 2.37). With his work, Hallé established a new scientific discipline of architectural studies in trees to provide tools for 'taxonomists, valuable for diagnosis, and sometimes more successful than floral characters for species identification in the tropical trees' (Hallé 1974).

In the late 1980s, de Reffye established the AMAP (Sec. A.2) workgroup at the Centre de coopération internationale en recherche agronomique pour le développement (CIRAD), Montpellier. His approach uses the meristem, a tissue that contains stem cells that produce new tissue within a bud, as only growth region within his model. Based on this and other botanical

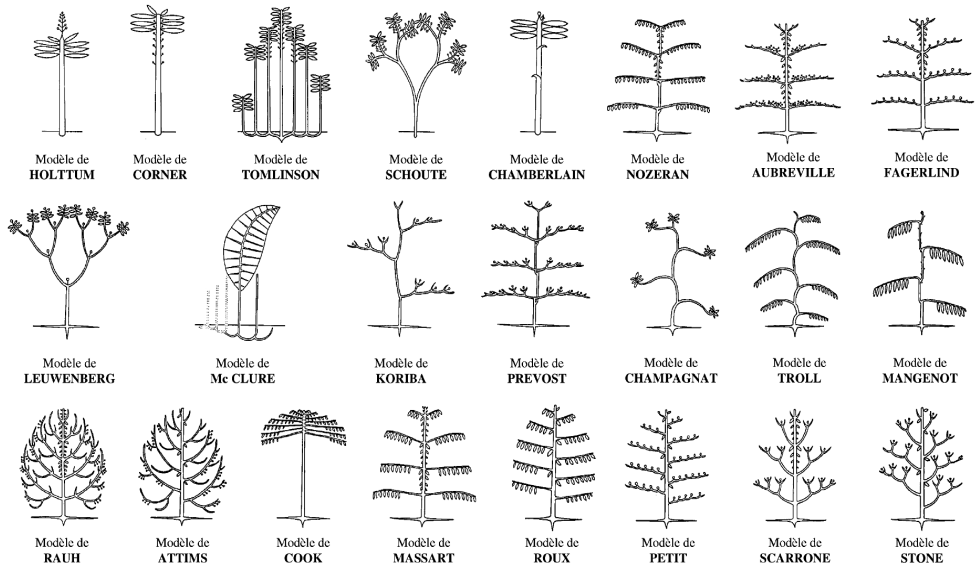


Figure 2.37.: 23 architectural tree patterns after Hallé *et al.* (1978) (modified).

laws, de Reffye *et al.* (1988) introduced a mathematical model which explains plant growth and architecture.

Shebell (1986) showed that principally all 23 tree architectures of Hallé could be simulated with L-systems. Prusinkiewicz and Remphrey (2000) propose a symbolic notation inspired by L-systems and a graphical representation based on Petri nets to formally describe architectural tree models introduced by Hallé and Oldeman.

2.2.2.2. Theoretical Computer Science

Most of the formalism used in FSPMs are based on mathematical concepts and formal languages originally developed by mathematicians and computer scientists. The probably best-known formalism used for plant modelling are Lindenmayer-systems (or short L-systems). It is a formal language, developed 1968 by Aristid Lindenmayer at Utrecht University. Cellular automata and iterated function systems are further formalisms that have their origins in theoretical computer science.

2.2.2.3. Theoretical Biology

During his time as head of the Theoretical Biology Group at the Utrecht University, Lindenmayer investigated cell division in general. He studied growth patterns of various types of algae, yeast and filamentous fungi. Today, L-systems are widely used in many disciplines but mainly in plant modelling. A comprehensive overview is given in the de-facto standard book of plant modelling: 'The Algorithmic Beauty of Plants' by Prusinkiewicz and Lindenmayer (1990).

2.2.2.4. Computer Graphics

Since their beginnings in the early 1970s, the emphasis in the creation of computer-generated plants and trees in computer graphics was always put on the rapid and comfortable generation of plants, or more precisely on the production of images, than on biological accuracy. While first approaches were simple 2-d models, it did not take long before the first 3-d models were published, e.g., 'virtual plants' by Room *et al.* (1996). CAD programs used CSG techniques and boundary representation to generate plant-like structures. Later, specialised software, like Xfrog (Deussen and Strothotte (2000) and Greenworks (2017), Sec. A.17) was developed as plugins for leading 3-d computer graphics software.

Today's driving force in this area are the computer gaming industry and animation studios working for the movie industry. Both have a huge demand for computer-generated plants and virtual reality and are always interested in the latest developments.

2.2.2.5. Others

Researchers from other scientific areas also designed plant models based on different concepts and ideas.

Forest ecologists and foresters used yield tables for more than 100 years to estimate growth for several forest tree species. Based on this huge data base, several forest simulations have been

developed. Based on single-tree models complex forest simulators are developed. BWINPro[1] (Nagel *et al.* 2006) and SILVA[2] (Pretzsch 2001) are two examples of current forest simulators.

Tree physicists investigated several aspects of tree morphology concerning mechanical properties like effects of wind exposure. In hydraulic models water and nutrient transport was modelled to investigate, e.g., the effects of water-stress (Fishman and Génard 1998).

Remote sensing is another area that had influence on plant modelling. Applications worth mentioning here are landscape planning through modelling and visualisation (Disney *et al.* 2006). Iovan *et al.* (2014) used remote sensing data to analyse and reconstruct urban vegetation using architectural tree models combined with model inputs estimated from aerial image analysis. Côté *et al.* (2009) used terrestrial light detection and ranging (LiDAR) data to reconstruct 3-d tree architectures, a method that promises to be robust and relatively insensitive to wind- and occlusion-induced artefacts.

2.2.3. Structure- and Space-Oriented Models

Other criteria for the classification of plant modelling approaches, apart from the scientific background and motivation of its protagonists, can be applied: Prusinkiewicz (1993) divided developmental biology models into structure- and space-oriented models (Fig. 2.38). Structure-oriented models use components, e.g., plant organs, to constitute the structure and describe the development. Structure-oriented models are, therefore, mainly based on endogenous mechanisms to control the structure by internal growth and elongation. In contrast to structure-oriented models, space-oriented models emphasise exogenous control. Space-oriented models usually allow growth only on the system boundaries while they capture influences of the entire environment of a growing plant.

2.2.4. Procedural and Rule-Based Modelling

The difference between procedural and rule-based modelling is more fundamental (Fig. 2.39). Procedural models are parameterised algorithms that are usually designed for the simulation

[1] https://www.nw-fva.de/index.php?id=475
[2] http://www.wwk.forst.tu-muenchen.de/research/methods/modelling/silva

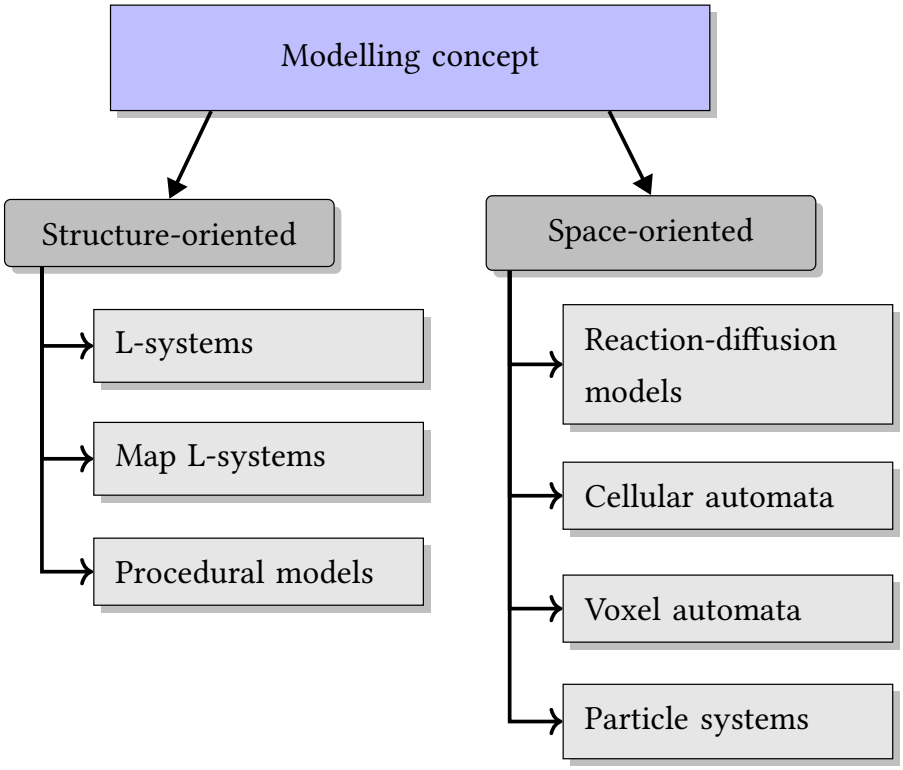


Figure 2.38.: Classification of modelling approaches into space- and structure-oriented models according to Prusinkiewicz (1993) and Prusinkiewicz *et al.* (1994) (non exhaustive).

of a plant type or a single species, respectively. They can be regarded as classical simulation programs, see Fig. 2.40. Rule-based modelling approaches, on the other hand, are using a formal system to define the model in terms of rules that are applied to an initial state until a complex final state is reached. These sets of rules can be seen as program that is being executed by the formal system.

While procedural modelling is usually more intuitive, the rule-based approach is more flexible and powerful. The combination of both approaches lead to the so-called rule-based object production, which was implemented in the Xfrog modelling system (Deussen and Lintermann 1997; Lintermann and Deussen 1999).

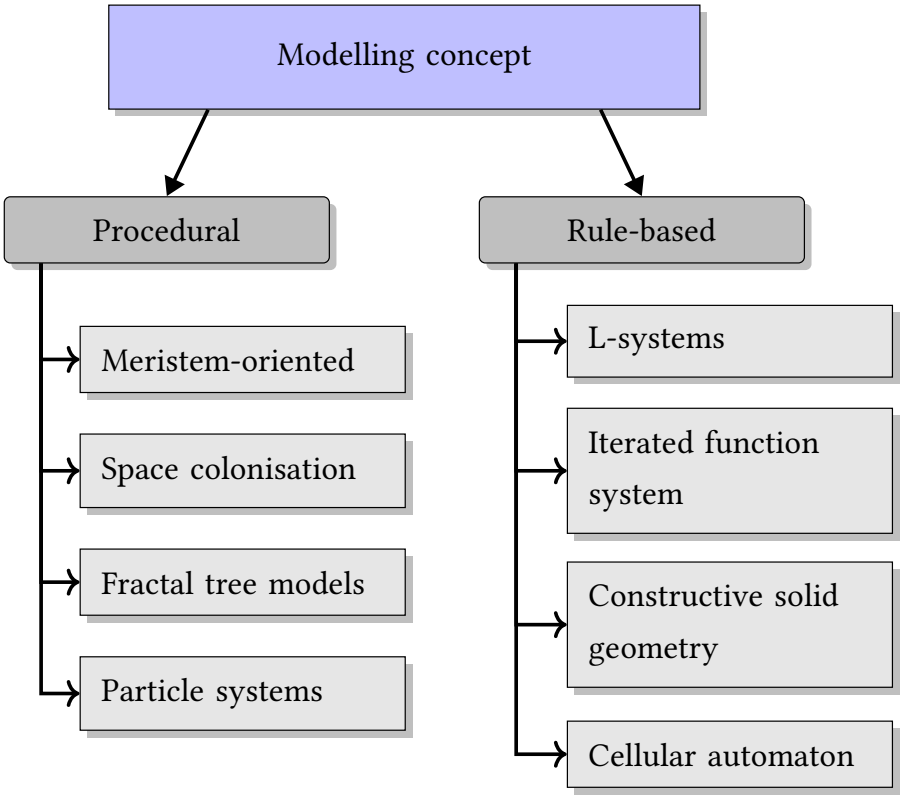


Figure 2.39.: Classification of modelling approaches into procedural and rule-based models (non exhaustive).

2.3. Functional-Structural Plant Modelling in Detail

As shown, current FSPM approaches are quite diverse. They follow various schools, use divers concepts and have different aims. What is missing are standards for modelling and best practises for design and implementation. The following attempt to develop such standards in FSPM is based on experience, empirical data and observations made during several (GroIMP-related) modelling projects of crops such as barley (Buck-Sorlin *et al.* 2007b, 2005), cut-rose (Buck-Sorlin *et al.* 2007a, 2011), tomato (de Visser *et al.* 2014), rice (Xu *et al.* 2011; Xu *et al.* 2010), cucumber (Chen *et al.* 2014), poplar (Buck-Sorlin *et al.* 2008), apple (Buck-Sorlin *et al.* 2012) and several related models, beech and spruce (Hemmerling *et al.* 2008), wheat (Evers *et al.* 2010), oilseed rape (Groer 2006; Groer *et al.* 2007). The knowledge gained during these diverse projects profoundly affected the development of the FSPM-P (Chapter 6, Henke *et al.* (2016)), a prototype for FSPM.

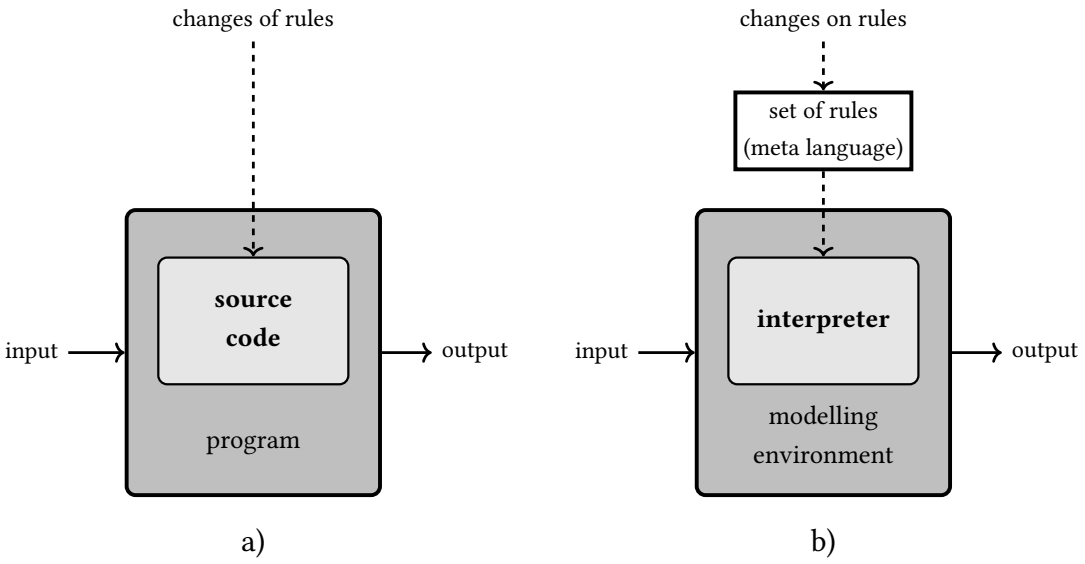


Figure 2.40.: Comparison of a) classical procedural simulation program with b) an interpreter of a rule-based model approach. While in a procedural approach the rules are usually hard coded (embedded) in the simulation program, the rule-based approach uses an interpreter for a meta language and the rules are defined in this meta language. As a consequence the rules can be easily changed.

2.3.1. Modelling Process

To arrive at an operational FSPM a procedure should be followed that consists of eight main activities including 1) planning, 2) data acquisition, 3) data processing, 4) modelling, 5) validation, 6) simulations, 7) evaluation, and 8) documentation (Fig. 2.41). All of these activities can be further decomposed into several sub-activities. The process of modelling with all its activities and sub-activities as given in Tab. 2.6 is only on the surface a straightforward process: actually a number of feedback loops exist, mainly for validation reasons, between different steps. For agent-based model (ABM), Grimm and Railsback (2005) introduced a 'modelling cycle' of six tasks: formulate the question, assemble hypotheses, choose model structure, implement the model, analyse the model, and communicate the model.

The planning or conceptual phase is a complex activity with several sub-activities that should not be underestimated. A conscientious preparation at this step of the project will always pay off later. Unfortunately, doing a good planning requires some experiences one only can have after finishing some modelling projects. This explains the relatively high failure rate of projects,

i.e. projects that do not deliver the expected result or, even worse, that can not be finished. On the other hand, this also shows the need for a guided modelling process and support in modelling helping the modeller to concentrate on important things instead of dealing with technical details.

Table 2.6.: The activities within a common modelling process.

Activity	Sub-activity	Description
planning	problem analysis	analysis and problem description, define inputs and outputs
	model objectives	derive questions to be answered
	hypotheses	define hypotheses
	scenarios	develop key scenarios
	elements	specify model aspects and boundaries
	data types, variables and scales	fix temporal and physical resolution, determine sample size as function of scale
	work plan	fix time schedule with individual tasks
	literature research	'State of the art': identify key literature (description of processes in reviews), search for available data on model parameters
data acquisition	measuring	measure the required input data using one or more of the methods described in Tab. 2.7
data processing	data preparation	clean measured data (e.g., outliers or missing values)
	data analyses	statistical examination and interpretation
modelling	model elaboration	design and define a numerical/mathematical model
	parameter estimation/fitting	find the right model parameters and estimate their values correctly (mean value, standard deviation, etc.)
	model implementation	design and implement the model formally in a programming language
	sensitivity test	check the influence of model parameters

validation	comparison: results vs. reality	check if the model reproduces sufficiently realistic output
	improvement	calibrate model (iterative process to enhance the model performance and improve realism)
operation phase	application of the model	perform simulations, scenario runs, ...
evaluation	test robustness	check if robust results could be obtained
	evaluation	evaluate the results
	conclusion	summarise the results
documentation	model documentation	document the whole model project with all sub-activities
maintenance	bug fixing	
	model porting	adapt the model to new platforms or systems

To obtain the required data base for a model strongly depends on the type of model. Usually, the literature does not provide data either in the required quality or quantity, so that own experiments and measurements are usually obligatory. The fact that many investigated crop species exhibit a seasonal limitation to their growth period can slow down the data acquisition process and add an additional complication. Reasons for the necessity to repeat part or all of the measurements (in the worst case scenario during a following season) to complete the data base are manifold and range from accidental or systematic mistakes committed during measurements, over lost measurements or plants, to the necessity to gain a better understanding of the model, or simply a modification in the aims and concepts due to the acquisition of new knowledge from the literature. Due to the nature of measuring of some, e.g., physiological data like flow rates, measuring itself can be quite challenging too. The large number of plants needed to be measured to become statistically 'significant' makes data acquisition increasingly difficult considering destructive and non-destructive measurements, especially for dynamic models or when multiple environmental factors have to be varied.

After the data acquisition is finished and the data base is complete the generated data need to be processed. Before the data can be analysed statistically, they need to be 'cleaned'. During this step corrupted or incomplete data sets are removed and often also extrema on the upper

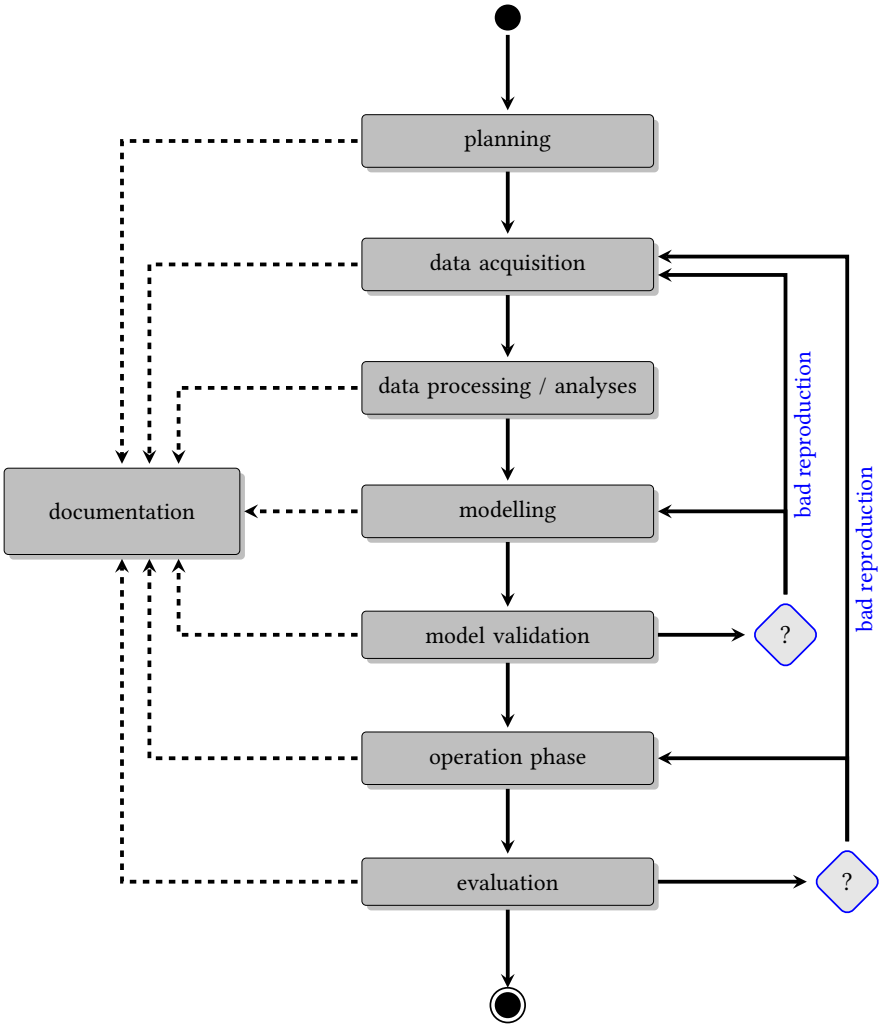


Figure 2.41.: Diagram depicting a simplified (idealised) common modelling project.

and lower range removed. The usual goal is to have a data set that describes a representative (not necessarily average) individual of the investigated species.

With the aim of the model, the research questions and hypotheses in mind, the actual model can be designed. This is often done in three (to four) steps: the first step consists in model elaboration, during which an appropriate description of the observed behaviour is searched and outlined using a formal language - at this stage often mathematical equations are used. In the second step, the model parameters are estimated, which involves parameter fitting or curve fitting. In an optional third step, the impact/influence of a single parameter or a set of parameters on the whole model is tested in a sensitivity test. This helps to identify parameters with a great

impact and on the other hand parameters that possibly can be neglected. After the right number of parameters and their correct values have been found the last step is the implementation of the model. The field of programming languages is large, ranging from classical imperative programming languages over specialised modelling languages to mathematical languages that work at a high level of abstraction.

After the model has been defined it needs to be validated to make sure the observed behaviour is reproduced correctly. This process is typically an iterative process of 'fine tuning' where in several repetitions parameters and parts of the sub-models are adapted to a stage of 'best fit' is reached. An important aspect of model validation is to use an independent data set, i.e. one that was not used to fit the model. To make a model operational is therefore usually a time-consuming task.

All the previous steps are more or less preliminary ones meant to set up and build a model that can ultimately be applied in practice. One characteristic application area are scenario tests: in these some parameters - often environmental ones - are changed, and the model simulations are repeated with this new context. Another aspect is to check whether the model can, e.g., produce some emergent growth pattern or show other behaviours that are not directly implemented in the model. Depending on the complexity of the model and the number of scenarios, performing simulations requires a significant amount of the project time.

The evaluation of simulation results will show if the model output is robust and able to answer the proposed research questions and if the hypotheses could be proven. Occasionally, it is necessary to repeat simulations or even to adapt the model when unexpected or unlikely results are obtained. This phase also includes the evaluation and conclusion of the project.

An often neglected activity is the documentation of the whole modelling project including all sub-activities. Documentation is an essential aspect for follow up projects, future model extensions and model maintenance. To bring it on one point: A model without a proper documentation is sentenced to die.

Due to the lack of a reliable estimation of the time requirements for the individual activities during the whole modelling process it is hard to give details but based on experiences the following observations given in Fig. 2.42 can be made. The planning phase typically requires one fifth of the whole project time. For the data acquisition it is hard to give an average value since this entirely depends on the amount of easily available data. If the main part of the data

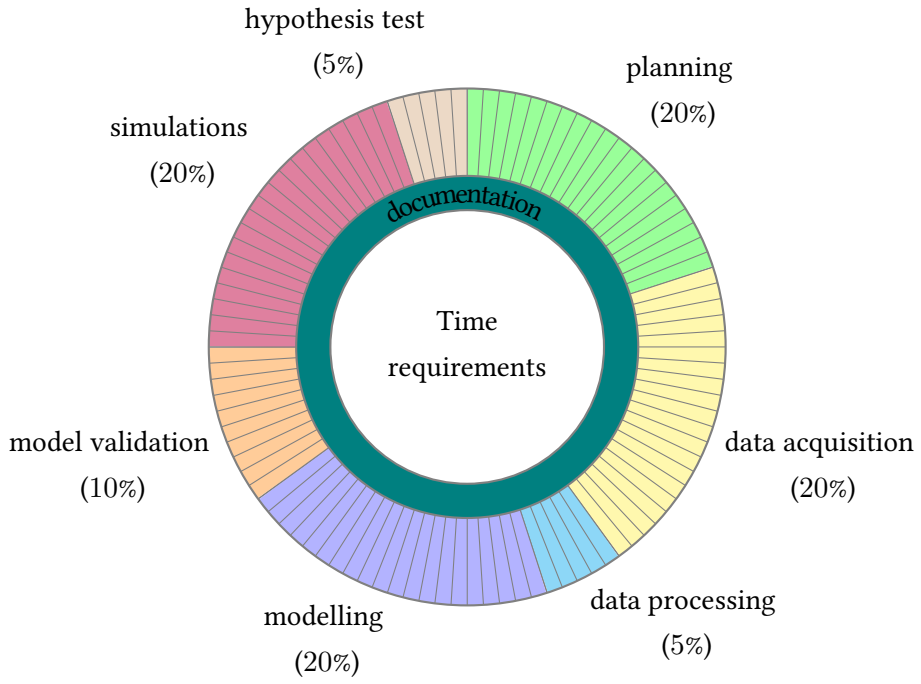


Figure 2.42.: Time wheel of a simplified (ideal) common modelling project.

needs to be measured in own experiments, this phase can be the most time-consuming activity. For the average chart we will account the data acquisition phase with 20 percent of the total project time. With some experience in basic statistics and the use of software tools the time requirements of the data preparation phase can be rather well estimated, as they will constitute only a small percentage of the total project time. The modelling phase, again, requires more resources while the validation typically does not require much time as it is quite straightforward (without the intention to decrease the importance of this activity). The time requirements of the simulation phase cannot be generalised. There is a large variety of scenarios that can be thought of, and the numer of scenarios will also depend upon the complexity of the simulated system, e.g., resolution and scale at which plants are modelled, the size of the modelled scene (number of plants and additional objects), the accuracy of light calculations, number and complexity of physiological and physical processes. All these uncertainties make it rather risky to issue a generalised statement. In the following, the simulation phase is assumed to taking up one fifth of the total needs. The hypothesis tests, finally, are not really time-consuming. The more demanding part is the documentation of each sub-activity that could - and ideally should - be estimated with 20 percent of the whole project time. For simplification, this time is already included within the single sub-activities.

2.3.2. Exemplary Model Design

Based on our experience in modelling and model design, we can identify a number of recurrent model designs that we consider fundamental. Three main classes of design aspects can be distinguished: 1) model structure, 2) model workflow, and 3) source/sink relations. The model structure (Sec. 2.3.2.1) describes how a model should be separated into different files / components / modules in order to obtain a clear model structure. The model workflow (Sec. 2.3.2.2) describes how operational sequences of tasks and sub-tasks are ordered and repeated. The last design pattern (Sec. 2.3.2.3) describes the typical transport movements of matter and information with respect to the origin (source) and target (sink) of a given type of matter/information.

2.3.2.1. Model Structure

A strictly modularised design should follow a number of principles in order to produce a clear model structure, that is easy to understand, to change and to extend. Such a design principle follows common concepts borrowed from software engineering (see Balzert (2009) and Gamma *et al.* (1995)). Parts that do not belong together, therefore, should not interweave each other. Thus one should always separate, e.g., species-specific information and the actual model infrastructure which latter is responsible for controlling the model workflow. Usually, there is only one main file needed to cover all items essential for the model infrastructure. Everything that is species-specific on the other hand, should be further sub-divided into three parts: 1) parameters, 2) definition of organs, and 3) definition of growth and development rules. An overview of a common model segmentation is given in Fig. 2.43.

Using the possibilities of modern object-oriented programming languages, the organ definition itself should follow a hierarchical structure with an organ superclass, providing basic variables and functions common to/essential for every organ. This ranges from simple counters for age, temperature sums to more complex pre-implemented functions for growth rates. Every concrete organ type extends this superclass and inherits its functionality. On the organ level, it is useful to separate the organ implementation with all the variables and functions from the 3-d visualisation of the organ shape.

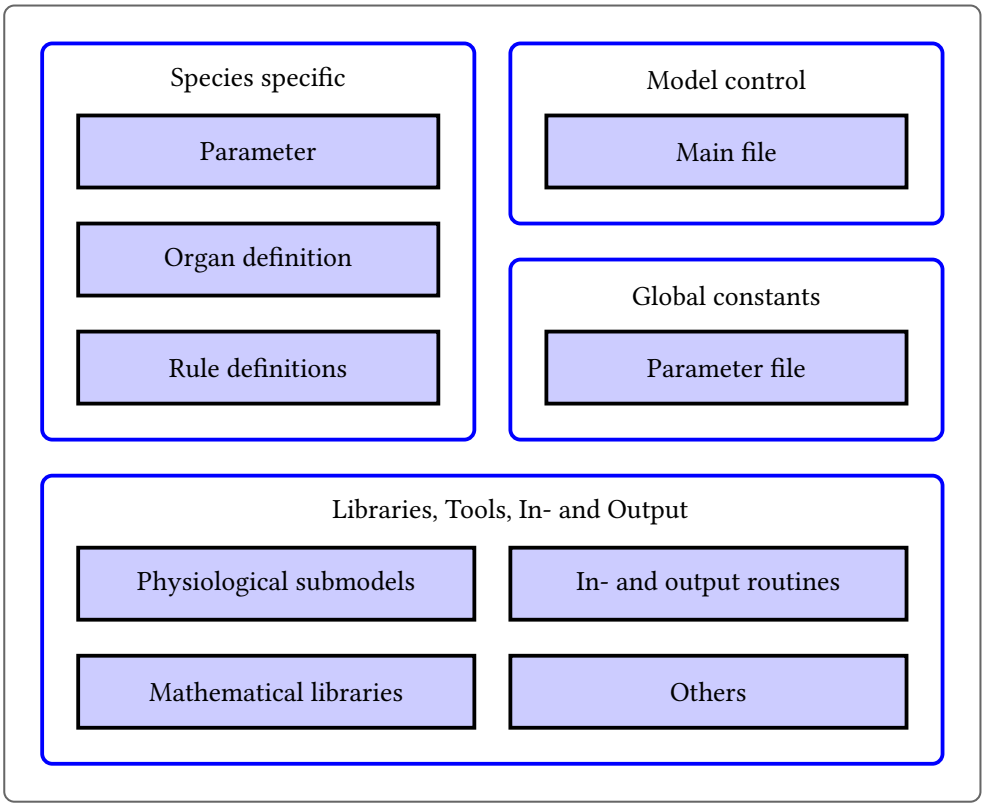


Figure 2.43.: Typical components of a modularised FSPM, considering the control by external parameter files.

It is also useful to store model parameters - both species and scenario-specific - in external parameter files, the latter allowing a rapid and easy exchange of parameters between users and platforms. With this technique, e.g., different scenario configurations can be stored in different files and applied to the model just by replacing one parameter file by another, without the need to modify the actual code.

2.3.2.2. General Model Workflow

Concerning the model workflow, static and dynamic models can be distinguished. While in the dynamic case, at each growth step the 3-d structure will change according to the changed parameters, in the static case the 3-d structure typically does not change. In the static case, typically, model parameters are changed to investigate the response of the 3-d plant structure. For

both cases, the general workflows are similar and have the same tasks. They can be generalised in three steps as follows: 1) initialisation, 2) simulation, and 3) evaluation.

Dynamic Model A typical general model workflow of a dynamic FSPM consists of three phases: 1) initialisation, 2) growth steps, and 3) summary (Fig. 2.44). During the initialisation global parameters are loaded and the whole scene is initialised. This includes the correct parameterisation of all light sources, e.g., the sky according to the measured values for a given day. Furthermore, the individual plant or plants are initialised and arranged in the scene, e.g., with a certain pattern and density.

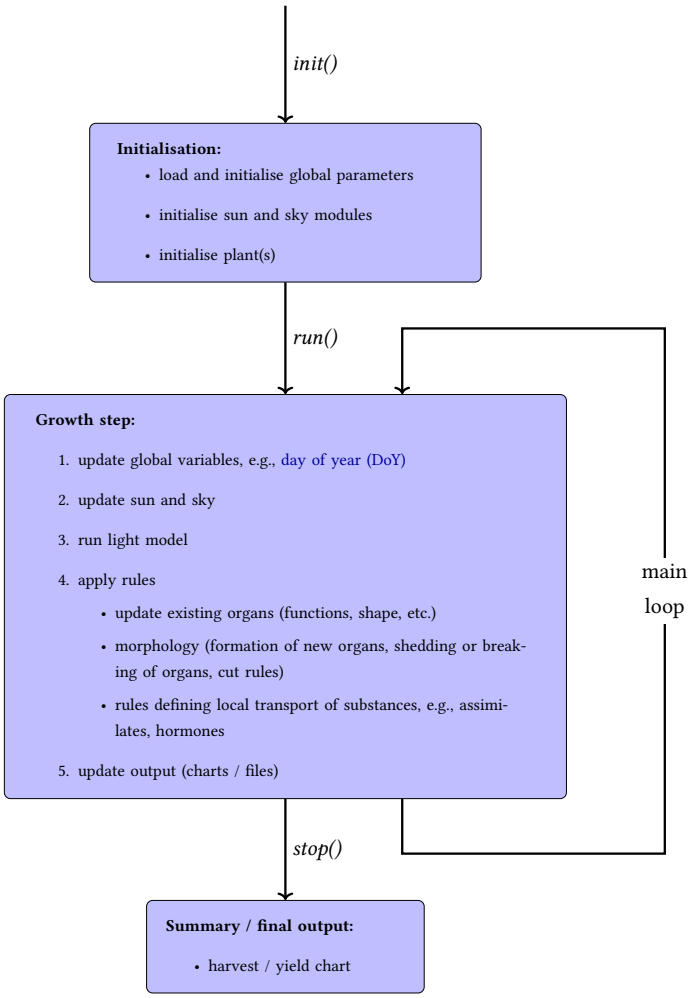


Figure 2.44.: General model workflow: after initialisation the model is executed during the main loop before final output is generated. Based on Henke *et al.* (2016).

Within the main loop, a discrete simulation step is performed, e.g., one growth step. At the beginning of each step, normally, some time counter (hour of day and day of year (DoY)) will be incremented. According to the new model time and date, the light sources (power output and new position/direction if it is a mobile light source) are updated and the light calculations are performed. With the information about light absorption, at the organ level, plant growth can be calculated and recorded. In the end, when the maximal number of simulation steps is reached, typically, some final (global) model data are collected and recorded.

Static Model The workflow of the static case is similar to the one of the dynamic case. During the initialisation, normally, whole plant structures are generated, usually based on real measurements of sample plants. In a static model the 3-d structure typically does not change while at each simulation step, e.g., environmental parameters like light or temperature are updated in order to simulate light absorption at plant level. This, for instance, is the case in scenario simulations where the response of a plant for different light sources, e.g., in a greenhouse is investigated (de Visser *et al.* 2014).

2.3.2.3. Modelling Source/Sink Relationships

The source/sink relation within a FSPM should, of course, try to simulate the relation that is observed in nature. In most cases, a source or sink function can be clearly assigned to an organ. However, when we include the developmental dynamics of an organ, there are exceptions, e.g., siliques of oilseed rape are in the juvenile stage sinks then become a source (i.e. they take over photosynthesis after the shedding of the main stem leaves). In fact, all juvenile organs, even leaves, are initially sinks with respect to assimilated sugar, while later on some of them turn into sources. Like in real plants, storage pools should be considered within the relations of source and sinks. This includes local storage pools within the leaves but also 'global' pools like those many species have inside their roots and stems. Such pools dynamically change their function between source and sink, depending on the developmental and respiratory needs of the plant. Figure 2.45 illustrates such an exemplary relationship for a model of a species starting at the stage of seed with a storage pool in the roots.

The question of transport is inseparable from the source/sink relationship. In an ideal model the transport should be realised by differential equations, describing the flow from organ to

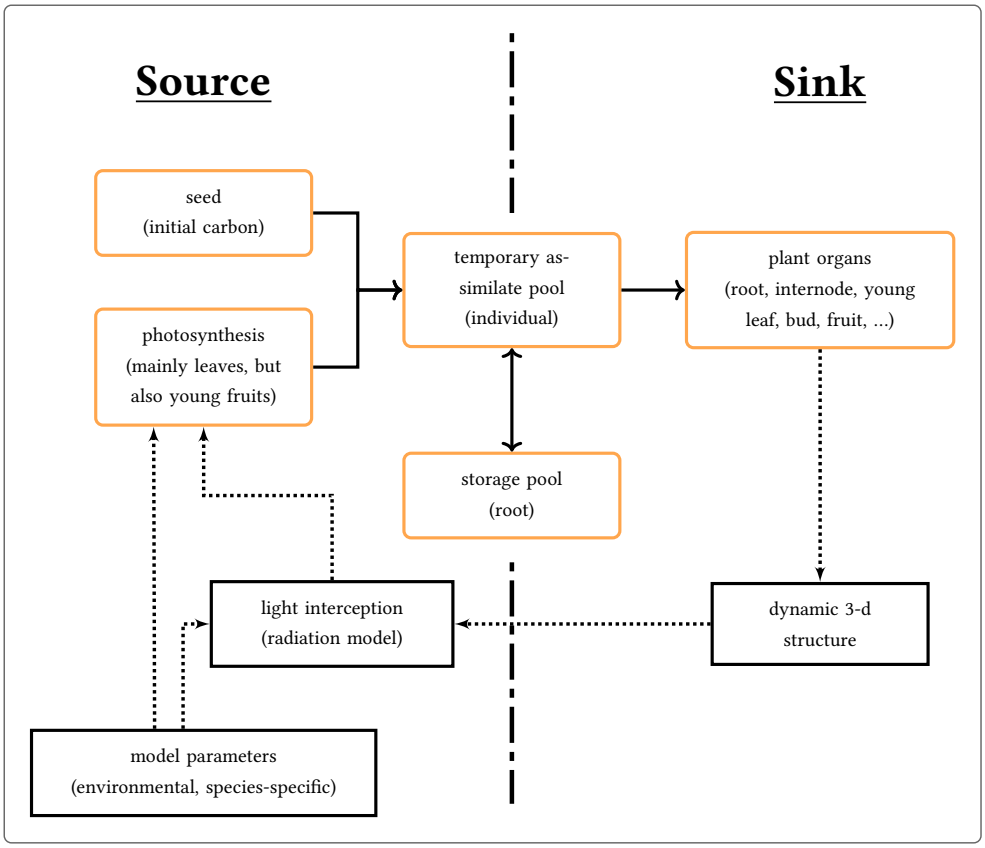


Figure 2.45.: Simplified source/sink relation within a typical FSPM. In this case an optional seed organ as sink and a storage pool within the roots are considered. Based on Henke *et al.* (2016).

organ. A 'global' or common pool, as being used in PBMs (see Sec. 2.2.1.2), neglects transport or sets transport resistance to zero.

2.3.3. Data Acquisition

All FSPMs have in common that they are typically quite data greedy. Before a FSPM can be used productively in simulations a lot of parameters need to be identified and their values fixed. To be statistically 'significant', measurements need to be repeated for a large number of plants - and for dynamic models over the whole growth period. Four main sources of model data can be identified:

- direct measurements,
- literature data,
- similar or comparable experiments, and
- parameter estimation from experimental data.

When other external parameters are changed the measurements need to be repeated for each parameter and normally for the permutation of the parameters. One typical example would be a temperature sensitive model. In this case, measurements are repeated under different temperature treatments. However, when a second parameter like soil humidity should be added, too, the measurements need to be repeated for a sufficiently large range of combinations of temperature treatments and humidity classes in order to get a significant data base for the model. This small example already permits to put forward the case for efficient methods for data acquisition.

The range of required data is directly related to the aim of the model. Figure 2.46 gives a categorised overview of data that are typically of interest for FSPM. Depending on the type of data, special techniques for measuring are required or preferable. Basically, four categories of methods for measuring morphological plant data can be distinguished (Tab. 2.7).

The still most widespread method is measuring by hand using a ruler and protractor. This method is still one of the most accurate but on the other hand also very time-consuming. The reasons for its 'popularity' are direct results of the throwbacks of other common methods. E.g., in point clouds produced by laser scanners, it is typically challenging to extract exact organ sizes, especially to identify the beginning and end points is problematic. Also, manual measurement allows to directly establish the identification of each organ as well as its topology.

Photogrammetry is another common method of data acquisition. It uses images captured by photographs or 2-d scanners as input. Images are processed to (semi)automatically identify and measure plant organs within them. Such methods are frequently used to measure leaf parameters, e.g., shape and surface area. After the actual image acquisition, the images are processed using dedicated image processing software: the user interactively selects threshold values for segmentation or identifies organ borders. One common drawback of photogrammetric methods are distortions when object and camera plane are not parallel. Therefore, such methods are

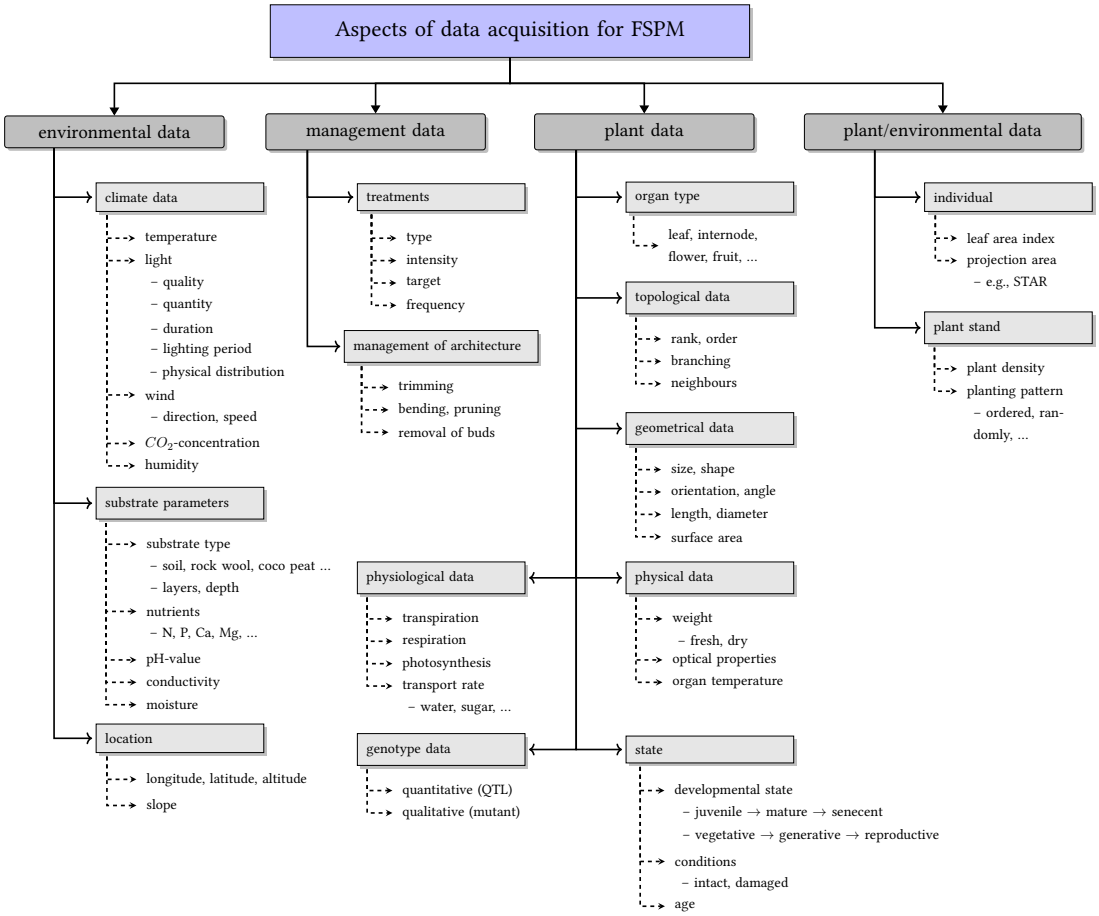


Figure 2.46.: Aspects of data acquisition for FSPM (non exhaustive).

typically destructive. In Henke *et al.* (2014a) (see Sec. 3) a non-destructive photogrammetric method for leaf observation was introduced.

A method widely used for digitising plants are electromagnetic digitisers. The principle of such a system is an electromagnetic field, emitted by a transmitter; a pen-shaped stylus is then used as a receiver within this field (usually one cubic metre large). The user can now iteratively measure every organ one by one. Technically caused such systems are quite sensitive to external influences, e.g., magnetic fields that would require calibrations when used indoors, or render this method useless in some cases, when used in the vicinity of iron heating pipes in the greenhouse.

Table 2.7.: An overview of measuring methods for morphological plant data, their advantages and limitations.

Method	Pros	Cons
hand	• accurate • (non) destructive	• very time-consuming
photogrammetric	• cost-effective	• destructive • problematic in 3-d
electromagnetic digitiser	• accurate • non-destructive	• sensitive to other influences • time-consuming
3-d scanner	• fast measuring • insensitive to sunlight • non-destructive • measuring the whole structure in one step	• poor resolution • expensive • problematic segmentation • occlusion

3-d measuring methods, e.g., structured light, LiDAR, laser line scanners, stereo vision, and other techniques for 3-d scanning, can result in the production of large amounts of data, so-called point clouds. Beside the technical problems of the single methods, their main problem lies in the segmentation of the derived data. To identify single plant organs is still a non-trivial problem. On the technical side, the biggest problem is caused by occlusion (i.e. overlapping leaves). What can not be 'seen' by the sensor will also be invisible in the 3-d image. To overcome this problem, recent developments combine multiple sensors into multi-view approaches. Depending on the used method most of the methods of this class, like laser line scanners, are insensitive to sunlight.

Other methods like ultrasound sensors and mechanical measuring arms are of minor importance in plant modelling.

A further important point is the temporal and spatial resolution of the measurements. Both have a great impact on the amount of generated data. Especially modern scanners produce large amounts of data that require novel techniques of data perpetration.

One fundamental question in data acquisition of plants is the question whether a measurement has to be destructive, thereby ending the life cycle of the plant, or non-destructive, thereby allowing the continued monitoring of one and the same plant until a natural end point. There are factors that can be measured only destructively, e.g., dry weight, and others, like transport or photosynthesis rates, that can be measured only non-destructively.

A critical point in data acquisition is the number of observed plants which needs to be large enough to get statistically significant data. Usually, a minimum of 20 to 35 plants (depending on plant size and complexity) is needed for a single treatment, to avoid the use of non-parametric statistical tests. During the planning phase, the loss of plants due to pests, mechanical destruction, and destructive measurements always has to be taken into account when estimating the initial demand for plants.

Based on the data obtained by one of the techniques mentioned above, a large set of secondary information can be derived. This includes several rates, e.g., of organ initiation, appearance, maturity and death. Also growth rates (organ extension rates) and bud break probabilities can be obtained in this way. Beside rates, important data about maxima, e.g., of organ size, branching levels and senescence classes can be generated.

2.3.3.1. Levels of Structure Description

As we have seen earlier, FSPMs can be classified into static and dynamic models, see Fig. 2.47. A static FSPM takes an unchanging 3-d structure, e.g., a plant at a certain date, as model input. Static FSPMs are using this structure in order to explain spatial heterogeneity in physiology, e.g., local light distributions within a single plant (Sarlikioti *et al.* 2011).

When changes in organ and plant structure in time are taken into account, or generally speaking when organ growth and development are simulated explicitly in space and time, then the model is referred to as being dynamic. Such processes are typically organ extension and formation of new organs including branching processes.

A dynamic FSPM is called context-free if it describes the ontogeny without considering external factors. One way to describe a context-free model are (context-free) L-systems. If external factors and conditions are considered such a model is referred to as being sensitive. Typical sensitivity factors are light and tropisms.

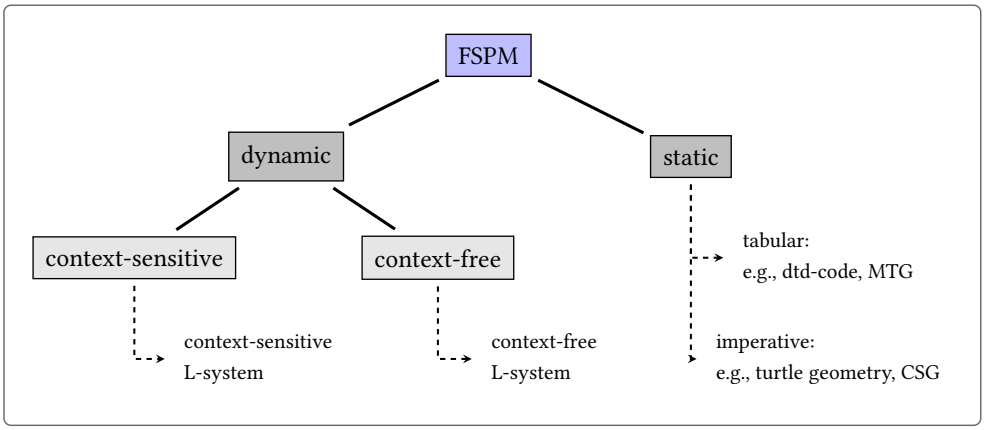


Figure 2.47.: Levels of structure description for FSPM approaches (grey boxes). The items connected by dashed lines are examples of ways in which such a model type can be described.

Following Godin (2000), plant architecture can be represented in three ways: 1) global, 2) modular, and 3) multiscale. In a global representation plant architecture is considered as a whole, and the fact that plants have different organs is not taken into account. In a modular representation plants are decomposed into modules that may be organ-based or can simply consist of a spatial subdivision. The plant is made up of a repetition of modules. In multiscale representations, the plant is described at a number of scales or hierarchical levels. See Godin (2000) for further details.

2.3.4. Application Areas

Models are used to abstract, describe, simulate, extrapolate and finally understand the function of complex systems. This also applies to FSPMs. The application areas related to FSPMs are as diverse as the backgrounds (Sec. 2.2) they are based on. One of the main applications is as a research tool to investigate functioning and relations within individual plants and plant stands.

Several models are focused on the increase in crop yield. FSPMs are widely used as planning and decision support tools. FSPM also play an increasingly important role in teaching students of different disciplines. Typically, FSPMs are used in the background of one of the following research areas: biology, agronomy, agriculture, horticulture, landscaping, forestry, and ecology. Following the complexity of the topic, interdisciplinary applications are typical.

Table 2.8.: Selected examples of GroIMP applications during the past years (non exhaustive).

Architecture	class with architecture students, Kniemeyer *et al.* (2008)
Biology	understanding of processes/functioning, Chen *et al.* (2014)
Computer graphics	stochastic path tracing on graphics cards, Huwe and Hemmerling (2010)
Crop science	FSPM as a new versatile tool in crop science, Vos *et al.* (2010)
E-Learning	Virtual Forester, a tool for teaching and decision support in forestry, Lanwert (2007)
Horticulture	cut-rose model, Buck-Sorlin *et al.* (2010, 2011)
Landscaping	reconstruction of historical gardens and parks, Smoleňová *et al.* (2010)
Plant physiology	a model of poplar linking physiology and morphology, Buck-Sorlin *et al.* (2008)
Plant production	simulation of wheat growth and development, Evers *et al.* (2010)
Scenario analysis	Climate KIC Innovation project: Carbon LED, LED based production systems – Calculation of light climate and scenario tests within 3-d plant models
Teaching	(nearly annual) GroIMP workshops and summer schools, GroIMP user and developer group meetings

GroIMP (Sec. A.9.1) as modelling platform has been used for several of the mentioned application areas (Tab. 2.8) since it has been released in 2003.

Several further application areas have been established during the past years. Plant vs. pest interaction, inter-cropping models, and soil/root models have moved into the focus of current research. An increasing demand also arose for linking genetic with (eco)physiological information, in order to better understand genotype $\times$ environment interaction, e.g., genes and quantitative trait locus (QTL) to FSPM. Recently, FSPMs and traditional crop models have begun to mutually impact each other (Fourcaud *et al.* 2008; Vos *et al.* 2010, 2007).

CHAPTER 3

First Paper

This paper is published as:
Henke M, Huckemann S, Kurth W, Sloboda B (2014) Reconstructing Leaf Growth Based on Non-destructive Digitizing and Low-Parametric Shape Evolution for Plant Modelling Over a Growth Cycle, *Silva Fennica*, 48(2), doi: 10.14214/sf.1019, (Creative Commons CC BY-SA 4.0 licence)

Authorship

- Stephan Huckemann re-wrote the manuscript based on a former version of Michael Henke.
- Winfried Kurth supported the writing of the manuscript.
- Branislav Sloboda supported the writing of the manuscript.

SILVA FENNICA

Silva Fennica vol. 48 no. 2 article id 1019
Category: research article
www.silvafennica.fi
ISSN-L 0037-5330 | ISSN 2242-4075 (Online)
The Finnish Society of Forest Science
The Finnish Forest Research Institute

Michael Henke[1], Stephan Huckemann[2], Winfried Kurth[1] and Branislav Sloboda[1]

Reconstructing leaf growth based on non-destructive digitizing and low-parametric shape evolution for plant modelling over a growth cycle

Henke M., Huckemann S., Kurth W., Sloboda B. (2014). Reconstructing leaf growth based on non-destructive digitizing and low-parametric shape evolution for plant modelling over a growth cycle. Silva Fennica vol. 48 no. 2 article id 1019. 23 p.

Highlights

- A complete pipeline for plant organ modelling (at the example of poplar leaves) is presented, from non-destructive data acquisition, over automated data extraction, to growth and shape modelling.
- Leaf contour models are compared.
- Resulting "organ" modules are ready for use in FSPMs.

Abstract

A simple and efficient photometric methodology is presented, covering all steps from field data acquisition to binarization and allowing for leaf contour modelling. This method comprises the modelling of area and size (correlated and modelled with a Chapman-Richards growth function, using final length as one parameter), and four shape descriptors, from which the entire contour can be reconstructed rather well using a specific spline methodology. As an improvement of this contour modelling method, a set of parameterized polynomials was used. To model the temporal kinetics of the shape, geodesics in shape spaces were employed. Finally it is shown how this methodology is integrated into the 3D modelling platform GroIMP.

Keywords non-destructive data acquisition; automated data extraction; image processing tool; growth modelling; leaf shape modelling; reusable modules; *Populus* x *canadensis*
Addresses [1] Department Ecoinformatics, Biometrics & Forest Growth, [2] Institute of Mathematical Stochastics, University of Göttingen, 37077 Göttingen, Germany
E-mail mhenke@uni-goettingen.de
Received 15 October 2013 **Revised** 26 February 2014 **Accepted** 12 March 2014
Available at http://dx.doi.org/10.14214/sf.1019

List of symbols

Abbreviation	Unit	Description
Plant material		
T_1	-	Set of leaves taken from tree one
T_2	-	Set of leaves taken from tree two
T_3	-	Set of leaves taken from tree three
$T_{1–3}$	-	$T_1 \cup T_2 \cup T_3$
t	d	Time
Size model		
l_0	m	Initial leaf length on its first day of measurement ($t=0$)
m	-	Parameter of the leaf length function
k	-	Parameter of the leaf length function
n	-	Parameter of the leaf length function
Proportional shape model		
l	m	Maximal leaf blade length
b_l	m	Maximal width of left half of leaf blade
b_r	m	Maximal width of right half of leaf blade
l_{m_l}	m	Normalized position on the midrib where the maximal left width is attained
l_{m_r}	m	Normalized position on the midrib where the maximal right width is attained
b_l/l	-	Shape parameter; ratio between maximal left width and length
b_r/l	-	Shape parameter; ratio between maximal right width and length
l_{m_l}/l	-	Shape parameter; ratio between l_{m_l} and length
l_{m_r}/l	-	Shape parameter; ratio between l_{m_r} and length
Geodesic model		
$x^{(t)}$		Modelled configuration landmark matrix at time t

1 Introduction

1.1 Motivation

Realistic modelling of plant growth is one of the key issues in ecoinformatics. In plant modelling, structural elements such as leaves play an essential role (Lecoustre et al. 1992; Prusinkiewicz et al. 1994). They are the main interface between the plant and its environment. Furthermore, they determine radiation interception and, thus, gas exchange and photosynthesis, which are the main factors of growth. Various functional-structural plant models (FSPMs) using several different software systems, e.g. GroIMP (Kurth 2007; Kniemeyer 2008; GroIMP Developer Group 2013), GreenLab (de Reffye et al. 1997; Hu et al. 2003), LIGNUM (Perttunen et al. 1996), AMAPStudio (Griffon and de Coligny 2012), LStudio (Biological Modeling and Visualization research group 2013) and Almis (Eschenbach 2000) have been developed for these issues.

Crop models are a common tool for estimating yield or biomass development in agri-, horti- and silviculture (Ruiz-Ramos and Mínguez 2006). However, classical crop models do not take into account plant architecture, let alone the growth of a single leaf. Plant structure gets into the focus in the development of new 3D crop models. Studies on leaf area already have a long history, going back to the beginning of the 20th century (Gregory 1921). Measurements were done manually (Gallagher 1979) for winter wheat and spring barley, and even recently, Chen et al. (2009) measured leaf lengths for over 1700 hours with a ruler, which is often the common way of obtaining data. Even when technical help was used for digitisation, the image segmentation was still done manually, e.g., Neto et al. (2006) extracted 510 leaves by hand. In order to obtain statistically meaningful results, a large number of measurements has to be done, which requires large human and financial resources.

1.2 Objectives and goals

The aim of this research is to provide:

a) a simple, effective, non-destructive and field-applicable method to acquire forms of leaves during a growth cycle,
b) an image processing tool to automatically extract leaf data from a large number of images,
c) a model for leaf shape development that is parsimonious with respect to the number of parameters yet as realistic as possible, and,
d) an integration of this model into an existing software environment allowing for *rule-based* (Kniemeyer 2004) realistic plant modelling.

In order to accomplish aim (a), we used a photogrammetric device consisting of a digital reflex camera within a rigid frame providing high accuracy of measurement while keeping every investigated leaf completely intact for frequently repeated measurements over its entire growth period. During such measurements, each leaf is measured / photographed several times. This has to be done for several leaves in order to obtain a statistically significant database, resulting in some hundred images, which need to be automatically extracted in a reproducible way. Thus, an ImageJ (ImageJ Developer Group 2013) macro as image processing tool (b) was developed and used. Concerning (c), reducing the number of parameters decreases the accuracy of the model, calling for search of a good trade-off. We propose to model the leaf forms with polynomials determined by a minimal set of 5 to 13 real-valued parameters. The set of these parameters defines form as size and shape in an abstract shape space. Fig. 1 illustrates all the phases of the workflow beginning with data collection, interpretation, representation and ending at the integration of the resulting model within new or existing FSPMs.

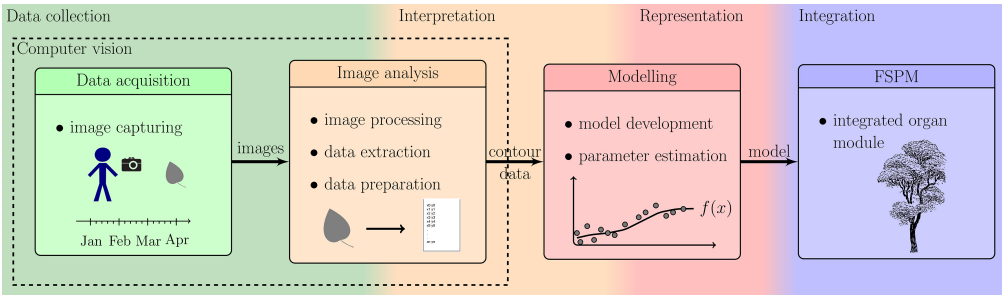


Fig. 1. Overview of the described workflow: showing the steps from photographing leaves to the integration of the resulting organ modules into an existing FSPM.

During the proof of concept phase, it turned out that the investigated leaves can be accurately modelled with the few parameters proposed. In the final step (d), the developed leaf models are implemented in the eXtended L-System modelling language (XL) (Kniemeyer 2004, 2008), a programming language created specifically for use in functional-structural plant modelling (Kniemeyer et al. 2007). It is an extension of Java that combines the advantages of an imperative and object-oriented language with those of a rule-based rewriting language. We use the modelling software GroIMP (Kurth et al. 2006; Kurth 2007; GroIMP Developer Group 2013) for implementation. Thus, the leaf components based on the above models are available for use in XL in different functional-structural models. These new plant organ modules facilitate the building of realistic complex models as they just need to be used and parameterized without the need to care about any implementation details.

1.3 Previous work

As far as we know, the task we set has not been tackled as a whole, while there is vast literature for several of these components, e.g., Dornbusch and Andrieu (2010) introduced the Lamina2Shape program, which was written in the commercial software MATLAB (The MathWorks, Inc.). They propose cutting the leaves and using a flatbed scanner for image acquisition. Using a scanner, of course, makes it impossible to observe growth of one individual leaf over the whole growth period which is often desirable (Maksymowych et al. 1973). This is particularly important when different temperature treatments influence leaf growth and, consequently, the total biomass production (Peacock 1975; Dennett et al. 1978). Use of commercially available leaf area meters (like the LI-3100, LI-COR Inc., Lincoln, Nebraska, USA) is another common destructive way of measuring leaf area (Routhier and Lapointe 2002; Reich et al. 2004).

These methods cannot be used to acquire data of an individual leaf over the whole growth cycle, which is a major limitation for the use in several FSPMs that have the aim to reproduce the dynamics of growth and morphological development in realistic detail. Further, most of these methods cannot be applied in the field, which restricts the number of species that can be investigated.

Electromagnetic 3D digitizers (e.g. Fastrak, Polhemus, USA) that are also used for digitisation of leaves (Wiechers et al. 2011) have several restrictions, too. First, it is difficult to avoid influencing the measurements by, e.g., touching the leaves while still aiming to be as close as possible in order to obtain correct data, and to avoid wind and breathing effects. Second, several points need to be recorded and repeatedly measured for each leaf. Thus, each point needs to be touched over and over again, so that marks on the leaf would be needed, which could alter leaf development. Apart from that, a thigmomorphogenesis (stunted growth due to touching) effect might ensue from repeated probing by the stylus and the hand of the digitizing person.

Regarding leaf growth models, several methods have been proposed in the literature. The most common way is to separate shape and size and model both aspects independently (Mosimann 1970). Functions or curves are used to generate a contour in most models in order to describe the shape (Chi et al. 2003; Dornbusch et al. 2011). State variables such as leaf length, leaf area or dry weight will change with time and are thus usually described using a growth function of time (e.g., Gompertz, logistic, Richards, Weibull, the beta growth function, or spline functions (Richards 1959; Richards 1969; Yin et al. 2003)). Note that there is usually no statistically significant difference between the fitted Richards curves and the Gompertz curve (Hackett and Rawson 1974).

Statistical methods have also been applied to model shape: Neto et al. (2006) used a chain encoded leaf contour as a basis for an elliptic Fourier descriptor. In this way (and using principle component analysis) they described the leaf shape, in order to reduce the total number of Fourier coefficients from $4h - 3$ (with $h = 30$ being the harmonic number). However, this method is still not practically applicable in an FSPM, as trees have hundreds or thousands of leaves of different ages. Finally, Chien and Lin (2005) used elliptic Hough transformations for shape description.

2 A simple and efficient photometric method

2.1 Recording equipment

In order to acquire realistic growth data from biological objects at hand, the method to be employed needs to be accurate, reliable, robust, and easily operable in field use under natural lighting conditions; non-destructive and not affecting natural growth since we want to frequently assess each object over a longer period of time at regular intervals. In particular, any impacts on natural growth during the data collection need to be minimized in order not to falsify the original data.

The framework presented here uses a digital reflex camera, Sigma SD9 with a Sigma 50 mm macro-objective (SIGMA Deutschland, Rödermark, Germany). The full image resolution of 2268 × 1512 pixels was used, which translated into 14.58 pixels per mm. As an advantage, the Sigma camera model features a Foveon X3 image sensor, which uses an array of photosites: These consist of three vertically stacked photodiodes for the three main colours (red, green, and blue), instead of arranging them next to each other, as is the state-of-the-art of common CMOS-sensors and which would always induce a slight measurement bias. Other types of camera equipment with appropriate resolution are suitable as well. For the photographs the camera is fixed on a rigid device by means of a threaded bush that otherwise is used to fixture a tripod. The device consists of a rectangular rigid frame with an attached screen on the front (covered with a glass pane) for fixing leaves (Fig. 2). The windowpane uses an antireflection coated glass. This simple but robust construction is easy to use and comparatively cheap.

An advantage of the quite inflexible construction is that the object plane is oriented parallel to the projection screen so that perspective distortions are nearly completely eliminated. In effect, the pictures extracted do not have to be equalized. The fixed distance as well as the fixed focal length allow for the use of a reproduction scale constant over the entire data extraction process. In particular there is no need for recalibration or rescaling for each image taken. Because of the usage of a macro lens there is no need for any correction of optical aberrations: the latter could be necessary when a normal lens is used at a relatively short distance such as the one between the object plane and the projection screen.

Black cotton velvet serves as background (Fig. 3b). On the one hand its soft and flexible tissue helps to avoid damage to the 3D structure of the leaf and lets it sink in smoothly while it is fixed non-destructively. On the other hand it absorbs nearly hundred per cent of the shadows cast

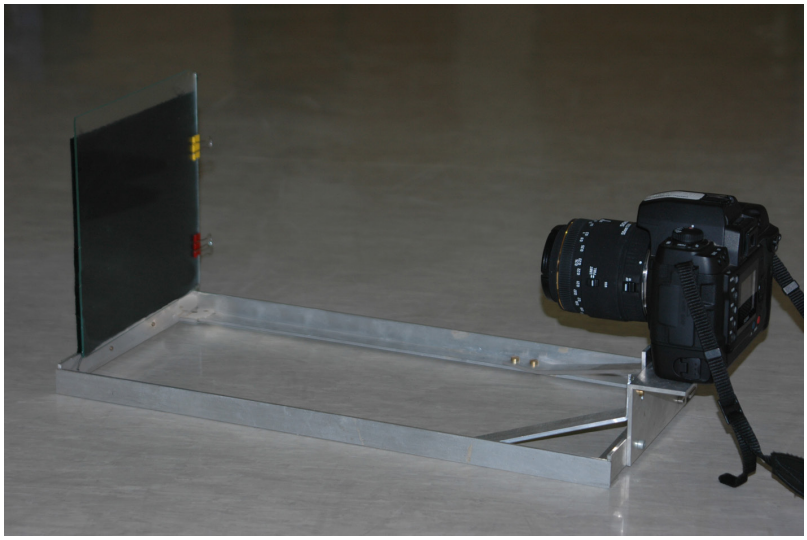


Fig. 2. Recording equipment. The use of an additional black board to cover the shining metal arms is recommended in order to prevent reflections, as well as the use of a second carrying strap fixed near the object plane to prevent toppling and to improve equilibrium for field use.

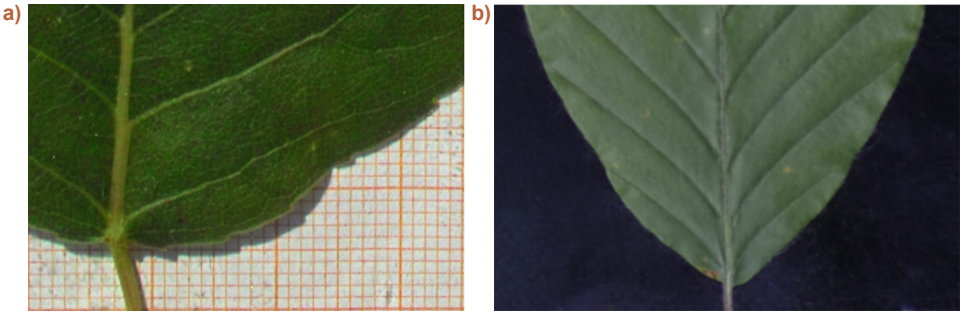


Fig. 3. Edge effects of different backgrounds. a) The photograph clearly shows a wide shadow that would cause problems in case of an automatic segmentation. b) A black velvet as background absorbs nearly all shadows and provides a soft structure allowing the leaf to sink in.

by the leaf's 3D structure (Fig. 3a), which are not negligible despite their small size. Without the velvet these shadows cast by the leaf would have to be corrected for in a laborious pre-processing step. Alternatively, in order to avoid shadows altogether the leaf contour would have to be fixed to the background, thereby possibly harming the protruding leaf veins. Since the scaling factor of our measuring apparatus was determined prior to data acquisition, the true size of leaves photographed in front of the velvet background can be easily determined.

2.2 The image processing tool

Of the images mentioned in the previous chapter, only the 2D contours bounding the leaves are of interest for the present application. Ultimately we would like to obtain from every acquired image a

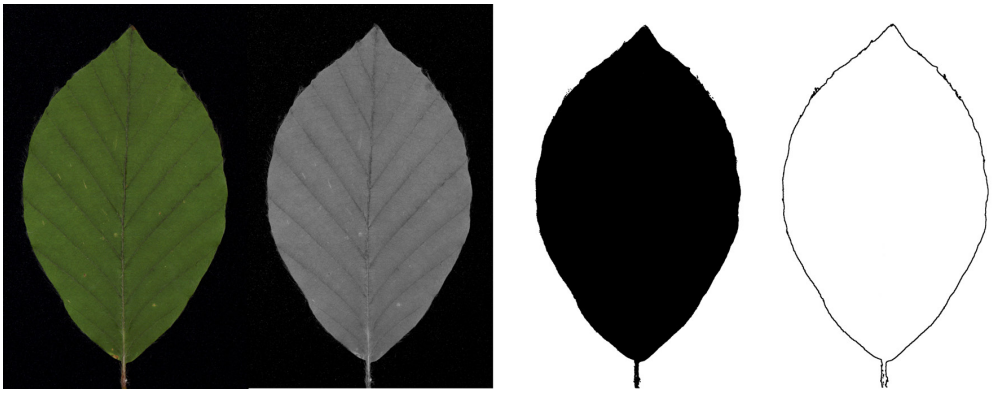


Fig. 4. Schematic workflow of contour extraction: Original image (colour picture) / green channel (grey-value image) / threshold image (black-white picture) / contour picture.

list of coordinate pairs (x, y) describing this contour (Fig. 4). Extracting contours is a typical image processing challenge which can be met by a variety of well-developed tools. Due to our specific recording equipment, the images are well prepared for this task as they are basically black (velvet) and green (leaf blade). When splitting the original image into its red-green-blue colour channels, only the green channel is used for further processing. This channel provides the highest contrast for the picture and facilitates segmentation, which in turn is realized by iterative adaptive thresholding. In the ensuing steps the picture is clipped, the background is cleared up and holes in the leaf if any are closed. In the next step the leaf petiole is removed from the obtained contour. Subsequently the leaf is rotated until its top and petiole entry point are aligned vertically. The resulting contour is stored as a list of 200 to 2500 coordinate pairs, depending on the size of the leaf, converted into mm with the leaf-base moved to the origin of the Cartesian coordinate system. As additional output values, maximum length, width and leaf area are obtained for each leaf.

The process described here was compiled as an image processing tool and implemented as an ImageJ macro (ImageJ Developer Group 2013). It permits the handling of complete directories into which all contours of collected images are extracted automatically as data files. However, two manual inputs are recommended: First of all, the user may opt for manual correction of an automatically determined threshold value, e.g. if a photograph was taken under unfavourable lighting conditions or if the ratio between leaf area and background is too variable, e.g., in the case of very big or very small leaves (as the automatic threshold finding is calibrated for medium-size leaves). Secondly, the user may want to specify the point at which the petiole is joined to the leaf blade. While this location can be automatically detected for most leaves, e.g. for all those depicted here, some petioles are running parallel to the base of the leaf blade or even pass in front of the leaf blade, making automatic extraction unreliable.

Bad image quality is in fact a common problem. Moreover in field use, lighting conditions are always problematic, e.g., sun reflections, shadows, blur due to wrong focus or aperture settings; illumination problems (over- and underexposure); and an exaggerated contrast. Tests and extended field use have shown that the image manipulation workflow presented here is surprisingly robust and resistant against these interferences which in a controlled laboratory environment would actually require no further attention. The script is free, open-source and available upon request from the first author.

Table 1. Summary statistics of the three trees that were used to build the data base for this study. From each tree a set of leaves was randomly chosen and observed over the whole period of growth.

Tree	Number of leaves	Number of measurements
T_1	20	262
T_2	12	162
T_3	11	158
Total	43	582

3 Plant material

For this study, we chose three trees of the Canadian Black Poplar (*Populus* x *canadensis* Moench), each representing a different clone, from an experimental stand at the University of Göttingen, Germany (51°31´N, 09°55´E). The study was conducted in 2008 on 5-year-old trees having a height of 140 to 250 centimetres (Table 1). From each tree during the entire growth period from spring to fall, 16 leaves were selected and measured, daily in the beginning, every two to four days subsequently. Some leaves had to be replaced by new leaves during the measurement period due to damage (scratches or big holes) or natural shedding. Leaves were sampled from different canopy heights to account for within-tree height effects, which were, however, not considered in this study.

Black poplar leaves were chosen for a number of reasons: first of all they do not resinate and are therefore not likely to soil the equipment; second, for the proposed photometric method we required flat, non-undulating leaf blades. Third, for advanced studies of inter clone differences planned in the future we needed a set of clones and a reference tree, which were only available for a limited set of species. Finally, in order to demonstrate our shape model at a simple example, a leaf with a simple contour was preferable. Leaves of, e.g., beech, birch, alder, or elm can be analysed in a similar way. For lobed leaf shapes such as maple or oak more sophisticated contour models are necessary.

In the following, the three trees are denoted by T_1, T_2 and T_3.

4 Leaf modelling

In our approach, we will neglect microstructure growth. Within a parameterized set of forms the statistically closest match is searched for, as in landmark-based shape analysis (Dryden and Mardia 1998). This paradigm lacks a premeditated biological model; rather biological data will be naturally associated with an appropriate model from a family of generic parsimonious models.

In particular we want to describe *form* by a tuple of one-dimensional *size* and multidimensional *shape*. As a common assumption for biological objects, we expect that growth affects both size and shape. A factorization into size and shape is in no way canonical (e.g. Mosimann (1970) for a broad discussion).

Statistical calculations and parameter fitting were done using the R language and environment (R-Project Developer Group 2013). The analyses presented here are restricted to single leaves, time series of individual leaves, and sets of leaves of individual trees.

4.1 Size modelling

In this work, we use the distance from petiole to tip, approximating the length of the main leaf vein, as the size variable. This attribute can easily be collected and has a simple geometric mean-

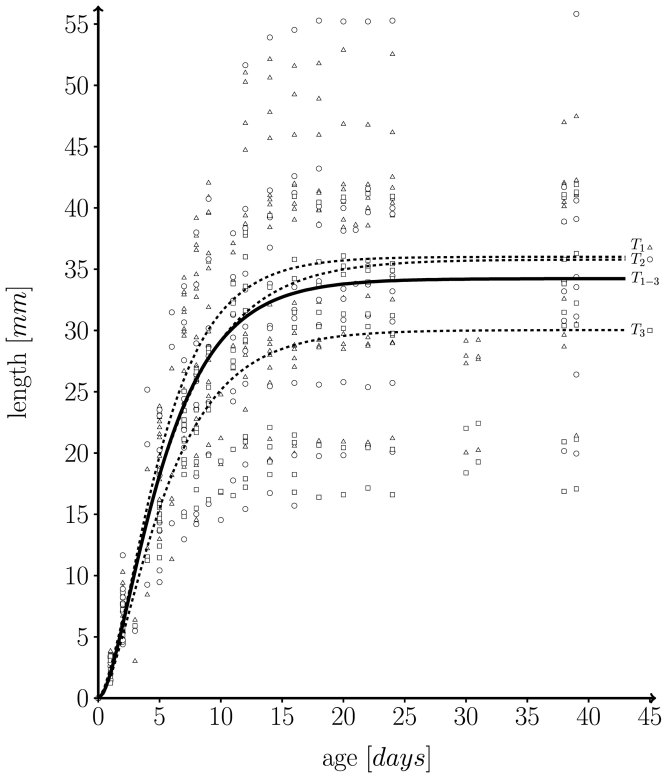


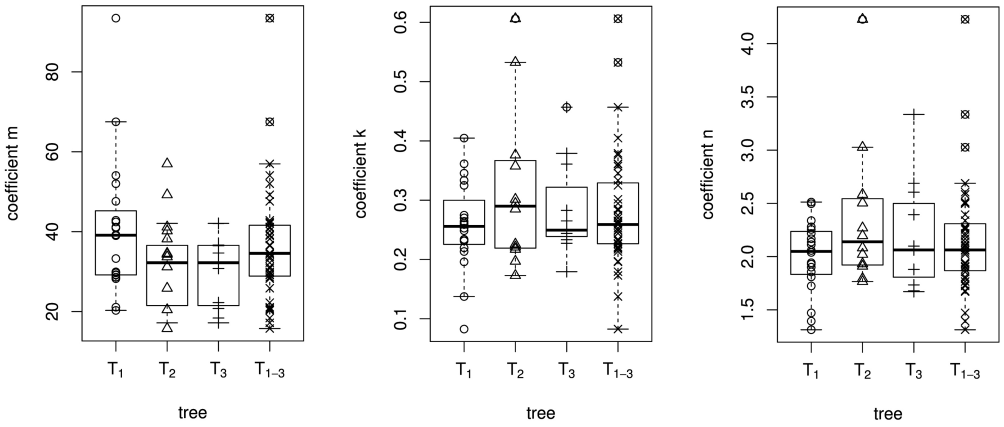
Fig. 5. Fitted length function, based on 582 measurements on a total of 43 individual leaves taken from three different trees (T_1=262, T_2=162, T_3=158 measurements) over one growth period. Solid line: fitted leaf length function over whole data set ($T_{1-3}=T_1 \cup T_2 \cup T_3$) (Eq. 1; m=34.224±0.595 (mean±SD), k=0.249±0.027, n=1.859±0.295, l_0=0); Dashed lines: fitted functions for each individual tree; T_1 (triangles), T_2 (dots) and T_3 (squares): measured leaf data.

ing. Size growth (Eq. 1) in time t [d] is modelled according to the Chapman-Richards growth function (Richards 1959), a standard growth function with parameters m, k and n determined by a least-squares fitting process.

$$leafLength(t) = m * \left(1 - \exp(-kt)\right)^n + l_0 \quad (1)$$

with l_0 being initial leaf length at t=0, and $m+l_0$ the maximum possible length which is approached asymptotically. In fact, the fitted growth functions represent individual temporal leaf kinetics rather well. Fig. 5 shows a function fitted to our whole data set.

The empirical distribution of coefficients m, k and n for the three trees is displayed in Fig. 6. The model was fitted for each leaf separately. It can be seen that the overall variation of coefficients within trees is rather large. As expected, the variation of the mean or median coefficients across trees is rather small (Fig. 5).



Coefficient m;
$m_{T1} = 40.4919 \pm 17.0568$,
$m_{T2} = 35.1405 \pm 11.4540$,
$m_{T3} = 30.3220 \pm 9.20957$,
$m_{T1-3} = 36.3969 \pm 14.2923$

Coefficient k;
$k_{T1} = 0.2574 \pm 0.0744$,
$k_{T2} = 0.3156 \pm 0.1347$,
$k_{T3} = 0.2843 \pm 0.0812$,
$k_{T1-3} = 0.2805 \pm 0.0971$

Coefficient n;
$n_{T1} = 1.9977 \pm 0.3362$,
$n_{T2} = 2.3595 \pm 0.6936$,
$n_{T3} = 2.1843 \pm 0.5224$,
$n_{T1-3} = 2.1464 \pm 0.5157$

Fig. 6. Overview of the size model coefficients' m, k and n in box and whisker plots (median, IQR, whiskers at 1.5 IQD or extremal values) broken down to the level of individual leaves (in total 43 leaves) and ordered by tree (T_1, T_2, T_3). The fourth column in each figure illustrates the combined results of all three trees taken together ($T_{1-3} = T_1 \cup T_2 \cup T_3$), see Table 1. Below we report the mean ± SD of the respective size model coefficients.

4.2 The proportional shape model

We begin our considerations for shape modelling with a minimal set of discriminative shape parameters (Fig. 7). Since leaf length accounts for size, clearly, maximal left and right leaf width b_l and b_r are the first descriptors for shape. Moreover, for meaningful shape discrimination the vertical locations l_{m_l} and l_{m_r} where the maximal left and right width is attained, respectively, seem essential.

For the leaves observed, the ratios of widths to length as well as of vertical locations to length were determined, for 83 leaves from one day of observation and for each tree separately over the growth period (see sub-figures 8a to 8c for some summaries).

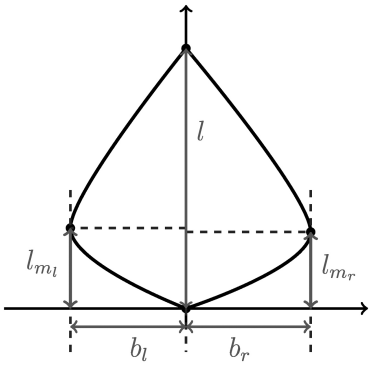
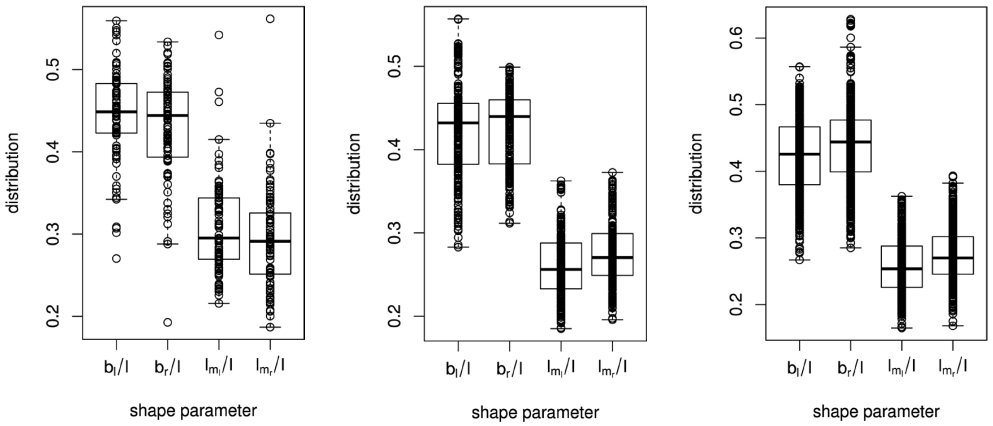


Fig. 7. Depicting the four shape parameters of the proportional model on a leaf contour. See list of symbols for further explanations.



a) 83 leaves from the three trees taken at the 103rd day of observation
$b_l/l = 0.4451 \pm 0.0583$,
$b_r/l = 0.4300 \pm 0.0619$,
$l_{m_l}/l = 0.3092 \pm 0.0581$,
$l_{m_r}/l = 0.2940 \pm 0.0605$

b) 262 leaves from one tree (T_1) taken over the whole period of observation
$b_l/l = 0.4229 \pm 0.0572$,
$b_r/l = 0.4254 \pm 0.0456$,
$l_{m_l}/l = 0.2608 \pm 0.0378$,
$l_{m_r}/l = 0.2761 \pm 0.0397$

c) 582 leaves from all three trees (T_{1-3}) taken over the whole period of observation
$b_l/l = 0.4226 \pm 0.0586$,
$b_r/l = 0.4391 \pm 0.0592$,
$l_{m_l}/l = 0.2584 \pm 0.0407$,
$l_{m_r}/l = 0.2752 \pm 0.0416$

Fig. 8. Distribution of shape parameters in box and whisker plots (median, IQR, whiskers at 1.5 IQD or extremal values) of a) 83 leaves measured on one single day, b) of one tree and c) of all three trees taken together over the whole period of observation. Below we report the mean±SD of the respective shape parameters. See list of symbols for further explanations.

Fig. 9 shows a linear regression of the shape variables to the size variable for both data sets, the one with 83 leaves from three trees taken at the 103rd day of observation (Fig. 9 a–d) and the one of 262 leaves from one tree (T_1) taken over the whole period of observation (Fig. 9 e–h). For all data sets we observed nearly constant shape parameters b_l/l and b_r/l (slightly positively correlated), as well as l_{m_l}/l and l_{m_r}/l (slightly negatively correlated) showing that the vertical position where the maximal width occurs moves with age from a position around one third of the total length towards one quarter of the leaf. In conclusion, we can say that length qualifies as a fairly good predictor of shape.

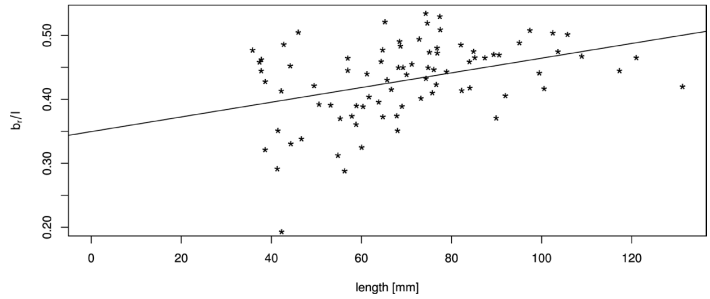
4.3 Contour models

4.3.1 Spline interpolation

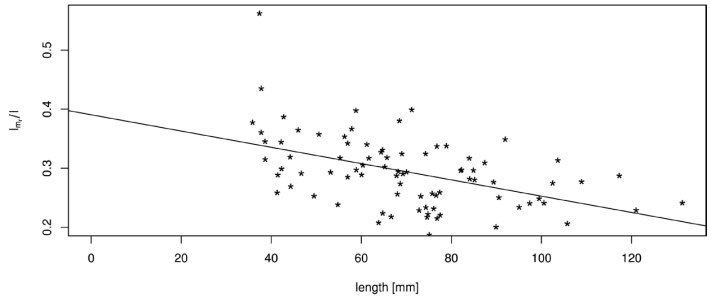
Here, we propose two models that allow reconstructing a realistic leaf contour from the four shape parameters and the single size parameter. In fact the first model uses only the four shape parameters of the proportional model; the second model learns its parameters from real contours.

We start with computing separate spline curves for the left and the right part of the leaf contour. More precisely, for each side, three control points S_0, S_1, S_2 are determined (Fig. 10; for the left side which was turned counter clockwise by 90°). Note that (b_l, l_{m_l}) and (b_r, l_{m_r}) are not extreme points of the resulting spline curve. In order to make them extreme we inserted an additional point between S_0 and S_1 (calculated as $S_{0b} = (l_{m_l}/2, 2b_l/3)$ for the left side and similarly for the right side), and possibly one more close to S_2, to sufficiently bend the curve downward to obtain a more globular bellied shape.

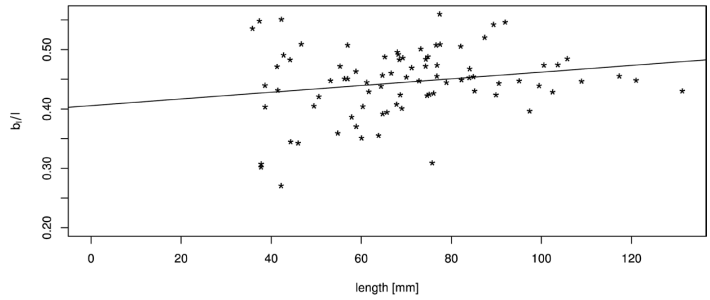
Alternatively, in order to guarantee that the points (b_l, l_{m_l}) and (b_r, l_{m_r}) were indeed extreme for the modelled leaf contour we calculated a parametric curve $C(s)$ (Eq. 4), the X and Y values of



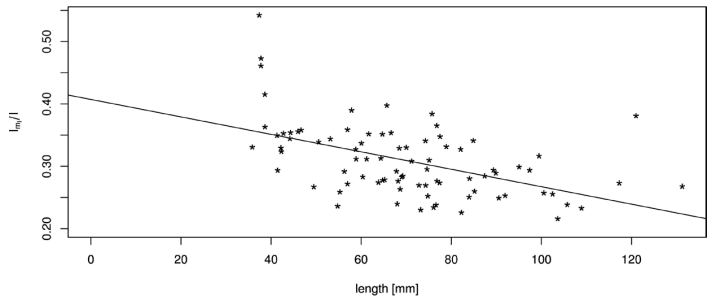
a) Relation b_r/l vs. l; $y=0.00115x+0.34959$, $r=0.3887$, $r^2=0.1511$



b) Relation l_{m_r}/l vs. l; $y=-0.00138x+0.39054$, $r=-0.4774$, $r^2=0.2279$

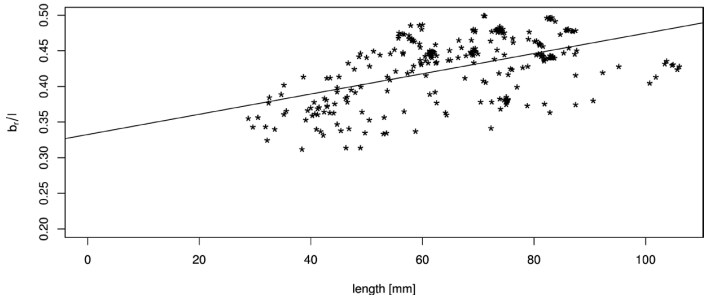


c) Relation b_l/l vs. l; $y=0.00056x+0.40557$, $r=0.2029$, $r^2=0.0412$

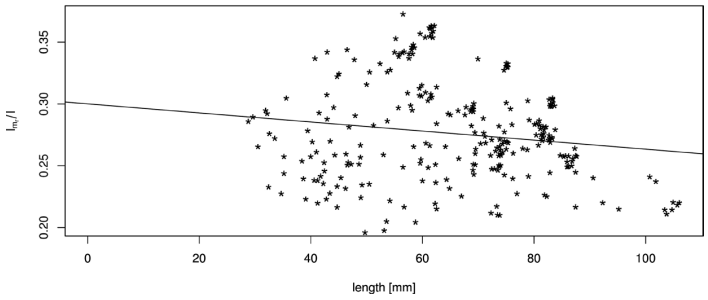


d) Relation l_{m_l}/l vs. l; $y=-0.00140x+0.40721$, $r=-0.5049$, $r^2=0.2549$

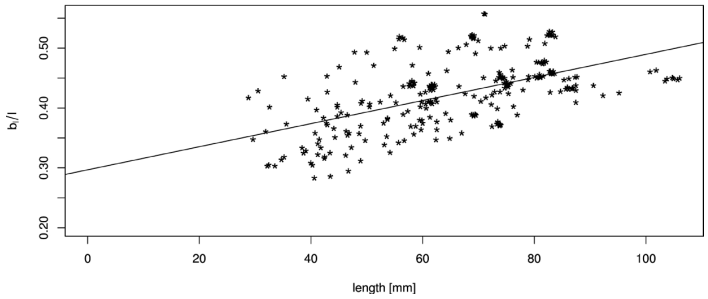
Fig. 9. Linear modelled relation between the model parameters b_l/l, b_r/l, l_{m_l}/l, l_{m_r}/l and the leaf length l of 83 leaves from three trees taken at the 103rd day of observation (sub-figures a–d) and of 262 leaves from one tree taken over the whole period of observation (sub-figures e–h). Correlation coefficient r and multiple R-squared r^2. See list of symbols for further explanations.



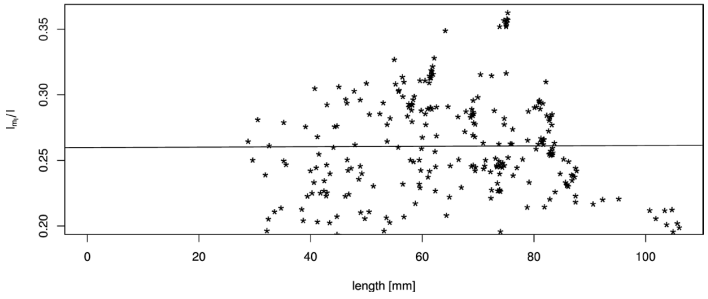
e) Relation b_r/l vs. l; $y=0.00142x+0.33246$, $r=0.5213$, $r^2=0.2717$



f) Relation l_{mr}/l vs. l; $y=-0.00037x+0.30009$, $r=-0.1543$, $r^2=0.0238$



g) Relation b_l/l vs. l; $y=0.00193x+0.29674$, $r=0.5629$, $r^2=0.3169$



h) Relation l_{ml}/l vs. l; $y=0.00001x+0.25980$, $r=0.0066$, $r^2=0.00004$

Fig. 9 continued.

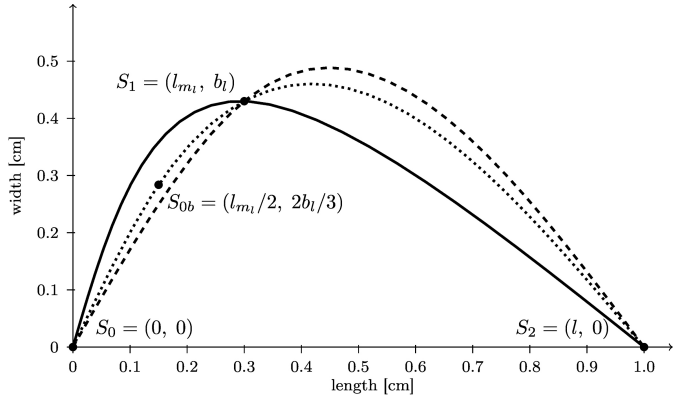


Fig. 10. Comparison of spline interpolation (dashed) between the three support points S_0, S_1, and S_2 which overshoots the maximum width with the bi-interpolation $C(s)$ (solid, black) interpolating splines for contour of the left side of the leaf. Additionally a spline interpolation between four support points (S_{0b}=(l_{m_l}/2, 2b_l/3)) is included (dotted).

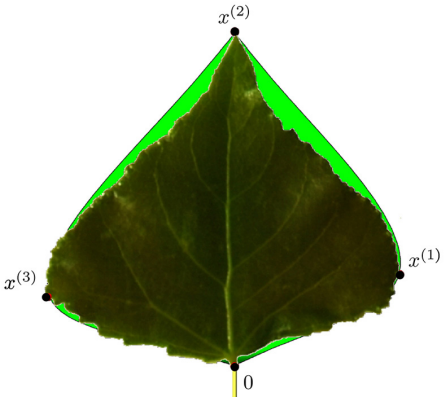


Fig. 11. Modelled contour approximation by bi-interpolation of the proportional model versus the original leaf contour. $x^{(i)}$ are the special contour points used in this model.

which are interpolated separately by two Hermite interpolations $sp1(s)$ and $sp2(s)$ between temporal supporting points T_{x0}, T_{x1}, T_{x2} used for $sp1(s)$ and T_{y0}, T_{y1}, T_{y2} for $sp2(s)$, with u_1 and u_2 which take for the left side of the leaf contour the form

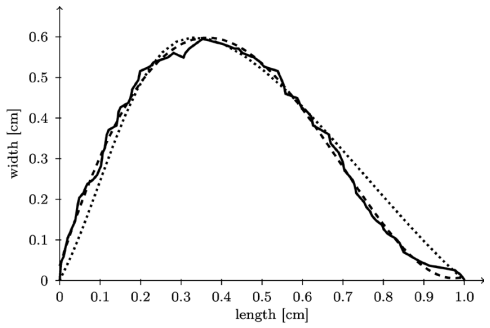
$$u_1 = \sqrt{S_{1_x} + S_{1_y}} = \sqrt{l_{m_r} + b_r} \tag{2}$$

$$u_2 = \sqrt{\left(S_{2_y} + S_{1_y}\right)^2 + \left(S_{2_x} + S_{1_x}\right)^2} = u_1 + \sqrt{(-b_r)^2 + \left(l - l_{m_r}\right)^2} \tag{3}$$

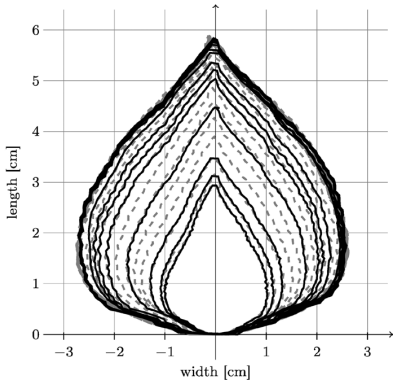
The final curve C is defined by Eq. (4), with $s \in [0,u_2]$

$$C(s) = \left(sp1(s), sp2(s)\right) \tag{4}$$

Similarly, we proceed for the right side of the leaf contour. Fig. 11 depicts a typical leaf reconstruction by the proposed bi-interpolation of the proportional model.



a) Comparison of the contour models with original (solid black): bi-interpolation based on the proportional model (dotted), polynomial (dashed).



b) Modelling by means of fifth order polynomials (dotted) vs. original (solid). The original leaf contour growth was recorded at 14 days of age; the corresponding polynomials were computed in daily intervals up to the age of 24 days. Eq. 1 was used as size model to determine the length of the leaf at time *t*.

Fig. 12. Reconstruction of a leaf contour over time and comparison of the contour models with original model.

4.3.2 A general polynomial fit

Once again, we can take advantage of the fact that for the poplar leaves investigated we may model each leaf side (left and right) separately. Moreover they exhibit the favourable property that each half-contour (left and right) can be viewed as a function of vertical height. We now model each half-contour separately by a fifth degree polynomial (Eq. 5) by pointwise fitting (Fig. 12a).

$$leafShape(x) = \sum_{i=0}^{5} c_i x^i \tag{5}$$

For every single leaf of a time series, meaning for every point in time $t_i, i \in [0,n]$, when the leaf image was captured, a tuple of coefficients $ct_{t_i} = (c_0, c_1, c_2, c_3, c_4, c_5)$ is fitted via least squares to the model (Eq. 5). The $n+1$ coefficient tuples are combined to a coefficient matrix M:

$$M = \begin{pmatrix} c0_{t_0} & c1_{t_0} & c2_{t_0} & c3_{t_0} & c4_{t_0} & c5_{t_0} \\ \vdots & & & & & \vdots \\ c0_{t_n} & c1_{t_n} & c2_{t_n} & c3_{t_n} & c4_{t_n} & c5_{t_n} \end{pmatrix} \tag{6}$$

In order to determine the coefficients at times between the moments when photos were taken, the coefficients of the same degree are spline interpolated.

As a further, optional enhancement step to model contours more realistically, another function is added to the leaf function, so that the sum will produce a slightly serrated edge, which matches the specific shape of poplar leaves quite well. As such a Fourier-series approximation of the so called saw-tooth function (Eq. 7) is used:

$$sawToothApprox(x) = b\left(2\sin(ax) - \sin(2ax) + 2/3\sin(3ax) - 1/2\sin(4ax)\right) \tag{7}$$

At this stage we use a generic function for all leaves of a species where the parameters a and b have been obtained from a fit to one representative leaf. Such a supplement might be seen at first

Table 2. Overview of all models with number of input parameters corresponding to their dimensionality.

Model	Input parameters		
	General	Symmetric	General temporal evolution
Proportional	1+2*2 = **5**	1+2 = **3**	1+2*4 = **9**
Polynomial	1+2*6 = **13**	1+6 = **7**	1+2*6*7 = **85**
Geodesic	1+2*2 = **5**	-	1+4+4 = **9**

sight only as an 'aesthetic correction' increasing computation time, as in the case of poplar leaves. However, applied to other species with more serrated leaf contours it will become significant for more realistic light interception and self-shading effects. Notably, the proposed saw-tooth function is a parsimonious approximation increasing the level of realism.

Fig. 12b depicts a typical reconstruction of a leaf contour over time.

In a generalising step the model's six coefficients were estimated for every observation of the entire data set of the 262 leaves used in Fig. 8b. For every coefficient, we interpolated dependencies on leaf length by a sixth order polynomial.

4.4 Comparison of contour models

In Fig. 12a we compare contours obtained by spline interpolation based on the proportional model and by polynomial fit based on length only with one another. Obviously the essentially eleven-dimensional (cf. Table 2) polynomial contour reconstruction performs rather satisfactorily. The difference between the two shape models and the original leaf contour (Fig. 13) shows that the polynomial model is underestimating the original leaf contour while the bi-interpolated model is overestimating it slightly. Though the proportional model comes with five parameters (Table 2) it performs rather well.

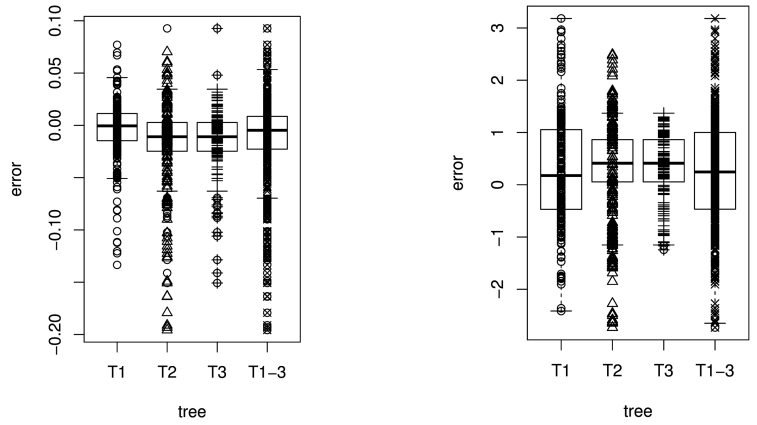
Recall that there is no fine-structure biological model underlying our approaches. Rather it can be said that in both approaches, an appropriate model emerges with a set of given parameters. In particular for the low-dimensional proportional model, these parameters have direct geometric meanings.

4.5 Geodesic shape interpolation

The geometric descriptors of shape and size defining a non-symmetrical *proportional model* as developed in Section 4.3 are identified as 3 two-dimensional *landmarks*:

$$x^{(1)} = \begin{pmatrix} b_r \\ l_{m_r} \end{pmatrix}, x^{(2)} = \begin{pmatrix} 0 \\ l \end{pmatrix}, x^{(3)} = \begin{pmatrix} b_l \\ l_{m_l} \end{pmatrix}. \quad (8)$$

Every such 3-landmark configuration $x=(x^{(1)}, x^{(2)}, x^{(3)})$ is then viewed as a matrix $x/\|x\|$ in the pre *shape space* $S^5 \subseteq R^{2\times 3}$ which carries the canonical structure of a non-flat 5-dimensional unit-sphere (Hotz et al. 2010). Within this setup, in order to model growth over one-dimensional time, most parsimonious one-dimensional data descriptors are sought for. Obviously *geodesics*, i.e. great circles on S^5 naturally qualify for this task, as they are generalizations of straight lines to a non-flat structure. A unit-speed geodesic is uniquely determined by initial offset $x_0 \in S^5$ and initial velocity $v_0 \in S^5$ orthogonal to x_0. Our *geodesic model* thus is specified as



a) Polynomial fitting
$T_1 = -0.0484 \pm 0.0291$,
$T_2 = -0.0219 \pm 0.0512$,
$T_3 = -0.0143 \pm 0.0327$,
$T_{1-3} = -0.0122 \pm 0.0381$

b) Bi-interpolated spline
$T_1 = 0.2563 \pm 1.0849$,
$T_2 = 0.0760 \pm 1.2358$,
$T_3 = 0.3208 \pm 0.6483$,
$T_{1-3} = 0.2229 \pm 1.0378$

Fig. 13. Error plots [mm^2] in box and whisker plots (median, IQR, whiskers at 1.5 IQD or extremal values) of the polynomial and the bi-interpolated spline shape model vs. the original leaf contour of the left leaf side for all individual trees and the average over all trees. The error is calculated as difference between the model and the original leaf contour. The model parameters are individually adapted for each leaf. Below we report the mean ± SD of the respective errors. See list of symbols for further explanations.

$$x^{(t)} = \lambda(t)\gamma_{x_0,v_0}(\tau(t)) \tag{9}$$

where $x^{(t)}$ is the modelled configuration landmark matrix at time t, $\lambda(t) > 0$ conveys size, $\gamma_{x_0,v_0}(\tau) = x_0 \cos(\tau) + v_0 \sin(\tau)$ is a great circular geodesic with initial velocity $\frac{d}{d\tau}\gamma_{x_0,v_0}(\tau(t))$ at $\gamma_{x_0,v_0}(0) = x_0$ and $\tau(t)$ relates real time with the speed of the geodesic $\left\|\frac{d}{dt}\gamma_{x_0,v_0}(\tau(t))\right\| = \left|\dot{\tau}(t)\right|$.

As has been shown previously in Hotz et al. (2010), shapes of poplar leaves during growth closely follow geodesics in shape space. As explained below, the proportional model and the geodesic model can be used in conjunction to model rather diversified growth and development.

4.6 Dimensionality of shape and size models

All models introduced here describe leaf form by using a tuple of a one-dimensional *size* and a multidimensional *shape*, as is common for biological objects. If we are to model temporal evolution then we may model size as a function of time given by (Eq. 1).

Since in the proportional model, b_l/l and b_r/l are nearly constant in size and l_{m_l}/l and l_{m_r}/l depend rather linearly on size (cf. Section 4.2), for temporal evolution in the proportional model, no new parameter is introduced (of course, using the linear evolution instead of a constant introduces two new parameters (slopes)).

In the polynomial model it turned out that the dependence on length of each of the six coefficients can be well modelled by a sixth order polynomial (cf. Section 4.3.2). In consequence, for temporal evolution, for each leaf side's shape we have seven parameters.

Although the geodesic model allows for less richness in the modelling of temporal evolution compared to the polynomial model, as with the proportional model, the parameters now have a geometrical meaning: Through every initial point x_0 determined by the four parameters of the proportional model (and the current size), there is a four-dimensional space (the tangent space of S^5) of possible geodesic directions through x_0. Every such direction can be thought of as an individual *shape-growing plan* of the individual leaf. Indeed, as shown in Hotz et al. (2010), shapes of poplar leaves tend to follow such geodesics rather closely.

The essence of the above considerations is condensed in Table 2. We have included a column for accordingly simplified *symmetric models* assuming that the right and left leaf side are mirrored.

As explained, all the proposed models have their specific advantages and disadvantages. They differ in number of parameters and the possibility of modelling leaf development over time. Therefore, a final recommendation of the most appropriate model to use in FSPM cannot be made, as it always depends on the specific requirements of the model. The integration of the geodesic as well as the polynomial model would be of great interest for a dynamic FSPM because they allow to model leaf growth over time. In particular the geodesic model is promising as it only requires 9 parameters while the polynomial one needs 85. With respect to computation time the geodesic model is clearly preferable because of fewer parameters. For a static shape model the proportional shape model with only two ratios will be nearly unrivaled.

5 Implementing leaf models as organ modules

5.1 Modelling software and language

For the implementation of our models we used the modelling software GroIMP (Kurth et al. 2006; Kurth 2007; GroIMP Developer Group 2013), with the integrated language XL (Kniemeyer 2004, 2008). This technique can be used for plant modelling in terms of organ modules, where each type of plant organ is implemented as a separate module. The organ modules are reusable program parts that permit a flexible use in different models and rid the modeller of repeatedly having to work out basic aspects such as geometry. These organ modules can be seen as predefined, ready-to-use components which only need to be parameterized appropriately. They lighten the load of low level programming work and thus permit to focus on modelling rather than on coding. Currently two models are available for use in XL in different functional-structural models, a static and a dynamic version.

5.2 Static version

The static leaf model has 10 to 18 input parameters which can be set by the user. First, the *tree number* has to be chosen to specify which data set is to be used as a base. Then the parameters for the size model (Eq. 1) according to the empirical distribution of Fig. 5 are selected (i). With *leaf age* as further input (ii) the size of the generated leaf is calculated. To estimate the shape (iii) either the polynomial (Eq. 5) or the proportional model (Sec. 4.2) are chosen. As a last parameter a flag *useSawtooth* indicates if the additional optical correction is turned on or not. Finally (iv) the contour is obtained according to Hermite or other polynomial interpolation as described above.

The used leaf profile and the trajectory that the leaf vein is following are currently constants but could easily be turned into stochastic functions. The extension of a single leaf (Fig. 14a) depicts as solid black line in the middle the course of the major axis (trajectory) along which the (horizontal) profile is shifted. This will turn the present 2D shape into a curved and bent shape in 3D. In GroIMP e.g. NURBS shapes can be used to visualise the described structure.

5.3 Dynamic version

For the dynamic version time evolution is included. We proceed as for the static version and adapt (iii) to either of:

a) pick initial shape and terminal shape parameters from the distribution underlying the proportional model and compute the corresponding geodesic in shape space;
b) pick initial shape parameters from the distribution underlying the proportional model and pick a geodesic (future work);
c) pick from a distribution underlying the interpolating functions for each parameter of the polynomial model.

Without going into details of the implementation in XL, this leaf module allows to estimate leaf growth over time in an elegant way by simply calling an *update()* function at each growth step. Internally the age will be increased, other parameters updated and the shape recalculated. For the functional part processes like photosynthesis or transpiration can be run and used to estimate the source and sink behaviour of each leaf independently. For a detailed description of the usage of XL see Kniemeyer (2004, 2008).

5.4 Illustration

To illustrate the possible applications two examples are given: The first is a static structural model (Fig. 14a) of a young poplar tree with a detailed enlargement of a single branch and a single leaf, for which the leaves were produced by the new leaf organ module. To obtain a more realistic 3D impression of the leaf shape a slightly curved bimodal profile was added following the main vein which was modelled as slightly bent. These profiles are based on empirical observation.

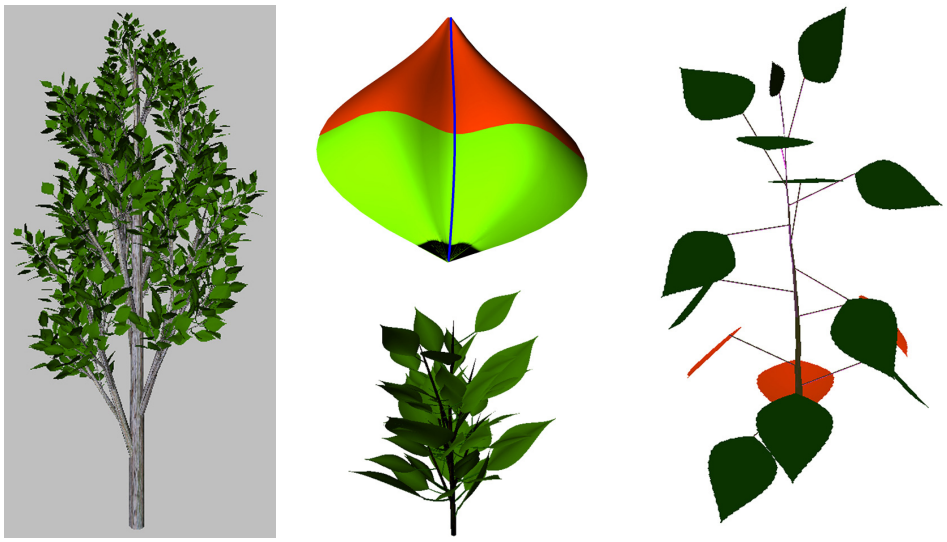
A second example (Fig. 14b), based on the model of a young, unbranched poplar (Buck-Sorlin et al. 2008), demonstrates the use of the leaf modules in a more complex functional-structural model, where an accurate leaf area surface is required to calculate the exact amount of light reaching the leaf surface and to estimate the quantity of produced assimilates. The leaf shape was automatically calculated from the age of each leaf.

6 Conclusions

In this paper a procedure for a non-destructive digitization was developed and demonstrated, using the example of *Populus* x *canadensis* leaves. The equipment presented is well suitable for field use, reaching its limits, however, when the investigated objects become too small. Otherwise, there are few restrictions and the methodology can be used for many other species without modification.

A stratified process of image processing was used to extract the contours of the digitized leaves. The image processing tool developed for this purpose, implemented as ImageJ macro, is a semi-automatic tool for contour extraction of leaves of a wide range of broad-leaved plants. Several models were developed based on the data that were collected during one growth cycle. These models are adapted to black poplar and accordingly fitted, but they can be used also for other species with similar leaf shapes, with only a few small changes. Model comparison and validation showed that leaf shape can be well modelled with a small set of parameters. Finally the models were implemented as GroIMP organ modules that can be used in different models as a component.

Providing such modularized systems of predefined leaf organs represents a first step toward a more user-friendly modelling workflow by ridding the modeller from low-level programming



a) Static GroIMP tree model including the new 3D leaf organ module with enlarged details of a branch and a single leaf; emphasis is on leaf profile and the main leaf-vein following a defined trajectory.

b) Generated 3D structure of a dynamic functional-structural plant model of a young poplar by Buck-Sorlin et al. (2008); enhanced by our leaf organ module.

Fig. 14. Two examples integrating the proposed leaf organ module.

work. It thus allows focusing on modelling rather than on coding. As reusable program parts, these organ modules will be included into the library of components of a component-based framework that we are currently working on.

One limitation of the non-destructive method of data acquisition introduced here is that it is only applicable to flat leaves (or those that can be easily flattened), which definitely excludes some species. However, it can be well employed for several other species, including many deciduous trees and crops.

Future work comprises linking the model to environmental parameters like temperature, light quantity and quality, and water. Furthermore, it is envisaged to model leaf damage due to insects or wind.

Acknowledgements

This research was funded by the German Research Foundation (DFG) under project identifier SL 11/12-1. Student assistants Philipp Wree, Julia Rudolph and Julia Knieriem gave support in data acquisition and extraction. S.H. gratefully acknowledges DFG Grant Hu 1575/2. The Department of Tree Physiology and Forest Botany of the University of Göttingen, led by Andrea Polle, provided the sample plants. All assistances are gratefully acknowledged.

References

Biological Modeling and Visualization research group. (2013). The virtual laboratory. [Internet site]. Available at: http://algorithmicbotany.org/virtual_laboratory. [Cited 15 Feb 2014].

Buck-Sorlin G.H., Kniemeyer O., Kurth W. (2008). A model of poplar (*Populus* sp.) physiology and morphology based on relational growth grammars. In: Deutsch A., Parra R.B. d. l. , Boer R.J. d., Diekmann O., Jagers P., Kisdi E., Kretzschmar M., Lansky P., Metz H. (eds.). Mathematical modeling of biological systems, Volume II. Modeling and Simulation in Science, Engineering and Technology. Birkhäuser, Boston. p. 313–322. http://dx.doi.org/10.1007/978-0-8176-4556-4_28.

Chen C.-C., Chen H., Chen Y.-R. (2009). A new method to measure leaf age: leaf measuring-interval index. American Journal of Botany 96(7): 1313–1318. http://dx.doi.org/10.3732/ajb.0800303.

Chi Y.T., Chien C.F., Lin T.T. (2003). Leaf shape modeling and analysis using geometric descriptors derived from Bezier curves. Transactions of the ASAE 46(1): 175–185.

Chien C.F., Lin T.T. (2005). Non-destructive growth measurement of selected vegetable seedlings using orthogonal images. Transactions of the ASAE 48(5): 1953–1961.

Dennett M.D., Auld B.A., Elston J. (1978). A description of leaf growth in *Vicia faba* L. Annals of Botany 42(1): 223–232.

De Reffye P., Fourcaud T., Blaise F., Barthelemy D., Houllier F. (1997). A functional model of tree growth and tree architecture. Silva Fennica 31(3): 297–311.

Dornbusch T., Andrieu B. Jan. (2010). Lamina2Shape – an image processing tool for an explicit description of lamina shape tested on winter wheat (*Triticum aestivum* L.) Computers and Electronics in Agriculture 70(1): 217–224. http://dx.doi.org/10.1016/j.compag.2009.10.009.

Dornbusch T., Watt J., Baccar R., Fournier C., Andrieu B. (2011). A comparative analysis of leaf shape of wheat, barley and maize using an empirical shape model. Annals of Botany 107(5): 865–873. http://dx.doi.org/10.1093/aob/mcq181.

Dryden I.L., Mardia K.V. (1998). Statistical shape analysis. Wiley, Chichester. ISBN 0-4719-5816-1.

Eschenbach C. (2000). The effect of light acclimation of single leaves on whole tree growth and competition – an application of the tree growth model ALMIS. Annals of Forest Science 57(5): 599–609. http://dx.doi.org/10.1051/forest:2000145.

Gallagher J.N. (1979). Field studies of cereal leaf growth. Journal of Experimental Botany 30(4): 625–636. http://dx.doi.org/10.1093/jxb/30.4.625.

Gregory F.G. (1921). Studies in the energy relations of plants. I. The increase in area of leaves and leaf surface of *Cucumis sativus*. Annals of Botany 35(1): 93–123.

Griffon S., de Coligny F. (2012). AMAPstudio – a software suite for plants architecture modelling. [Internet site]. Available at: http://amapstudio.cirad.fr/doku.php?id=start. [Cited 15 Feb 2014].

GroIMP Developer Group. (2013). Web page of GroIMP. [Internet site]. Available at: http://www.grogra.de. [Cited 15 Feb 2014].

Hackett C., Rawson H. (1974). An exploration of the carbon economy of the tobacco plant. II. Patterns of leaf growth and dry matter partitioning. Australian Journal of Plant Physiology 1: 271–281. http://dx.doi.org/10.1071/PP9740271.

Hotz T., Huckemann S., Gaffrey D., Munk A., Sloboda B. (2010). Shape spaces for pre-aligned star-shaped objects in studying the growth of plants. Journal of the Royal Statistical Society, Series C 59(1): 127–143.

Hu B.G., de Reffye P., Zhao X., Yan H.P., Kang M.Z. (2003). GreenLab: a new methodology towards plant functional-structural model – structural aspect. In: Hu B.G., Jaeger M. (eds.). International Symposium on Plant Growth Modeling, Simulation, Visualization and Applications [PMA03], 2003, Beijing (China PRC), 13–16 October 2003. Tsinghua University Press

and Springer, Beijing. p. 21–35.
ImageJ Developer Group. (2013). ImageJ. [Internet site]. Available at: http://rsbweb.nih.gov/ij/index.html. [Cited 15 Feb 2014].
Kniemeyer O. (2004). Rule-based modelling with the XL/GroIMP software. In: Schaub H., Detje F., Brüggemann U. (eds.).The logic of artificial life. Proceedings of 6th GWAL. AKA Akademische Verlagsges Berlin. p. 56–65.
Kniemeyer O. (2008). Design and implementation of a graph grammar based language for functional-structural plant modelling. PhD thesis. BTU Cottbus.
Kniemeyer O., Buck-Sorlin G.H., Kurth W. (2007). GroIMP as a platform for functional-structural modelling of plants. In: Vos J., Marcelis L.F.M., deVisser P.H.B, Struik P.C., Evers J.B. (eds.). Functional-structural plant modelling in crop production. Proceedings of a workshop held in Wageningen (NL), 5.–8. 3. 2006. Springer, Dordrecht. p. 43–52.
Kurth W. (2007). Specification of morphological models with L-systems and relational growth grammars. Image – Journal of Interdisciplinary Image Science 5(1): 50–79.
Kurth W., Buck-Sorlin G.H., Kniemeyer O. (2006). Relationale Wachstumsgrammatiken: Ein Formalismus zur Spezifikation multiskalierter Struktur-Funktions-Modelle von Pflanzen. Modellierung pflanzlicher Systeme aus historischer und aktueller Sicht. Symposium zu Ehren von Prof. Dr. Dr. h.c. E.A. Mitscherlich. Schriftenreihe des Landesamtes für Verbraucherschutz, Landwirtschaft und Flurneuordnung Brandenburg Reihe Landwirtschaft Band 7(1): 36–45.
Lecoustre R., de Reffye P., Jaeger M., Dinouard P. (1992). Controlling the architectural geometry of a plant's growth – application to the *Begonia* genus. In: Magnenat Thalmann N., Thalmann D. (eds.). Creating and animating the virtual world. Springer-Verlag New York Inc., New York. p. 199–214.
Maksymowych R. (1973). Analysis of leaf development. Cambridge University Press. ISBN 0-5212-0017-2.
Mosimann J.E. (1970). Size allometry: size and shape variables with characterizations of the log-normal and generalized gamma distributions. Journal of the American Statistical Association 65(330): 930–945.
Neto J.C., Meyer G.E., Jones D.D., Samal A.K. (2006). Plant species identification using elliptic Fourier leaf shape analysis. Computers and Electronics in Agriculture 50(2): 121–134. http://dx.doi.org/10.1016/j.compag.2005.09.004.
Peacock J.M. (1975). Temperature and leaf growth in *Lolium perenne*. I. The thermal microclimate: its measurement and relation to crop growth. Journal of Applied Ecology 12(1): 99–114.
Perttunen J., Sievänen R., Nikinmaa E., Salminen H., Saarenmaa H., Väkevä J. (1996). LIGNUM: a tree model based on simple structural units. Annals of Botany 77(1): 87–98. http://dx.doi.org/10.1006/anbo.1996.0011.
Prusinkiewicz P., Remphrey W.R., Davidson C.G., Hammel M.S. (1994). Modelling the architecture of expanding *Fraxinus pennsylvanica* shoots using L-systems. Canadian Journal of Botany 72(5): 701–714. http://dx.doi.org/10.1139/b94-091.
Prusinkiewicz P., Lindenmayer A. (1990). The algorithmic beauty of plants. Springer, New York. ISBN 0-3879-7297-8.
Reich A., Holbrook N.M., Ewel J.J. (2004). Developmental and physiological correlates of leaf size in *Hyeronima alchorneoides* (Euphorbiaceae). American Journal of Botany 91(4): 582–589. http://dx.doi.org/10.3732/ajb.91.4.582.
Richards F. (1959). A flexible growth function for empirical use. Journal of Experimental Botany 10(2): 290–300. http://dx.doi.org/10.1093/jxb/10.2.290.
Richards F. (1969). The quantitative analysis of growth. In: Steward F.C. (ed.). Plant physiology: a treatise. Analysis of growth. Vol. 5. Series: A. Behaviour of Plants and their Organs. Academic

Press, New York. p. 3–76.

Routhier M.-C., Lapointe L. (2002). Impact of tree leaf phenology on growth rates and reproduction in the spring flowering species *Trillium erectum* (Liliaceae). American Journal of Botany 89(3): 500–505. http://dx.doi.org/10.3732/ajb.89.3.500.

R-Project Developer Group. (2013). The R project for statistical computing. [Internet site]. Available at: http://www.r-project.org. [Cited 15 Feb 2014].

Ruiz-Ramos M., Mínguez M.I. (2006). ALAMEDA, a structural-functional model for Faba Bean crops: morphological parameterization and verification. Annals of Botany 97(3): 377–388. http://dx.doi.org/10.1093/aob/mcj048.

Wiechers D., Kahlen K., Stützel H. (2011). Evaluation of a radiosity based light model for greenhouse cucumber canopies. Agricultural and Forest Meteorology 151(7): 906–915. http://dx.doi.org/10.1016/j.agrformet.2011.02.016.

Yin X., Goudriaan J., Lantinga E.A., Vos J., Spiertz, H .J. (2003). A flexible sigmoid function of determinate growth. Annals of Botany 91(3): 361–371. http://dx.doi.org/10.1093/aob/mcg029.

.

Total of 40 references

CHAPTER 4

Second Paper

This paper is published as:
Henke M, Sloboda B (2014) Semiautomatic tree ring segmentation using Active Contours and an optimised gradient operator, *Forestry Journal*, 60(3)

Authorship

- Branislav Sloboda supported the writing of the manuscript.

Lesn. Cas. For. J. 60 (2014) 185–190

REFERÁT – DISCUSSION PAPER

Lesnícky časopis - Forestry Journal
http://www.nlcsk.sk/fj/

Semiautomatic tree ring segmentation using Active Contours and an optimised gradient operator

Poloautomatizovaná segmentácia ročných radiálnych kruhov stromov s využitím metódy „Active Contours“ a optimalizovaného gradientového operátora

Michael Henke*, Branislav Sloboda

Department Ecoinformatics, Biometrics and Forest Growth, University of Göttingen, Büsgenweg 4, 37077 Göttingen, Germany

Abstract
This paper presents an easy and effective method to extract tree rings completely from images of tree discs independent of their source. The method uses Active Contours, often used in medical image processing to detect organs, in combination with an optimised image filter based on the Sobel operator to automatically outline the tree rings. Special attention is given to eliminate critical physical irregularities caused by branches, cracks or colourisations. The work resulted in the implementation of a platform independent, free and open source software solution for semiautomatic tree ring segmentation. Comparison to manual measurements shows that the system is dependable and the results are reproducible. The system has been applied to several conifer species.
Keywords: stem disc analysis; filter optimisation; segmentation tool; image analysis

Abstrakt
Článok prezentuje jednoduchú a efektívnu metódu pre extrakciu radiálnych ročných prírastkov z obrázkov získaných skenovaním diskov odobratých zo stromov. Metóda využíva techniku „Active Countours“, ktorá je často používaná pri spracovaní obrazu pre detekciu orgánov, v kombinácii s optimalizovaným obrazovým filtrom založeným na „Sobel“ operátore pre automatizované zvýraznenie ročných kruhov. Špeciálna pozornosť sa venuje eliminácii kritických fyzikálnych nepravidelností spôsobených vetvami, puklinami alebo zafarbením. Výsledkom práce je voľne dostupný softvér pre poloautomatizovanú segmentáciu ročných radiálnych prírastkov na odobratých diskoch. Porovnanie s manuálnym meraním ukázalo, že systém je spoľahlivý a výsledky sú reprodukovateľné. Systém bol preskúšaný na niekoľkých ihličnatých drevinách.
Keywords: analýza kmeňových diskov; filtrová optimalizácia; nástroj pre segmentáciu; analýza obrazu

1. Introduction

Tree rings can tell us a lot about climate variations over the last several thousand years. Generally, the precise determination of tree rings of a tree disc is an arduous and time-expensive matter. Therefore, one is in most cases satisfied with measuring the radii of the annual rings along single rays, e.g. obtained by a core sample. This is often done in a computer-assisted way and with the aid of a macroscope (Taube & Sloboda 1992). Other external systems like measuring stages, encoders, and readout units enable linear encoding of measurements. Such systems provided, e.g., by Metronics, Boecker boxes, Acu-Rite or Measucron are directly supported by commercial software systems like MeasureJ2X. For a great number of applications this restriction can lead to acceptable solutions, for example in the case of dendrochronological investigations where a destructive treatment of historical samples has to be avoided as far as possible. Later developments can process high-resolution images of tree discs or core samples. Commonly used, commercial representatives are WinDENDRO (Guay 2012) and LIGNOVISION™ (RINNTECH e.K.).

However, if one is interested in precise statements about stem growth, e.g., to explore how a predominating wind direction affects the stem or how a slope situation is balanced by asymmetric growth, a complete extraction of all tree rings can hardly be avoided. Therefore, digitizing and processing of whole tree discs are required. Image analysis techniques have been applied already in several systems like TRESS (Conner 1999, Gopalan 2000). A watershed based transformation in combination with other morphological operations for measuring the areas of annual tree rings was introduced by Soille and Mission (2001). Zhou et al. (2012) presented another method based on watershed segmentation to detect and count tree rings. Norell (2011) used image filter based methods to analyse wood quality by counting the number of annual rings.

Since stem positions are fixed to the ground, asymmetric tree growth is an adaption in response to environmental conditions, e.g., plant density what influences light conditions or topo-graphical site conditions. The question of which influencing factor most prominently define such differences in growth is not trivial. Field experiments were already done by Gaffrey (2004) to investigate how influences of the elastomechanical behaviour of the stem, as well as the distribution of the assimilate crown production will affect the growth behaviour. The aim was to analyse, and then describe and model the expected resulting change in stem growth. In

*Corresponding author. Michael Henke, e-mail address: mhenke@uni-goettingen.de, phone: +49 (0)551 39 12109

functional-structural plant modelling of trees the focus also lies to physiological processes, because they are the driving force that determine growth and shape development of the stem in order to guarantee both mechanical stability and sufficient water supply to the foliage.

Calculations of ring area and the annual increment and furthermore estimations of volume and volume increment, what is possible when several sample disks are taken at different sections of the tree, are important information not only from a modelling or forest economic point of view.

2. Material and methods

The method which will be presented here and which was implemented in a software tool enables the complete extraction of tree ring information on the basis of images of tree discs. Data can be retrieved step by step from the disc to determine the precise dimensions of the annual rings.

In order to extract the tree rings from image data, well-known methods from image processing are used. First, the image will be improved qualitatively, then the proper extraction of the rings is carried out by means of Active Contours (Kass & Witkin & Terzopoulos 1987). This image processing method of course gives the best results if the tree rings are already well visible and distinguishable in the original sample. A (preferentially sharp) colour gradient between subsequent rings, however, does not occur in all tree species. In fact, in most deciduous tree species there is frequently the situation that some tree rings can hardly be identified. Thus our method is preferentially to be used for conifer species.

2.1. Description of the procedure

The primal material for our system are pictures of cross sections of a tree, in short tree discs, independent of which source they are origin, e.g., X-ray, photographs or scanned. The surface of the disc should be smooth and free of unevenness. Therefore, the discs should be planed or sanded using a grid 200 or higher in order to remove the damages caused by sawing. For pictures taken by a camera you need to make sure that the object plane is parallel to the camera in order to obtain an undistorted image.

Preparation of the base material is as important as image pre-processing including blurring and noise reduction can be to produce better results. A set of basic operations are also implemented for this reason in our software.

Besides quality of the image material, resolution of the image (dot density: number of individual dots per inch – dpi) plays a major role. This density directly limits the amount of informations that can be stored within an inch of an image and so it limits the number of rings and the minimal ring width that can be resolved. In order to distinguish more rings per inch a high resolution is recommended. At a density of 300 dots per inch 1 millimetre corresponds to 11.81 pixels.

A further problem that cannot be solved in such a simple way is a too weak contrast between early and late wood, which occurs frequently in some tree species. If there is the possibility before the images are taken, it is recommended to pre-apply colouring to the tree discs with specialised indicator colours, e.g., with a solution of hydrochloric acid and phloroglucinol, which colourises latewood darker than earlywood because of the higher concentration of Lignin. For coloured images it can be an advantage to split the original image into RGB channels and to use only the channel with the highest contrast for further analysis. Likewise, wood discolorations caused by fungal infection, e.g., by the Blue Stain, can cause errors in ring boundary detection. Besides colour and contrast problems, physical irregularities like branches or cracks in the wood represent further challenges. Another nontrivial problem comes from rings that are located so narrowly next to each other that with unaided eye their course is hardly tractable. In order to get as few problems as possible and thus to avoid time-expensive manual post-processing of the images, already in the phase of sample selection there should be paid attention to choose discs as immaculate as possible.

The described problems require new procedures for the semi-automatic tree ring extraction which exceed the standard image processing operations since the latter fail in difficult situations.

For the improvement of picture quality it is absolutely helpful to enhance the images using appropriate software in order to improve contrast or brightness before the extraction is carried out.

The extraction is based on the so-called edge image which is generated by means of gradient operators. This edge image serves as the basis for the segmentation by Active Contours.

The process of tree ring extraction can thus be split in the following steps: preparation of the disc → digitization → image pre-processing → edge recognition → segmentation → data post-processing.

2.2. Tree ring extraction = edge recognition

In order to identify a tree ring in a picture of a tree disc in a computer-assisted way, the here presented procedure uses the difference in colour or brightness, respectively, that occurs between the darker late wood and the brighter early wood. In the field of image processing, such an abrupt change is called an edge.

Technically seen, the problem of tree ring extraction can thus be reduced to that of edge recognition. If a grey-level gradient from black to white (Fig. 1a) is considered and the corresponding brightness values are plotted in a diagram where the value 0 is assigned to black and 1 to white, one receives Figure 1b. An edge can thus in the continuous case be defined by using the derivative of the brightness (or colour) function. The location at which the first derivative is maximal while the second derivative is zero defines the edge, see Figure 1.

From image processing a number of methods are known that approximate the derivative of an image, the so-called gradient procedures. Corresponding to the possibility to define an edge by primarily using the first or the second derivative, some gradient procedures approximate the first and some the second derivative.

In most procedures, one or several matrices, so-called convolution kernels, are shifted step by step over the initial

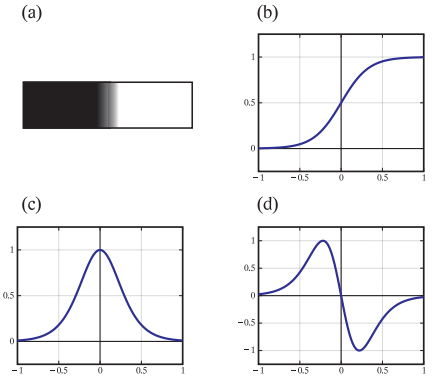


Fig. 1. Definition of an edge using derivatives. (a) Brightness gradient, (b) Brightness gradient in the diagram, (c) First derivative, (d) Second derivative.

image, and in each step a new pixel of the edge image is generated. A higher brightness value in the obtained edge image corresponds to a stronger gradient in the initial image. The edge (or gradient) image thus shows the positions and, via brightnesses, the strengths of the edges which occur in the initial image.

A simple but effective representative of such a discrete differentiation operator which approximates the first derivative is the Sobel operator method (Gonzalet & Woods 1992). In this case two convolution kernels are used, one (G_h) for the detection of horizontal edges and one (G_v) for vertical edges. In our implementation we use 3×3 kernels:

$$G_h = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}, \quad G_v = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad [1]$$

The two values are afterwards combined using one of the two variants in equation 2 in order to calculate the actual value.

$$|G| = \sqrt{G_h^2 + G_v^2}, \quad accurate$$
$$|G| = |G_h| + |G_v|, \quad approximation \qquad [2]$$

The application of the Sobel operator for example to the picture of a coast fir disc (*Abies grandis*) (Fig. 2a) yields the corresponding edge picture (Fig. 2b).

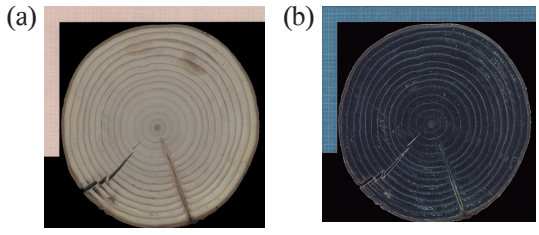


Fig. 2. Sobel operator applied to the picture of a coast fir disc: (a) Original disc, (b) Result of the application of the standard Sobel operator.

A more precise look at the edge image, however, reveals several unpleasant features which render the image almost useless for automatic edge detection. On the one hand every edge in the initial image is emphasized, including those generated by possibly existing branches or cracks in the wood (Fig. 2a). On the other hand sometimes double edges are generated (Fig. 3): This happens if the late wood appears only as a thin line that is enclosed by the brighter early wood. So the application of the Sobel operator gives two edges when the late wood is approached, one when coming from the centre and one when coming from outside.

These disadvantages make an optimisation of the filter necessary.

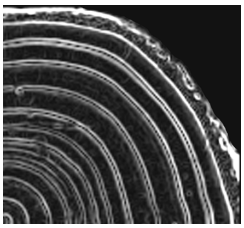


Fig. 3. Magnification of detail: A spruce disc after application of the standard Sobel operator illustrates double edges which can occur with wide late wood parts.

2.3. Filter optimisation

Purpose of the optimisation is to emphasize only the desired edges, in this case those corresponding to annual rings, and to weaken or to delete all other edges. Furthermore, the double edges should be reduced to a single edge. The basic idea of our filter optimisation is to consider the angles of the detected edges and to emphasize only those edges which are orthogonal to a ray through the marrow of the tree disc, taking a certain tolerance zone into account (Fig. 4). This condition is based on the property that tree rings expand in a more or less circular way around the pith and thus have tangents orthogonal to a ray through the centre.

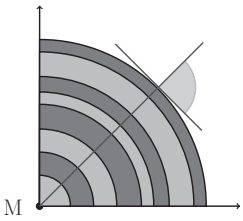


Fig. 4. Edge orthogonal to the ray through the centre.

As the basis of the filter optimisation the Sobel operator is used. While the convolution kernels are applied to the initial image step by step, in each pixel the angle between the tangent in this pixel and the ray through the centre is calculated according to equation 3.

$$\theta = \begin{cases} \arctan\left(G_h / G_v\right), & G_v \neq 0 \\ 0°, & G_v = 0, G_h = 0 \\ 90°, & G_v = 0, G_h \neq 0 \end{cases} \qquad [3]$$

If the angle between the tangent and the ray to the centre lies now in a tolerance zone of up to five or ten degrees, the point will be intensified by 20 percent, otherwise the point is not included in the edge image. The reverse orientation of the dark and bright side of a "false" edge has the desired consequence that these false edges are refused by the modified operator. Thus the artefact of "double edges" (as in Fig. 3) is automatically avoided. Figure 5 shows the result of the optimization, applied to the example shown in Figure 2.

Precondition for the method is that the centre of the tree disc, more precisely: the position of the pith, is known. The exclusive strengthening of those edges which follow a nearly circular course around a centre has led us to the name "circular Sobel operator" for the optimized operator.

In the process of extraction of tree rings, the next step is the segmentation, i.e., the assignment of the detected edges to single tree rings and the extraction of their coordinates. The procedure which we have applied here is known under the name "Active Contours".

Fig. 5. Application of the circular Sobel operator to the tree disc of Fig. 2. Problematic edges caused by cracks and branches as well as noise are removed.

2.4. Active Contours

The concept of Active Contours, also known as Snakes, was introduced by Kass (1987). A lot of optimizations and derived methods were subsequently developed, and their applications are today widespread. Basically the aim is to determine the contour of an object. A special feature of the method is its robustness against disturbances and noise in the initial image. Hence it is possible to identify even objects with very weak contours. This feature has led to a particularly widespread usage of the method in medical image processing where objects like organs or venation are to be identified in CT or MRT images. The method is also widely used in computer-aided object tracking and in face recognition.

The method makes use of a parametric curve, which is in most cases initialized manually. The slope of this curve is controlled by so-called internal and external energies. The internal energies are calculated solely from the form of the contour. They determine the tension and thus the tendency to the formation of loops, as well as the stiffness of the curve, or, expressed in a positive sense, its ability to adapt itself to fine details of the contour. The initial image determines the external energy via the edge (or gradient) image. An iterative optimization, which seeks to minimize the sum of the energies, deforms the contour until a stable state is obtained. The Snake curve thus seeks in the gradient image for maximal brightness values and adapts itself to their locations in the best possible way, taking the internal energies into account.

2.5. Tree Ring Segmentation Tool – TriST

As a basis for the implementation of our software we have used the Java Extensible Snake System, JESS for short, which has been developed by Tim McInerney and his team at Ryerson University, Toronto (McInerney & Sharif & Pashotanizadeh 2005). It offers a hierarchically designed structure with various Snake implementations and a simple graphical user interface. Furthermore, the system allows an interactive manipulation of form and parameters while the Snake curve is optimized. To control the course of the curve, additionally so-called magnets can be defined which have impact on the form of the curve independently from the edge image and from the parameters.

We have extended JESS by the circular Sobel filter and by the necessary infrastructure for tree ring extraction. This includes all functionality that is used for processing the recognized rings, for automatic initialization of the new curve in relation to the recognized ring, and, ultimately, for storage of the recognized rings. Additionally, a toolbox of standard image processing routines for pre-processing of the scanned rings was implemented, which will not be discussed further here.

For standard image manipulation the free Java Advanced Imaging Library – JAI (version 1.1.3, Oracle Corporation) was used. Therefore, all common image formats, e.g., TIFF and PNG, are supported by our software. Our tool is platform independent and as open source software it is free of charge, available upon request by the first author.

3. Results

3.1. Tree Ring Segmentation

The process of tree ring extraction is started by the initialization of the first curve outside the innermost ring and by the definition of the centre. During the process of adaption of the curve the user can interact and manipulate the pathway of the curve at any time by simple pressing the mouse at one point where the curve should cross. When the first ring is seized correctly by the curve, the user gives a confirmation. Subsequently, the recognized ring is stored and a new curve, positioned in relation to the old one, is initialized. In addition approximations of average radius, circumference and the area enclosed are calculated (Fig. 6). So each ring, successively from the innermost to the outermost one, is processed until the bark is reached. Finally, the data can be saved in different plain text formats (e.g. coordinate based and polar coordinates). Images can be archived with or without their analysis.

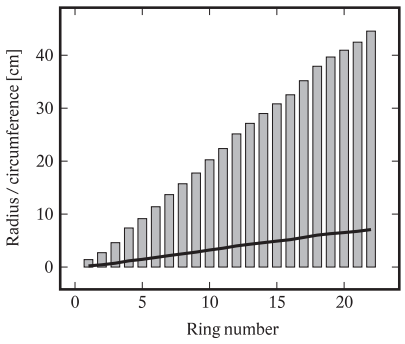


Fig. 6. Approximation of average radii (line) and circumference (bars) of a spruce disc with 22 rings (Fig. 8).

Branch scars, injuries or contaminations, e.g., caused by fungal infestation, are a common problems which usually causes trouble and requires manual intrusions during the extraction of tree rings. Figure 7a shows a disc from a coast fir with a branch scar and a crack at the bottom. It is clearly

visible how the curves (highlighted in blue) have adapted to the courses of the tree rings, without having been significantly influenced by the branch scar or the crack.

The following example (Fig. 7c–d) of a spruce disc with 22 rings shows in a direct comparison the original disc and the extracted tree rings.

It is also possible to extract the rings from discs with a diameter of 50 centimetres and more with the here presented software TRiST. The problem in this case rather lies in the process of digitizing, which is restricted by the size of the maximal scanning area. A solution is offered by scanning in several steps with subsequent joining, the so-called stitching, of the partial images to a complete image, which then serves as the basis for tree ring extraction as described.

The largest tree disc which we have processed until now originated from a 21 years old coast fir (*Abies grandis*). It had dimensions of 45 to 38 centimetres and had to be scanned in four steps. Figure 7b shows the original disc as well as the extracted tree rings.

3.2. Accuracy of measurement and expenditure

For verifying the accuracy of measurement, a tree disc was measured exemplarily by hand and the results were compared with the data obtained from computer-assisted extraction. The object for this test was a disc from an approximately 25 years old coast fir (*Abies grandis*), taken from a height of 9.5 meters, with 9 tree rings. Starting with a fixed direction, at intervals of 10 degrees the radii of all tree rings in relation to the pith were determined.

The sample thus included a total of 324 measuring points (36 directions, each with 9 measurements). To check the quality of extraction, the differences between the manually measured points and the points obtained from computer-assisted extraction were calculated and plotted in Figure 8. The average deviation is about −0.184 mm and shows a systematic error, which was probably caused by the conservative measuring by hand. The obtained errors are in an interval between −0.99 and 0.95 millimetres. Altogether, 86% all of all differences are within a deviation of 0.5 millimetres or less around the mean.

Several test runs gave an average extraction time per ring of 60 seconds, with the required time increasing with larger radius. This results from the increase in perimeter and the resulting longer control time. For the complete analysis of a disc with 20 tree rings we got an average processing time of approximately 20 minutes.

4. Discussion

The key issue for successful, accurate and reliable measurements of whole tree rings with our system depend mainly on the quality and type of input material. As discussed above, the quality and success of extraction are proportional to the visibility of each ring within the image, consequently species where early- and latewood are clearly distinguishable

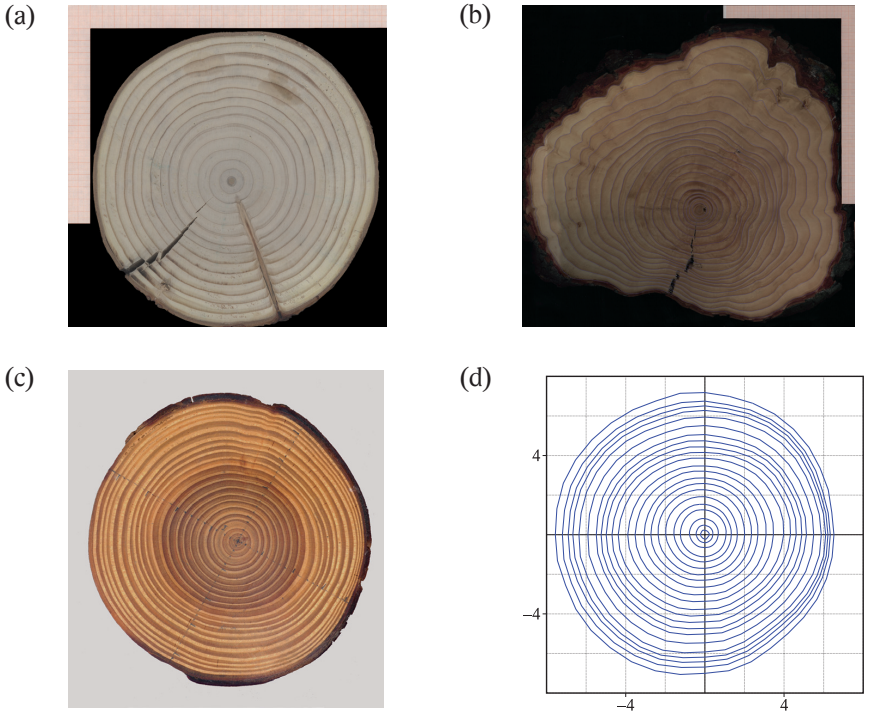


Fig. 7. (a) Tree disc of a coast fir with a branch scar and a crack. The extracted tree rings are highlighted in blue. **(b)** Example of a 21 years old coast fir (*Abies grandis*), the largest tree disc analysed so far by our method. The extracted tree rings are highlighted in blue. (c–d) Example of a spruce disc with 22 tree rings. **(c)** Original disc, **(d)** Extracted rings.

are preferred. In order to reduce false detections during the segmentation process caused by traces of the cutting process, e.g. sawing scratches, precedent preparations of the tree disc itself are advisable before scanning. During image processing common strategies to eliminate noise and enhance the contrast can be applied by the system if required. For our purpose, we can conclude that the technique of Active Contours produce reproducible and reasonable results.

The reliability of our system was evaluated by comparison with manually measurements performed on 324 measuring points (36 directions, each with 9 measurements). From Figure 8 we can see that the errors does not follow any pattern, while with increased ring number the variation are larger. The overall accuracy of measurement of our system might be low compared with touchstones applied in dendrochronology. While systems used in dendrochronology normally using light microscopes and consider only radial ring-width measurements, our system is designed to extract whole tree rings, what can compensate the errors to a certain extent. For deducing growth behaviour, the difference has no significant consequences.

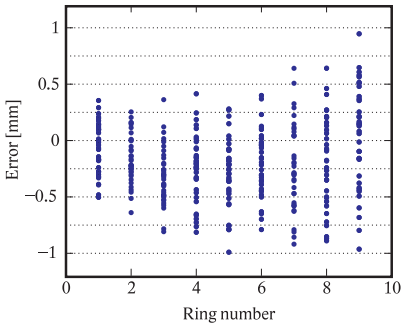


Fig. 8. Differences between the manually measured points and the points received from computer-assisted ring recognition. Mean = −0.184 mm, standard deviation = 0.336 mm.

While for common edge detection operators applied to wide latewood parts double edges, so called pseudo rings, are produced, it is found that our optimised Sobel operator does not face this problem. Cracks as well as branches can be nearly eliminate in the same way.

Our system reaches his limits when the contrast between early- and latewood is to low and following no edge can be identified. The same applies when a gradient instead of a sharp switch in colour is given. Very thin rings, low ring width and large image noise can lead to more user interaction and so increase the time for extraction.

5. Conclusions

A new software tool for the semiautomatic tree ring extraction by using Active Contours was developed. In order to enhance the tree ring recognition a new filter was designed and integrated into the software. The system is suitable for all kind of input images that fulfils a minimum requirement of a certain contrast, while the size of tree disks is only limited by the used recording equipment. To improve the accuracy of the interactive measurement process and to reduce its complexity, some exemplary studies have been conducted which gave promising results. The system provides an efficient, time-saving way for tree ring extraction. The resulting data can deliver plenty of information on how trees adapt growth to environmental conditions that further can be used to analyse wood quality or to describe and model changes in stem growth.

Acknowledgments

This research was funded by the German Research Foundation (DFG) under grant SL 11/12-1. Student assistant Julia Rudolph gave support in data extraction. All support is gratefully acknowledged.

References

Conner, W. S., 1999: A Computer Vision Based Tree Ring Analysis and Dating System, Thesis, The University of Arizona.

Gaffrey, D., Sloboda, B., 2004: Modifying the elastomechanics of the stem and the crown needle mass distribution to affect the diameter increment distribution: A field experiment on 20-year old *Abies grandis* trees. Journal of Forest Science 5: 199–210.

Gonzalez, R. C., Woods, R. E., 1992: Digital image processing. 3rd edition, Addison Wesley, 954 p.

Gopalan, G., 2000: An Interactive Image Analysis System for Dendrochronology, Thesis, The University of Arizona.

Guay, R., 2012: WinDENDRO 2012: User's Guide, Regent Instruments, Inc., Quebec, Canada.

Kass, M., Witkin, A., Terzopoulos, D., 1987: Snakes: Active contour models. International Journal of Computer Vision, 1(4): 321–331.

McInerney, T., Sharif, M. R. A., Pashotanizadeh, N., 2005: JESS: Java Extensible Snakes System. SPIE International Symposium, Medical Imaging 2005: Image Processing; San Diego, 5747(12–17):1985–1992.

Norell, K., 2011: Automatic counting of annual rings on *Pinus sylvestris* end faces in sawmill industry. Computers and Electronics in Agriculture 75(2):231–237.

Soille, P., Misson, L., 2001: Tree ring area measurements using morphological image analysis. Canadian Journal of Forest Research, 31(6):1074–1083.

Taube, D., Sloboda, B., 1992: Polarkoordinatenmeßsystem zur interaktiven und automatischen Analyse forstlicher Objekte, insbesondere von Stammscheiben. Schrift. Universität Göttingen und Nordwestdeutsche Forstliche Versuchsanstalt (NW-FVA), 106:6–11.

Zhou, H., Feng, R., Huang, H. H., Lin, E.-P., Yu, J.-L., 2012: Method of tree-ring image analysis for dendrochronology. Optical Engineering 51(7): 077202-1–077202-7.

CHAPTER 5

Third Paper

This paper is published as:
Henke M, Kniemeyer O, Kurth W (2017) Realization and extension of the Xfrog approach for plant modelling in the graph-grammar based language XL, *Computing and Informatics*, 36(1), 33-54, doi: 10.4149/cai_ 2017_ 1_ 33

Authorship

- Ole Kniemeyer supported the writing of the manuscript.
- Winfried Kurth wrote parts fo the methods section and supported the writing of the manuscript.

Computing and Informatics, Vol. 36, 2017, 33–54, doi: 10.4149/cai_2017_1_33

REALIZATION AND EXTENSION OF THE XFROG APPROACH FOR PLANT MODELLING IN THE GRAPH-GRAMMAR BASED LANGUAGE XL

Michael Henke

Department Ecoinformatics, Biometrics and Forest Growth
Büsgenweg 4
University of Göttingen
37077 Göttingen, Germany
e-mail: mhenke@uni-goettingen.de

Ole Kniemeyer

Antonianger 27
31061 Alfeld (Leine), Germany
e-mail: ole.k@arcor.de

Winfried Kurth

Department Ecoinformatics, Biometrics and Forest Growth
Büsgenweg 4
University of Göttingen
37077 Göttingen, Germany
e-mail: wk@informatik.uni-goettingen.de

Abstract. Two well-known approaches for modelling virtual vegetation are grammar-based methods (L-systems) and the Xfrog method, which is based on graph transformations expanding "multiplier" nodes. We show that both approaches can be unified in the framework of "relational growth grammars", a variant of parallel graph grammars. We demonstrate this possibility and the synergistic benefits of the combination of both methods at simple plant models which were processed using our open-source software GroIMP.

Keywords: Plant modelling, object instancing, graph grammar, Xfrog, XL, GroIMP

Mathematics Subject Classification 2010: 68-U05

1 INTRODUCTION

Modelling the detailed structures of plants with a custom interactive 3D modeller is very time-consuming. Several algorithmic solutions have been implemented using general-purpose programming languages to construct realistic vegetation structures automatically. Beyond these ad-hoc solutions, there are two approaches offering a more generic framework for the specification of the architecture of individual plants: the string-based formalism of Lindenmayer systems, realized, e.g., in the software LStudio [18], and the graph-based interactive approach proposed by Lintermann and Deussen [16] and realized in the software Xfrog [7].

Other approaches for modelling trees, like the one introduced by Pirk [19], use skeleton-based geometries extracted from images or laser scanners to generate 3D structures. The produced dynamic models can react on environmental influences, a feature which was in the past only possible when using growth models such as L-systems. On the other hand the simplicity of the models allows a creation on the fly so that they can be used in real-time scenarios such as games or simulations.

Ijiri [11] presented a sketch-based technique, a combination of rule-based and image-based techniques on procedurally created trees. Stroke inputs are used in L-systems to control the overall model appearance and the depth of recursion.

These approaches are mainly focused on fast production of realistically looking images of plants. They work with interactive design tools and simplified structures and do not claim to be botanically correct in any case. Modelling plant functions like transport processes is not considered.

Here we present a combination of the object instancing approach, as implemented in Xfrog, and rule-based modelling. Our modelling system consists of three components: Relational Growth Grammars (RGG) as formal basis, the programming language XL (eXtended L-system language) enabling an easy use of RGG and at the same time extending the well-known object-oriented language Java, and the software GroIMP (Growth-grammar related Interactive Modelling Platform), providing interactive facilities, rendering, and a full-scale development environment for XL.

2 STATE OF THE ART

Lindenmayer systems (L-systems for short) are systems of replacement rules operating on strings. Developed in the context of formal grammar theory, they can be used to specify the growth of vegetative structures according to botanical rules,

which has been demonstrated in various papers and books, the most prominent by Prusinkiewicz and Lindenmayer [20]. To this purpose, the strings generated by the grammar mechanism have to be traversed from left to right and must undergo an interpretation by turtle geometry (see [20] for details).

Sequential graph grammars have previously been used in various fields of application, most often in software engineering, but also in pattern recognition and image analysis. Graph grammars with parallel mode of operation, as in our case, were theoretically investigated in the seventies (for references see [12]), but then got out of focus for a while, and their use to generate 3D scene graphs is new. Besides L-systems with interpretation [12] and structural factorization [21] the Xfrog multiplier nodes are another concept for object instancing that can be used within GroIMP. With the combination of these concepts we provide new ways for fast prototyping and model development.

The Xfrog approach, on the other hand, allows to interactively edit a graph made up of component prototypes, the so-called p-graph (see [3] for details). Among the nodes of this graph there are not only shape nodes representing graphical primitives, but also various sorts of *multipliers* which have the semantics of copying the structures encoded by their descendant nodes and placing them in specified positions. E.g., a "Wreath" node generates a circular arrangement of copies. The p-graph is then expanded to a tree, the so-called i-tree, having instances of the prototypes as nodes, and this tree is then traversed, similar to a scene graph [6], in order to build the geometrical model of the plant (Figure 1, cf. [3]). The interactive access to the p-graph requires a medium level of abstraction and allows a quick feedback from the resulting rendered model to the editing process, thus enabling a quite intuitive working. The portfolio of components (node types) is, however, restricted, and there is no natural way to simulate processes of growth and development in this framework – or even to include biologically-inspired process-based models (e.g., of plant hormonal effects controlling flowering), which is relatively easy in L-systems (cf. [20]). The current version of Xfrog is implemented as plugin for the 3D computer graphics softwares Cinema 4D and Maya.

3 METHODS

In the following, an introduction of Relational Growth Grammars (RGG) is given and the application of two sorts of RGG rules, generative and instantiation rules, will briefly be explained. After that, a short introduction of the interactive modelling platform GroIMP is given.

3.1 Relational Growth Grammars

While L-systems are widely used, they have still some drawbacks – not so much concerning their theoretical power, but with respect to transparency and simple use when complex, multi-level plant models including functional aspects and/or

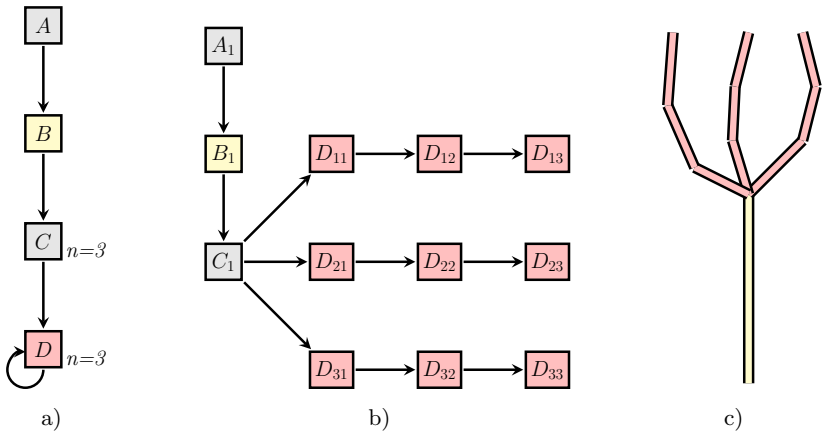


Figure 1. The Xfrog workflow. In this example the nodes B and D have a geometric interpretation while the node C is a replicator without a geometry. The node C replicates its subgraph three times. In this case the subgraph consists only of node D. The node D replicates itself also three times. a) i-tree, b) p-graph, c) one possible geometrical interpretation; colouring is in the corresponding graph colours.

genetic control are required. One critical feature is: L-systems operate basically on strings, which have to be translated into 3D-structures (representing plants or plant communities), the latter being the actual objects of modelling (see Figure 2 a)).

We use the concept of RGG, a graph grammar formalism, and its implementation in the language XL (eXtended L-Systems) to overcome this and other drawbacks. In XL, nodes are objects in the sense of object-oriented programming, they generalize the symbols in classical L-system strings and can be associated with Java classes. Edges can represent arbitrary, user-defined relations, they generalize the sequential order of symbols in strings. Hence the extra description level of strings can be dispensed of in the rewriting process (Figure 2 b)); we use strings only for writing down the rules.

Advantages:

- **Complex relationships** such as genotype-phenotype relations can now be represented with the same simplicity as a topological neighbourhood in classical L-systems,
- the same holds for **multiscale** plant descriptions [17],
- arbitrary sorts of **context** can easily be defined,
- the representation of **networks**, including feed-back loops, is possible in our formalism in an intuitive way (as graphs),
- the interface between rule-based model description and **procedural modelling** becomes more elegant by incorporating Java classes (as nodes) and scripts in the rules,

- **strings, trees** and **multisets** are **subcases** of our graph data structure, thus our RGG have at least the same descriptive power as the rewriting systems operating on these restricted structures.

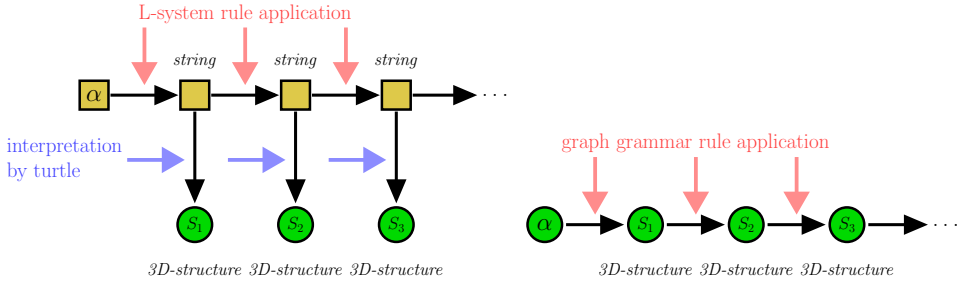


Figure 2. Functioning of a) a classical L-system, compared with b) a relational growth grammar. α is the start symbol (axiom). The developmental steps of a plant or plant community are represented by 3D-structures $S_1, S_2, \ldots$

Details about RGG, XL and GroIMP have been described in a thesis [12] and are used in the field of biological modelling of plants [2, 9, 23]. RGGs are parallel rewriting systems operating on typed attributed graphs (instead of strings) and form thus a variant of graph grammars. Their exact definition can be given in terms of algebraic graph-grammar theory [13] and captures the "dynamic component" which is inherent in L-systems but lacking in the Xfrog approach. For instance, an RGG rule given in XL in the form

```
Axiom ==> Cylinder(2,3) Cylinder(20,1) Sphere(3);
```

will replace a default initial node called `Axiom` by a graph consisting of three successive nodes (connected by successor edges – the blanks delimiting the components of the right-hand side of the rule are used to construct edges of type "successor"), and these nodes are interpreted as parts of a scene graph, namely as a cylinder with length 2 and radius 3, on top a cylinder with length 20 and radius 1, on top a sphere with radius 3. All three nodes can again be replaced by other nodes if there are corresponding additional rules. Furthermore, auxiliary nodes like `A` or `B(1)` without geometrical meaning are allowed, similar to L-systems. Edges of other types than "successor" can be specified using the notation "`-edgetype->`".

3.2 Generative Rules within XL

The "normal" type of rules used in an XL program is a generative RGG rule, a straightforward generalization of an L-system rule. In the example given above in the Introduction, one node of type `Axiom` is replaced by a new subgraph, consisting of two nodes of type `Cylinder` and one of type `Sphere`. These node types are predefined as Java classes for the scene graph of the modelling platform GroIMP; as Java

classes they have encapsulated attributes, amongst them `radius` and (for `Cylinder`) `length`, and their appearance as part of the right-hand side of a rule is analogous to a constructor invocation in Java. Node types can also be defined by the user, and they can inherit from other node types. Besides conventional Java class declarations, a shorthand notation called module declaration is possible: for instance,

```
module A(int i, super.radius) extends Sphere(radius);
```

defines a new user-defined extension of the `Sphere` node class inheriting the `radius` parameter but with an additional parameter `i` with integer values as well as the geometrical shape of the `Sphere` node class.

In XL it is possible to insert imperative commands in right-hand sides of rules; thus an alternative, but equivalent rule to the example given in the Introduction would be

```
Axiom ==> Cylinder(2,3) Cylinder(20,1)
          s:Sphere { s.setRadius(3); };
```

where the created `Sphere` node is labelled `s` and the assignment of its `radius` attribute is done *a posteriori*. (It is also possible to have more than one node on the *left-hand side* of an RGG rule and thus to replace non-trivial subgraphs, but we will not use this possibility in the following.) Transformation nodes like `Translate(x,y,z)` or `Scale(u)`, as known from scene graphs, are also defined. With rules of this sort, classical L-systems as well as many of their extensions published in the plant-modelling literature can be emulated. However, what is still missing is the possibility to copy whole subgraphs, as it is required during the transformation of the Xfrog p-graph to the i-tree. This can be done in a generative XL rule like that shown above by introducing a user-defined node type (here called `Replicator`) and invoking the `cloneSubgraph` method provided by GroIMP. To connect the subgraph which is to be replicated to the replicator, we use an extra edge type called `multiply`. The method `getFirst` yields the subgraph beginning with the first node accessible via this edge from the replicator. Figure 3 shows the respective graphs where the initial state consists only of a default node (`Root`) and the initial node `Axiom`.

```
module Replicator;

public void run() [
        Axiom ==>
                Cylinder(2,3) Replicator -multiply->
                Cylinder(20,1) Sphere(3);

        r:Replicator ==>
                [cloneSubgraph(r.getFirst(multiply))]
                Translate(10, 0, 0)
                [cloneSubgraph(r.getFirst(multiply))];
]
```

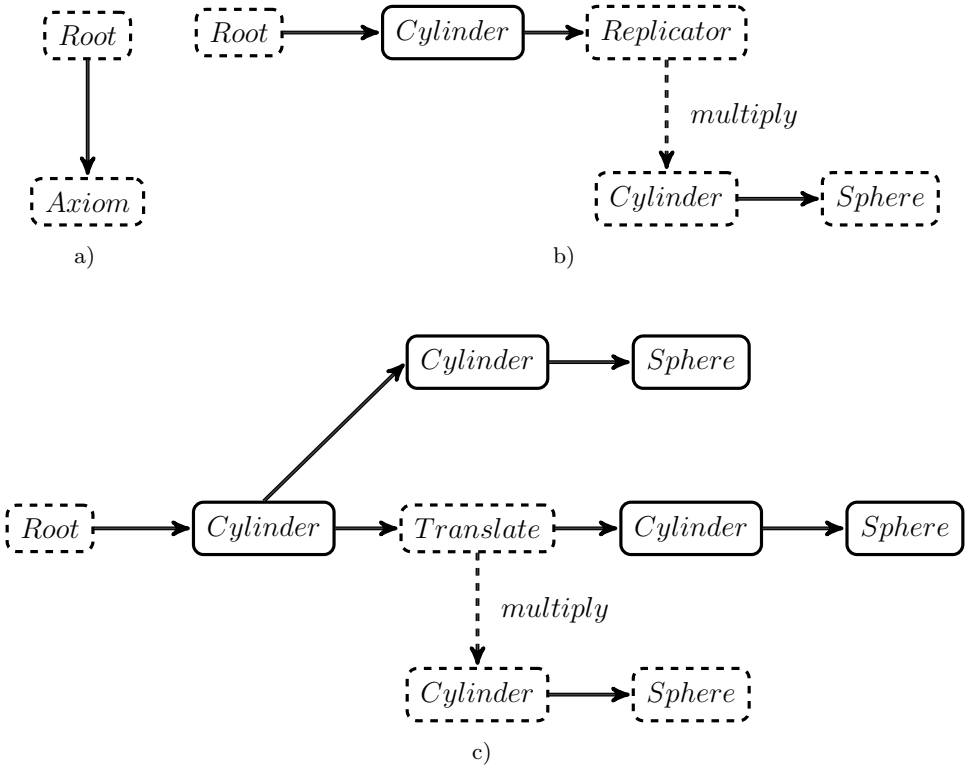
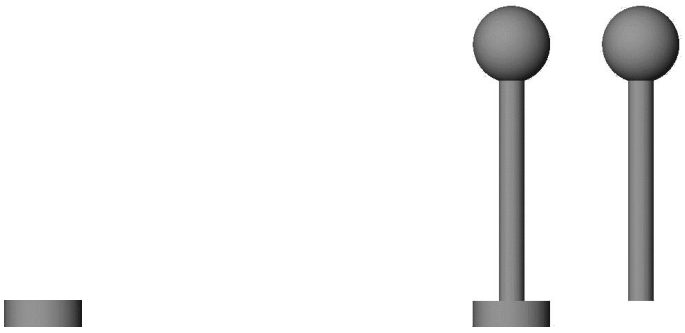


Figure 3. Graphs produced by the "Replicator" grammar from the text, a) for the initial state, b) after a single time step and c) after two time steps. Dashed nodes are not visible in the 3D view of the structure.

The visible result of this small XL program (Figure 4), consisting of two RGG rules which are first applied to the default start node `Axiom`, is after the first time step only the first cylinder (with length 2 and radius 3), because the `Replicator` node has no visual interpretation and the `multiply` edge is not traversed during geometrical interpretation of the graph (see Figure 4 a)). In the second time step, however, the replicator is replaced via application of the second rule by two copies of the subgraph consisting of the long cylinder and the sphere, which are separated (because of the `Translate` node) by 10 units in x direction, see Figure 4 b).

3.3 Instantiation Rules

By using relational growth grammars as described above, we can construct a scene graph consisting of nodes for geometric primitives and further nodes which may describe non-geometric states of the underlying (botanical) model. However, there are cases where it is advantageous to assign a set of primitives to a single node of the graph. For example, a single entity of the model may need several primitives for its 3D representation, and then it would be cumbersome, a waste of memory



a) b)

Figure 4. Graphical result of the "Replicator" grammar from the text, a) after a single time step and b) after two time steps

and a violation of the principle of separation of concerns if one has to include these primitives in the same graph as the entities of the model.

Therefore, the language XL defines *instantiation rules* which can be assigned to node classes and which may expand a single node to a set of primitives when they are invoked by GroIMP as part of the 3D visualization of the node. Although these rules resemble generative rules in syntax, they do not modify the graph and are only activated during visualization. A formal definition is given in [21]. For an instantiation rule, we have to use a module declaration and add the nodes which shall be used for visualization after an arrow symbol as in

```
module Stem(float len) ==> Cylinder(len,1);
```

which represents a stem as a cylinder without the requirement that the class `Stem` inherits from `Cylinder`. The right-hand side may also contain references to other parts of the graph so that instantiation rules provide a simple means to specify object instancing, i.e. multiple occurrences of the same 3D structure at different locations. This can be used for the replicator example from above: if we use for the `Replicator` node an instantiation rule instead of the simple generative rule, we can dispose of the `cloneSubgraph` invocation.

```
module Replicator ==>
       [ getFirst(multiply) ]
       Translate(10, 0, 0)
       [ getFirst(multiply) ];

public void run() [
       Axiom ==>
             Cylinder(2,3) Replicator -multiply->
             Cylinder(20,1) Sphere(3);
]
```

The graphical result of this XL program is the same as above in Figure 4 b).

3.4 The Software GroIMP

The open-source 3D modelling platform GroIMP [8] has been developed together with the formalism of relational growth grammars and the language XL to have an integrated environment for rule-based 3D modelling. GroIMP provides a rich set of 3D node classes including simple ones like spheres, boxes and cylinders, but also NURBS surfaces, height fields and CSG operations. GroIMP maintains a current graph which is interpreted as a scene graph for visualization and may be transformed by rules specified in the language XL. The user may select a node in the 3D visualization and inspect or modify its attributes. Depending on the underlying rules of the model, interactive modifications by the user like the removal of branches of a plant may influence the further development of the structure. GroIMP contains an OpenGL visualization and an integrated raytracer with the option to use path tracing. Besides being a 3D modelling platform, GroIMP also contains a source code editor and an XL compiler to facilitate rule-based modelling with the language XL. Figure 5 shows a screenshot of the GroIMP modelling platform.

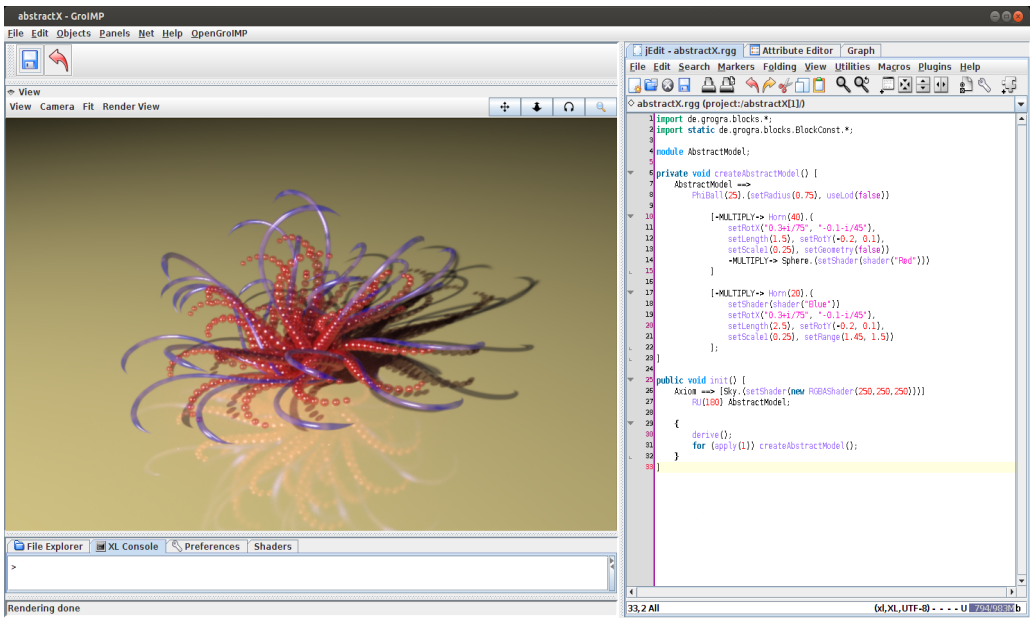


Figure 5. Screenshot of the GroIMP software displaying an “alien plant” in the 3D-view. The editor window on the right hand side displays the corresponding model code.

4 APPLICATIONS

This section will demonstrate the emulation of Xfrog’s multiplier components in the language XL, but also some extensions of the Xfrog functionality which result in a natural way from the embedding in the new framework. A discussion of the

advantages, possible applications and future extensions of our modelling approach will close the paper.

4.1 Simple Models

At the example of the `Wreath` component of Xfrog we want to show how to put into practice the Xfrog components in the modelling language XL. The `Wreath` component multiplies its child structure in a ring around a centre point. The radius of the ring where the instances are generated as well as their number can be controlled by the user. In our implementation of a wreath instantiation module we will use the following four attributes:

`number`	number of generated instances
`rX`	radius in x direction
`rY`	radius in y direction
`scale`	a uniform scaling factor

These attributes are included in line 1 of the following code specifying an instantiation rule. In the body of the subsequent loop we have two parts: one Java part (lines 3–10) with some calculations and one instancing part (lines 11–14) which produces the geometry. Like in the replicator example in the Introduction, the method `getFirst` (line 13) returns the first node which is attached to the current `Wreath` node by a `multiply` edge. This node represents the root of the subgraph to be multiplied.

```
1  module Wreath (int number, float rX, float rY, float scale)
2      extends Point ==> {
3          float delta = (float)(2 * PI / number);
4      }
5  for (int i = 0; i < number; i++) (
6      { // java part
7          float w = i * delta;
8          float x = rX * (float) Math.cos(w);
9          float y = rY * (float) Math.sin(w);
10     }
11     [ // graph part
12         Translate(x, y, 0) Scale(scale)
13         getFirst(multiply)
14     ]
15 );
```

The following XL code produces an elliptic distribution of 15 cone nodes as shown in Figure 6, using the `Wreath` class defined above. The elliptic shape depends on the different radii for x and y axis, here 6 for x and 4 for y. The scaling factor in this example is for all instances one.

```
Axiom ==> Wreath(15,6,4,1) -multiply-> Cone;
```

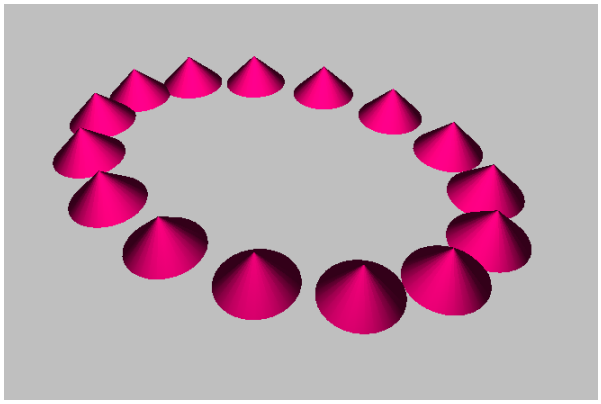


Figure 6. Graphical result of the application of the `Wreath` class to a cone, demonstrating the possibility to model an Xfrog component using the language XL.

In [10] the main Xfrog components were made available for GroIMP: `Tree`, `Horn`, `Hydra`, `Wreath` and `PhiBall`. As extensions there are also some new components: `Variation`, `BlockScale`, `BlockColor` and `Arrange`.

Because the user does not need to care about the technical realization of these multiplier nodes, they were predefined and collected in a "3D-construction-set package" (3D-CS) which is integrated as a library in the modelling platform GroIMP. They are called instantiation components or blocks. The following instantiation components are available in GroIMP:

Tree: Basic component for trees, creates the geometry of a stem and multiplies subsequent components as branches. Parameters are the distribution of branches, their scale, angle etc.

Horn: A component that places other components on a user-defined curve. It is used for stems, twigs, etc.

Hydra: Multiplies subsequent components on a curve with any direction relative to the direction of the parent component.

Wreath: The functionality is integrated in the Hydra component (as in Xfrog v.4.0.).

PhiBall: Multiplier that distributes all connected structures on a section of an ellipsoid according to the golden angle.

Arrange: Main component for arranging large numbers of instances on an area according to user-defined terrain data.

Variation: Allows in combination with any multiplier to vary the generated instances in a sequential, spread, exceptional or random way.

BlockScale: Scaling component, enables scaling depending on internal variables.

BlockColor: Enables colouring depending on internal variables.

With this set of instantiation components it is possible to model a huge number of not only organic structures. Such models are specified by two parts. The first part is a graph whose nodes are instantiation components and graphical primitives and the second part is a set of attributes for each component. The graph describes the physical structure of the model. For example, a tree consists of a stem with some levels of branches, and on the last level the leaves are located. The attribute set determines the properties of a component, e.g., for the tree, the number of leaves that a branch generates.

4.2 The Arrange Component

With great amounts of objects to be distributed on a terrain in a realistic manner, only in the rarest cases an individual positioning can be done manually. Hence efficient procedures for modelling of whole populations must be found. [4] already introduced a system built around a pipeline of tools that address this task. However, it was never implemented as Xfrog component. The `Arrange` component offers a wide range of possibilities to specify distributions on a terrain. They can be separated into four classes:

geometric arrangement generates a strict geometric arrangement, e.g., following lines or circles.

probability arrangement arranges the objects according to probability distributions (Poisson or normal distribution).

halftoning arranges the objects according to halftoning methods allowing for a user-defined density field [22, 15].

additional operations collection of operations like tilings or iterative methods like Voronoi-Lloyd [5].

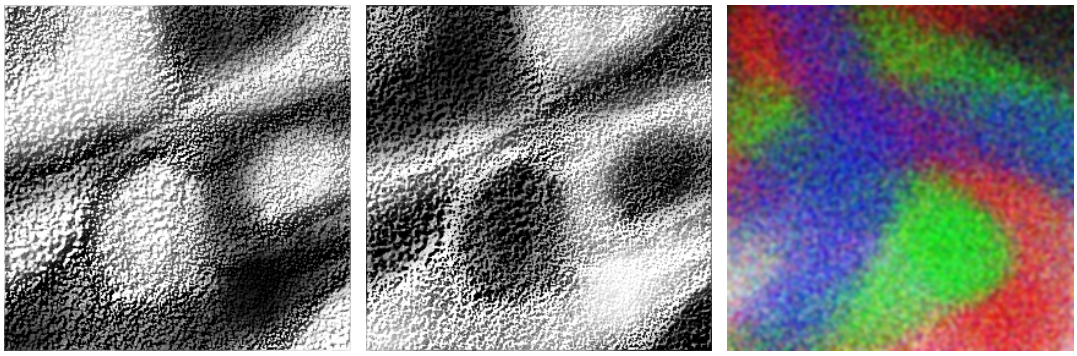
In addition to arrangements of objects on a plane, the user can define terrain data and location parameters for the whole area to be filled. The information is available in every multiplied structure and can easily be changed just by exchanging an underlying image file.

The following small example generates an `Arrange` field with 50 uniformly distributed `Horn` instances.

```
Axiom ==>
        Arrange(50) -multiply->
        BlockScale("0.01+n1", "0.01+n1", "0.5")
        BlockColor("10", "n2*256", "10")
        Horn(5).(setLength("0.05+n3*0.2"));
```

Scaling, colouring and length of instances depend on user-defined location parameters, which are given by an image (see Figure 7). Each channel of this image, usually interpreted as RGB colour specification, can be accessed by using one of the variables `n1`, `n2` or `n3`, and so they can easily be fed into a model. In the above

example, the colour of the instances depends on the variable `n2` which is used in the `BlockColor` component to set the green channel multiplied by 256.



a) height field b) density field c) location parameter field

Figure 7. User-defined location parameters for an `Arrange` component are defined by image files

The attributes as well as the inclusion of the location parameter field can be inspected and modified in an attribute editor. A part of the attribute editor for the arrange component is shown in Figure 8. In the lower part one can see how the location parameter field is included.

Starting from the initial state shown in Figure 9 a) we can now utilize the user-defined fields shown in Figure 7. Figures 9 b), 9 c) and 9 d) show the individual applications of user-defined fields. Figure 9 e) reflects the joint application of all three fields.

Besides the already mentioned attributes of the `Arrange` component, the possibility of scaling, rotation as well as random modification of positions of the produced instances is implemented.

4.3 Combination with Grammar-Based Models

Modelling with the 3D-construction set permits a fast and easy way to produce attractive models. However, because instantiation rules are a purely structural concept, modelling of functionality with them is not possible. Functional-structural models of plants, including dynamics of growth, can be obtained by a combination of our 3D-CS with generative RGG rules. So it is possible to instantiate complex plant organs like blossoms and use them in generated structures. They do not have to be generated using a complex derivation process. On the other hand, structures described by RGG-based models can be multiplied and/or positioned by instantiation components.

The first example does not use generative rules at all, but only a conventional control structure for iteration. The XL code "for (int i :(1: n)) (X)" generates `n` replications of X, where X can be any object or graph structure. Thus the following

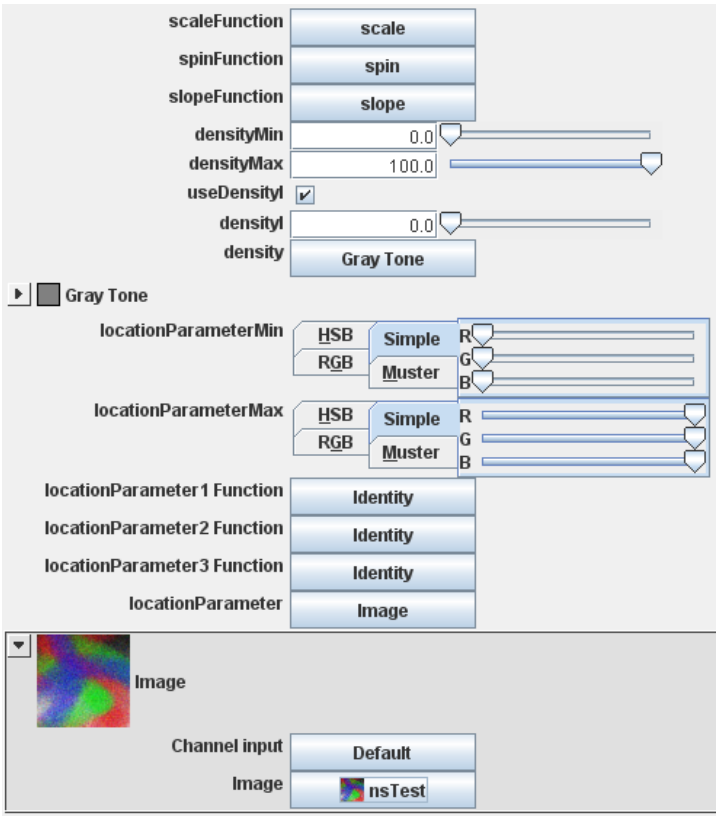


Figure 8. Panel for the attributes of the `Arrange` component

code section generates 15 `Horn` instances and arranges them at random positions on a 10×10 square field (Figure 10).

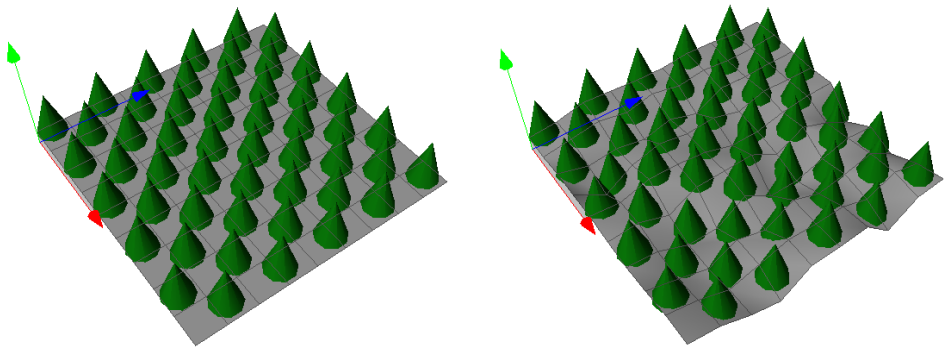
```
Axiom ==> for (int i:(1:15)) (
  [
        Translate(random(0, 10), random(0, 10), 0)
        Horn().(setLength(0.01+i/30))
  ]
);
```

Additionally the length of the generated instance depends on its iteration index.

The next example demonstrates how a blossom produced by instantiation can be used as a module in an RGG generated structure. The branching structure is derived from these generative rules:

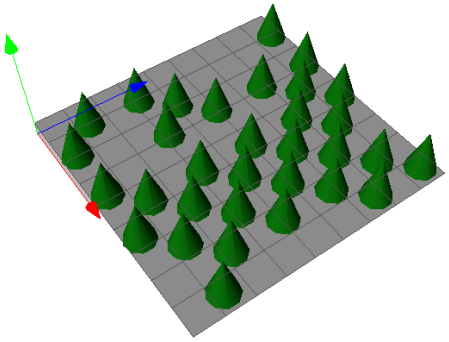
```
Axiom ==> Shaft(3.5, 0);

Shaft(x, a) ==>
        D(x/5) F(x)
        [
```

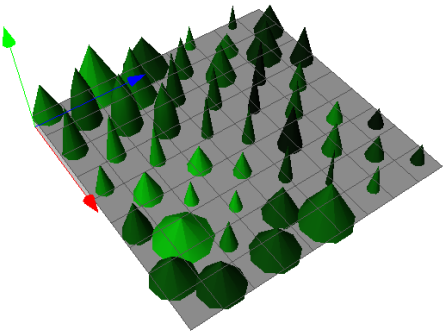


a) initial base

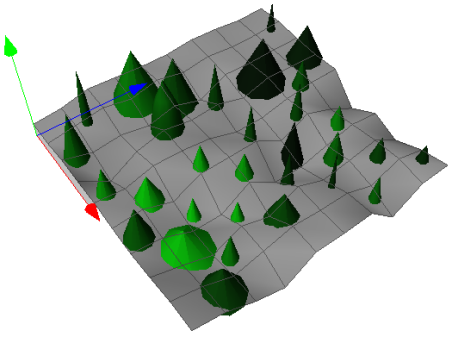
b) height field



c) density field



d) location parameter field



e) result, all fields applied

Figure 9. Example of a configuration of parameterised objects generated with the `Arrange` component and employing the user-defined fields from Figure 7. Figures a)–e) show variants where different parameter fields are switched on or off.

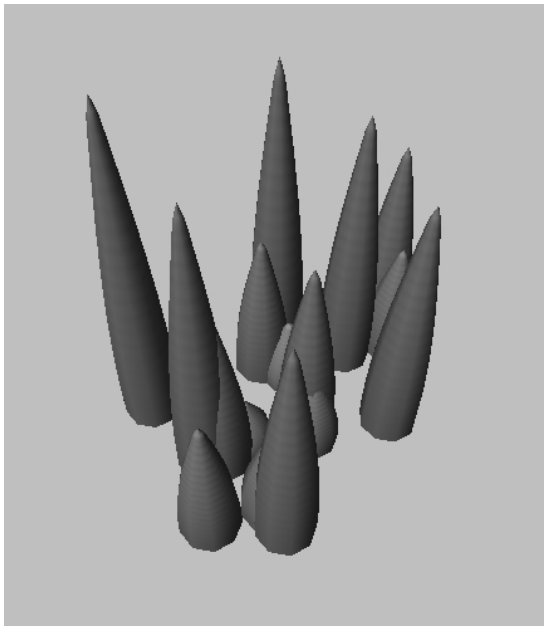


Figure 10. Graphical result of the simple "for" loop example: 15 randomized arranged `Horn` instances with increasing length

```
        RH(a) RU(45) LeafA(x) Branch(0.3*x, 20)
      ]
      Shaft(x*0.9, a+137.5);

Branch(x, age) ==>
      D(0.6*x) F(x) RU(-3) Branch(x*0.9, age+5);

LeafA(x) ==> LeafA(x*1.12);
```

The rule for `Shaft` generates the shaft and arranges the branches around it. There we use two parameters `x` and `a`. `x` is the actual length and `a` the actual branching angle. The next two rules for `Branch` and `LeafA` define instantiation modules. `LeafA` just generates the green leaf and lets it grow by a factor of 1.12 per step. The `Branch` module instantiates a branch-like structure:

```
module Branch(float x, int age) ==> {
      makeGraph ==>
        rootS:Scale(age/50)
        PhiBall(age/3).(setRadius(0.5),
          setFan1(0.2), setScale(1.0, 0), useLod(false)
        ) -multiply-> Rotate(0, age, 90) Petal();
}
rootS;
```

Starting with a `Scale` node a `PhiBall` multiplies a `Rotate` node followed by a `Petal`. The important parameter to control the blossom is the age. It controls the scaling factor, the number of petals as well as their opening angle. A petal itself generates a bent NURBS surface with a petal texture (code not shown). The result of the model is shown in Figure 11.

Figure 11. Blossoms generated by instantiation rules

The last example demonstrates the reverse method, an instantiation component, in this case an `Arrange` component, multiplies a generated structure. Here we used a biological model of a young unbranched poplar taken from [1]. This model includes methods for biosynthesis and transportation of phytohormones as well as process-based calculations of photosynthesis depending on light interception. The instantiation is quite easy: In a first initial step five different trees will be instantiated and saved in an array. To arrange them, all to do is to place an `Arrange` component before the tree model. In XL code, this could look like

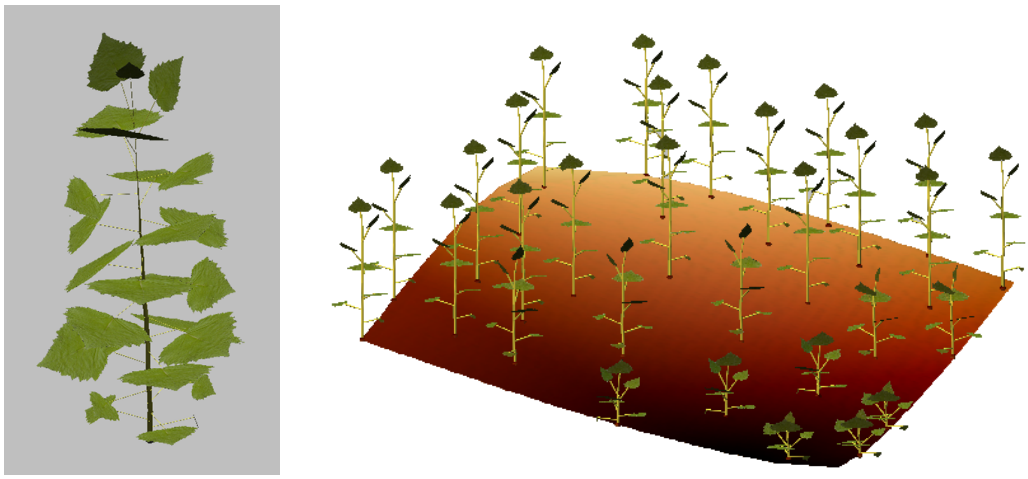
```
Axiom ==> Arrange.(
                setWidth(2, 2),
```

```
        setArrangeMethod(new AdditionalArrange(30))
    ) -multiply-> PoplarTree;
```

With `setArrangeMethod` we define the arrange method. The class `Additional-Arrange` includes some additional methods available in the arrangement process, where the default method is "DartThrowing", a random based distribution. The constructor argument 30 sets the number of generated instances. The module `PoplarTree` finally returns one of the predefined trees depending on the location parameters of the current position.

As a modification of the output of the original poplar model, the growth potential of the initial meristem of each individual is made dependent on its height over ground level, which can be determined by the location parameters of the `Arrange` component. Thus, small trees appear in lower regions and larger trees in the higher parts.

Figure 12 b) shows the result: 30 poplars arranged by an `Arrange` component.



a) original single tree model b) arranged trees; using the new `Arrange` component

Figure 12. Model of young unbranched poplar taken from [1]

5 CONCLUSION

Modelling with instantiation components is a powerful technique to get results quickly. Although this modelling technique has no botanical basis as it does not capture the growth process, the results are visually satisfying. It is hard to translate data taken form nature to the parameters of the component. A relation between reality and model can only be subsequently produced by measurements at the archetype and comparison with the model. In the rarest cases this gives botanically correct models. The degree of lifelikeness lies completely in the responsibility of the modeller.

The generated graphical results are nevertheless more than only nice pictures. With them, new possibilities arise for example in the visualisation of ecological data. In landscape planning, the results of interventions in ecological systems can be represented using the `Arrange` component. Decision makers are thus put into the situation to move around in a planned virtual landscape and then to consider the alternatives.

Further fields where the potential applications are promising are architecture, driving and flight simulators, films as well as games.

Using the combination of instantiation with generative relational growth grammars, some of the restrictions of pure instantiation-based modelling can be compensated. It is now possible to model significant causal aspects of processes of growth, their control being realized by a botanically-tested growth grammar. In this framework it is, e.g., possible to represent biochemical reactions and metabolic reaction networks; see [14, 2] for details. These highly-detailed modelling approaches can now easily be combined with the instantiation-based Xfrog approach for specifying virtual plants.

As further extension, an interactive graphical rule editor would be one next step to develop. This would free the user from the necessity to specify the rules and instantiation components by writing XL code.

Acknowledgements

Research was partially funded by the German Research Foundation (DFG) under the project identifier Ku 847/6-1. All support is gratefully acknowledged.

REFERENCES

[1] Buck-Sorlin, G. H.—Kniemeyer, O.—Kurth, W.: A Model of Poplar (Populus Sp.) Physiology and Morphology Based on Relational Growth Grammars. In: Deutsch, A., Parra, R. B., de Boer, R., Diekmann, O., Jagers, P., Kisdi, E., Kretzschmar, M., Lansky, P., Metz, H. (Eds.): Mathematical Modeling of Biological Systems, Volume II, Modeling and Simulation in Science, Engineering and Technology. Birkhäuser Boston, 2008, pp. 313–322.

[2] Buck-Sorlin, G. H.—Kniemeyer, O.—Kurth, W.: Barley Morphology, Genetics and Hormonal Regulation of Internode Elongation Modelled by a Relational Growth Grammar. New Phytologist, Vol. 166, 2005, No. 3, pp. 859–867, doi: 10.1111/j.1469-8137.2005.01324.x.

[3] Deussen, O.: Computergenerierte Pflanzen. Technik und Design Digitaler Pflanzenwelten, Springer, Berlin, 2003 (in German), doi: 10.1007/978-3-642-55822-1.

[4] Deussen, O.—Hanrahan, P.—Lintermann, B.—Měch, R.—Pharr, M.—Prusinkiewicz, P.: Realistic Modeling and Rendering of Plant Ecosystems. Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '98), ACM, New York, NY, USA, 1988, pp. 275–286.

[5] Du, Q.—Gunzburger, M.: Grid Generation and Optimization Based on Centroidal Voronoi Tessellations. Applied Mathematics and Computation, Vol. 133, 2002, No. 2-3, pp. 591–607, doi: 10.1016/s0096-3003(01)00260-0.

[6] Foley, J. D.—van Dam, A.—Feiner, S. K.—Hughes, J. F.: Computer Graphics. Principles and Practice. Addison-Wesley, Reading, Massachusetts, 1997.

[7] Greenworks: Xfrog Modelling Software. Web Site. Available on: http://xfrog.com, 2017.

[8] GroIMP Developer Group: Web Site. Available on: http://www.grogra.de, 2017.

[9] Hemmerling, R.—Kniemeyer, O.—Lanwert, D.—Kurth, W.—Buck-Sorlin, G. H.: The Rule-Based Language XL and the Modelling Environment GroIMP Illustrated with Simulated Tree Competition. Functional Plant Biology, Vol. 35, 2008, pp. 739–750, doi: 10.1071/fp08052.

[10] Henke, M.: Design and Implementation of a Modular System for 3D Visualization of Plants Using GroIMP Instantiation Rules – Entwurf und Implementierung eines Baukastens zur 3D-Pflanzenmodellierung in GroIMP. Diploma thesis (in German), BTU Cottbus, 2006. Available on: http://www.uni-forst.gwdg.de/~wkurth/cb/html/henke_dipl_final.pdf.

[11] Ijiri, T.—Owada, S.—Igarashi, T.: The Sketch L-System: Global Control of Tree Modeling Using Free-Form Strokes. Smart Graphics, 2006, pp. 138–146, doi: 10.1007/11795018_13.

[12] Kniemeyer, O.: Design and Implementation of a Graph Grammar Based Language for Functional-Structural Plant Modelling. Ph.D. thesis, BTU Cottbus, 2008. Available on: http://opus.kobv.de/btu/volltexte/2009/593/.

[13] Kniemeyer, O.—Barczik, G.—Hemmerling, R.—Kurth, W.: Relational Growth Grammars – A Parallel Graph Transformation Approach with Applications in Biology and Architecture. In: Schürr, A., Nagl, M., Zündorf, A. (Eds.): Applications of Graph Transformations with Industrial Relevance. Springer Berlin Heidelberg, Lecture Notes in Computer Science, Vol. 5088, 2008, pp. 152–167, doi: 10.1007/978-3-540-89020-1_12.

[14] Kurth, W.: Specification of Morphological Models with L-Systems and Relational Growth Grammars. Computational Visualistics and Picture Morphology (Themenheft zu IMAGE 5), Vol. 5, 2007, No. 1, pp. 50–79.

[15] Lau, D.—Arce, R. G.: Modern Digital Halftoning. Marcel Dekker, Inc., New York, Basel, 2001.

[16] Lintermann, B.—Deussen, O.: A Modelling Method and User Interface for Creating Plants. Computer Graphics Forum, Vol. 17, 1998, No. 1, pp. 73–82, doi: 10.1111/1467-8659.00216.

[17] Ong, Y.—Streit, K.—Henke, M.—Kurth, W.: An Approach to Multiscale Modelling with Graph Grammars. Annals of Botany, Vol. 114, 2014, No. 4, pp. 813–827.

[18] Prusinkiewicz, P.—Karwowski, R.—Měch, R.—Hanan, J.: L-Studio/cpfg: A Software System for Modeling Plants. In: Nagl, M., Schürr, A., Münch, M. (Eds.): Applications of Graph Transformations with Industrial Relevance. Springer Berlin

Heidelberg, Lecture Notes in Computer Science, Vol. 1779, 2000, pp. 457–464, doi: 10.1007/3-540-45104-8_38.

[19] Pirk, S.—Stava, O.—Kratt, J.—Said, M. A. M.—Neubert, B.—Měch, R.—Benes, B.—Deussen, O.: Plastic Trees: Interactive Self-Adapting Botanical Tree Models. ACM Transactions on Graphics, Vol. 31, 2012, No. 4, pp. 50:1–50:10.

[20] Prusinkiewicz, P.—Lindenmayer, A.: The Algorithmic Beauty of Plants. Springer, New York, 1990, doi: 10.1007/978-1-4613-8476-2.

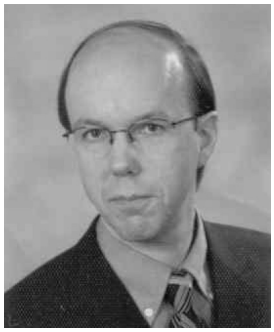
[21] Smoleňová, K.—Kurth, W.—Cournède, P.-H.: Parallel Graph Grammars with Instantiation Rules Allow Efficient Structural Factorization of Virtual Vegetation. Electronic Communications of the EASST, Vol. 61, 2013, p. 17.

[22] Ulichney, R.: Digital Halftoning. MIT Press, Cambridge, 1987.

[23] Xu, L.—Henke, M.—Zhu, J.—Kurth, W.—Buck-Sorlin, G. H.: A Functional-Structural Model of Rice Linking Quantitative Genetic Information with Morphological Development and Physiological Processes. Annals of Botany, Vol. 107, 2011, pp. 817–828, doi: 10.1093/aob/mcq264.

Michael Henke Diploma degree in computer science from the Brandenburg University of Technology Cottbus – 2007 Ph.D. student at the University of Göttingen, Department Ecoinformatics, Biometrics and Forest Growth – 2009–2010 Visiting scholar at Zhejiang University, Hangzhou, China – 2013 Researcher at French National Institute for Agricultural Research – 2014–2016 Researcher at Wageningen UR, The Netherlands – Assistant Lecturer at Brandenburg University of Technology Cottbus and University of Göttingen – GroIMP developer – research fields: functional-structural plant modelling, simulation.

Ole Kniemeyer 1997–2002 Studies of physics and mathematics at the University of Göttingen – 2002 Diploma degree in theoretical physics at the University of Göttingen – 2002–2006 Ph.D. student at the University of Technology in Cottbus – 2006–2008 Visiting researcher at the University of Göttingen, Institute of Forest Biometry and Informatics – 2008 Ph.D. in computer science at the University of Technology in Cottbus – since 2008 software developer at MAXON Computer GmbH.



Winfried Kurth Diploma degree in mathematics and Ph.D. in theoretical computer science at the University of Technology in Clausthal – subsequently junior researcher at the Universities of Göttingen and Bayreuth – 1992 research internship at CIRAD, Montpellier – venia legendi in forest biometrics and computer science obtained in 1999 at the University of Göttingen – 2000–2001 Heisenberg scholarship – 2001–2008 Professor for practical computer science/graphics systems at the University of Technology in Cottbus – since 2008 Professor for computer graphics and ecological informatics at the University of Göttingen – research fields: rule-based languages, representation of 3-d data, functional-structural plant models, simulation.

CHAPTER 6

Fourth Paper

This paper is published as:
Henke M, Kurth W, Buck-Sorlin GH (2016) FSPM-P: Towards a general Functional-Structural Plant Model for efficient model development, *Frontiers of computer science*, 10(6), 1103-1117, doi: 10.1007/s11704-015-4472-8

Authorship

- Winfried Kurth supported the writing of the manuscript.
- Gerhard H. Buck-Sorlin developed the model together with me and supported the writing of the manuscript.

Front. Comput. Sci., 2016, 10(6): 1103–1117
DOI 10.1007/s11704-015-4472-8

RESEARCH ARTICLE

FSPM-P: towards a general functional-structural plant model for robust and comprehensive model development

Michael HENKE (✉)[1], Winfried KURTH[1], Gerhard H. BUCK-SORLIN[2]

1 Department of Ecoinformatics, Biometrics and Forest Growth, University of Göttingen, Göttingen 37077, Germany
2 UMR1345 Institut de Recherche en Horticulture et Semences (IRHS), AGROCAMPUS OUEST Centre d'Angers, Angers 49045, France

Abstract In the last decade, functional-structural plant modelling (FSPM) has become a more widely accepted paradigm in crop and tree production, as 3D models for the most important crops have been proposed. Given the wider portfolio of available models, it is now appropriate to enter the next level in FSPM development, by introducing more efficient methods for model development. This includes the consideration of model reuse (by modularisation), combination and comparison, and the enhancement of existing models. To facilitate this process, standards for design and communication need to be defined and established. We present a first step towards an efficient and general, i.e., not species-specific FSPM, presently restricted to annual or bi-annual plants, but with the potential for extension and further generalization.

Model structure is hierarchical and object-oriented, with plant organs being the base-level objects and plant individual and canopy the higher-level objects. Modules for the majority of physiological processes are incorporated, more than in other platforms that have a similar aim (e.g., photosynthesis, organ formation and growth). Simulation runs with several general parameter sets adopted from the literature show that the present prototype was able to reproduce a plausible output range for different crops (rapeseed, barley, etc.) in terms of both the dynamics and final values (at harvest time) of model state variables such as assimilate production, organ biomass, leaf area and architecture.

Received October 24, 2014; accepted September 16, 2015

E-mail: mhenke@uni-goettingen.de

Keywords functional-structural plant model, prototyping, modelling standards, teaching / learning FSPM, GroIMP

1 Introduction

Current crop growth models are often based on a selection of general processes describing the mechanisms of primary production. Generally, in these models factors determining potential, attainable and actual crop growth are distinguished, allowing the same model to be used for a variety of crop species, given the availability of a standard set of crop parameters [1].

In contrast to these process-based models, functional-structural plant modelling (FSPM) has its origin in purely structural modelling, and within this paradigm models are developed in a much more *ad hoc* way. Developers of such models are often plant biologists who are keen to explore the impact of plant architecture (organ geometry and topology) on a limited range of physiological effects, e.g., the effect of leaf angle distribution on canopy radiation interception. These workers are often lacking experience in programming yet have a clear overview of the structure and scope of their model. Another group consisting of programmers and computer scientists who are interested in biological systems considers it as a challenge for the application of the rule-based paradigm. Thus, while plant biologists use an FSPM approach to study the effect of a static architecture on light interception and leaf photosynthesis, computer scientists study the way complex tree architectures could be created using a

very limited set of production rules. Most physiological functions that are currently used in crop models could be used in the same general way in FSPM, and structures, such as plant organs, could be defined generally and then implemented for a crop species.

Current FSPMs of crop plants (e.g., for peach [2, 3]; rice [4]; cut-rose [5]; rape [6]; barley [7–9]) contain common components and recurring parts (e.g., for photosynthesis, growth and extension of organs, build-up of the structure through formation of phytomers at the shoot tip and through branching), which could be generalized and re-used as subsystems. One possible solution to benefit from former models is a prototype as a base for new models.

FSPMs with a generic character are not numerous. AmapSim [10] is in its core a purely structural model, allowing the linking of functional components as external programme modules. Based on the notion of physiological age, it was primarily adapted to trees. However, it does not inherently support the feedback of carbon assimilation on growth and structural development, which makes it less useful for crop plant simulations. GreenLab [11] uses the concept of physiological age in its structural part; it was used to model several crop plant species. Furthermore, the feedback of assimilates on structural growth was included in the advanced version GL3 [12]. Because of the simplified description of source functions, it was considered as "intermediate between FSPM and (purely) process-based models" [12]. Breckling [13] designed an FSPM for a generic, modular plant and implemented it in the object-oriented language Simula. However, to adapt it to real crop, the Simula source code has to be modified. Finally, LIGNUM [14] uses annual time steps for growth and was designed for Scots pine in its first version; later it was adapted to other tree species. These adaptations require changes in the code again.

Here we present an FSPM prototype which goes a step further than the FSPM approaches described in the previous paragraph: while plant architecture is still largely descriptive (i.e., organ geometry and arrangement is input to the model), the majority of processes related to the functioning of sources and sinks are implemented in a generic way, allowing the computation of resource allocation according to the demand of each organ. The model is written in the rule-based language XL and implemented on the software platform GroIMP (see Section 2.1). This model uses an object library in which each botanical object is provided with predefined state variables and methods representing internal processes (photosynthesis, growth, maintenance and growth respiration, storage and remobilisation of assimilates). Because of its structured, object-oriented design, modular set-up and a user manual provided with it, it is easy to parametrize, use and extend. The prototype has not yet reached its final degree of generality; some default values and procedures are chosen arbitrarily in order to allow the user to get started rapidly and will be replaced by more general or exchangeable parts in a future version. In its current version, our model and this paper are meant to provide scientists and students of the plant sciences with an easy access to the FSPM paradigm, which might be a valuable additional tool for hypothesis testing.

2 Materials and methods

2.1 Modelling language and platform

The present FSPM-P (FSMP-Prototype) is written using the modelling language XL (eXtended L-System modelling language) [15], a rule-based language which supports the specification of graph grammars generalizing L-systems [16, 17] and which is at the same time a superset of the language Java. Hence each Java programme can easily be embedded in an XL programme. The modelling platform GroIMP is platform-independent, open-source and freely available[1)]. GroIMP is employed for model implementation and visualisation. It is designed as an integrated platform which incorporates modelling, simulation, visualisation and user interaction, and provides a compiler and development tools for XL [15].

2.2 General features of the FSPM-Prototype project

The FSPM-Prototype project comprises two elements, the FSPM-Prototype model (FSPM-P) (current version: 0.4) and a user manual as free download from the model gallery at `www.grogra.de`. The model is subdivided into separate modules: a main file for model initiation and control; a file for defining objects (such as plant organs) and their properties; a library of photosynthesis rate models to be coupled with leaf objects; global parameter definition; a file containing auxiliary tools and functions like charts. To make FSPM-P an accessible and comprehensible tool, an extensive user manual which provides a detailed *Model description* was written.

The FSPM-P is a fairly extensive set of XL modules and Java implementations comprising the description of a fairly comprehensive set of biophysical and physiological processes such as radiation interception, photosynthesis, growth

[1)] https://sourceforge.net/projects/groimp/

and development. The hierarchical scale at which the model is implemented is the same as that of the organ, but processes can also be aggregated at the plant individual scale.

In the following sections, we will describe some features of the current model: definitions for plant organs, work flow, growth and development, the latter being based on source (local photosynthesis of assimilates, storage of assimilates locally and in a central pool) and sink functions (reallocation of assimilates for growth as a function of sink strength, i.e., relative potential growth rate with the source/sink ratio used to steer growth and branching).

2.3 Plant definition

Within the FSPM-P model a plant species is defined by three files, 1) a parameter file, with species-specific parameters mainly for growth and photosynthesis, 2) all rules for morphology, cutting, transport, and organ update etc., are collected in a rule file, 3) and a module file listing predefined plant organs. In addition, there are different hierarchical scales within the plant organ definition: basic organs (seed, root, meristem, bud, leaf, internode, flower, fruit, etc.) and organ aggregations (individual and shoot). They contain, e.g., standard variables and summary functions based on XL-queries to get fast information about the internal plant state.

The object-oriented design of the FSPM-P with its strict separation of species-, parameter-, and infrastructure-specific parts, allows to simulate more than one species at once, which can be done by adding an additional file set for the new species and to activate its simulation in the main model loop. Besides, further things like arrangement of the individual plants and their interactions needs to be defined by the end user. Currently, shading effects between different species are the only emergent process that the FSPM-P provides. Other processes like sensing, independent of the above or below ground, competition for nutrients or any stress effects need to be implemented by the user.

Plant structure and topology are based on measurements. They are defined by morphological rules and therefore input to the model. For the following description of the prototype, hypothetical "observations" based on real data are used.

2.4 Model work flow

During initialisation, global parameters and variables are loaded, direct and diffuse light sources (sun and sky light) as well as a single plant or plant stand are put into the scene with their initial parameters. In a main loop (method `run()`), a single growth step is repeated until the user stops it manually or after a predefined time. For each growth step, four sub-steps are carried out: update the sun and sky module; run light model; apply rules; update output. Finally, some statistical outputs, e.g., amount of harvested biomass, are generated and pasted into a chart. The work flow in the model is summarized in Fig. 1.

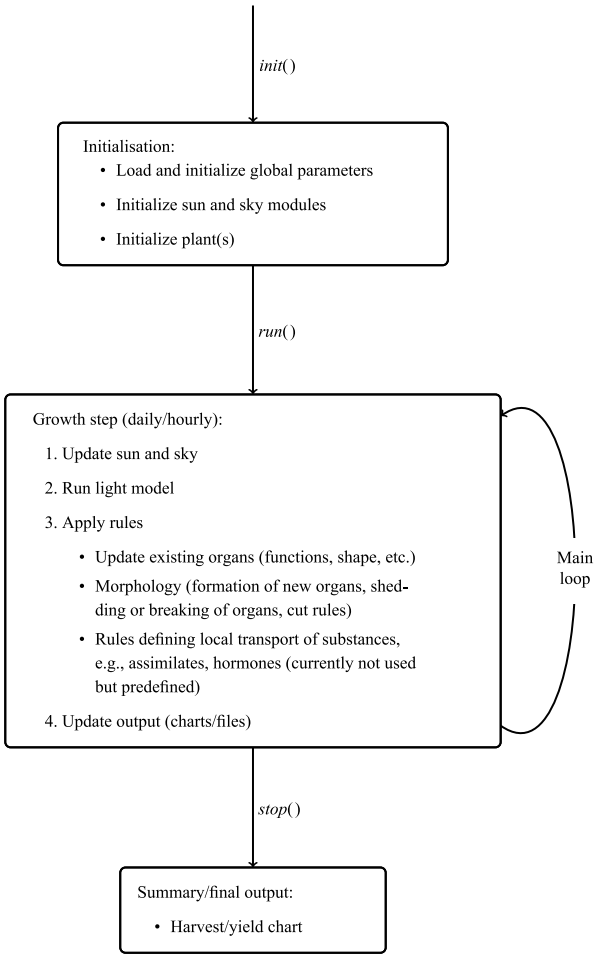


Fig. 1 General model work flow: after the initialisation, the model will be executed during the main loop before final output is generated

The `applyRules()` function is the only species-specific function within the main loop. For each species a user wishes to simulate one such function call needs to be included. Consequently, all simulated species are sharing the same scenario and environment condition, while the type and number of processes as well as their temporal resolution does not have to be the same.

To improve legibility of the code, the `applyRules()` function is also clearly subdivided.

The different methods invoked are described in the following section.

```
protected void applyRules() {
    morphologyRules();
    cutRules();
    transportRules();
    organUpdates();
    otherRules();
}
```

2.5 General processes

According to their different functionalities, there are rules for morphology (formation of new phytomers at the tip of an axis, and branching), cutting/abscission of organs, transport, organ updates (of internal parameters, e.g., length, diameter, mass, as well as processes, e.g., growth and maintenance respiration), and other rules (mainly for information about the current state of the model).

The function `morphologyRules()` comprises the following rules:

1) Germination If conditions for germination are satisfied, replace seed with root and a meristem (containing the shoot apical meristem). The meristem has three parameters: the plant individual that it belongs to, the rank (running number of phytomers in the shoot, counted from the base), and the branching order. The two last parameters are initialized with 1.

2) Development The corresponding rule finds all meristem objects that fulfil certain conditions, and replaces them with a phytomer, i.e., an internode, a leaf, and a new meristem or bud. The final rules are analogous to the first bud rule, but replace the bud with a flower, and the flower with a fruit, respectively, if the conditions for these processes are met.

The conditions for bud break are 1) topological: rank and order; 2) light: a bud must absorb more light than a threshold; 3) temperature: mean air temperature must be in a suitable range; 4) the average source/sink ratio of the plant has to be bigger than a user-defined threshold. The latter condition ensures that the plant currently has sufficient reserves to form new phytomers; 5) a bud break probability model, e.g., by a semi-Markov chain; and 6) phyllochron. Finally, as an exceptional case for formation of new sinks in a situation of overproduction of assimilates, sleeping (dormant) buds can be reactivated when a specific average source/sink ratio is reached.

A newly-formed meristem is initialized with a species-specific phyllochron (measured in thermal time units), which expresses the developmental phase between bud initiation and bud break to form a new phytomer; this internal variable is decreased at each organ update by the actual average temperature. When the phyllochron is counted down to zero (or has a negative value), one condition for phytomer production is fulfilled and the rule may be executed.

Growth and development are based on source (leaf photosynthesis of assimilates and release from a storage pool) and sink functions (reallocation of assimilates for growth as a function of relative sink strength, storage in the pool).

Photosynthesis in the model is restricted to leaf blades; photosynthesis of other green organs such as sheaths, stems and walls of immature fruits is currently not considered (however, this would be possible without problems as all these organs implement the organ superclass).

Simplified transport of water is implemented to illustrate the usage of the ordinary differential equation (ODE) framework of GroIMP [18]. An inexhaustible water reservoir provides the water that can be absorbed by the root. The absorbed water is piped through internodes and leaves driven by a temperature sensitive transpiration function within each leaf.

2.6 Radiation model and light interception

GroIMP provides two ways for calculation of light interception: 1) a central processing unit (CPU) based implementation [15, 19] and 2) an implementation able to use multiple devices in parallel inclusive of the graphics processing unit (GPU) called GPUFlux [20]. The user has to choose the method that is used to simulate light distribution and local light interception. These methods are based on a reversed path tracer algorithm with Monte-Carlo integration [21] and use light sources and geometric objects placed into a scene. The selected radiation model is invoked once per simulation step, and is applied to a scene created within the modelling environment GroIMP. GroIMP provides several types of light sources. As default setting, we use a directional light source to simulate direct sun light whereas diffuse sky light is simulated using an array of 72 directional lights positioned regularly in a hemisphere in six circles with twelve lights each, with emitted power densities being a fixed function of the elevation angle [22, 23]. As alternative sky model, an implementation based on Preetham [24] is integrated into GroIMP too. It is planned to provide several established sky models as alternative choices in a future version of FSPM-P. Both the sun and the sky object are dynamically updated at each step as function of the Julian day of the year and the time of the day [h]. The light model is run with two parameters: total number of rays produced by all light sources in the scene, and

the number of times a reflected or transmitted ray is traced. In the default configuration, we recommend to use at least ten million rays for the CPU ray tracer in the daily, a twenty-fourth of it in the hourly run mode and a recursion depth of ten. For the much faster GPU ray tracer, the number of rays can be easily increased up to 200 millions and even more, in order to enhance accuracy of the obtained light distribution.

Once a leaf is formed, it is identified with a label, and its absorbed radiation is determined as a spectrum at a run of the light model. This spectrum is converted from $[W/m^2]$ to Photosynthetic Photon Flux Density PPFD $[\mu molPPFD/(m^2 s)]$ by multiplication with a conversion factor (2.275 in the case of daylight [1]).

To simulate the distribution of direct PAR during the day, the position of the sun is computed according to Goudriaan and van Laar [1], and the normal vector representing that position is transformed into a vector representing the orientation of the directional light source, updated at an hourly rate.

The advanced GPUFlux ray tracer [20] supports multiple devices for simultaneous calculations, e.g., all threads of a CPU and, in addition, a GPU, which reduces the time for light calculation dramatically. Besides this significant acceleration, the GPUFlux ray tracer provides the possibility to calculate the full spectrum of light, which opens new application areas, as discussed in Subsection 3.3.

2.7 Source implementation

The main carbon sources for a plant in our model are the leaves (after the carbon stored in the seed has been consumed during germination). Intermediate storage and remobilization of starch is considered only in the root organ, where at each time step a small amount (1–2.5%) of the produced assimilates is stored. This storage pool is used as source only in the last developmental stage, during fruit development, and during times where environmental conditions are unfavourable for growth. (For convenience, it is located in the root organ, though in reality it might be distributed all over the plant).

Integrated into the model is a library of photosynthesis rate models (differing in complexity from simple light-response curves to biochemical Farquhar-type models), which can be selected with a global parameter (see Section 2.8).

At the level of the individual, all produced assimilates of all leaves, minus a certain fraction local demand LD which is stored in the local pool of the leaf for its own growth, are collected only for calculation purposes in a temporary assimilate pool AP $[g/time]$:

$$AP_t = (1 - LD) \times \sum_{i=1}^{n} PS_i. \quad (1)$$

This is done automatically at each time step by calling the `update()` function in the Individual module, see FSPM-P user manual for more details.

In the current setting, the dynamics of the source is characterized by five phases. In the first phase, initial carbon is provided by the seed, which is rapidly exhausted during germination in the second phase. After unfolding of the first leaves, photosynthesis commences. During the third phase, source and sink are in balance, and the temporary assimilate pool AP is emptied at each step (source/sink ratio fluctuates around one). In the fourth phase of vegetative establishment, source strength is bigger than sink demand and assimilate reserves are stored in the storage pool. During the fifth phase of maturity, fruit formation takes place, and for this the storage pool is used as a further source in addition to the assimilates provided by photosynthesis at each step, but which are declining due to leaf ageing. Feedback inhibition of the photosynthesis rate, due to a local excess in assimilates (low sink strength), has not been implemented.

2.8 Photosynthesis models

The temporal resolution of our model can currently be switched by the user between daily and hourly run mode. The required weather file is automatically loaded and used as an input to the photosynthesis model, containing daily or hourly values of mean temperature, global radiation, and relative humidity. If only daily totals of global radiation are available, the expected value for a given hour of the day can be estimated using a sine function [1], assuming atmospheric transmissivity to be a function of global daily radiation and solar elevation, as described by Gijzen [25].

The model's runtime is only restricted by the availability of weather data. Currently, the model provides only a single weather file with daily values from the weather station Haarweg, Wageningen, and the Netherlands[2)] (366 days recorded from January 1, 2008 to December 31, 2008), but weather files comprising several years can be used, too.

The method `getPAR()` defined in the leaf module is used to calculate the photosynthetically active radiation PAR $[\mu molphotons/(m2s)]$, by taking the actual absorbed radiation and dividing it by the leaf area.

The FSPM-Prototype provides a portfolio of nine photosynthesis rate models, three versions of biochemical leaf photosynthesis rate models considering leaf temperature, PAR,

[2)] http://www.met.wau.nl/haarwegdata/

CO2 concentration and leaf energy balance: the LEAFC3-N photosynthesis model [26] with consideration of nitrogen [27, 28], Baldocci [29], and the model by Kim and Lieth [30]; furthermore, models based on simple light-response curves [31–37] are included.

The user can select one of these photosynthesis models in the global parameter file, to be used in the model runs.

2.9 Sink activities and their relationship to the source

The timing and growth duration of active sinks drives the conversion of assimilates to harvestable dry matter. In our FSPM approach, the overall control of sink activity is prescribed by growth and development rules, and the overall biomass production is an emergent property of the integration of these rules applied to the growing structure over simulated time, see Fig. 2. In addition, the rate of extension of each organ is described by a sigmoid growth function, e.g., the beta growth function [38]:

$$w_t = w_{\max}(1 + \frac{t_e - t}{t_e - t_m})(\frac{t}{t_m})^{\frac{t_e}{t_e - t_m}}, \quad (2)$$

with $0 \leqslant t_m \leqslant t_e$, where $w_{\max}$ is a maximum value of w_t, reached at time t_e, and t_m is the time when growth rate reaches its maximum.

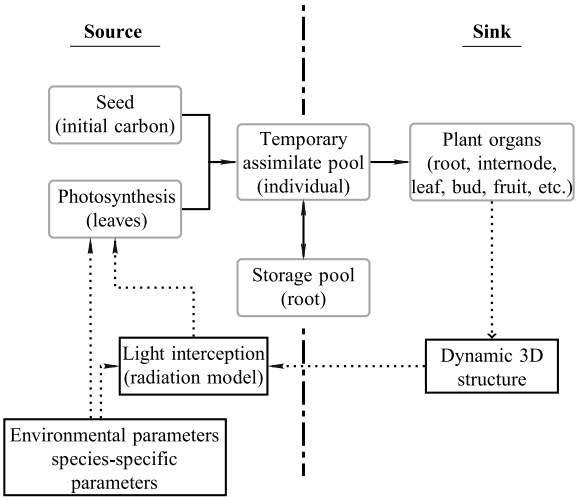


Fig. 2 Schematic overview of source/sink relationship used within the model. After the seed storage is exhausted and the first leaves are developed, photosynthesis takes over as main source process

Here $w_{\max}$, t_m, and t_e are organ-dependent input values, which should be based on real measurements for a given species. Typically, such a growth function also depends on the (acropetal) rank of the leaf or internode (as has been shown for barley by Buck-Sorlin 2002). For FSPM-P, we use hypothetical but realistic values, as shown in Fig. 5(b) for internodes.

The sink strength of a growing organ i at time t can be approximated by its potential growth rate $PGR_{i,t}$, which is the instantaneous increment in dry matter w and can be described by the derivative of the above function:

$$PGR_{i,t} = \frac{dw_{i,t}}{dt} = c_{\max}(\frac{t_e - t}{t_e - t_m})(\frac{t}{t_m})^{\frac{t_m}{t_e - t_m}}, \quad (3)$$

where $c_{\max}$ is the maximum growth rate at time t_m [38]. The method `getPGR()` is used to compute the potential growth rate in each organ [$drymass/time$].

As an alternative, other preimplemented growth functions such as Chapman-Richards [39] or a logistic function are provided.

Global sink demand sd [$drymass/time$] is defined as the sum of all potential growth rates PGR of concurrently growing organs:

$$sd = \sum_{i=1}^{n} PGR_{i,growingorgan}. \quad (4)$$

The relative sink strength RSS [−] is calculated for each organ i by:

$$RSS_i = PGR_i / sd. \quad (5)$$

Multiplication of RSS with the temporary assimilate pool AP [$g/time$] results in the actual/realized growth rate AGR [$g/time$] thereby assuming that AGR cannot be bigger than PGR:

$$AGR_i = \min(PGR_i, RSS_i * AP_t). \quad (6)$$

In the model, this is implemented for each organ in the `getAGR()` method, where AP is calculated using the method `getTemporaryAssimilatePool()` of the associated individual.

Once growth of an organ takes place, the actual growth AG is added to the *dryWeight* of each organ, and the temporary assimilate pool is updated accordingly. The unused assimilates at time t is the difference in all assimilates available for growth and sum of respiration losses at the same time step t:

$$AP_{t+1} = AP_t - \sum_{i=1}^{n} R_{i,t}, \quad (7)$$

where respiration R for an organ i at time t is:

$$R_{i,t} = MR_{i,t} \times DW_{i,t} + GR_{i,t}. \quad (8)$$

Maintenance respiration MR is computed as an organ-specific fixed proportion of structural biomass, whereas growth respiration GR is defined as the amount of assimilates [g] respired when producing one gram of new biomass [40]. It can be conveniently expressed as a conversion factor, (g[$glucose$]/g[$newdryweight$]), i.e., the total amount of assimilates per gram new biomass [−]. Thus GR is proportional

to the growth rate as described in Goudriaan and van Laar [1]. Both terms are subtracted from the temporary assimilate pool at each step.

If the temporary assimilate pool is not completely exhausted, the excessive assimilates will be added to the storage pool. This storage pool will be activated if the environmental conditions cause an emergency situation for the plant or the fruit formation.

Each plant organ module implements an `update()` function with two parameters: the amount of absorbed radiation and the current mean temperature. At each call of this function, the internal age counter is increased and the carbon budgets are updated as described above.

2.10 Vegetative and generative development

To simulate vegetative and generative development, a small set of growth, developmental and branching rules is repetitively applied to a Bud module and all of its ensuing organs, leading to the visible phenotype. This type of repetitive application of rules is straightforwardly implemented in the rule-based language XL which supports the specification of graph grammars generalizing L-systems [16, 17]. The structural framework created thus is used to simulate and analyse the dynamics of assimilate flow as dictated by local (potential) growth rates and assimilate availability in the temporary assimilate pool. The model simulates phenology, including germination, seedling stage, juvenile (vegetative) and adult (generative) plant, and finally harvest maturity.

Formation of a new organ from a meristem occurs after some intrinsic delay (phyllochron). The main stem and tillers are created within the limits given by topological parameters (i.e., maximum rank and order). A new leaf is formed with an initial dry weight which is converted to the initial length and diameter, plus a new bud initiated at the tip of the shoot, and the rank increased by one. At the same time, the phyllochron is set to its initial value (as specified by a species-specific parameter PHYLLOCHRON).

The potential extension and final dimension of organs (leaves, internodes, etc.) depend upon their rank and age, while the actually achieved dimensions are also a function of sink competition and assimilate availability, as described in Section 2.9. Leaf dimensions are determined using the beta growth function [38], calculating dry matter increment as a function of time, and dry matter is then converted into leaf shape (length and width) using a constant conversion factor for simplicity.

Once the generative stage is attained, flower formation takes place, followed by fruit formation according to a user-defined fertilisation rate which uses simple stochastic mechanisms. Fruits / seeds formed from flowers will grow and change their colour according to the stage of maturity attained, limited by potential growth rate.

2.11 Source/sink ratio for model regulation

A dynamically calculated average source/sink ratio *SSR* (calculated over a number of previous steps), which exhibits a range of values (usually between 0.1 and 1.1), is used to control the carbon budget in the model [41–44].

The idea is to keep source and sink in balance and to up- or-down regulate the average *SSR* in such a way that it stays at roughly a value of one. Depending on the value that *SSR* attains, sink or source regulation in the model takes place in different ways: if the *SSR* gets too high, the source strength is decreased by decreasing photosynthetic efficiency. Alternatively, sink strength is increased by increasing the number of growing organs (bud break) or their potential growth rate, and by increasing storage of assimilates in the temporary carbon pool.

When source capacity exceeds global demand (i.e., by all growing sinks), a possible measure is the down-regulation of the source, specifically the photosynthetic efficiency, by multiplying the result of the photosynthesis function with a factor. This regulation factor is based on the difference between the average source/sink ratio and one (1-avg(*SSR*)). Otherwise, if environmental conditions turn very unfavourable and if then, as a consequence, assimilate production is strongly reduced, photosynthetic efficiency cannot be up-regulated again to counterbalance the unfavourable conditions.

Another possibility for a plant to react to a surplus of assimilates is to produce new sinks by increasing the rate of bud break, thus creating new shoots. Conversely, as a reaction to a low source/sink ratio (high sink demand or low source capacity, or both), the photosynthetic efficiency can be increased (see above), and weak sinks can be removed from the plant (e.g., flower or fruit abortion), or PGR of organs reduced.

2.12 Implementation of processes and module communication

All processes are implemented as functions inside each organ definition with an organ-specific parameterization (Table 1).

According to the organ superclass, all organs are having processes implemented in a standardised way, which makes it easy to define an equal function for all organ types and to use it for organ update (see Section 2.9 for details).

Table 1 Implemented processes for different organ types

Process	Seed	Root	Bud	Internode	Leaf	Flower	Fruit
Maintenance respiration	-	+	+	+	+	+	+
Growth respiration	+	+	+	+	+	+	+
Photosynthesis	-	-	-	-	+	-	-
Potential growth	-	+	+	+	+	+	+
Actual growth	-	+	+	+	+	+	+

With this technique and the combination of the powerful graph query language integrated into XL [15], it is possible to get information about plant state variables like dry weight of all organs which can be determined by:

```
sum( (* Organ *).getDryWeight() )
```

in an elegant way. The graph query `(* Organ *)` searches all instances of the type `Organ` within the graph and returns them. In the second step, the function `getDryWeight()` is called for each object found. Finally, all results are aggregated by the sum function. Most of the functions implemented in the individual module are defined according to this scheme:

```
public float getDryWeight() {
 return sum(
   (* x:Organ, (x.getIndiID()==indiID) *)
   .getDryWeight()
 );
}
```

In case more than one individual is initialized, the condition `(x.getIndiID()==indiID)` makes sure only to output the dry weight of organs of the same individual, i.e., with the matching individual identification number (`indiID` is a constant defined in the parameter file).

Another principle that we use is known from object-oriented programming as “encapsulation” or “information hiding”, where information or data are protected from direct access from outside. All conditions for use inside the rules are implemented as functions of organs, e.g., the conditions for a seed to germinate are implemented in the form of a boolean function `isGerminationConditions()` inside the seed module definition, and are used as a simple function call in the rule definition.

```
s:Seed, (s.isGerminationConditions()) ==>
    Root(s[indi]) s Bud(1, 1, s[indi]);
```

2.13 Adding a new process to the model

Two main procedures can be applied to add a new process to the FSPM-P. The first procedure involves the linking of a programme which describes the process to be added and which is written in another language. XL being an extension of Java, such a programme could be wrapped using a Java interface allowing the inclusion of libraries (e.g., Apache Commons[3] or JScience[4]), packages and implementations from other languages. Though possible, this is not part of the philosophy of FSPM-P, because other approaches like OpenAlea [45] are much more tailored to conduct “gluing” of heterogeneous models (besides, GroIMP has already provided an http-based interface Open GroIMP, which is used to communicate with OpenAlea).

The second and preferable procedure to integrate new processes into the model is to implement them directly in the FSPM-P code. The object-oriented approach used in FSPM-P facilitates the implementation of a new function. By implementing the new function in the definition of the general organ superclass, it becomes available for all organ types, and then this new function can be adapted or modified in the definition of the concrete organ type if required. For example, growth respiration is always calculated taking the actual growth rate and multiplying the latter with a constant, organ-type specific factor. The general function for growth respiration and its actual implementation for each organ type are stated in the definition of the organ superclass:

```
// growth respiration [g]
public float getGrowthRespiration() {
  return getActualGrowth()
    * ASSIMILATE_LOSS_GR_ORGAN
      [getOrganType-()];
}
```

In cases where the growth respiration as defined in the organ class does not fit, it can be overwritten in the definition of the specific organ type.

The main temporal resolution of our model is either daily or hourly run mode. To manage different time steps between different processes, e.g., to compute morphological rules each day and light interception at each hour between 6am and 8pm, the user can add conditions for the execution timing of each process:

```
if(hourOfDay == 12) morphology();
if(hourOfDay >= 6 && hourOfDay <= 20) {
  lightInterception();
}
```

[3] http://commons.apache.org
[4] http://jscience.org

2.14 Visualization

FSPM-P implements a general model, which means that it is not associated with a fixed plant species and thus also not parametrized for a certain species. The parametrization is chosen such that plausible (qualitatively realistic) growth and development will be generated. Figure 3 shows the generated 3D structure at different ages. Additionally, a measurement ruler for visual comparison has been inserted, as well as a black, one square meter large patch as ground which serves for verification of the light model, i.e., to determine the amount of light reaching the ground.

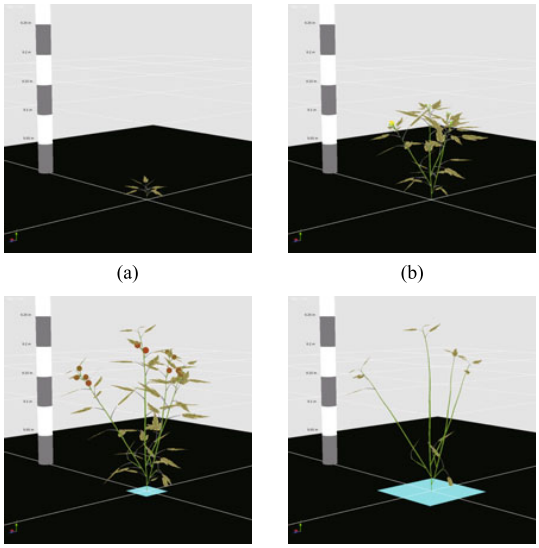


Fig. 3 Generated 3D structure of the FSPM-P model at different development stages. (a) Age 25: juvenile plant; (b) age 50: young plant, first reproduction organs (flowers) occur; (c) age 75: adult plant, fruits at different maturity levels have developed; (d) age 110: terminal stage, fruits have dropped (or been harvested), most basal leaves have been shed due to leaf mortality

3 Results and discussion

3.1 Simulated model output

To monitor and document the dynamics of growth and development processes, a variety of charts have been implemented, e.g., dynamics of organ dry weight and length.

Even without a proper parametrization for a certain crop species, the model has already exhibited general patterns similar to those found in plants, with respect to the phenology of growth stages or stem extension dynamics. Figure 4(a) shows simulated dry weight of leaves as a function of leaf rank. It can be seen that most leaves do not reach their potential dry weight, probably because of the competition for substrate among too many concurrently unfolding leaves while source leaves are still limiting.

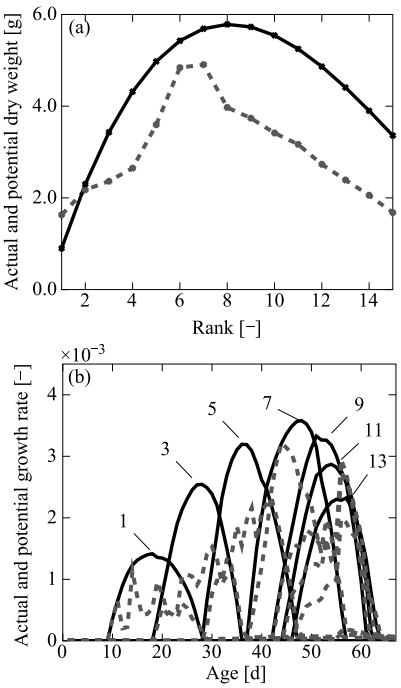


Fig. 4 Simulated model output concerning leaf blades. (a) Final dry weight given as input for the potential growth rate (solid curve) and simulated final dry weight of main leaf blades (dashed curve); (b) potential (solid lines) and actual (dashed lines) growth of main stem leaf blades (rank 1, 3, 5, 7, 9 11 and 13) input

In the current implementation, the final dry weight is rank-dependent for leaves and internodes, while for other organ types it is considered to be equal for each rank. The values used in the FSPM-P are hypothetical, to be subsequently replaced with real measurements. For this prototype, which is a showcase, we took sample data which can be described by a polynomial function.

On the other hand, basal and median leaves nearly reach their potential dry weight: early leaves have little competition with other organs, whereas growth of late leaves is supported by the source strength of many older leaves and (at least before onset of fruit growth (a strong sink)) again experience little competition with other growing organs. The potential and actual growth rate of main stem leaf blades is shown in Fig. 4(b). It can be seen that the realized growth is almost always smaller than potential growth. Since the model is not specific for a certain crop, this has no further meaning. However, if the model had been parametrized for a species, this could mean that the assimilation rate given by the photosynthesis model is too low (due to insufficient photosynthetic

efficiency). As plant growth is almost never reaching its potential but is limited by a shortage of nutrients, water, or light, suboptimal temperatures or pests and diseases, the measured growth rate by definition can not surpass the potential growth rate.

Figure 5(a) demonstrates the simulated dynamics of the carbon assimilation (dry weight) of main stem internodes. According to the parabolic shape of measured final dry weight of internodes (Fig. 5(b)) as a function of rank used as input to calculate potential growth, the simulated final dry weight (Fig. 5(a)) shows the same pattern of internodes with same weight.

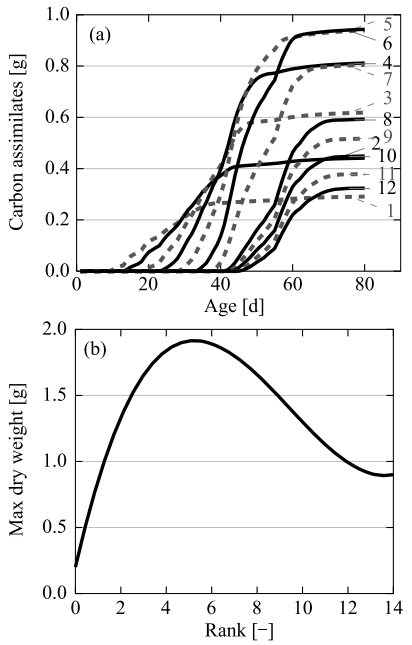


Fig. 5 Comparison of final dry weight of internodes and the measured input function. (a) Simulated dynamics of the carbon assimilation for internodes of rank 1–12; (b) hypothetical "observations" of maximal dry weight as function of rank used as model input

3.2 Importance of a prototype approach

The growing recognition of the FSPM approach, as a logical continuation of the crop modelling tradition [46] (see also the other articles in that special issue on FSPM), necessitates the provision of possibilities for efficient model development as well as for maintenance, support and enhancement. An FSPM, like any other computer programme, can draw substantial benefit and advantage from computer science techniques, mainly software engineering, e.g., object-oriented programming, modularisation, design patterns, software re-usability and basic programming standards [47]. This can enhance both the models themselves and the development process, turning it more structured, efficient, and clearer.

By applying such good practices, models will become easier to understand and better comparable, and submodels can be replaced more easily. Development, combination, implementation, calibration and validation of models can equally benefit from such good practices. The establishment of the best practice in FSPM is a solution for recurring problems, and would rationalise work and enhance productivity as it reduces time for coding, testing and documentation. A predefined and consistent solution like a prototype can also provide standards for testing of parts or the whole model.

A related approach, OpenAlea [45], is a distributed collaborative effort to develop Python libraries and tools that address the needs of current and future work in Plant Architecture modelling. OpenAlea includes modules to analyse, visualize and model the functioning and growth of plant architecture. However, the difference is that OpenAlea essentially links different programmes (potentially written in different languages and exhibiting different compilation states: dll, source code, etc.), whereas our approach is a core FSPM that runs a priori, and that has already included the main functional elements (light interception, photosynthesis, etc.), in the same programming environment and language (GroIMP and XL).

The GreenLab approach [11] is comparable to the present model, as it provides a fully runnable model that can be parametrized for different species. However, its source function being based on radiation use efficiency and lacking internal transport, it falls short of the generality which we consider as necessary for an extensible FSPM. In this respect, it is closer related to CANON [48], in which a composite design pattern was implemented at the phytomer level for use in a FSPM.

FSPM-P can be seen as the first step to a general FSPM which, in its first version, is presented as a conceptual model including a user manual with explanations about experiment set-up, measurement protocols, data processing, model description and parametrizations, and the model itself.

In terms of a model classification, e.g., the pedigree of "de Wit" models [1, 49], our approach is not strictly comparable as it explicitly considers structure in 3D. However, it can be classified according to the (fairly large number of) processes it describes and the level of detail it provides, as a potential production model working at the physiological level of detail [1]: it neither consider the effect of limitations of water and nutrients (for this, an extension to a root-soil interface model would be necessary) nor the effect of pests and diseases on crop production, yet it considers three of the four main ecophysiological processes listed by Goudriaan and van

Laar [1] — carbon assimilation, plant development, and respiration, disregarding plant transpiration. However, since the LEAFC3 model, which is provided in our library of photosynthesis models, also computes potential evapotranspiration, an extension to cover plant transpiration is within reach.

3.3 Possible further application areas

An important feature of FSPM-P is the fact that it already constitutes a running model, which can thus be used straight away. Its use as a departure point for developing a dedicated FSPM of a certain crop is thus obvious.

A further, immediate application is its utilization in teaching and for presentations in the plant sciences where it is often necessary to demonstrate a process in a general way. Crop models without a consideration of plant architecture like LINTUL [50] or SUCROS [51] have been successfully used for teaching purposes [1]. Our approach currently permits the modelling of both individual plants and plant stands (canopies), where the latter are potentially consisting of a mixture of two or more different species. The possibility to model mixed stands makes it suitable for application in intercropping. The more or less concurrent cultivation of two crops in the same field is a very important technique, e.g., in Chinese agriculture. While it seems to be more resource-use efficient than conventional mono-cropping, it also poses substantial challenges with respect to understanding the underlying mechanisms. The most common advantage of intercropping is the production of greater yield on a given piece of land [52]. Furthermore, Ouma [53] also took risk minimization and reduction of soil erosion into consideration, and increased food security as advantages of intercropping. Both publications illustrate clearly the high potential of intercropping as sustainable alternative. An FSPM-P adapted for an intercropping system could be used to investigate and analyse competitive and facilitative relationships between the crop species involved in detail, both above-ground and below-ground, and to elucidate dynamic interactions in space and time at the level of plant organs (e.g., leaves and roots). Calculations done using the FSPM-P would thus help to explain eco-efficiencies in field experiments on the basis of causal ecological mechanisms, and could then be used to explore opportunities for improved intercrop performance by modified system design (species choice, sowing date, planting pattern, irrigation and fertilisation).

Modelling interaction between root systems of different species with each other and with the soil, is an essential element in the investigation of intercropping systems. Modelling the soil requires a discretization of space into cubes with properties such as nutrition, resistance, or water status, and of the dynamics of water and nutrient movement. We have conducted a preliminary implementation study [54], in which we have created a simple yet modular root-soil interaction model. Such a model could thus be used as a generic module in a larger intercropping model system.

3.4 Possible extensions

The range of possible extensions is quite diversified and could encompass the following:

- More detailed carbon storage and transport concepts The central carbon pool concept is an extreme simplification and biologically not well founded. A transport-based concept with local organ pools would be more realistic. The latter is a concept which is very relevant for our approach. Once this extension is implemented, our model could be used to test hypotheses from plant physiology, such as central versus local pools, ranges and modes of transport (e.g., diffusion, convection and active transport).
- Extension to a general tree model The current version of FSPM-P is mainly adapted to small plants with a vegetation period of less than one year. However, it would be interesting and not difficult to extend and change FSPM-P to simulate perennial and polycarpic trees.
- Component-based plant model The ultimate objective of this project is to design a user-friendly general FSPM with generic modules representing functions and processes, plant organs, architectural characteristics or communication and transport which can be used as components and simply combined to a model using a kind of graphical editor. Such an approach with independently developed, verified and reusable components can further facilitate the comparison and exchange of submodels as well as their evaluation and standardization.
- Calculation of spectral light The use of the GPUFlux ray tracer provides several opportunities to not only simulate light distribution over the full spectrum of light, but also allow to calculate, e.g., relations between red and far red light. In combination with simulations of artificial light sources with specific spectral power distributions and physical light distributions, common light conditions, e.g., found in greenhouses or climate chambers, can be reproduced and further used for functional-structural plant modelling.

4 Outlook and conclusions

The model presented here is the first step towards establish-

ing a general model with standardised modules, processes and communication structure, which enables a clear model design, and is easy to parametrize (see Appendix B), understand and extend.

This systematic approach provides all the necessary infrastructure and documentation to develop efficient FSPMs based on their own measurements for different target groups (with or without knowledge of programming or modelling) and could also be useful for professional FSPM developers as a basic framework.

FSPM-P is nevertheless open for arbitrary extensions by rule-based coding in the language XL, thus its application is not restricted to a predefined range of parameter values or to a preselected portfolio of shapes or processes. Finally, a prototype like the one presented here will facilitate communication between modeller, programmer and experimentator, which can be mutually beneficial and helpful in establishing FSPM as a tool for research, development and education in the plant and crop sciences.

Acknowledgements We thank J. H. Lieth and J. Müller for providing the source code of their photosynthesis models, and also for their valuable discussions. Thanks are also due to L. Marcelis for helpful advice. Michael Henke received a ten-month scholarship from the Chinese Government (China Scholarship Council (CSC), http://www.csc.edu.cn), enabling him a research stay at the Institute of Crop Science, Zhejiang University, China. This research was also supported by the German Research Community DFG (Ku 847/8-1), the Dutch Science Foundation STW (07435) and the Product Board for Horticulture PT.

Appendixes

Appendix A List of abbreviations

Abbreviation	Description	Unit
AG	actual growth	g
AGR	actual growth rate	g/s
AP	temporary assimilate pool	g
CPU	central processing unit	
CSV	comma-separated values	
FSPM	functional-structural plant model	
FSPM-P	FSPM-Prototype	
GPU	graphics processing unit	
GR	growth respiration	
LD	local demand	g/s
L-systems	Lindenmayer-systems	
MR	maintenance respiration	
ODE	ordinary differential equation	
PAR	photosynthetically active radiation	Wm^{-2}
PPFD	photosynthetic photon flux density	$\mu molP/m^{-2}s$
PGR	potential growth rate	g/s
RSS	relative sink strength	
XL	eXtended L-System modelling language	

Appendix B External parameter files

The external parameter files are an elegant way to easily (re-)configure the FSPM-P. For example, for scenario tests, each configuration is stored in an individual file, where the user can switch between scenarios by changing only one entry.

The parameter files follow the syntax of common property files, which are widely used to configure software. Property files are simple text files, which can have maximal one entry per line. An entry consists of a key / identifier followed by an equals sign and the actual value of this entry:

$$< key > \; = \; value.$$

The *key* is a string used to identify this entry. We use the same name here as used later in the model code to make it traceable and transparent. In the current implementation, *value* can be one of the following types: *String*, *Integer*, *Double*, *Boolean*, or an array of one of them.

Below you can find a part of the *scenario.ini* file, which is used to configure the whole configuration for one specific scenario. Here it can be defined, e.g., which climate file, which photosynthesis model has to be used, or the start day for the simulation.

```
// species parameter
SPECIES_PARAMETER_FILE = speciesParameters.ini
// climate data:  Meteostation Haarweg 2008
CLIMATE_DATA_FILE = climateHaarweg2008Daily.csv
// number of values in CLIMATE_DATA_FILE
CLIMATE_DATA_VALUES = 366

// environment data
ENVIRONMENT_DATA_FILE = environment.ini

// debug mode
DEBUG_MODE = true

// show benchmark informations at each step
BENCHMARK = false

// activate data logging
USE_LOG_FILE = false

// determines the run modi of the model
DAILY_RUN_MODE = true

// day of the year; model starts at:  April 1st
START_DAY = 121

// select the photosynthesis model
(LEAFC3N2010=0,
//LIETHPASIAN = 1, KIMLIETH = 2, THORNLEY = 3,
```

```
//THORNLEYN = 4, MARSHALLBISCOE = 5,
//JOHNSONTHORNLEY = 6, HOST = 7, BALDOCCHI = 8)
PHOTOSYNTHESIS_MODEL = 8

// light model (CPU = 0, GPU = 1)
LIGHT_MODEL = 1
...
```

References

1. Goudriaan J, Van Laar H H. Modelling Potential Crop Growth Processes: Textbook with Exercises. Dordrecht: Kluwer Academic Publishers, 1994
2. Lopez G, Favreau R P, Smith C, Costes E, Prusinkiewicz P, DeJong T M. Integrating simulation of architectural development and source-sink behaviour of peach trees by incorporating Markov chains and physiological organ function submodels into L-PEACH. Functional Plant Biology, 2008, 35(10): 761–771
3. Allen M T, Prusinkiewicz P, DeJong T M. Using L-systems for modeling source-sink interactions, architecture and physiology of growing trees: the L-PEACH model. New Phytologist, 2005, 166(3): 869–880
4. Xu L F, Henke M, Zhu J, Kurth W, Buck-Sorlin G H. A rule-based functional-structural model of rice considering source and sink functions. In: Proceedings of the 3rd International Symposium on Plant Growth Modeling, Simulation, Visualization and Applications. 2009, 245–252
5. Buck-Sorlin G H, de Visser P H B, Sarlikioti V, Burema B S, Heuvelink E, Marcelis L F M, van der Heijden G W A M, Vos J. SIMPLER: an FSPM coupling shoot production, human interaction with the structure, morphogenesis, photosynthesis and light environment in cut-Rose. In: Proceedings of the 6th International Workshop on Functional-Structural Plant Models. 2010, 222–224
6. Groer C, Kniemeyer O, Hemmerling R, Kurth W, Becker H, Buck-Sorlin G H. A dynamic 3D model of rape (Brassica napus L.) computing yield components under variable nitrogen fertilization regimes. In: Proceedings of the 5th International Workshop on Functional-Structural Plant Models. 2007
7. Buck-Sorlin G H, Kniemeyer O, Kurth W. Barley morphology, genetics and hormonal regulation of internode elongation modelled by a relational growth grammar. New Phytologist, 2005, 166(3): 859–867
8. Buck-Sorlin G H, Kniemeyer O, Kurth W. A grammar-based model of barley including genetic control and metabolic networks. In: Vos J et al., eds. Functional-Structural Plant Modelling in Crop Production. Dordrecht: Springer, 2007, 243–252
9. Buck-Sorlin G H, Hemmerling R, Kniemeyer O, Burema B, Kurth W. A rule-based model of barley morphogenesis, with special respect to shading and gibberellic acid signal transduction. Annals of Botany, 2008, 101(8): 1109–1123
10. Barczi J F, Rey H, Caraglio Y, Reffye P D, Barthélémy D, Dong Q X, Fourcaud T. AmapSim: a structural whole-plant simulator based on botanical knowledge and designed to host external functional models. Annals of Botany, 2008, 101(8): 1125–1138
11. Hu B G, Reffye P D, Zhao X, Yan H P, Kang M Z. GreenLab: a new methodology towards plant functional-structural model — structural aspect. In: Hu B, Jaeger M, eds. Plant Growth Modeling and Applications. Beijing: Tsinghua University Press and Springer, 2003, 21–35
12. Letort V. Analyse multi-échelle des relations source-puits dans les modèles de développement et croissance des plantes pour l'identification paramétrique. Cas du modèle GreenLab. Dissertation for the Doctoral Degree. Châtenay-Malabry: École Centrale Paris, 2008
13. Breckling B. An individual based model for the study of pattern and process in plant ecology: an application of object oriented programming. EcoSys, 1996, 4: 241–254
14. Perttunen J, Sievänen R, Nikinmaa E, Salminen H, Saarenmaa H, Väkevä J. LIGNUM: a tree model based on simple structural units. Annals of Botany, 1996, 77(1): 87–98
15. Kniemeyer O. Design and implementation of a graph grammar based language for functional-structural plant modelling. Dissertation for the Doctoral Degree. Cottbus: Brandenburg University of Technology, 2008
16. Kurth W. Morphological models of plant growth. Possibilities and ecological relevance. Ecological Modelling, 1994, 75: 299–308
17. Prusinkiewicz P, Lindenmayer A. The Algorithmic Beauty of Plants. New York: Springer Science & Business Media, 2012
18. Hemmerling R. Extending the programming language XL to combine graph structures with ordinary differential equations. Dissertation for the Doctoral Degree. Göttingen: University of Göttingen, 2012
19. Hemmerling R, Kniemeyer O, Lanwert D, Kurth W, Buck-Sorlin G H. The rule-based language XL and the modelling environment GroIMP illustrated with simulated tree competition. Functional Plant Biology, 2008, 35(9/10): 739–750
20. Van Antwerpen D G. Unbiased physically based rendering on the GPU. Dissertation for the Master Degree. Delft: Delft University of Technology, 2011
21. Veach E. Robust Monte Carlo Methods for Light Transport Simulation. Dissertation for the Doctoral Degree. Palo Alto: Stanford University, 1998
22. Buck-Sorlin G H, Hemmerling R, Vos J, de Visser P H. Modelling of spatial light distribution in the greenhouse: Description of the model. In: Proceedings of the 3rd International Symposium on Plant Growth Modeling, Simulation, Visualization and Applications. 2009, 79–86
23. Evers J B, Vos J, Yin X, Romero P, Van Der Putten P E L, Struik P C. Simulation of wheat growth and development based on organ-level photosynthesis and assimilate allocation. Journal of Experimental Botany, 2010, 61(8): 2203–2216
24. Preetham A J, Shirley P, Smits B. A practical analytic model for daylight. In: Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques. 1999, 91–100
25. Gijzen H. Development of a simulation model for transpiration and water uptake and an integral growth model. AB-DLO Report 18. 1994
26. Nikolov N T, Massman W J, Schoettle A W. Coupling biochemical and biophysical processes at the leaf level: an equilibrium photosynthesis model for leaves of C3 plants. Ecological Modelling, 1995, 80: 205–235

27. Müller J, Wernecke P, Diepenbrock W. LEAFC3-N: a nitrogen-sensitive extension of the CO2 and H2O gas exchange model LEAFC3 parameterised and tested for winter wheat (Triticum aestivum L.). Ecological Modelling, 2005, 183: 183–210
28. Müller J, Braune H, Diepenbrock W. Photosynthesis-stomatal conductance model LEAFC3-N: specification for barley, generalised nitrogen relations, and aspects of model application. Functional Plant Biology, 2008, 35: 797–810
29. Baldocchi D. An analytical solution for coupled leaf photosynthesis and stomatal conductance models. Tree Physiology, 1994, 14: 1069–1079
30. Kim S H, Lieth J H. A coupled model of photosynthesis, stomatal conductance and transpiration for a rose leaf (Rosa hybrida L.). Annals of Botany, 2003, 91(7): 771–781
31. Lieth J H, Pasian C C. A simulation model for the growth and development of flowering rose shoots. Scientia Horticulturae, 1991, 46: 109–128
32. Thornley J H M. A model to describe the partitioning of photosynthate during vegetative plant growth. Annals of Botany, 1969, 33: 419–430
33. Thornley J H M. Dynamic model of leaf photosynthesis with acclimation to light and nitrogen. Annals of Botany, 1998, 81(3): 421–430
34. Johnson I R, Thornley J H M. Dynamic model of the response of a vegetative grass crop to light, temperature and nitrogen. Plant, Cell and Environment, 1985, 8(7): 485–499
35. Marshall B, Biscoe P V. A model for C3 leaves describing the dependence of net photosynthesis on irradiance I. Derivation. Journal of Experimental Botany, 1980, 31(1): 29–39
36. Marshall B, Biscoe P V. A model for C3 leaves describing the dependence of net photosynthesis on irradiance II. Application to the analysis of flag leaf photosynthesis. Journal of Experimental Botany, 1980, 31(1): 41–48
37. Rauscher H M, Isebrands J G, Host G E, Dickson R E, Dickmann D I, Crow T R, Michael D A. ECOPHYS: an ecophysiological growth process model for juvenile poplar. Tree Physiology, 1990, 7: 255–281
38. Yin X Y, Goudriaan J, Lantinga E A, Vos J, Spiertz H J. A flexible sigmoid function of determinate growth. Annals of Botany, 2003, 91(3): 361–371
39. Richards F J. A flexible growth function for empirical use. Journal of Experimental Botany, 1959, 29(10): 290–300
40. Thornley J H M. Growth, maintenance and respiration: a re-interpretation. Annals of Botany, 1977, 41(6): 1191–1203
41. Bertin N, Gary C. Évaluation d'un modèle dynamique de croissance et de développement de la tomate (Lycopersicon esculentum Mill), TOMGRO, pour différents niveaux d'offre et de demande en assimilats. Agronomie, 1993, 13: 395–405
42. Marcelis L F M. A simulation model for dry matter partitioning in cucumber. Annals of Botany, 1994, 74(1): 43–52
43. Marcelis L F M. Sink strength as a determinant of dry matter partitioning in the whole plant. Journal of Experimental Botany, 1996, 47: 1281–1291
44. Qi R, Ma Y T, Hu B G, de Reffye P, Cournède P H. Optimization of source-sink dynamics in plant growth for ideotype breeding: a case study on maize. Computers and Electronics in Agriculture, 2010, 71(1): 96–105
45. Pradal C, Dufour-Kowalski S, Boudon F, Fournier C, Godin C. OpenAlea: a visual programming and component-based software platform for plant modelling. Functional Plant Biology, 2008, 35(10): 751–760
46. Vos J, Evers J B, Buck-Sorlin G H, Andrieu B, Chelle M, de Visser P H B. Functional-structural plant modelling: a new versatile tool in crop science. Journal of Experimental Botany, 2010, 61(8): 2101–2115
47. Wilson G V. Where's the real bottleneck in scientific computing? American Scientist, 2006, 94(1): 5–6
48. McMaster G S, Hargreaves J N G. CANON in D(esign): composing scales of plant canopies from phytomers to whole-plants using the composite design pattern. NJAS - Wageningen Journal of Life Sciences, 2009, 57(1): 39–51
49. Bouman B A M, Keulen v H, Laar v H H, Rabbinge R. The 'school of de Wit' crop growth simulation models: A pedigree and historical overview. Agricultural Systems, 1996, 52(2): 171–198
50. Spitters C J T. Crop growth models: their usefulness and limitations. ISHS Acta Horticulturae 267: VI Symposium on the Timing of Field Production of Vegetables. 1990, 349–368
51. Van Keulen H, Penning de Vries F W T, Drees E M. A summary model for crop growth. In: Penning de Vries F W T, van Laar H H, eds. Simulation of plant growth and crop production, Wageningen: Centre for Aqricultural Publishing and Documentation, 1982
52. Lithourgidis A S, Dordas C A, Damalas C A, Vlachostergios D N. Annual intercrops: an alternative pathway for sustainable agriculture. Australian Journal of Crop Science, 2011, 5(4): 396–410
53. Ouma G P J. Sustainable horticultural crop production through intercropping: the case of fruits and vegetable crops: a review. Agriculture and Biology Journal of North America, 2010, 1(5): 1098–1105
54. Henke M, Sarlikioti V, Kurth W, Buck-Sorlin G H, Pagès L. Exploring root developmental plasticity to nitrogen with a three-dimensional architectural model. Plant and Soil, 2014, 385(1): 49–62

Michael Henke received his Diploma degree in computer science from the Cottbus University of Technology, Germany. Currently, he is working on his PhD in applied computer science at the Department of Ecoinformatics, Biometrics and Forest Growth, University of Gottingen, Germany. From 2009 to 2010, he was a visiting scholar at Zhejiang University, China. He worked as an assistant lecturer at Cottbus University of Technology and University of Gottingen, Germany, and as a researcher at French National Institute for Agricultural Research, Angers, France in 2013 and 2014, and also worked in Wageningen UR, the Netherlands from 2014 to 2016. His research interests are functional-structural plant modelling and light calculation.

Winfried Kurth received his Diploma degree in mathematics, and PhD in theoretical computer science from Clausthal University of Technology, Germany. Subsequently, he was a junior researcher at the Universities of Göttingen and Bayreuth. From 2001 to 2008, he was a professor in practical computer science and graphics systems at Cottbus University of Technology, Germany. Since 2008, he is a professor in computer graphics and ecological informatics at University of Göttingen, Germany. His research fields include rule-based languages, representation of 3D data, functional-structural plant models, and simulation.

Gerhard H. Buck-Sorlin received his Diploma degree in biology at the University of Göttingen, Germany, and his PhD in biology at the University of Wales in Bangor, UK in 1997. Subsequently, he worked as a postdoctoral scientist at the Institute of Plant Genetics and Crop Plant Research in Gatersleben, Germany, and at Cottbus University of Technology, Germany from 1997 to 2007, and as a guest professor at the Zhejiang University, China from 2005 to 2009. Between 2007 and 2011, he worked as a senior scientist at Wageningen UR, the Netherlands. Since 2011, he is a professor in Fruit Tree Culture and Modelling at Agrocampus Ouest, Centre d'Angers, France. His research fields include ecophysiology of crop plants, and functional-structural plant modelling.

CHAPTER 7

Fifth Paper

This paper is published as:
Henke M, Sarlikioti V, Kurth W, Buck-Sorlin GH, Pagès L (2014) Exploring root developmental plasticity to nitrogen with a three-dimensional architectural model, *Plant and Soil*, 385(1-2), 49-62, doi: 10.1007/s11104-014-2221-7

Authorship

- Michael Henke implemented the model and supported the writing of the manuscript.
- Vaia Sarlikioti wrote the manuscript and performed the scenario runs.
- Winfried Kurth supported the writing of the manuscript.
- Gerhard H. Buck-Sorlin supported the writing of the manuscript.
- Loïc Pagès developed the model base and supported the writing of the manuscript.

Plant Soil
DOI 10.1007/s11104-014-2221-7

REGULAR ARTICLE

Exploring root developmental plasticity to nitrogen with a three-dimensional architectural model

Michael Henke · Vaia Sarlikioti · Winfried Kurth · Gerhard H. Buck-Sorlin · Loïc Pagès

Received: 30 March 2014 / Accepted: 25 July 2014

Abstract
Background and aims Root plasticity is a key process affecting the root system foraging capacity while itself being affected by the nutrient availability around the root environment. Root system architecture is determined by three types of plastic responses: chemotropism, spacing of lateral roots, hierarchy between laterals and their mother root.
Methods We attempt a systematic comparison of the effect of each mechanism on the whole root plasticity when the root is grown under four distinct nutrient distribution scenarios using a functional-structural root model. Nutrient distributions included i) a completely random distribution, ii) a layered distribution, iii) a patch distribution, and iv) a gradient distribution. Root length, volume, total uptake, uptake efficiency as well as the soil profiles are given as model outputs.
Results Root uptake was more efficient in a soil with a gradient nutrient distribution and less so in a patch distribution for all mechanisms. In terms of mechanisms uptake was more efficient for the spacing (elongation) mechanism than the hierarchy (branching) mechanism.
Conclusions Root mechanisms play a different role in the foraging of the root with chemotropism being a global tracking mechanism, whereas spacing and hierarchy are ways to proliferate in a zone with locally available nutrients.

Keywords Root plasticity · 3D architecture · Nutrient uptake · Chemotropism · Root growth strategies · Functional-structural plant modelling (FSPM)

Abbreviations

FSPM Functional-structural plant modelling

Introduction

Root system plasticity is known to be a key process for enhancing foraging ability (Hodge 2004; Malamy 2005; de Kroon et al. 2009), being highly dependent on soil properties, such as variations in nutrient availability. The interaction of the root system with the soil can affect root growth at the local level, thus having an effect on total root architecture (Fitter 1994; Hutchings and de Kroon 1994). Previous studies have shown that roots

Michael Henke and Vaia Sarlikioti contributed equally to this work.

Responsible Editor: Angela Hodge.

M. Henke · G. H. Buck-Sorlin
UMR1345 Institut de Recherche en Horticulture et Semences (IRHS), Agrocampus Ouest, Centre d'Angers,
2, rue André le Nôtre, 49045 Angers cedex 01, France

V. Sarlikioti · L. Pagès
INRA Centre d'Avignon, UR 1115, PSH, Site Agroparc,
84914 Avignon cedex 9, France

M. Henke (✉) · W. Kurth
Department of Ecoinformatics, Biometrics and Forest Growth, University of Göttingen,
Büsgenweg 4, 37077 Göttingen, Germany
e-mail: mhenke@uni-goettingen.de

Published online: 03 August 2014

may proliferate when they reach nutrient-rich zones or patches in the soil, by increasing the root volume per unit soil area (Robinson 1994). The plastic response of the root seems to depend on the nutrient distribution and on the nutritional status of the plant (Ericsson 1995; Zhang and Forde 2000). Robinson (2001) showed that the root growth rate is gradually acclimated to the distribution of nutrients around the root.

Different growth and branching strategies have been indicated in the past to affect global plasticity of root architecture, alone as well as in interaction (Fitter 1994; Wijesinghe and Hutchings 1997; Einsmann et al. 1999). Individual roots tend to re-orient themselves towards the position of the nutrient cluster (Epstein and Bloom 2005). This phenomenon is known as chemotropism, and is related to the precision strategy of root foraging for maximizing nutrient and water uptake, especially when the root is growing under stress conditions (Campbell et al. 1991). The preferential root growth towards nitrogen has been established under numerous split root experiments (Scott and Robson 1991; Marina et al. 2002). It has also been shown that this directional growth is a result of the root's ability to sense a gradient (Monshausen and Gilroy 2009). Other studies revealed (Barlow 2002) that chemotropism positively affects cell elongation.

An increase of the linear branching density along the root segment that results in roots with smaller inter-branching distances and more lateral roots, is a known root strategy observed in many species when the supply of nutrients is localised. In contrast, at a low nutrient supply the roots are more elongated and fewer lateral roots are produced (Robinson 1994; Casper and Jackson 1997). In grass crops inter-branching distance was dependent on the plant species as well as the plant strategy for quick soil proliferation or for a slower root growth with specific plasticity (Campbell and Grime 1989). Finally an important root strategy is the hierarchical modification between the growing root and its laterals. In nutrient poor zones, main roots can have thin and short laterals, whilst they tend to develop more vigorous branches (thicker and longer) in nutrient enriched zones (Granato and Raper 1989; Pagès 1995; Bingham et al. 1997).

All mentioned strategies work in parallel in the root system. Nevertheless no systematic comparison has been made to identify the relative advantage of these mechanisms on nutrient uptake under different conditions of nutrient availability, and the effect on root architecture. It is probable that mechanisms that can produce more lateral roots as the hierarchy strategy will demonstrate a higher nutrient uptake in rich soils, while a chemotropic strategy will be more advantageous when nutrients are sparse and randomly distributed, as it can help turn the roots towards the higher gradient.

Field description of root plasticity is very challenging and data collection is hardly ever accurate. New techniques of 3D visual reconstruction in situ with laser scanners (Fang et al. 2009) and X-rays (Fang et al. 2012) that are highly promising for spatial studies are prohibitory in experimentation because of their high cost. Another approach for studying the root plasticity is through 3D plant modelling. Functional-structural plant models (FSPM or virtual plants) are defined as models that couple a selection of physiological processes that result in an explicit 3D plant structure, often supplied with a mutual feedback between physiology and structure (Vos et al. 2007; Buck-Sorlin 2013). This type of modelling can provide a valuable complementary solution when studying plastic development in response to soil heterogeneity.

Few root models describe the plasticity of the root system in relation to soil heterogeneity. Dunbabin et al. (2004) simulated nitrogen uptake in uniform and non-uniform nitrogen distribution scenarios for two theoretical root architectural types. Different root models that have been proposed during the last years (Fitter et al. 1991; Berntson 1994; Lynch et al. 1997; Pagès 2011) focus on the spatial distribution and the topological characteristics of the root systems, where the root system is considered as a binary tree in which nodes connect root parts. These models, although helping with the understanding of the foraging efficiency of the root, do not consider the heterogeneity of the soil and, therefore, fail to capture the effect of different nutrient distributions to root plasticity. Chen et al. (2013) investigated the influence of phosphorus distribution on root system development.

The objective of the present study was to test the effect of recognized plasticity strategies on root architecture and uptake efficiency, using a simplified root FSPM that was deliberately not parameterized for a specific crop or plant species but designed as a generalized model meant to illustrate different plasticity mechanisms. These included i) the perception of nutrient rich zones, and orientation of root growth towards them (chemotropism mechanism), ii) adaptation of the linear branching density (spacing

Table 1 Model parameters

Model parameter	Value	Unit	Description
Maximum steps	40	Days	Maximum number of simulation steps
Root			
Main root initial radius	0.0005	m	Root radius at the initiation of the root
D_{min}	0.0001	m	Minimum diameter of the root. If meristem radius is lower than this value no growth is possible
D_{max}	0.001	m	Maximum diameter of the root. The value regulates the potential growth rate
E	8	–	Slope of the relationship between root diameter and the growth rate
Lrs	0.002	m	Root segment length. The parameter is constant for all scenarios except spacing; Fitter et al. (1991), Pagès (2011)
Biomass cost	0.25	g	Parameter used for calculating the root biomass demand
Primordia growing period	4	Days	Number of days after the initiation of primordium that they start to grow
Primordia_phyllotactic angle	137	°	Rotation around the z- axis of the primordium initiation
Primordia branching angle	55	°	Rotation around the x-axis of the primordium initiation
Branching angle variation	5	°	Random variation of the primordium initial branching angle
Uptake			
Root uptake distance	0.01	m	Maximum distance of a source particle in order for it to be sensed (and eventually absorbed by the growing root)
Tropisms			
Gravitropic intensity	0.15	–	Intensity with which the root turns towards the ground
Opening angle (CA)	65	°	Opening angle of the cone within which the root is searching for the nutrient sources
Chemotropic intensity	0.75	–	Intensity with which the root turns towards the nitrogen sources
Spacing mechanism			
S_{Lf}	−0.0001	–	Slope of the relationship between root segment length and local uptake (0 on other scenarios); Fitter et al. (1988), Rose (1983)
Hierarchy mechanism			
D_P	0.004	m	Initial primordium diameter
D_{BRMo}	0.65	–	Slope of the relationship between mother and daughter root diameter; Toky and Bisht (1992)
D_{BRMf}	0.03	–	Slope of the relationship between RMDB and local uptake (0 on other scenarios); Levang-Brilz and Biondini (2002), Lynch and Brown (2001)
σ	1.75	–	Constant that modulates the variance of the distribution; Pagès (2011)
R	−1 to 0	–	Random number; Pagès (2011)
Resources parameters			
Nitrogen			
P_N	10500	–	Number of nitrogen sources distributed in the scene
Nitrogen source distance	0.05	m	Distance of the root meristem from the nitrogen source in order to be taken up

mechanism) and iii) modification of the relative growth rate of the lateral roots relative to the mother root (hierarchy mechanism). The effects of such strategies on architecture and uptake at the root level

are explored on four scenarios of soil nutrient distribution.

Materials and methods

Overview and scope of the model

The simple dynamic root model proposed by Pagès (2011) was translated into the modelling language XL, using the open-source GroIMP platform (www.grogra.de) and was used as a base. The simplicity of the previous model that used only a reduced number of parameters was retained. On this basis, the model was developed further in order to include developmental processes representing plastic responses and to consider soil-root interactions.

During the initialization of the model the 3D scene is created and filled up with a defined constant number of nutrient particles ("nitrogen") following one of four distribution patters. A single root meristem is placed on the top of the ground. During each time step a set of replacement rules are applied to the generated structure. These rules follow an extension of the well-known L-system syntax and semantics. For each type of organ we defined one or more rules with certain conditions, which were limiting the number of cases where the rule(s) can be applied. Thus these rules define the outer environmental variables that in turn determine the growth behaviour of modelled roots.

While the rules are fixed, only the soil distributions are varied. Each simulation generates a new environment in which the above mentioned parameters give a different result given the initial parameter set allowing the comparison between mechanisms.

Allometric relationships between roots and shoots as well as assimilate supply by shoots were neglected for simplicity; however, development was limited by introducing a maximal biomass increment at each time step. The time step of the model was one day. All values of the model parameters are presented in Table 1. Parameter value ranges were adopted from Pagès (2011) as well as from other literature as listed in Table 1.

Soil representation

For the construction of the soil, nitrogen as a resource was considered. Nitrogen source particles were represented as sphere objects (with a given diameter) in the scene within a virtual soil box of 0.25 m^3 (0.5*0.5*1 m). In order to investigate the effect of nutrient distribution on root plasticity four different nutrient distribution scenarios were created in order to mimic contrasted types that can be found in the field (Fig. 1): i) a random distribution (R) in which nitrogen was randomly distributed in the volume of the box, ii) a layered distribution (L) in which 90 % of nitrogen was distributed in dense layers of 30 cm depth with only 10 % of nitrogen in between layers, iii) a patch distribution (P) in which the particles were equally distributed in spheres of a certain diameter *d* and where these spheres were in their turn randomly distributed within the box, and finally iv) a gradient distribution (G) where nitrogen was gradually reduced from the top to the bottom of the box with 50 % of nitrogen distributed within the first 25 cm of the box.

Nutrient uptake

Nutrient uptake was simulated as function of the distance between the root meristem and a nutrient source. When the root meristem was positioned within a threshold distance to a nutrient source then this source was removed from the scene and its content added to the local nutrient pool of the root meristem. In the current version of the model, the nitrogen uptake is not regulated. The total cumulative uptake at the root level was given as an output, representing the count of the number of sources of nitrogen removed from the scene. For simulating plastic response (see below) we also calculated the local transient uptake by each meristem as the average uptake of the last three days.

Root system development

The root system is composed of three types of components: the root meristem, the root segment and the root primordium. Plant growth is controlled by a Lindenmayer system (L-system) (Prusinkiewicz and Lindenmayer 1990) which consists of a set of production rules, that when applied, expand each symbol on the left-hand side of the rule by the sequence of symbols on the right-hand side. Hence, a root meristem is replaced applying a general substitution rule

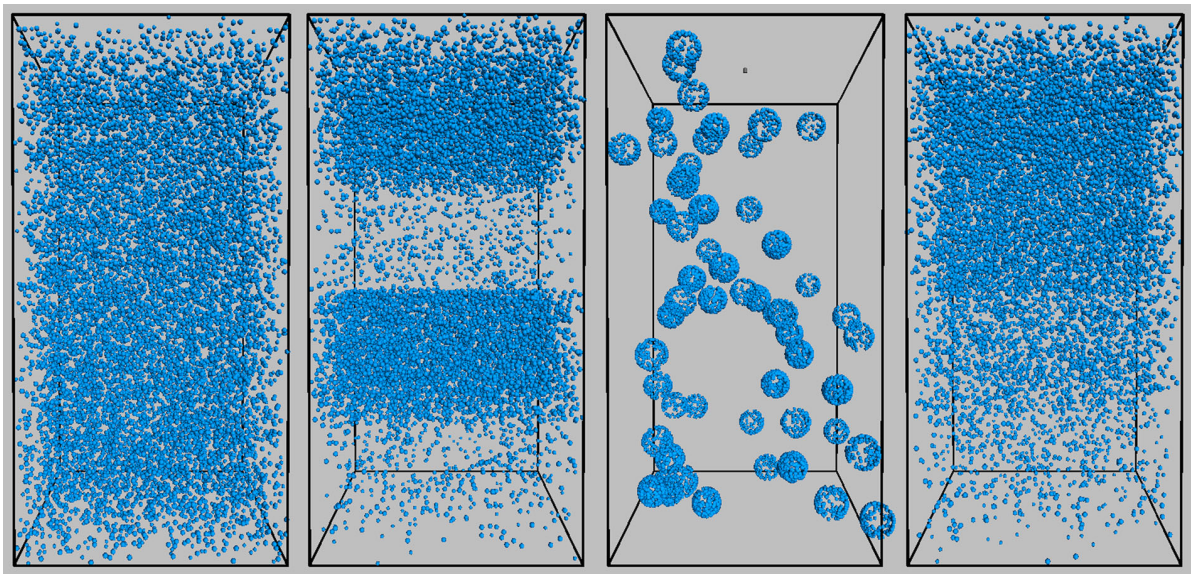


Fig. 1 Representation of different soil scenarios. From left to right: random distribution (R), layer distribution (L), patch distribution (P), and gradient distribution (G)

rM : RootMeristem⇒(RootSegment[RootPrimordium])$^{+}$rM

by a sequence of pairs consisting of one new RootSegment followed by a branching RootPrimordium that will produce axillary roots as long as RootMeristem length is higher than the maximal length of a segment (Fig. 2). Finally, the rule foresees a replacement of the RootMeristem (rM) to the tip of the new segment, in order to allow application of the rule in the next step. Rank and branching order are not limited by fixed thresholds, but constrained indirectly by setting a threshold minimal diameter. Each root segment is characterized by its length L and diameter D. At initiation the root consists of one root meristem. The potential growth rate of the root meristem is given by the following equation (Pagès 2011):

$$G_P = \begin{cases} 0 & , \quad D \le D_{min} \\ E * (D - D_{min}) & D > D_{min} \end{cases} \tag{1}$$

where D_{min} is the minimal diameter below which no elongation is possible, D is the current diameter and E represents the slope of the linear function.

The actual growth rate (G_A) was obtained by multiplying the potential growth rate (G_P) with a satisfaction ratio. This satisfaction ratio (R_S) was calculated as:

$$R_S = \frac{A_V}{D_V} \tag{2}$$

where A_V (volume allocation) referred to the assimilates imported into the root from the aerial part of the plant. For the sake of simplicity, a given maximal volume was assigned as a function of time. By D_V (volume demand) we defined the amount of assimilates needed per root volume. The biomass demand was calculated by multiplying the root volume with a constant (biomass cost).

After the application of the growth rules each meristem is potentially substituted by one or more root segments with a new root meristem at the tip (Fig. 2). Root segment length L_{RS} is given by the equation:

$$L_{RS} = L_S + S_{Lf} * U_l \tag{3}$$

where U_l represented the local uptake, L_S (segment length) was the root segment length in the absence of uptake that represented the threshold value of the equation and S_{Lf} was another constant that represented the slope of what was assumed to be the linear regression function between local uptake and segment length. The value of S_{Lf} was set to zero when chemotropism and the spacing mechanisms were studied.

With every new root segment also a root primordium was initiated. The insertion angle between the root segment and the initiated root primordium was stochastic

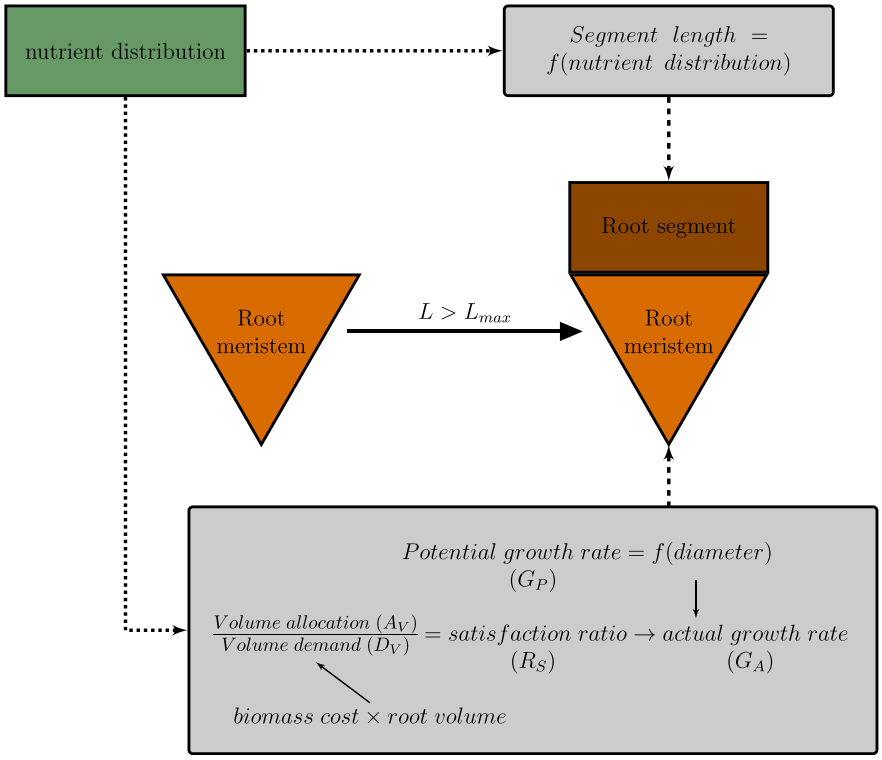


Fig. 2 Flow chart explaining the principle of the model

with a given variation around a mean value. The primordium started growing 4 days after the time of its initiation. The diameter of the primordium (D_p) was dependent upon the diameter of the mother root from which it was produced according to the function:

$$D_p = D_{MR} * \left(D_{BRM_0} + D_{BRM_f} * U_l\right) e^{\sigma * R} \quad (4)$$

where D_{MR} represented the mother root diameter, D_{BRM_0} the initial diameter of the branch root relative to its mother and D_{BRM_f} the slope of the linear relationship between the local uptake and the mother root diameter. Assuming a locally constant root hair density and a correlation between the total number of (functional) root hairs per unit root length and uptake rate, then local uptake rate is proportional to root diameter, which is expressed by this slope. σ was a constant that modulated the variance of the distribution, while R was a random number between −1 and 0. While studying chemotropism and spacing mechanisms, D_{BRM_f} was set to zero.

Chemotropism was defined as a function of the distance, the local position of the root meristem and the attracting force exerted by the nutrient sources upon the future orientation of the root meristem (Fig. 3). Under field conditions, the root will turn within its sensing radius towards the greater concentration of sources available. In order to implement the sensing mechanism in the model, we supposed that each root meristem perceived nutrient sources within a cone-shaped volume, with the root meristem being situated at the top of the cone, which itself had a constant opening angle (*CA*, see Table 1). Inside the cone-shaped sensing volume all nutrient sources at a distance smaller than, or equal to, a constant threshold value were found. The mean position of all the sources inside the cone was calculated and the root meristem turned towards that mean location which represented the highest gradient, with a certain pulling strength given by a constant (Root Intensity Tropism).

Design of the simulation experiment

For each soil scenario (4) and each mechanism (3) five replicate simulations were run, resulting in a total of 60 simulation runs. The number of days for root growth

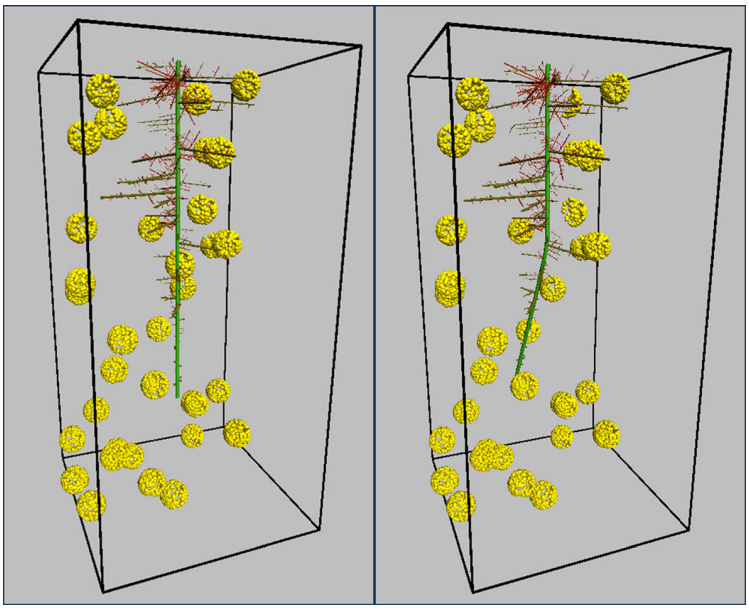


Fig. 3 3D root representation when no mechanism is applied (*left*) and when chemotropism mechanism is applied (*right*) for a patch nutrient distribution

was set to 40. For each simulation the soil distribution for each soil scenario was different based on a randomization factor in the model. The set of parameters used for each simulation can be found in Table 1. Data were analyzed by analysis of variance (ANOVA) using R (version 2.14.2; www.r-project.org). Mechanisms were compared at 5 % probability level using least significant differences based on Student's t-test (P=0.05). The t-tests was performed at the 40 days data points, i.e. the last simulated day.

Results

Different plasticity mechanisms as well as different soil distributions had an impact on both root plasticity and morphology and therefore on nutrient uptake.

The longest roots were obtained in the L and G distributions, whereas at the R and P distributions the shortest roots were produced (Fig. 4). As expected, on average the chemotropic mechanism exhibited the shortest root system (16.25 m±1 m) while the spacing mechanism producing the longest (45.15 m±10 m). No significant differences were observed between the spacing and the hierarchy mechanism at the random and gradient distributions. In the case of layer distribution all mechanisms were significantly different from each other. Chemotropism differed significantly from the other two mechanisms at all distributions except at the P distribution where no differences were found between chemotropism and hierarchy mechanism.

The total root volume in general differed significantly between all plasticity scenarios, with the hierarchy mechanism always achieving the highest volume and the chemotropic mechanism the lowest (Fig. 5). The difference between the mechanisms depended a lot on the nutrient distribution. With the R, L and G distribution (Fig. 5) significant differences were observed between all mechanisms. Root volume did not differ greatly for chemotropism between scenarios. However, in the spacing mechanism the root volume increased by 40 % between the R distribution and L distribution and by 30 % between R and G, respectively. This increase was even greater at the hierarchy mechanism with an increase of 50 % and 68 % in the L and G distribution, respectively. No significant differences were observed between the three mechanisms when a P nutrient distribution was used (Fig. 5c).

The soil scenarios and the plasticity mechanism had a strong effect on the distributions of the root organs in the soil. In order to investigate exactly how organ production was affected by the different mechanisms, we chose to produce root profiles where the number of organs at various soil depths was expressed as a percentage of the total number of organs (Fig. 6). No differences were observed between mechanisms at the R distribution.

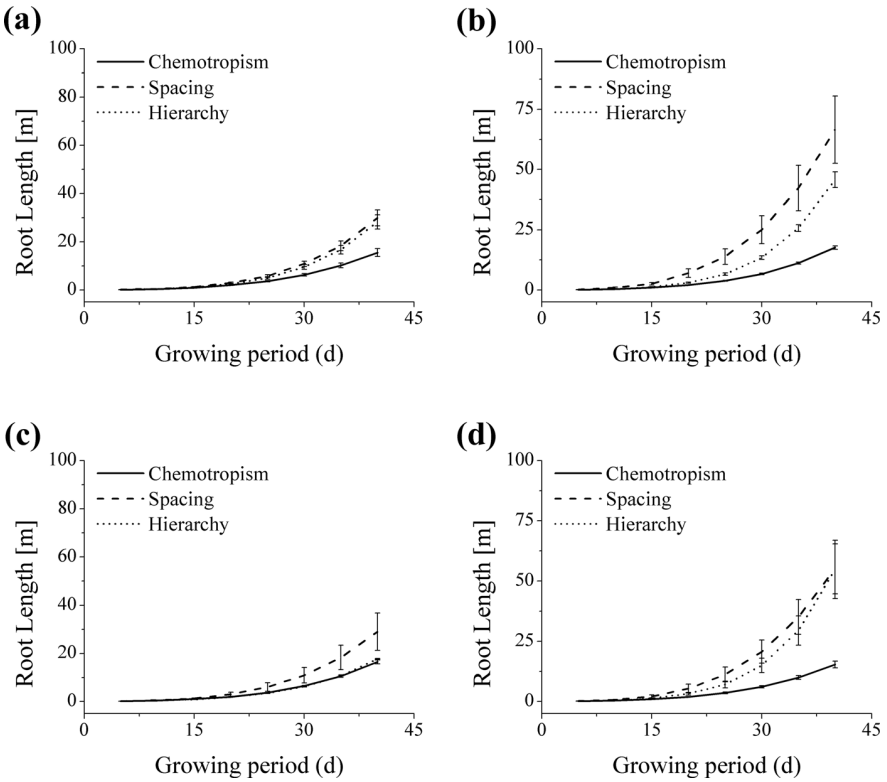


Fig. 4 Total root length for a growing period of 40 simulation steps for (**a**) random, **(b)** layer, (**c**) patch and **(d)** gradient distributions. The *continuous line* represents the chemotropism mechanism, the *dashed line* the spacing mechanism and the *short dashed line* the hierarchy mechanism. The *vertical bars* represent the standard error of the mean for five replications

When an L distribution was applied, with the spacing mechanism 80 % of the organs were positioned within the upper 0.25 m of the soil, while this figure was only 65 % and 50 % for the hierarchy and chemotropic mechanisms, respectively (Fig. 6b). With the same distribution the chemotropic mechanism lead to a deeper soil penetration, up to 0.5 m. In the P scenario a distinctive peak was observed with the spacing mechanism at 0.2 m depth, which was at 0.3 m depth with the hierarchy mechanism, while no distinctive peaks were found for chemotropism (Fig. 6c). When a G distribution was applied significant differences were found between the three mechanisms at the top 30 cm of the soil box, with a higher concentration of organs at the spacing mechanism and the lowest for chemotropism.

The differences observed in morphology and root plasticity were also found with respect to nutrient uptake. In general, roots with the chemotropic mechanism had a significantly lower uptake than roots under the other two mechanisms for all scenarios except the patch distribution (Fig. 7c). In the random distribution scenario no differences were found between the spacing and hierarchy mechanism with both root systems having taken up 26 % of the available nitrogen (Fig. 7a). In the L distribution there was a significant difference between the hierarchy mechanism (45 % uptake) and the spacing mechanism (52 % uptake). In the patch distribution the uptake was around 15 % for all mechanisms (Fig. 7b). In the gradient distribution the highest uptakes were found with the hierarchy mechanism (uptake of 62 % of the nitrogen) and the spacing mechanism (55 %), but the difference found between the two mechanisms was not significant (Fig. 7d).

We define uptake efficiency (η_{U_r}) according to Eq. 5:

$$\eta_{U_r} = \frac{\sum U_l}{\sum P_N} \tag{5}$$

where $\sum P_N$, is the number of nitrogen particles, and $\sum U_l$ is the sum of all local uptake events during a time step.

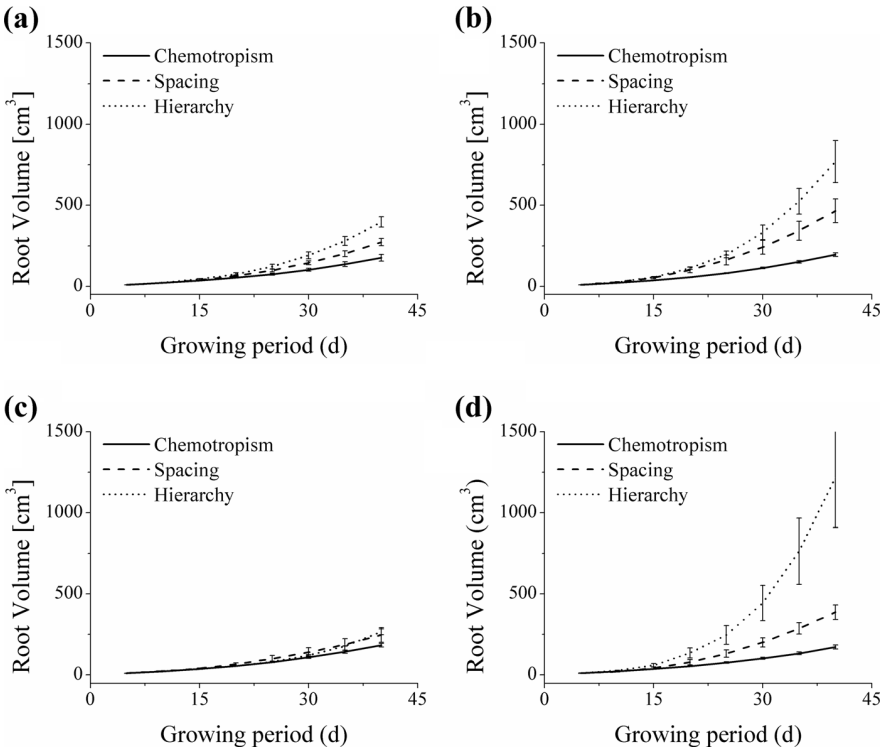


Fig. 5 Root volume (cm^3) for a growing period of 40 simulation steps for (**a**) random, (**b**) layer, (**c**) patch and (**d**) gradient distributions. The *continuous line* represents the chemotropism mechanism, the *dashed line* the spacing mechanism and the short dashed line the hierarchy mechanism. The *vertical bars* represent the standard error of the mean for five replications

High root uptake did not correlate directly with highest uptake efficiency. Our simulations showed that the highest uptake efficiency was found with the spacing mechanism (except for the patch distribution where chemotropism was superior), with the hierarchy mechanism displaying a very low efficiency (Fig. 8).

Discussion

The aim of this paper was to assess, by way of simulation, the effect of different nutrient distributions on the uptake and root morphological characteristics when three distinctive plasticity mechanisms were considered. These mechanisms have previously been described through experimentation, as an adaptation strategy of the root system in order to increase its foraging capacity (Forde and Lorenzo 2001). The incorporation of all these mechanisms in one root model in combination with a spatially explicit representation of the nutrient distribution has not been previously attempted. In this work we presented such a model as well as a systematic comparison of the effects of initial nutrient distribution in space on the root architecture.

Since the intention of the current work is to simulate a number of simplified systems with a limited number of parameters, in order to understand the specific plastic response of each mechanism, simulations of only those mechanisms were performed. A combination of all mechanisms at the same moment simulating a "real" plant was not performed as in the current form it would not further the understanding of the proposed mechanisms.

We focused on nitrogen as an exemplary highly soluble nutrient for our simulations and adapted our parameter set to this nutrient in terms of root response and nutrient mobility. However, our approach is general enough to be used for the representation of other nutrients like phosphorus or potassium, with the given parameters adjusted accordingly. With respect to potassium it should be noted that its presence does apparently not induce localized root proliferation, contrary to what has been implemented in our model.

Roots are restricted in movement in terms that the root cannot turn backwards if it senses a source in that

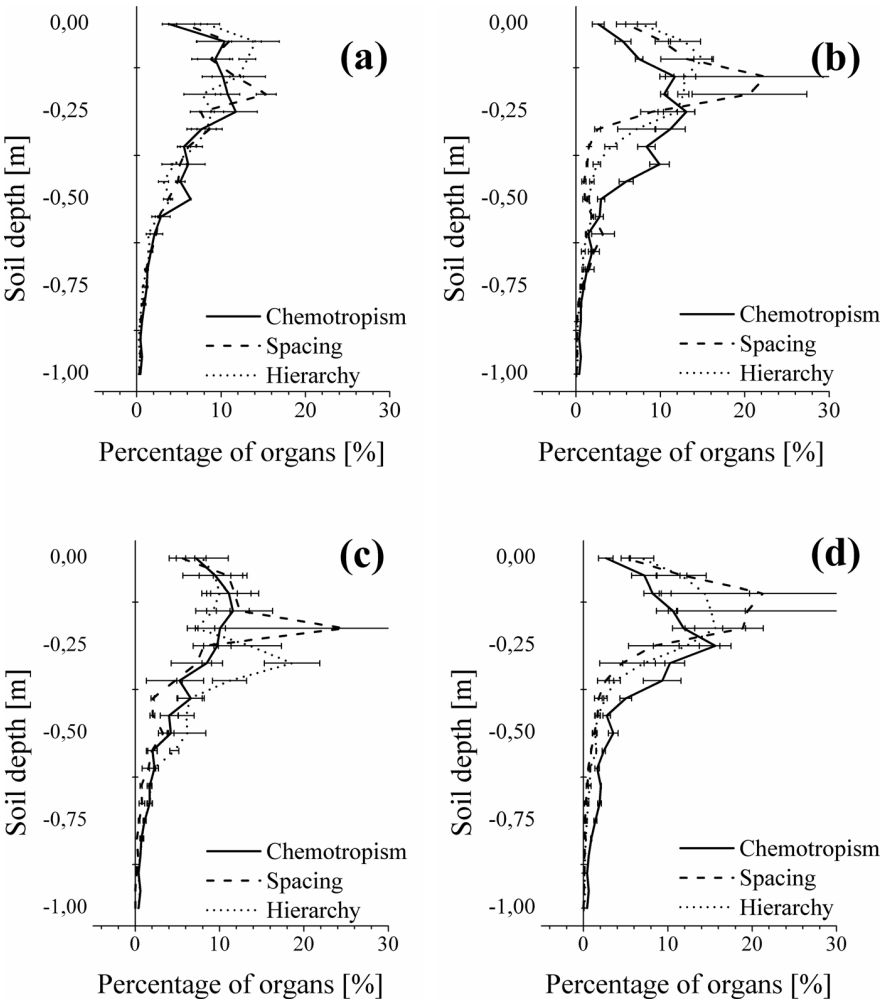


Fig. 6 Root components distribution for a growing period of 40 simulation steps for (**a**) random, (**b**) layer, (**c**) patch and (**d**) gradient distributions. The *continuous line* represents the chemotropism mechanism, the *dashed line* the spacing mechanism and the *short dashed line* the hierarchy mechanism. The *vertical bars* represent the standard error of the mean for five replications

direction. Another important point is that in our model within the cone the root turns towards the largest gradient. In the literature on water tropism that we looked at, the assumption was that roots move towards directions that indicate a "sensing cone" viewing angle rather than a sphere.

The importance of the chemotropic mechanism can be understood in unfertile soils or when the nutrients are randomly clustered in patches. Our simulations showed that in those cases the nutrient uptake efficiency increased by 50 % in comparison to a random nutrient distribution. Although the existence of the chemotropic mechanism has been well established, there are in the literature (to our knowledge) as yet no data available that help quantifying this tropic movement. Therefore, a number of model approaches have been proposed in the last years (Tsutsumi et al. 2003), more recently an implementation was proposed by Leitner et al. (2010) who also used a three-dimensional model. Our approach differed from that one in that we based chemotropism on the distance of the nutrient supply from the root tip as well as on the number of nutrient sources within the "viewing angle" of the root. In that way, we allowed the root to sense the available nitrogen within a certain range and to grow towards the higher supply.

In nutrient rich soils or soils in which the fertilization is applied to the top layer of the soil the formation of

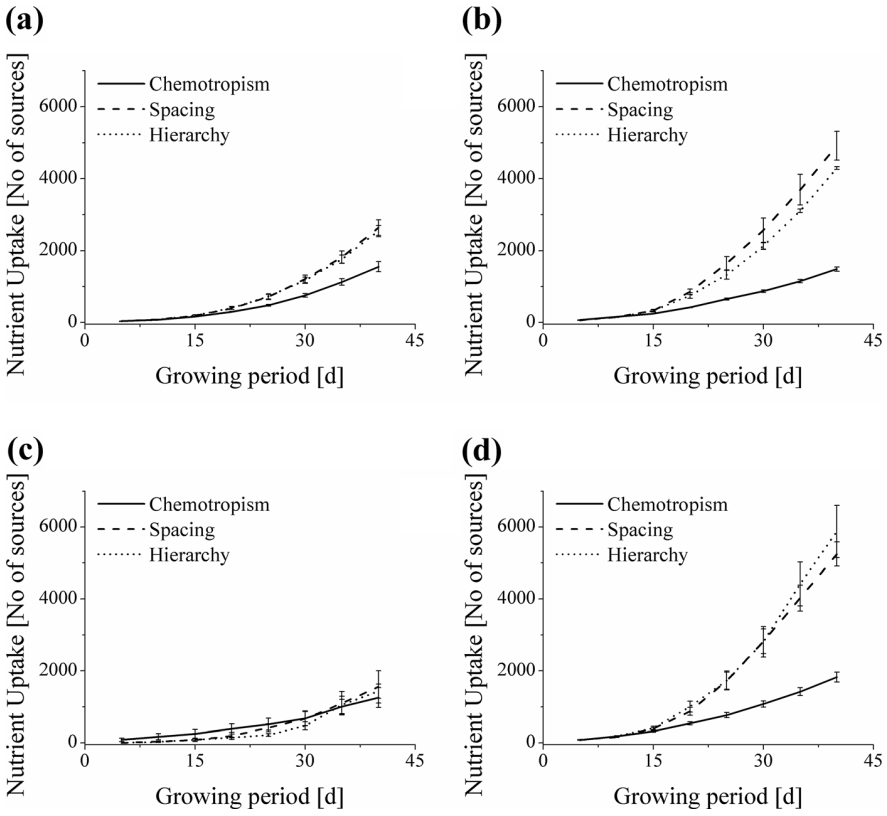


Fig. 7 Resources uptake measured in number of sources for a growing period of 40 simulation steps for (**a**) random, (**b**) layer, (**c**) patch and (**d**) gradient distributions. The *continuous line* represents the chemotropism mechanism, the *dashed line* the spacing mechanism and the *short dashed line* the hierarchy mechanism. The *vertical bars* represent the standard error of the mean for five replications

short, compact roots with high branching density has been observed (Forde and Lorenzo 2001). Our model calculations showed that when the nitrogen was mainly available at the top of the soil box the roots produced had shorter organs and were located on the top end of the box as well, when the spacing mechanism was applied. Furthermore, this scenario showed that the nutrient uptake per root volume was much higher in comparison to a patchy or a completely random nutrient distribution. Similar results were also reported in previous studies (Fitter et al. 1991; Berntson 1994; Nielsen et al. 1994). The spacing mechanism was also more efficient in comparison with both other plasticity scenarios. Its efficiency can be attributed to the fact that this mechanism enables the root system to optimize the uptake by changing the root morphology and concentrating its roots on the nutrient rich patches of the soil. This effect has been shown also in practice mainly by patch experiments (Robinson 1994) or competition experiments between neighboring plants (Hodge et al. 1999).

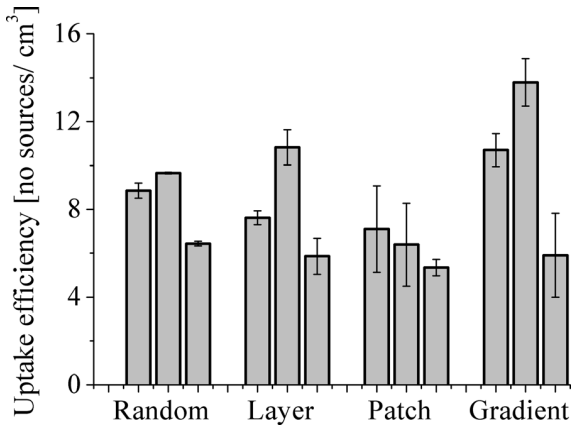


Fig. 8 Uptake efficiency expressed in number of taken-up nitrogen particles per cm^3 of root, for chemotropism, spacing and hierarchy mechanisms and different nutrient distributions (random R, layer L, patch P, and gradient G)

The change of the lateral root diameter depending on the nutrient conditions was also studied. It has been shown that in nutrient deficient soils finer roots are found (Fitter et al. 1991). In our simulations it was shown that this scenario is the most efficient in terms of absolute nutrient uptake. When a simulated root met a nutrient rich patch thicker roots were created increasing the root volume and therefore increasing the root surface available for nutrient uptake (Bilbrough and Caldwell 1995) so that the roots were able to branch. It has been shown that this scenario, though more demanding in terms of assimilate resources, in fact increases the uptake and transport capacity of the root (Fitter 1987). Nevertheless, it has been found that this mechanism in many species does not change significantly with nutrient availability (Hutchings and de Kroon 1994) and is usually seen only in fast growing species (Ryser et al. 1997) showing that the root only has a small time window in which nutrients are available for uptake. This aspect is dealt with in the model by considering only an average of three days' uptake in order for the root to adapt its diameter.

Our simulation study showed that uptake efficiency was dependent mainly on the root plasticity mechanism and only in the second place on nutrient distribution. The adaptation of the linear branching distance gave a distinct advantage to foraging when the root came into contact with the nutrient sources. When the sources were dispersed a mechanism of locating the nutrient sources conveyed a higher foraging ability. In reality the different mechanisms do not exist independently from each other but rather act synergistically inside the root system. However, the current literature provides no clear references of how these mechanisms interact and of the role they play in root foraging. In contrast to this, the present work seems to indicate that a hierarchy exists among the functioning of these mechanisms. Root plasticity during foraging is affected by root morphology as well as nutrient availability. Studies have shown that when a rich nutrient patch is available, a bigger root will have the higher absolute number of lateral roots in the patch while a smaller root will have a higher number of younger roots within the patch (Campbell et al. 1991; Wijesinghe et al. 2001). The first scenario can be associated with the spacing mechanism while the second is linked with the hierarchy mechanism. Therefore, we can propose that the root uses chemotropism as a global sensing device to locate the available nutrient source. After the nutrients are located depending on the morphology of the root and the quality of the nutrient patch found the root is employing for foraging either the spacing or the hierarchy mechanism.

Our current model version considers root development and growth but neglects root mortality. This could be considered a deficit, especially with respect to the role of the finest terminal roots, which have a rapid turnover rate (i.e. a relatively short life span) and a significant influence on the capacity of the root system to exploit nutrients locally. However, the time frame that we chose for the root development in our model was limited to 40 days and the expected root mortality was thus not expected to affect the simulation results.

In the current work we used a number of parameters to describe each scenario. However, the lack of information in the literature concerning the accuracy of these parameters calls for a sensitivity analysis enabling us to quantify the impact of parameters modulating the magnitude of the plastic responses. Conducting a meaningful sensitivity analysis would require a reparameterisation of our model with parameter values for a specific crop. A methodological framework for sensitivity analysis has recently been developed by Wu (2012) and could be applied to our model. It would be interesting to see whether the conclusions we have drawn from our generalized model still hold for the parameter sets of the different species or not. Furthermore, in our model the nutrient sources are static with no mobility allowed. Therefore, future work should also take into account the effect of nutrient movement in the simulated root architecture.

The modelling methodology used to implement the present root model was inspired by our ongoing work on a generalized modular modelling of morphological and physiological processes as simplified and uniform prototypes that can be principally applied to a number of species without the need to write basic code, which can be advantageous to the average botanical user who has no prior knowledge in informatics. First results of this effort have been presented by Henke et al. (2010) using the GroIMP modelling platform (Kniemeyer 2008).

The model presented here could be potentially relevant to address questions in domains like plant ecology or crop breeding: Representing basic mechanisms of root plasticity in response to nutrient resource availability, the present model or its extensions could be useful for investigating mechanisms of plant competition in monocultures or intercropping systems. 3D model-based design of

ideotypes with respect to certain relevant traits of the aerial architecture of a crop (e.g. light interception, stability), has recently been proposed as a valuable new method in horticultural crop breeding (e.g., Sarlikioti et al. 2011). Finding an optimal root architecture with respect to minimization of nitrogen loss or to maximization of N uptake in a given layered soil might be worthwhile applications of our model.

Acknowledgments V. Sarlikioti was funded by a grant of French National Institute of Agronomic Research (INRA, EA department).

References

Barlow P (2002) The root cap: cell dynamics, cell differentiation and cap function. J Plant Growth Regul 21:261–286. doi:10.1007/s00344-002-0034-z

Berntson GM (1994) Modeling root architecture: are there tradeoffs between efficiency and potential of resource acquisition? New Phytol 127:483–493. doi:10.1111/j.1469-8137.1994.tb03966.x

Bilbrough CJ, Caldwell MM (1995) The effects of shading and N status on root proliferation in nutrient patches by the perennial grass Agropyron desertorum in the field. Oecologia 103:10–16. doi:10.1007/BF00328419

Bingham IJ, Blackwood JM, Stevenson EA (1997) Site, scale and time-course for adjustments in lateral root initiation in wheat following changes in C and N supply. Ann Bot 80:97–106. doi:10.1006/anbo.1997.0412

Buck-Sorlin GH (2013) Functional-structural plant modeling. In: Dubitzky W, Wolkenhauer O, Cho K, Yokota H (eds) Encyclopedia of systems biology, doi:10.1007/978-1-4419-9863-7

Campbell BD, Grime JP (1989) A comparative study of plant responsiveness to the duration of episodes of mineral nutrient enrichment. New Phytol 112:261–267. doi:10.1111/j.1469-8137.1989.tb02382.x

Campbell BD, Grime JP, Mackey JML (1991) A trade-off between scale and precision in resource foraging. Oecologia 87:532–538. doi:10.1007/BF00320417

Casper BB, Jackson RB (1997) Plant competition underground. Annu Rev Ecol Syst 28:545–570. doi:10.1146/annurev.ecolsys.28.1.545

Chen YL, Dunbabin VM, Postma JA, Diggle AJ, Siddique KHM, Rengel Z (2013) Modelling root plasticity and response of narrow-leafed lupin to heterogeneous phosphorus supply. Plant Soil 372(1–2):319–337. doi:10.1007/s11104-013-1741-x

Dunbabin V, Rengel Z, Diggle A (2004) Simulating form and function of root systems: efficiency of nitrate uptake is dependent on root system architecture and the spatial and temporal variability of nitrate supply. Funct Ecol 18:204–211. doi:10.1111/j.0269-8463.2004.00827.x

Einsmann JC, Jones RH, Pu M, Mitchell RJ (1999) Nutrient foraging traits in 10 co-occurring plant species of contrasting life forms. J Ecol 87:609–619. doi:10.1046/j.1365-2745.1999.00376.x

Epstein E, Bloom AJ (2005) Mineral nutrition of plants: principles and perspectives, 2nd edn. Sinauer Associates, Sunderland

Ericsson T (1995) Growth and shoot: root ratio of seedlings in relation to nutrient availability. Plant Soil 168–169:205–214. doi:10.1007/978-94-011-0455-5_23

Fang S, Yan XL, Liao H (2009) 3D reconstruction and dynamic modeling of root architecture in situ and its application to research on rice phosphorus acquisition. Plant J 60:1096–1108. doi:10.1111/j.1365-313X.2009.04009.x

Fang S, Clark R, Liao H (2012) 3D quantification of plant root architecture in situ. Measuring roots. Springer Berlin 135–148. doi:10.1007/978-3-642-22067-8_9

Fitter AH (1987) An architectural approach to the comparative ecology of plant root systems. New Phytol 106:61–77. doi:10.1111/j.1469-8137.1987.tb04683.x

Fitter AH, Nichols R, Harvey ML (1988) Root system architecture in relation to life history and nutrient supply. Funct Ecol 2:345–351. doi:10.2307/2389407

Fitter AH, Stickland TR, Harvey ML, Wilson GW (1991) Architectural analysis of plant root systems. 1. Architectural correlates of exploitation efficiency. New Phytol 118:375–382. doi:10.1111/j.1469-8137.1991.tb00018.x

Fitter AH (1994) Architecture and biomass allocation as components of the plastic response of root systems to soil heterogeneity. In: Caldwell MM, Pearcy RW (eds) Exploitation of environmental heterogeneity of plants. Academic, New York, pp 305–323

Forde B, Lorenzo H (2001) The nutritional control of root development. Plant Soil 232:51–68. doi:10.1023/A:1010329902165

Granato TC, Raper CDJ (1989) Proliferation of maize (*Zea mays* L.) roots in response to localised supply of nitrate. J Exp Bot 40:263–275. doi:10.1093/jxb/40.2.263

Henke M, Kurth W, Buck-Sorlin GH (2010) A general FSPM for prototyping, intercropping and education. In: DeJong T and Da Silva D (eds) Proceedings of the 6th International Workshop on Functional-Structural Plant Modeling, University of California, Davis, U.S.A., 12–17 September 2010, p 264

Hodge A, Robinson D, Griffiths BS, Fitter AH (1999) Why plants bother: root proliferation results in increased nitrogen capture from an organic patch when two grasses compete. Plant Cell Environ 22:811–820. doi:10.1046/j.1365-3040.1999.00454.x

Hodge A (2004) The plastic plant: root responses to heterogeneous supplies of nutrients. New Phytol 162:9–24. doi:10.1111/j.1469-8137.2004.01015.x

Hutchings MJ, de Kroon H (1994) Foraging in plants: the role of morphological plasticity in resource acquisition. Adv Ecol Res 25:159–238. doi:10.1016/S0065-2504(08)60215-9

Kniemeyer O (2008) Design and implementation of a graph grammar based language for functional-structural plant modelling. Dissertation, BTU Cottbus

de Kroon H, Visser EJW, Huber H, Mommer L, Hutchings MJ (2009) A modular concept of plant foraging behaviour: the interplay between local responses and systemic control. Plant Cell Environ 32:704–712. doi:10.1111/j.1365-3040.2009.01936.x

Leitner D, Klepsch S, Bodner G, Schnepf A (2010) A dynamic root system growth model based on L-systems. Tropisms and coupling to nutrient uptake from soil. Plant Soil 332:177–192. doi:10.1007/s11104-010-0284-7

Levang-Brilz N, Biondini ME (2002) Growth rate, root development and nutrient uptake of 55 plant species from the Great Plains Grasslands, USA. Plant Ecol 165:117–144. doi:10.1023/A:1021469210691

Lynch J, Nielsen K, Davis R, Jablokow A (1997) Simroot: modeling and visualization of botanical root systems. Plant Soil 188:139–151. doi:10.1023/A:1004276724310

Lynch JP, Brown KM (2001) Topsoil foraging—an architectural adaptation of plants to low phosphorus availability. Plant Soil 237:225–237. doi:10.1023/A:1013324727040

Marina GG, Brown JS, Gersani M (2002) Intra-plant versus inter-plant root competition in beans: avoidance, resource matching or tragedy of the commons. Plant Ecol 160:235–247. doi:10.1023/A:1015822003011

Malamy JE (2005) Intrinsic and environmental response pathways that regulate root system architecture. Plant Cell Environ 28:67–77. doi:10.1111/j.1365-3040.2005.01306.x

Monshausen GB, Gilroy S (2009) The exploring root—root growth responses to local environmental conditions. Curr Opin Plant Biol 12:766–772. doi:10.1016/j.pbi.2009.08.002

Nielsen KL, Lynch J, Jablokow AG, Curtis PS (1994) Carbon cost of root systems: an architectural approach. Plant Soil 165:161–169. doi:10.1007/978-94-017-0851-7_16

Pagès L (1995) Growth patterns of the lateral roots of young oak (Quercus robur) tree seedlings. Relationship with apical diameter. New Phytol 130:503–509. doi:10.1111/j.1469-8137.1995.tb04327.x

Pagès L (2011) Links between root developmental traits and foraging performance. Plant Cell Environ 10:1749–1760. doi:10.1111/j.1365-3040.2011.02371.x

Prusinkiewicz P, Lindenmayer A (1990) The algorithmic beauty of plants. Springer, New York, ISBN: 0-387-97297-8

Robinson D (1994) The responses of plants to non-uniform supplies of nutrients. New Phytol 127:635–674. doi:10.1111/j.1469-8137.1994.tb02969.x

Robinson D (2001) Root proliferation, nitrate inflow and their carbon costs during nitrogen capture by competing plants in patchy soil. Plant Soil 232:41–50. doi:10.1023/A:1010377818094

Rose DA (1983) The description of the growth of root systems. Plant Soil 75:405–415. doi:10.1007/BF02369974

Ryser P, Verduyn B, Lambers H (1997) Phosphorus allocation and utilization in three grass species with contrasting response to N and P supply. New Phytol 137:293–302. doi:10.1046/j.1469-8137.1997.00807.x

Sarlikioti V, de Visser PHB, Buck-Sorlin GH, Marcelis LFM (2011) How plant architecture affects light absorption and photosynthesis in tomato: towards an ideotype for plant architecture using a functional-structural plant model. Ann Bot 108(6):1065–1073. doi:10.1093/aob/mcr006

Scott BJ, Robson AD (1991) The distribution of Mg, P and K in the split roots of subterranean clover. Ann Bot 67:251–256

Toky OP, Bisht RP (1992) Observations on the rooting patterns of some agroforestry trees in an arid region of north-western India. Agrofor Syst 17:245–263. doi:10.1007/BF00123320

Tsutsumi D, Kosugi K, Mizuyama T (2003) Root-system development and water-extraction model considering hydrotropism. Soil Sci Soc Am J 67:387–401. doi:10.2136/sssaj2003.3870

Vos J, Marcelis L, de Visser P, Struik P, Evers J (2007) Functional–structural plant modelling in crop production – Adding a dimension. In: Vos J, Marcelis L, de Visser P, Struik P, Evers J (eds) Functional–structural plant modelling in crop production. Springer, Berlin, pp 1–10

Wijesinghe DK, Hutchings MJ (1997) The effects of spatial scale of environmental heterogeneity on the growth of a clonal plant: an experimental study with Glechoma hederacea. J Ecol 85:17–28

Wijesinghe DK, John EA, Beurskens S, Hutchings MJ (2001) Root system size and precision in nutrient foraging: responses to spatial pattern of nutrient supply in six herbaceous species. J Ecol 89:972–983. doi:10.1111/j.1365-2745.2001.00618.x

Wu QL (2012) Sensitivity analysis for functional-structural plant modeling. Dissertation, École Centrale Paris

Zhang H, Forde BG (2000) Regulation of Arabidopsis root development by nitrate availability. J Exp Bot 342:51–59. doi:10.1093/jexbot/51.342.51

CHAPTER 8

Sixth Paper

This paper is accepted as:
Henke M, Buck-Sorlin GH; Using a full spectral raytracer for the modelling of light microclimate in a functional-structural plant model, *Computing and Informatics*

Authorship

- Gerhard H. Buck-Sorlin supported the writing of the manuscript.

Computing and Informatics, Vol. XX, XXXX, 1–31, V 2017-Mar-28

USING A FULL SPECTRAL RAYTRACER FOR CALCULATING LIGHT MICROCLIMATE IN FUNCTIONAL-STRUCTURAL PLANT MODELLING

Michael Henke, Gerhard H. Buck-Sorlin

IRHS, INRA, AGROCAMPUS-Ouest, Université d'Angers, SFR 4207 QUASAV, 42 rue Georges Morel, 49071 Beaucouzé cedex, France
e-mail: mhenke@uni-goettingen.de, gerhard.buck-sorlin@agrocampus-ouest.fr

Abstract. Raytracers that allow the spatially explicit calculation of the fate of light beams in a 3-d scene allow the consideration of shading, reflected and transmitted light in functional-structural plant models (FSPM). However, the spectrum of visible light also has an effect on cellular and growth processes. This recently created the interest to extend this modelling paradigm allowing the representation of detailed spectra instead of monochromatic or white light and to extend existing FSPM platforms accordingly. In this study a raytracer is presented which supports the full spectrum of light and which can be used to compute spectra from arbitrary light sources and their transformation at the organ level, by absorption, reflection and transmission in a virtual canopy. The raytracer was implemented as an extension of the FSPM platform GroIMP.

Keywords: Full spectral raytracing, light modelling, FSPM, GPU, photosynthesis, GroIMP

Mathematics Subject Classification 2010: 68-U05

1 INTRODUCTION

Accurate computation of light flux in a plant canopy and thus of its light micro-climate should be a prerequisite for every crop model, whether it considers a single plant individual or an entire canopy, since light is the single-most important input parameter for a photosynthesis model, and photosynthetic activity controls plant growth and

development. Functional-structural plant modelling (FSPM) refers to a paradigm for the description of a plant by creating a (usually object-oriented) computer model of its structure and selected physiological and physical processes, at different hierarchical levels: organ, plant individual, canopy (a stand of plants), and in which the processes are modulated by the local environment [6]. By better describing the heterogeneity of the micro-environment and considering physiological processes that are modulated by it, FSPM have become increasingly realistic. Correspondingly, on the functional side, implemented processes have become much more complex. Most approaches for light computation have been focusing on the quantity of photosynthetically active radiation (PAR) reaching different parts of a plant. Light quality is as important as quantity, but much harder to estimate quantitatively. Instead of a single ray with only one power value the entire spectral composition of each ray needs to be traced, with reflection, absorption, and transmission being different for each wavelength. Such complex and computationally demanding processes become manageable with the development of highly parallel computing techniques on the graphics card (GPU) [39]. Light quality exerts a significant influence on canopy development [19, 1, 3]. Light quality, via photomorphogenesis, influences shoot architecture and source/sink ratio, and thus indirectly plays a major role for, e. g., fruit quality [5, 20]. Furthermore, reflection and transmission spectra varied considerably among light- and shade-adapted leaves in different apple cultivars [35].

In the past 25 years, several approaches to estimate the light environment have been developed. Greene [21] considered the entire sky as a hemispherical diffuse light source and computed the local light environment within a plant canopy using raycasting. Another early approach was the one by Kanamaru [25]: here, the amount of light reaching a given sampling point was calculated by assuming that it was at the centre of projection, and by subsequently projecting all leaf clusters of a tree onto a hemisphere surrounding this point. The Transrad model by Dauzat [14] simulates multiple scattering of light and returns the complete radiative balance of a canopy. Mech [37] introduced a light environment model based on Monte Carlo (MC) path tracing of photons, with the possibility of interfacing it with virtual plants created using open L-systems [38]. Besides allowing the computation of the absorbed power this approach was also capable of calculating the spectral composition of light. The LIGNUM model implemented two approaches: a raycasting based approach called "mutual shading of segments" [40] and a voxel space method described in [44]. Disney [17] reviewed the use of MC methods in optical canopy reflectance modelling. He predicted a good deal of potential for MC based methods but also adjusted advantages for current analytical methods in cases where speed, invertibility, or a generalised statement of parameter influences are key. Estimation of canopy light interception by using the Beer-Lambert law is a simplified method used in many crop models. This method only accounts for leaf area index (LAI) and leaf angle distribution (LAD) without considering the crop's structural heterogeneity in space. Certain modelling approaches that are intermediate between process-based and functional-structural plant models, e. g. GreenLab [13] used this simplified approach. Wang [49] introduced light interception based on photon mapping to replace the Beer-Lambert law in the

Qingyuan software, a GreenLab clone. The CARIBU model implemented radiosity for light sampling [11]. CARIBU was subsequently made a part of the OpenAlea software package [42, 10]. AmapStudio, Simeo and AmapSim [34] used the MMR model implemented in the Archimed simulation platform [15, 16]. MMR performs calculations in three steps (1) MIR calculates the incident radiation intercepted by plant organs; (2) MUSC calculates the scattering of light within the canopy which is divided into horizontal layers and (3) RADBAL combines the previous results according to radiative conditions provided by a meteorological data file. The model outputs provide the irradiation of plant organs and a map of radiation reaching the ground. The Xplo software used this approach, too [45]. Cieslak [12] used a randomised quasi-Monte Carlo (RQMC) sampling method (QuasiMC) and confirmed that RQMC offers advantages in speed and/or accuracy improvement over MC.

A common work flow for most approaches is to follow a multi-stage process of exporting the 3-d scene to a format that can be imported by an external renderer (library/software) and to then reimport the results of the light computation into the core model for further use. Working directly on the generated structure and in this way making the steps of exporting and reimporting redundant would be an obvious way to save computation time, given that the whole work flow is already computationally expensive. The GroIMP platform was among the first model environments that included a Monte-Carlo radiation model [23].

Based on these developments of MCRT methods for FSPM [23, 12], we have published a number of articles [7, 8, 48] describing applications and validations of these existing light modelling methods to concrete cropping situations (rose and tomato production under controlled conditions in the greenhouse), thereby also showing up the gaps and weak points associated with these approaches. The present study describes the latest extension of GroIMP allowing full spectral raytracing powered by parallel computing on the GPU. To our knowledge, in the past seven years, no meaningful progress has been made in the field of light modelling methodology for FSPM. Therefore, the present paper is an attempt to catch up with the needs for progress in light modelling identified from own applications and from enquiring within the community of FSPM modellers.

2 MATERIAL AND METHODS

In order to make full spectral raytracing available for FSPM, and to allow the computation of the spectra from arbitrary emitting light sources and their transformation at organ level by absorption, reflection and transmission in a virtual canopy, a framework that supports the following fundamental aspects is required: 1) a global illumination model (light model), 2) light sources, and 3) a local illumination model (shader) (Fig. 1).

The features presented in this work have been implemented and integrated in the framework of the modelling software GroIMP [22, 32, 31], with the integrated language XL [29, 30]. The hardware requirement to perform GPU-based raytracing

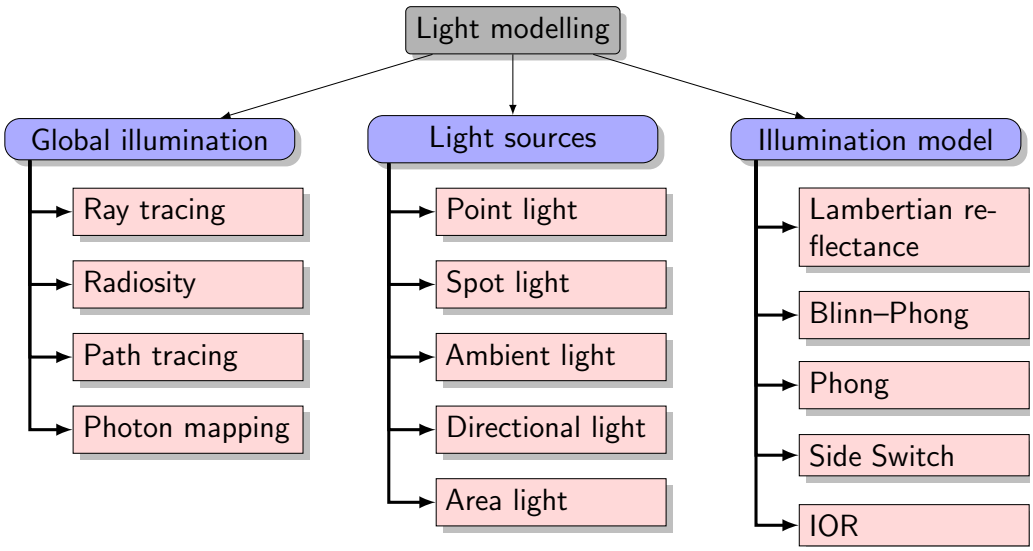


Fig. 1: Main computation techniques, light sources, and local illumination models used in computer graphics. The list of examples is not exhaustive.

is a programmable graphics card with OpenCL support (SSE > 4.1, [33]).

2.1 Light model

Fig. 2 illustrates the overall work flow of light transport simulation within a 3-d scene. The light model acts as the overall control unit: Depending on the method used for light calculation it performs different steps to estimate the light distribution. For standard raytracing a defined number of rays is emitted by one or several light sources. Each ray is traced throughout the scene and in case it hits an object is treated according to the local optical properties of the hit object, cf. Sec. 2.3 and Fig. 7. For each object in the scene the amount of absorbed light is collected.

GPUFlux, an integrated light model implemented by [47], is a high-performance light model that uses OpenCL [28] to directly access the processor of the graphics card (graphical processing unit - GPU) as well as CPUs that support SSE > 4.1 [33]. Since GPUs are designed to perform highly parallel computation, the computation time can be reduced at least by factor ten and up to more than one hundred times (depending on the compared CPU and GPU). This and the fact that multiple devices, e. g. several GPUs and all threads of a CPU, are supported in parallel considerably speeds up light computation. As a further feature, the full spectrum of light is supported with a minimal optical resolution of 1 nm, over an arbitrary spectrum, but with default values ranging between 300 nm and 800 nm - the range is not limited by the system, however, far outside the visible spectrum it will not produce correct results, as the physical properties of such rays will be too different from those within the visible spectrum. Finally, three types of illumination models are implemented: a

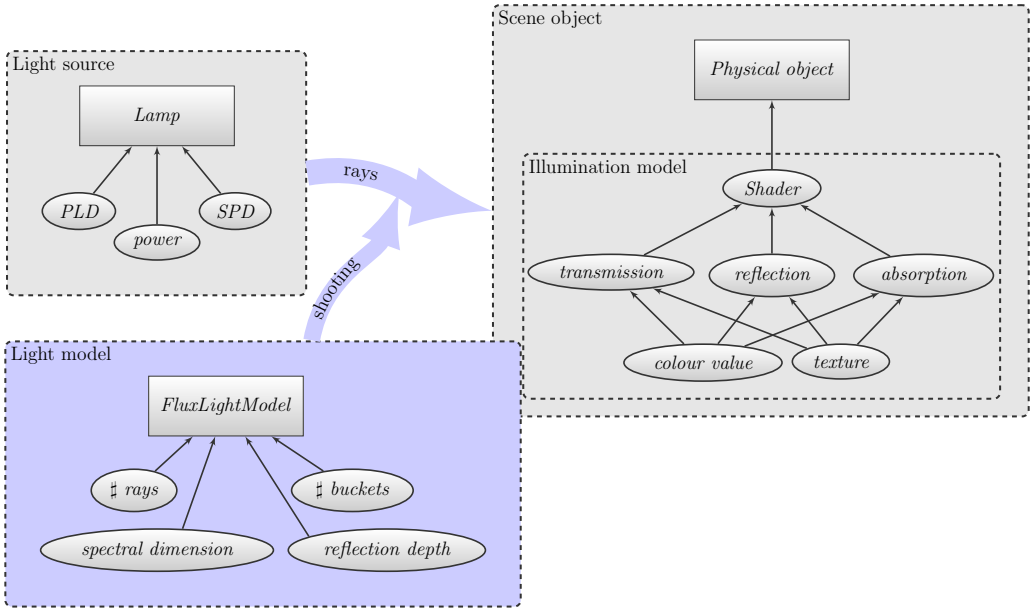


Fig. 2: Diagrammatic representation of a common system for light modelling. The light source and the objects are situated in a virtual scene: After invoking the light model all light sources emit virtual rays into the scene, and their paths are traced until one of the cut-off criteria is reached: ray leaving the scene; ray spectral power below a threshold value; maximum number of reflections reached.

path tracer, a bidirectional path tracer and spectral renderer based on a spectral Monte-Carlo light tracer.

Each ray is traced through the scene until one of the following cutoff conditions is triggered: 1) minimal power of a ray is lower than a predetermined threshold power, 2) the maximal depth of reflections is reached, or 3) the ray leaves the bounding box of the scene. To control precision of calculation and computation time the cutoff power and maximal recursion depth can be defined by the user.

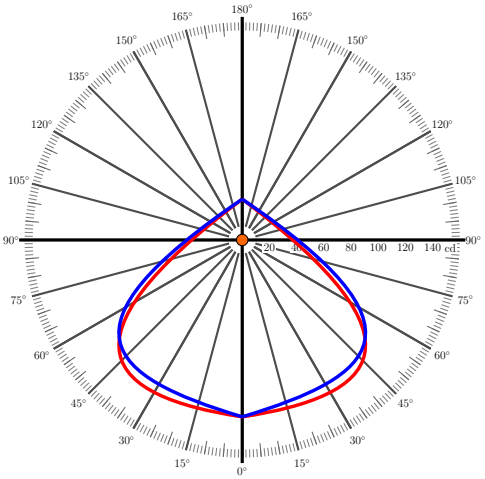
2.2 Light sources

An arbitrary light source can be defined by two parameters:

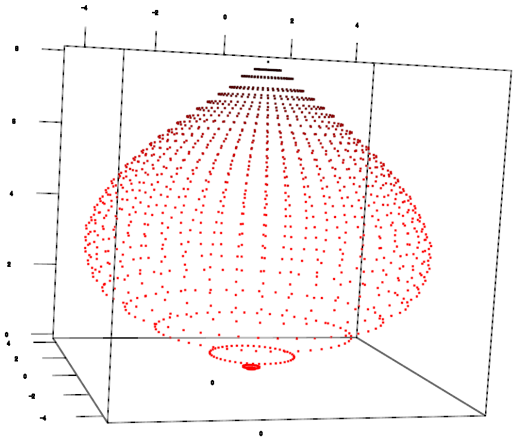
1. physical light distribution (PLD),
2. spectral power distribution (SPD).

The physical light distribution describes the luminous intensity, i.e. the measure of the wavelength-weighted power emitted by a light source in a particular direction per unit solid angle (cf. Fig. 3a), based on the luminosity function, a standardised model of the sensitivity of the human eye, over the whole sphere (cf. Fig. 3b). It is usually measured in candela (cd) per steradian beam (st) and measured using a

goniophotometer for light sources such as light bulbs. Most manufacturers of light sources provide this information on their websites for public use (see Sec. 3.1 for the conversion from candela to watt).



(a) A polar distribution diagram (also called polar curve) showing the luminous intensity values with increasing angles from two imaginary axes of the lamp which is placed in the centre. Red: $0 - 180^\circ$ plane, blue $90 - 270^\circ$ plane.



(b) 3-d Visualisation of the same light source. The colour of each point (gradient from black to bright red) as well as the distance to the light source both indicate the power emitted by a light source in a particular direction per unit solid angle.

Fig. 3: Two visualisations of a physical light distribution of an not further defined light bulb. The 2D case shown in sub-figure (a) is the common case usually provided by manufacturers.

One common file format to describe a PLD is called IES and has been introduced by the Illuminating Engineering Society [24]. Another common format is LUM. Both can be directly imported by GroIMP and both are simple ASCII files that can be converted into each other without problems. Fig. 4 shows a GroIMP snapshot of a set of common predefined light sources provided by GroIMP. By turning on an option of these lights the physical light distribution can be visualised through simple lines. The rendered result of a lamp demo model which is a default example included in GroIMP is shown in Fig. 5.

The spectral power distribution (SPD) measurement describes the power per unit area per unit wavelength of an illumination. The ratio of spectral concentration (irradiance or exitance) at a given wavelength to the concentration of a reference wavelength provides the relative SPD, as shown in Fig. 6 for a high-pressure sodium (SON-T) lamp, which is commonly used as additional growth light source in greenhouses.

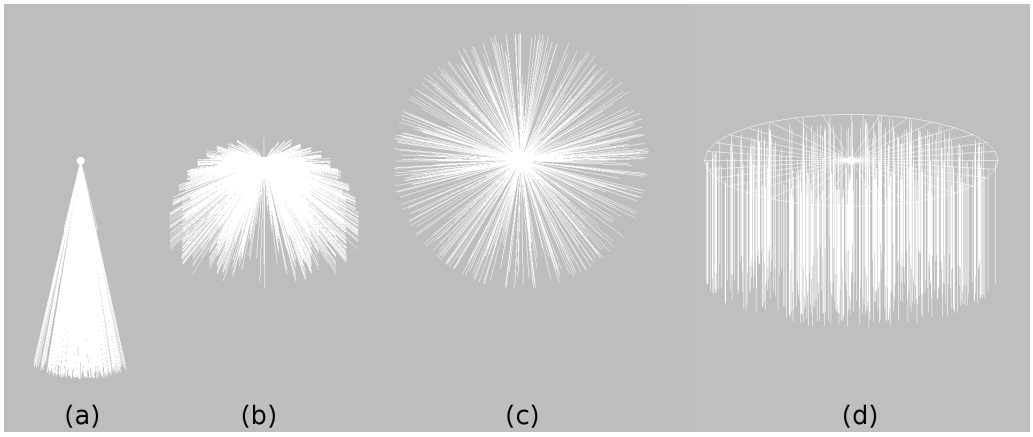


Fig. 4: Visualisation of physical light distributions of different light sources: a) spot light, with a defined opening angle; b) user defined distribution; c) point light, equally distributed; d) directional light, equal distribution over an area.

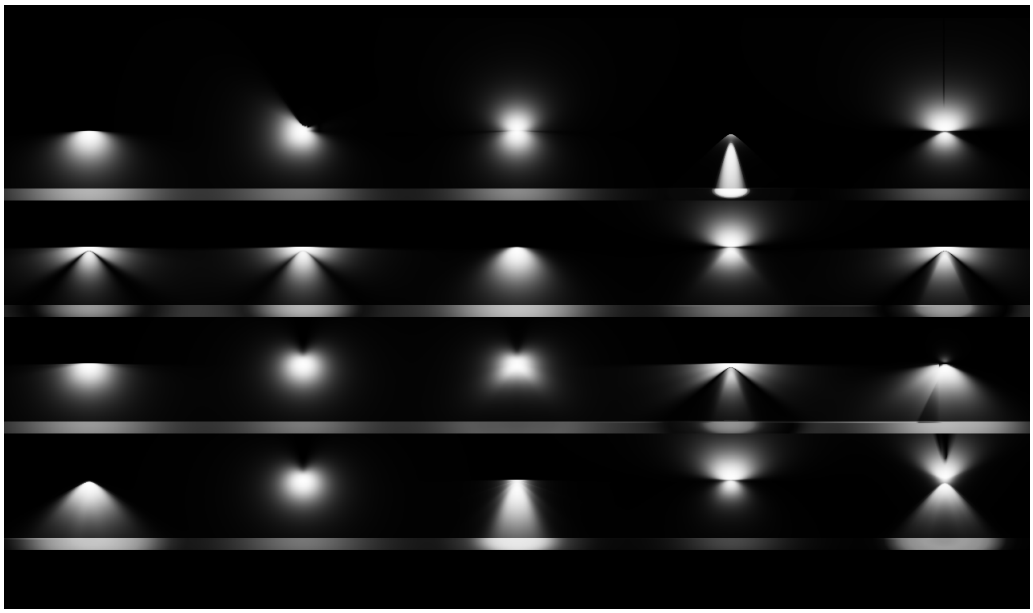


Fig. 5: GroIMP snapshot of a lamp demonstration model (rendered image), simulating a set of 20 lamps placed on a wall. The camera is looking from the side showing the distribution of light reflected from the wall and a part of the pattern produced on the ground. Model source is available in the example gallery of GroIMP 1.5.

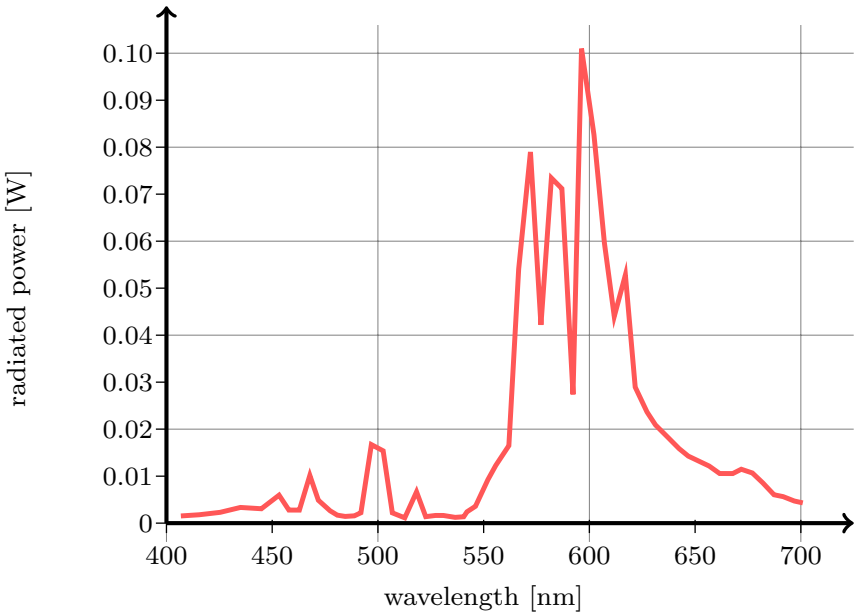


Fig. 6: Spectral power distribution of an EYE Lighting - Sunlux® LU400 lamp. The resulting line on the graph is the Spectral Power Distribution (SPD) Curve, and shows the power distribution across the visible spectrum.

2.3 Illumination model

An illumination model describes the local illumination, more generally the local optical properties of an object at a certain point on the surface (Fig. 7), which includes the three basic properties, absorption, reflection, and transmission.

In computer graphics the optical properties of an object are defined by a shader which is mapped onto the surface of an object. The most common shaders are the Lambertian reflectance [2], which only supports diffuse reflection, and the Phong shader, developed by Phong [41], which is an advanced shader that supports ambient, diffuse and specular reflection (Fig. 8).

In GroIMP (Code 2.3), each property of each type of shader can be defined separately. The implemented Phong shader allows the definition of values for: shininess, transparency, ambient and specular reflection, emission, diffuse reflection, transparency shininess, and diffuse transparency. Each one of these properties can be defined either as constant for each colour value (graytone) or for each base colour independently (RGB colour). Additionally, for spectral raytracing spectral power distributions can be applied.

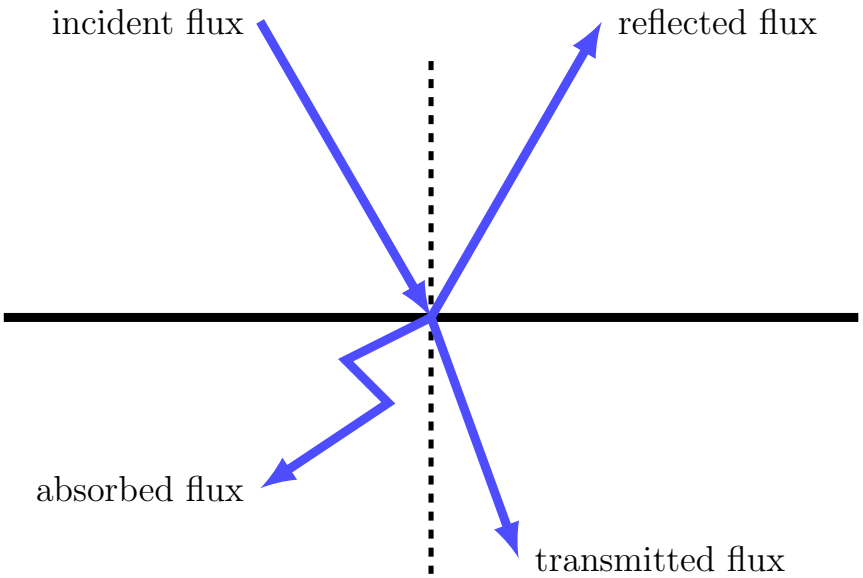


Fig. 7: Definition of local illumination. On impact at the surface of a material, the incoming flux is split up into a reflected, absorbed and transmitted flux. Both the transmitted and reflected flux can be further subdivided into a direct and diffuse part.

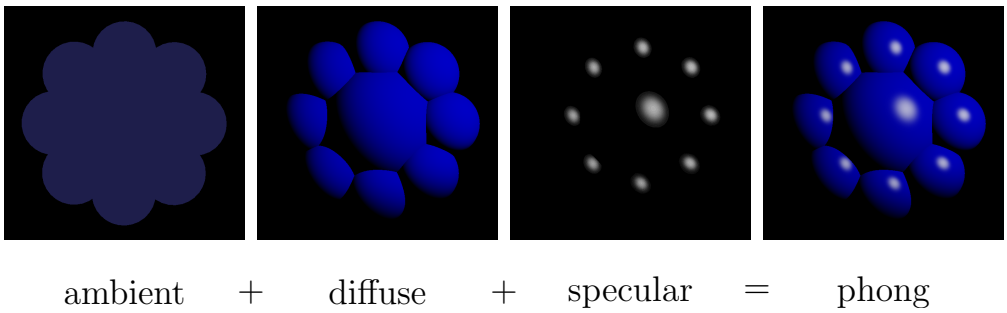


Fig. 8: Phong shader, consisting of a combination of ambient, diffuse and specular reflection.

2.4 Verification

As described above three aspects, a global illumination model (light model), light sources, and an illumination model (shader), are needed to perform spectral raycasting for FSPM. Whenever the light climate in a complex system is to be evaluated it is appropriate first to verify each part separately.

Only a few parameters of the light model can be evaluated directly. One thing to validate is whether the total amount of power absorbed by all objects of a scene

is equal to the total power output of all light sources. In GroIMP this can be done using what is called an execution rule, identified by the operator "::>". This type of rule will leave the graph structure unchanged (i. e. the topology of nodes and edges), while modifying one or more parameters associated with the node or subgraph; the node, or subgraph, identified by the search pattern on the left-hand side of the rule is thus not replaced by anything on the right-hand side of the rule as is the case in normal L-system rules. In this example the light model is invoked to calculate the amount of absorbed radiation by a specific object before this amount is added to the global counter:

```
float total = 0;
x: ShadedNull ::> { total += LM.getAbsorbedPower(x).integrate(); }
```

For each object found the light model (LM) is invoked to obtain the amount of light absorbed. Here *ShadedNull* is a super class of all visible objects in GroIMP. Since the result is returning a spectrum it needs to be integrated before it can be added to the total sum (done with the method *integrate()* of LM). An alternative way, assuming the absorbed power has already been calculated by the light model and stored within an attribute *absorbedPower* of the objects, would be the use of graph queries:

```
float total = sum( (* ShadedNull *).absorbedPower );
```

that can directly be called on the GroIMP console. The part "(* *pattern* *)" indicates a graph query searching a specific pattern within the graph (in this case of all nodes of type *ShadedNull*). All parts of the graph matching the pattern will be returned by the query and are available for further actions. Here we query the amount of power absorbed by each object. The results for each object are aggregated by the *sum* function to obtain the final result.

For light sources 1) the spectral power distribution, 2) the physical light distribution as well as 3) the total power output are of interest. Regarding 1) and 3) the verification is easy: 1) each ray is initialised by the light model per default with the defined spectrum, 3) the total power is equally distributed over all emitted rays. To check this, a sphere with a black shader can be used as test object, as this will absorb all incident light. The light source is placed in its centre and the light model will be executed once to obtain the total amount of light absorbed by the sphere. The verification of the PLD (2) requires more effort: since RT is a stochastic process the actual light distribution depends on the number of rays used. It is converging with increasing number of rays to the distribution given by the light source. To get an idea of how many rays are needed to obtain a converging distribution, we implemented a small test environment simulating a goniophotometer with the same number of sensor nodes as defined in the PLD-file, arranged in a sphere around the light source (Fig. 9). In this scenario only direct light was registered. For a real 3-d scene with all the light interaction a much larger number of rays is needed, making it difficult to give a recommendation. The number of rays is proportional to the complexity of the scene. However, experience showed that increasing the number of rays used improved the results obtained, without any saturation effect being observed.

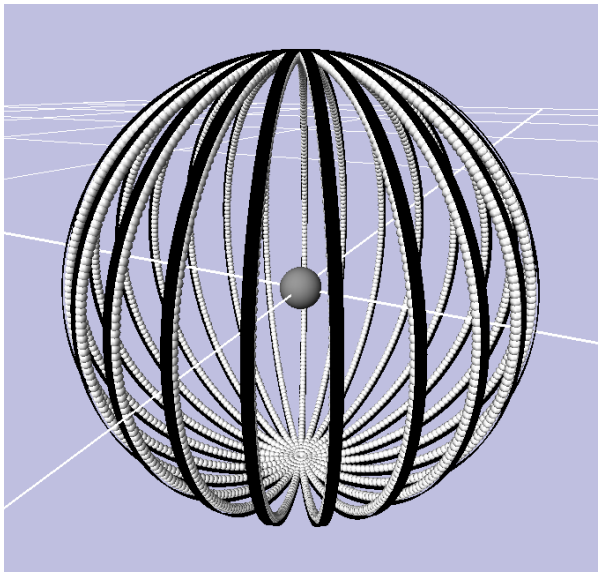


Fig. 9: 3-d visualisation of a sensor sphere used for verification of the PLD. The light source is placed at the centre of the sphere. The virtual sensor nodes (*SensorNode*) of the model are as many as, and at the same locations as, the real light sensors used to obtain the PLD.

The accuracy of a reconstructed light distribution as a function of number of used rays is given in Fig. 10. While the increment of computation time is linear, the MSE only decreases proportionally. At 20 million rays the MSE for the light source used in this test is around 15 cd (round 0.022 W), which would be small enough to be negligible for most applications. However, with up-to-date graphics hardware the number of rays can and should be increased to 100 or 200 million rays - which still requires only seconds.

The main factors influencing computation time are listed below:

- hardware: programmable graphics card; optionally a CPU that supports SSE > 4.1, [33]
- complexity of the scene (number of objects, complexity of objects $\Longleftrightarrow$ number of triangles / facets)
- number of light sources (play a role for rendering not for light modelling)
- resolution and range of investigated spectrum (number of buckets)
- recursion depth
- optical properties of objects (shader: e. g. transparency, emission)

To test the influence of numbers of rays on the reproducibility of the results of the light calculation a simple sensitivity test was performed using two scenarios, a simple scene with only one object and a complex scene with 2000 objects randomly

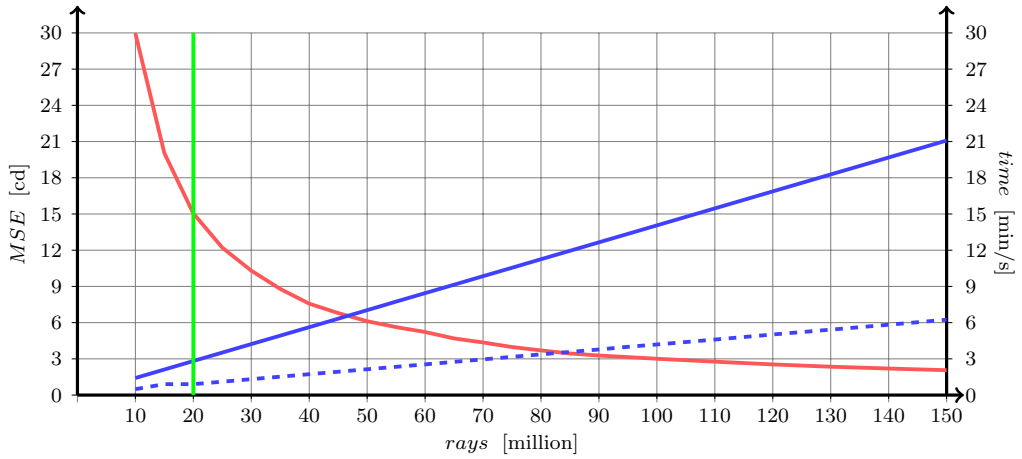


Fig. 10: Model accuracy as a function of number of rays used (red line). The objective was to obtain a realistic physical distribution with minimal computational investment. Here, realistic means a minimal mean square error (MSE). The vertical green line represents the recommended minimum number of rays (20 million), whereas 50 million rays will minimize the mean standard error to 6 cd. Error values measured in candela [cd] - 1 cd $\approx$ 1/683 W at 555 nm, see Sec. 3.1. Additionally, the computation times needed on a Nvidia Quadro FX 1700M (Date of Announcement: 01.10.2008) (blue line) and on a Nvidia GeForce GTX 880M (Date of Announcement: 12.03.2014) (dashed blue line) are given. – Note that for the Quadro card the unit is minutes while for the GeForce it is only maximally six seconds.

distributed in a box with five meter edge length. In both tests one single spotlight with an outer angle of 30 degrees was used as light source. It was placed ten metres above the ground where the test objects were placed. Each test object had an edge length of ten centimetres. Its shader was set to black for scenario 1 and in order to get reflections it was set to 50 per cent black for scenario 2. The recursion depth was set to ten for the complex scenario while for the simple scenario it was set to one. The standard deviation of absorption of the object was calculated for 25 repeated runs of the light model, while for each repetition the light model was initialised with a different random seed, thus resulting in different ray distributions. If the variation of the 25 repetitions is small it can be shown that the distribution is reproducible. However, it does not tell much about the quality of the obtained light level. Therefore, the test needs to be repeated with an increasing number of rays and the obtained mean standard deviation and variance need to be compared. It can be expected that when more rays are used the variance will become smaller and the mean standard deviation will converge. The blue line in Fig. 11 shows the standard deviation for 5 million up to 1.5 billion rays for the simple scene. For this very basic test scenario it can be observed that above 500 million rays the standard deviation nearly does not decrease further. In the second scenario, the complex scene, the absorption of each

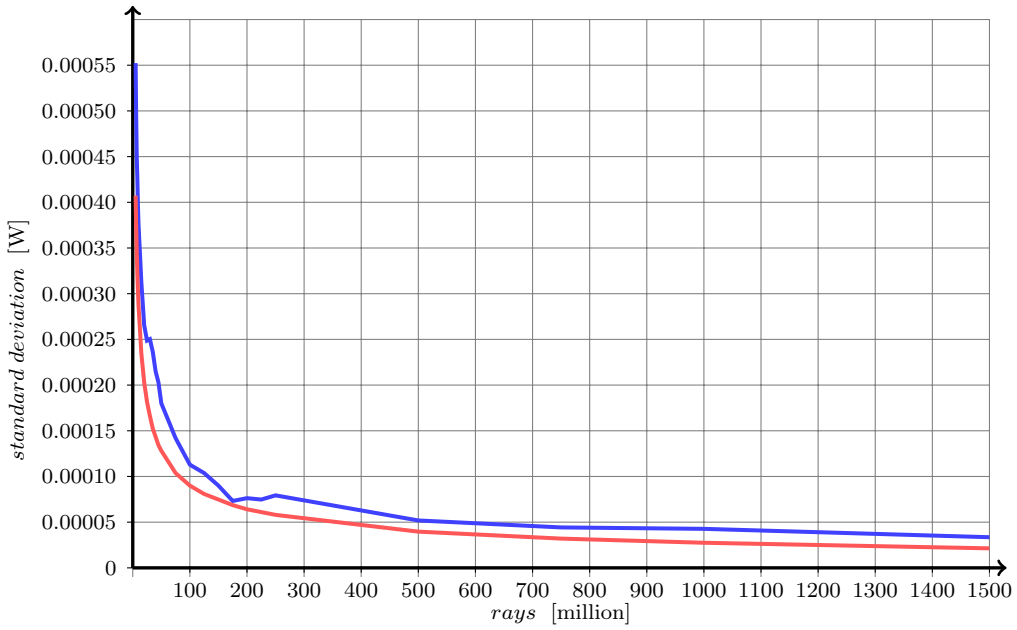


Fig. 11: Standard deviation of 25 repeated light model runs for an increasing number of rays (five million to 1.5 billion rays). Blue: simple scene; red: complex scene; (Note that one run with 1.5 billion rays for the complex scene with 2000 objects took 61 seconds on a Nvidia GeForce GTX880M.)

object was measured for 25 repetitions and the standard deviation was calculated. Afterwards, the average standard deviation of these 2000 standard deviations was calculated (red line in Fig. 11). While at first sight the results look similar they were on average 25 per cent better than for scenario 1. This can be explained by the simple fact that the surface area of the objects in the complex scenario is several times higher than in the simple scenario. With an increase in area the possibility of a ray to hit a particular surface increased, too, resulting in an equalisation of the average absorption during repeated tests. To sum up, it was observed that a minimum number of rays is needed to guarantee a satisfactory reproduction of a particular physical and spectral light distribution while a much higher number of rays are needed to obtain a qualitatively good light distribution simulation. With this in mind, a realistic light distribution - a prerequisite for a realistic plant simulation - requires no less than 50 million rays while any number below this is not acceptable. We recommend to use around 200 million rays to obtain a good compromise between computation time and quality of the obtained light distribution. These statements are made for a recursion depth of 10 reflections. With fewer reflections the number of rays needs to be higher.

To check the proper functioning of the shader a simple test environment (Fig. 12) was created: this consisted in two cylinder objects to define the boundaries, and a

virtual frame fixing the test shader in the middle and dividing the upper and lower cylinder. A spot light with a very small opening angle directly facing the shader was placed at the ground of the lower cylinder.

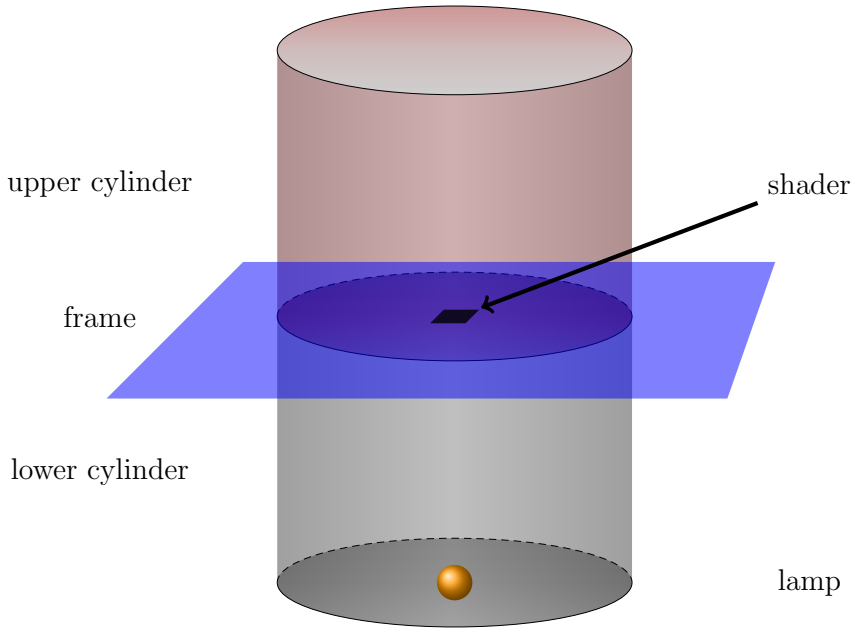


Fig. 12: Illustration of the shader test environment. It provides a self-contained system for testing the optical properties of a shader, mainly: absorption, reflection, and transmission.

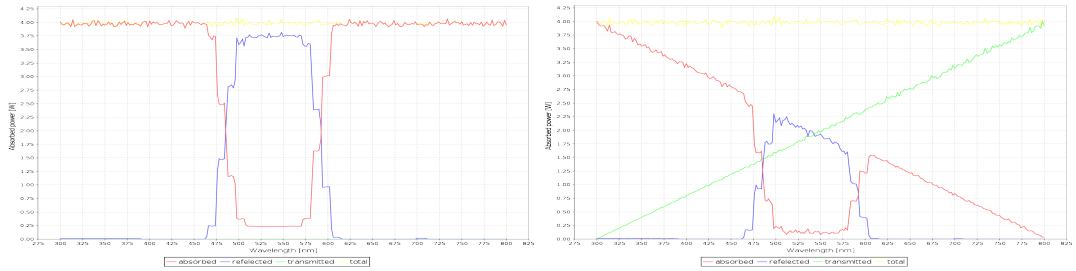
In the default configuration the power output of the spot light is set to 1000 W, the spectral resolution it set to 5 nm wide buckets in the range 380 nm to 780 nm, and the number of rays is set to 1 M. An example output of a pure green shader is displayed in Fig. 13.

3 ILLUSTRATION

To illustrate the possibilities of the full spectral RT presented here we chose two examples: The first one is a visualisation of a dispersion effect while the second one represents photosynthesis, one of the most important physiological processes. Regarding the latter, a wavelength dependent photosynthesis model will be presented.

3.1 Dispersion

In optics, dispersion is the phenomenon in which the phase velocity of a wave depends on its frequency [4]. One familiar consequence of dispersion is the change in the angle of refraction of different colours of light as commonly illustrated by the spectrum



(a) power absorbed by the shader (red) = 797.87 W; power reflected by the shader and therefore absorbed by the lower cylinder (blue) = 201.25 W; power transmitted by the shader and therefore absorbed by the upper cylinder (green) = 0 W, sum of power absorbed in the scene (yellow) = $a + r + t$ = 999.12 W

(b) power absorbed by the shader (red) = 396.326 W; power reflected by the shader and therefore absorbed by the lower cylinder (blue) = 106.56 W; power transmitted by the shader and therefore absorbed by the upper cylinder (green) = 496.23 W, sum of power absorbed in the scene (yellow) = $a + r + t$ = 999.12 W

Fig. 13: GroIMP snapshot of a chart showing the absorbed power of one object broken down to 5 nm buckets produced by the shader test environment for a) a pure green shader, b) a pure green shader with a transmission increased linearly over the whole spectrum. Initial power output of the lamp was set to 1000 W and the number of rays used for the simulation was 100 K.

produced by a dispersive prism. The same phenomenon can be observed with crystals as simulated in Fig. 14.

Fig. 14: GroIMP snapshot, produced with the Flux renderer, showing the "rainbow" effect produced by the dispersion of white light. The dispersion effect on this image was artificially enhanced for illustration reasons only.

3.2 Spectral light reaction - Photosynthesis

In order to model photosynthesis with a high level of accuracy several aspects such as temperature, humidity, CO_2 concentration and light microclimate need to be considered.

Photosynthesis, more specifically the light reaction, depends upon the absorption of light by pigments in the leaves. The diagram in Fig. 15 shows that the percentage of absorbed radiation varies depending upon its wavelength, with the lowest absorption in the green waveband (550 nm), and peaks in the red (650 nm) and blue (450 nm) waveband. The principal pigments responsible for the absorption are chlorophyll *a* and *b*. However, the absorption rate in the green waveband is not zero: about 20 % of it is absorbed due to other leaf pigments, e. g., beta-carotene and chlorophyll *b*. The plot of the absorption spectra of the chlorophylls plus beta carotene correlates well with the observed photosynthetic output, Fig. 15.

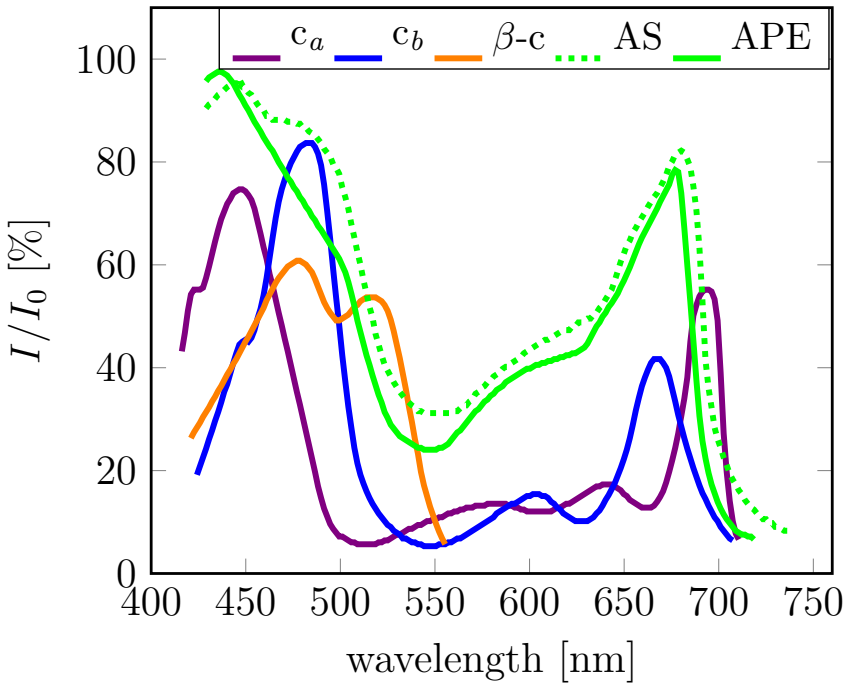


Fig. 15: Measured photosynthesis rate as a function of absorbed wavelength. The photochemically active pigments determine the action spectrum and thus photochemical efficiency. Legend: c_a = chlorophyll *a*, c_b = chlorophyll *b*, β-c = β-carotene, AS = absorption spectrum = relative light absorption, APE = actual photochemical efficiency = action spectrum. I/I_0 refers to the radiation absorbed, transmitted or reflected relative to incident radiation at the same wavelength. Data taken from [27].

Figure 16 shows measured values for reflection and transmission of light by a typical soybean leaf [26]. First of all, a shader was parameterised with these values

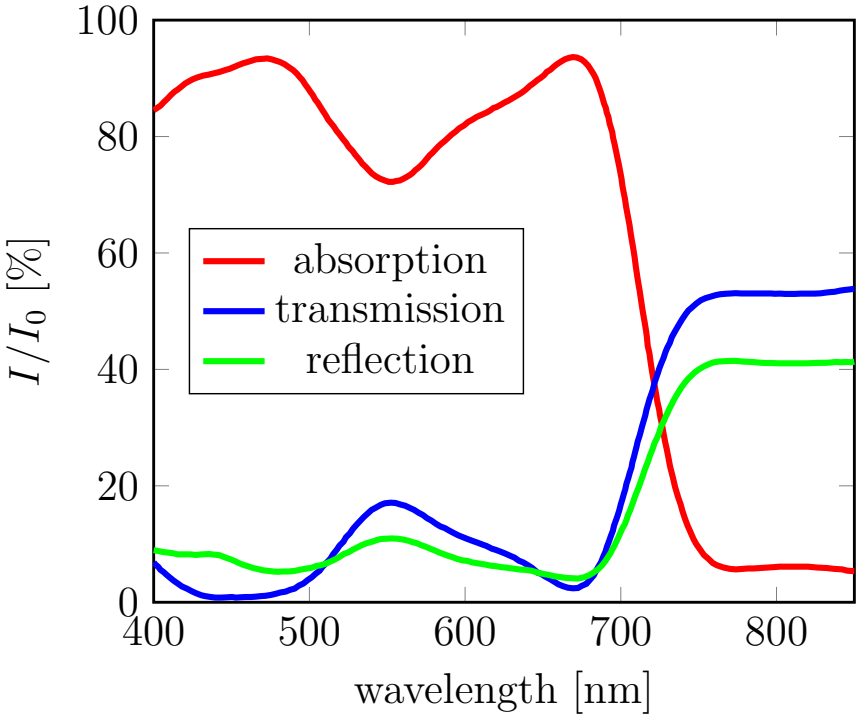


Fig. 16: Absorption, reflection and transmission of light by a typical soybean leaf. I/I_0 refers to the radiation absorbed, transmitted or reflected relative to incident radiation at the same wavelength. Data taken from [26].

(reflection and transmission) and placed into the shader test environment (Fig. 12) to verify the simulated values for absorption. This was repeated for two types of light sources, with a constant spectrum, and simulated daylight. The results are shown in Fig. 17 and Table 1.

The McCree Curve (Fig. 18) [36] was used to correct the absorbed radiance in order to obtain the usable amount of light per waveband.

Up to this point we have estimated and verified the absorbed spectrum of our simulated soybean leaf. In the next step we show how an absorbed spectrum can be used to calculate the corresponding assimilate production.

3.3 Calculation of Photosynthesis

Common photosynthesis models (PSM) use the integral of all absorbed wavelengths as photosynthetically active radiation (PAR) [Wm^{-2}] or photosynthetic photon flux density (PPFD) [µmol photons $m^{-2}s^{-1}$] as input to calculate assimilate production. The problem with this integration over all wavelengths is that the different wavelengths can not be treated as having equal efficiency, since the absorbing pigments (e. g. chlorophyll *a* and *b*, beta carotene) exhibit clear light spectrum-dependent

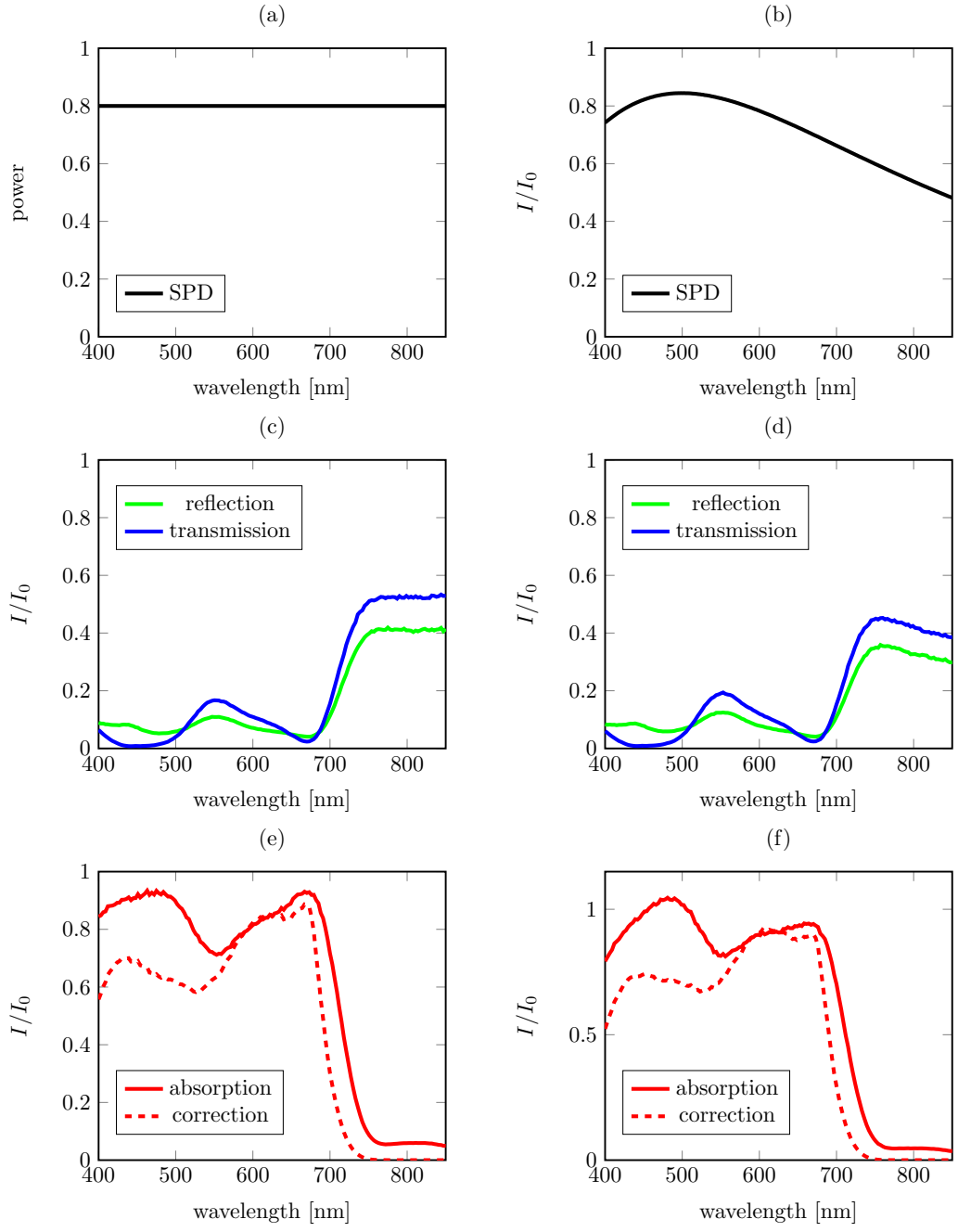


Fig. 17: Direct comparison of two different input light spectra and their behaviour in the presence of the soybean leaf shader described above: a) constant spectrum (case a), b) simulation of sunlight (case b); c) and d) reflected and transmitted spectrum; e) and f) absorbed spectrum and McCree corrected spectrum.

	original	case a	case b	case a/**ori**	case b/**ori**
transmission	31.585	30.979	27.653	0.981	0.876
reflection	25.923	25.724	23.191	0.992	0.895
absorption	93.714	93.247	99.110	0.995	1.058
corrected absorption	72.321	71.863	77.320	0.994	1.070

Table 1: Integrated absolute values for transmission, reflection and absorption of the simulated soybean leaf, Fig. 17, for a constant spectrum (a) and simulated sun light (b) in Watt. The last two columns represent the ratio between simulated and measured / original values.

differences in absorption efficiency (see above).

To our knowledge there is no spectral PSM available in the literature. The idea of a spectral PSM is to calculate the assimilates for each bucket independently. The actual photochemical efficiency (APE) from Fig. 15 is normalised so that the integral is one. This provides the wavelength efficiency distribution. Then a common PSM is used to calculate assimilate production for the given temperature, leaf age and integral of absorbed power. The result of this calculation is used to scale the normalised APE, the latter then yielding a distribution that follows the APE. Finally, the intersection of the scaled APE and the absorbed spectrum are calculated. The intersection describes the maximal possible assimilate production at the absorbed spectrum according to the estimated assimilate production that was calculated by a common PSM. Fig. 19 provides a schematic overview of the whole calculation.

In GroIMP, each object within the 3-d scene can be "queried" for the amount of light it has absorbed at a given time step. According to the light model used and its parameterisation the results differ. For this example the spectral RT, called GPUFlux, was used, with 5 million rays emitted by a single light source. The observed spectrum had 80 buckets, ranging from 300 nm to 700 nm in 5 nm steps.

The amount of absorbed power A_p of an object x of the 3-d scene is determined using a library method *getAbsorbedPowerMeasurement* of the light model (Code 2.1), which returns the spectrum as an array of absorbed radiation divided into buckets in $\mathrm{Wobj}^{-1}_{\mathrm{surface\_area}}\mathrm{bucket}^{-1}$.

For the calculation of the net photosynthesis A_n the model of Thornley [46] was used. As input parameter the integral of absorbed light converted into yielded photon flux (YPF) was used; a default temperature *temp* of 25 degrees C and a leaf age *age* of 30 days were assumed, the latter corresponding to full functionality.

Before the absorbed power could be used for further calculation it was necessary to consider photochemical efficiency. This required the distinction between different wavelength/buckets. The McCree Curve was used to weight PAR values according to the photosynthetic response at each wavelength and thus YPF approximated [36, 3]. The McCree Curve, also known as the Plant Sensitivity Curve, describes the action spectrum for an average plant. Fig. 18 shows the graph of the McCree Curve in the

range 325 nm to 775 nm.

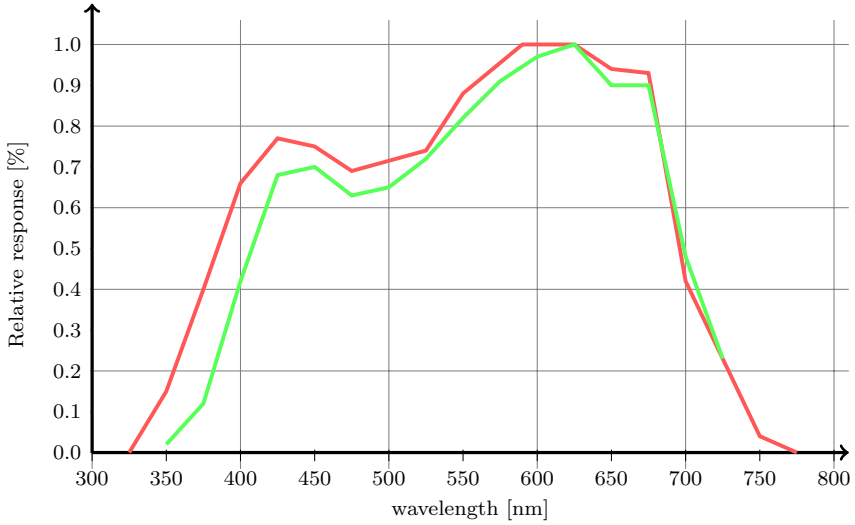


Fig. 18: The relative quantum efficiency curve, also known as the McCree Curve, as determined by the average plant response for photosynthesis rate; Red: (Wikipedia.org: Photosynthetically_active_radiation). Green: Mean relative quantum yield of field plant species [36], Tab. IV.

4 DISCUSSION

In this work we demonstrated how at present commonly available computer hardware can be used to drastically reduce computation time for calculation of light fate in a 3-d scene using inverted Monte Carlo raytracing. Furthermore, not only performance can be increased with our implementation but also the quality of light computation can be raised to a higher level by including information about the light spectrum. The simulation of natural light, including spectral information, opens up new possibilities for application in the domain of FSPM, such as realistic simulations of additional light sources, e.g. in greenhouses, climate chambers, or indoor farming, the latter in the context of urban horticulture, which will be playing an important role for local food production in the near future. Furthermore, the presented approach enables the modelling of an arbitrary diversity of specific light sources, e. g. High Pressure Sodium, Mercury Vapor, and LED lamps, in combination with commercially available reflectors and panels, e. g. double-ended wing, or raptor. The effect of certain wavebands on physiological processes is another domain in which the present model could be used: processes affecting plant growth or fruit quality traits are often affected by the strength of certain light signals; the most common example is the ratio of the red to far-red waveband which is instantly translated in the plant into a ratio of phytochromes. This in turn can initiate complex processes such as etiolation

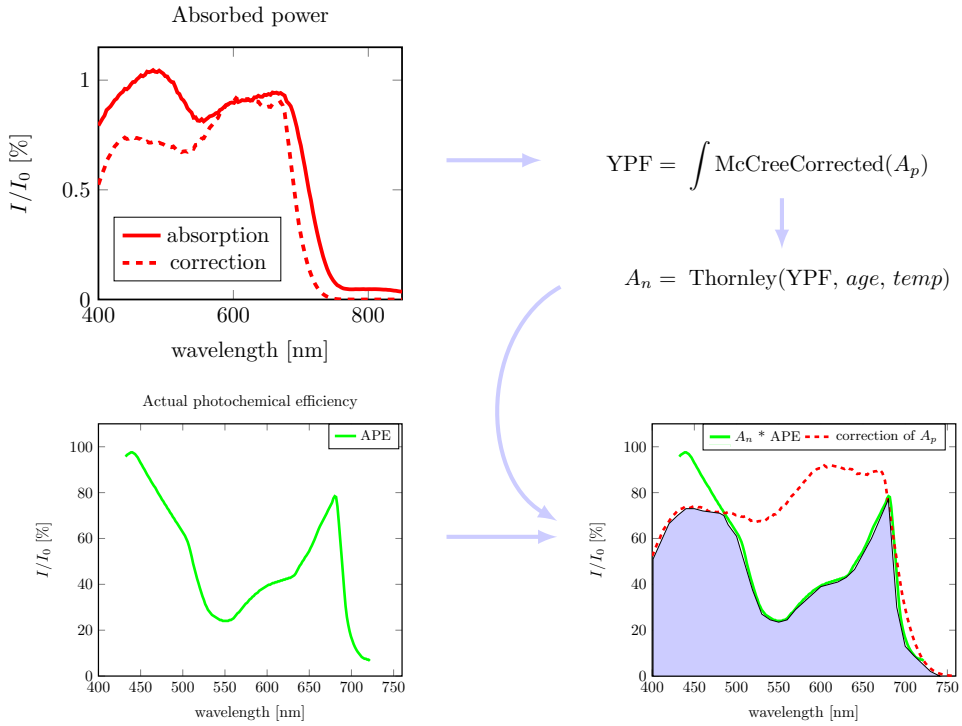


Fig. 19: Schematic overview of the spectral photosynthesis model. The grey-blue area of the lower right chart represents estimated assimilate production.

or germination via transcription factors [9]. On a more practical note, differences and changes in light quality within a canopy can be influential for product quality, notably homogeneity or yield stability, e. g. the red colour of apple fruit which is due to sunlight induced anthocyan formation in the skin of the apple [18]. In crops that grow in height or in different layers the lower layers receive much less light, due to shading by upper structures. The lower parts benefit from a higher percentage of diffuse light, as this allows light rays to penetrate deeper into the canopy than direct light. However, the process of light scattering also goes along with a shift in the spectrum which might have an effect on the lower leaves yet which can not be considered in models that neglect spectral information.

5 ACKNOWLEDGEMENTS

The authors thank Dietger van Antwerpen for the implementation of the GPU raytracer. The work presented in this paper has been partially funded by the "Pathfinder" project "LED – Realizing carbon footprint reduction in food and non-food chain by new productions systems based on LED lighting" in the frame of the

Climate-KIC (Knowledge and Innovation Community). The financial support is gratefully acknowledged.

A LIST OF SYMBOLS

Abbreviation	Description	Unit
APE	actual photochemical efficiency	
AS	absorbed spectrum	
CPU	central processing unit	
FSPM	functional-structural plant model	
GPU	graphics processing unit	
LAD	leaf angle distribution	
LAI	leaf area index	
LED	light-emitting diode	
LM	light model	
LPI	leaf position index	
MC	Monte-Carlo	
MCRT	Monte-Carlo raytracer	
MSE	mean square error	
PAR	photosynthetically active radiation	Wm^{-2}
PLD	physical light distribution	
PPFD	photosynthetic photon flux density	µmol photon $m^{-2}s^{-1}$
PS	photosynthesis	
PSM	photosynthesis model	
RQMC	randomised quasi-Monte Carlo	
RT	raytracing / raytracer	
SPD	spectral power distribution	
YPF	yield photon flux	Wm^{-2}

B CODE

2.1 Light model

Code 1: Setup of the Flux Light Model including the following steps:

```
import de.grogra.gpuflux.tracer.FluxLightModelTracer
    .MeasureMode;
import de.grogra.gpuflux.scene.experiment.
    Measurement;


const int RAYS = 10000000;
const int DEPTH = 10;
const FluxLightModel LM = new FluxLightModel(RAYS,
    DEPTH);


protected void init () {
  LM.setMeasureMode(MeasureMode.FULL_SPECTRUM);
  LM.setSpectralBuckets(81);
  LM.setSpectralDomain(380, 780);
}
```

1. import needed classes
2. initialisation: with 10 million rays and a recursion depth of 10
3. parameterisation: 400 nm divided into 80 buckets of 5 nm

Code 2: Run the light model and determine the amount of sensed radiation or of absorbed power for an object-type. Steps:

```
public void run () [
  {
    LM.compute();
  }

  x:SensorNode ::> {
    Measurement spectrum = LM.
        getSensedIrradianceMeasurement(x);
    float absorbedPower = spectrum.integrate();
    ...
  }

  x:Box ::> {
    Measurement spectrum = LM.
        getAbsorbedPowerMeasurement(x);
    float absorbedPower = spectrum.integrate();
    ...
  }
]
```

1. computation: runs the light model
2. evaluation of sensor objects
3. evaluation of objects of type Box

2-3 and integrate the whole absorbed spectrum

Code 3: Demonstration of the method for the determination of the amount of absorbed power per bucket or integration over a certain spectral range.

```
Measurement spectrum = LM.
    getAbsorbedPowerMeasurement(x);

//absorbed power for the first bucket: 380-385 nm
float ap380_385 = spectrum.data[0];

//accumulate absorbed power for the first four 50
     nm buckets
float b0 = 0, b1 = 0, b2 = 0, b3 = 0;
for(int i:(0:10)) {
   b0 += spectrum.data[i];
   b1 += spectrum.data[i + 10];
   b2 += spectrum.data[i + 20];
   b3 += spectrum.data[i + 30];
}

//integrate the whole spectrum
float ap = spectrum.integrate();
```

1. obtain the absorbed spectrum
2. store the first bucket in a variable
3. build four integrals, each of 50 nm (10 buckets of 5 nm) and sum up the first 40 buckets
4. calculate the integral over the whole spectrum

2.2 Light sources

Code 4: Parameterisable light node - with explicit definition of physical light and spectral power distribution by simple arrays.

```
import de.grogra.imp3d.spectral.
    IrregularSpectralCurve;

const double[][] DISTRIBUTION = {
  {131.25, 131.67, 132.37, ...},
  {131.36, 131.81, ...},
  ...
};

static const float[] WAVELENGTHS = {380,385, ...};
static const float[] AMPLITUDES = {0.000967,
    0.000980, ...};

module MyLamp extends LightNode() {
  {
    setLight(new SpectralLight(
      new IrregularSpectralCurve(WAVELENGTHS,
          AMPLITUDES)).(
        setPower(10), //[W]
        setLight(new PhysicalLight(DISTRIBUTION))));
  }
}
```

1. import of required classes
2. definition of the physical light distribution
3. definition of the spectral power distribution
4. definition of a lamp using the specified parameters

Code 5: Parameterizable light node - Using file references for physical and spectral power distribution.

```
const LightDistributionRef DISTRIBUTION = light("
    distribution1");
const SpectrumRef SPECTRUM = spectrum("equal");

module MyLamp1 extends LightNode {
 {
   setLight(new SpectralLight(new PhysicalLight(
       DISTRIBUTION), SPECTRUM, 10)); // 10 W
 }
}

module MyLamp2 extends LightNode {
  {
    setLight(
      new SpectralLight().(
        setPower(10), //[W]
        setLight(new PhysicalLight(DISTRIBUTION)),
        setSpectrum(SPECTRUM)));
  }
}
```

1. Definition of a file reference. This file can be included or linked to a project.
2. This reference can be applied to a concrete lamp via the constructor,
3. or by using the set-methods of the *lightNode* class.

2.3 Illumination model

Code 6: Definition of a Phong shader by textures, by colours and by a user-defined spectral power distribution.

```
Phong myShader = new Phong();
ImageMap image = new ImageMap();
image.setImageAdapter(new FixedImageAdapter(image(
    "leaf").toImageAdapter().getBufferedImage()));
myShader.setDiffuse(image);


Phong myShader = new Phong();
myShader.setDiffuse(new RGBColor(0,1,0));
//myShader.setSpecular(new Graytone(0.5));
//myShader.setShininess(new Graytone(0.5));
myShader.setDiffuseTransparency(new RGBColor
    (0.5,0,0));
//myShader.setAmbient(new Graytone(0.5));
//myShader.setEmissive(new Graytone(0.5));


 ChannelSPD MySPD = new ChannelSPD(new
     IrregularSpectralCurve(
    new float[] {400,410, ....,740,750} //
        WAVELENGTHS,
    new float[] {0.1,0, .... ,0.4,0.25} //
        AMPLITUDES
  ));
Phong myShader = new Phong();
myShader.setDiffuse(MySPD);
```

1. Set an image as texture: Values for reflection depend on the colour of the texture at each pixel of the image.

2. Set specific properties: At this configuration green will be reflected to 100 % and red will be transmitted to 50 % for the whole surface of the object.

3. A user-defined SPD is used to define the diffuse colour.

C CALCULATIONS

3.1 Conversion: Candela $\Longleftrightarrow$ Watt

Candela (cd) indicates the brightness of a light in a given direction. It is defined as lumen lm per steradian sr. A steradian is the standard unit solid angle in three dimensions. At a given wavelength of 555 nm one candela can be approximated by 1/683 watt. For an arbitrary wavelength λ the luminous intensity $I_v(\lambda)$ can be obtained by Eq. 1:

$$I_v(\lambda) = 683.002 * \bar{y}(\lambda) * I_e(\lambda) \tag{1}$$

where $\bar{y}(\lambda)$ is the luminous intensity function by Sharpe [43], which describes the visual perception of brightness by the human eye (Fig. 20) between 390 and 830 nm.

To estimate the total power over a defined range of wavelengths $[w_{lower}, w_{upper}]$, the luminous intensity function needs to be integrated over the whole spectrum of wavelengths.

$$\int_{\lambda=w_{lower}}^{w_{upper}} I_v(\lambda) \tag{2}$$

with the approximation Eq. 3 using n nm buckets:

$$\sum_{\lambda=w_{lower}}^{w_{upper}} I_v(\lambda),\ \lambda \equiv 0\,(mod\,n) \tag{3}$$

The values for $I_e(\lambda)$ can be obtained directly for each object *obj* by calling the spectral raytracer (getAbsorbedPowerMeasurement(*obj*)).

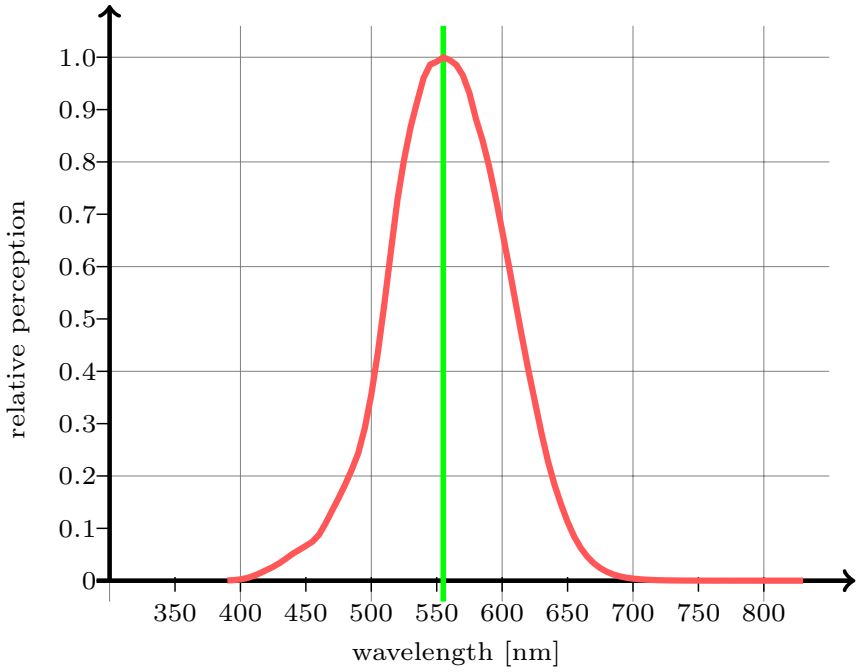


Fig. 20: Luminous intensity function by Sharpe [43], which describes the visual perception of brightness by a human eye. The human eye is most sensitive for green light at a wavelength of 555 nm.

REFERENCES

[1] Alabadí, D., Gil, J., Blázquez, M.A., and García-Martínez, J.L.: Gibberellins repress photomorphogenesis in darkness. Plant Physiology, Vol. 134, 2004, No. 3, pp. 1050–1057.

[2] Angel, E.: Interactive computer graphics - a top-down approach using OpenGL (4. ed.). Addison-Wesley, 2006.

[3] Barnes, C., Tibbitts, T., Sager, J., Deitzer, G., Bubenheim, D., Koerner, G., and Bugbee, B.: Accuracy of quantum sensors measuring yield photon flux and photosynthetic photon flux. HortScience, Vol. 12, 1993, No. 28, pp. 1197–1200.

[4] Born, M., Emil, W.: Principles of Optics. Cambridge: Cambridge University Press. pp. 14–24, 1999.

[5] Brown, C.S., Schuerger, A.C., and Sager, J.C.: Growth and photomorphogenesis of pepper plants under red light-emitting diodes with supplemental blue or far-red lighting. Journal of the American Society for Horticultural Science, Vol. 120, 1995, No. 5, pp. 808–813.

[6] Buck-Sorlin, G.H.: Functional-structural plant modeling. In Dubitzky, W., Wolkenhauer, O., Cho, K.-H., and Yokota, H., editors, Encyclopedia of Systems Biology, pages 778–781, Springer New York, 2013.

[7] Buck-Sorlin, G.H., de Visser, P.H.B., Henke, M., Sarlikioti, V., van der Heijden, G.W.A.M., Marcelis, L.F.M., and Vos, J.: Towards a functional-structural plant model of cut-rose: simulation of light environment, light absorption, photosynthesis and interference with the plant structure Annals of Botany, Vol. 108, 2011, No. 6, pp. 1121–1134.

[8] Buck-Sorlin, G.H., Hemmerling, R., Vos, J., and de Visser, P.H.B.: Modelling of Spatial Light Distribution in the Greenhouse: Description of the Model. In Li, B. et al., editor, Proceedings of the third International Symposium on Plant Growth Modeling, Simulation, Visualization and Applications [PMA09], 2009, pp. 79–86, Beijing (China), Nov. 2009. IEEE, Los Alamitos 2010.

[9] Casal, J.J., Candia, A.N., and Sellaro, R.: Light perception and signalling by phytochrome a. Journal of Experimental Botany, Vol. 65, 2014, pp. 2835–2845.

[10] Chelle, M., Hanan, J.S., and Autret, H.: Lighting virtual crops: the CARIBU solution for open L-systems. In Godin, C. et al., editor, Proceedings of the Fourth International Workshop on Functional-Structural Plant Models [FSPM04], 2004, pp. 194–194, Montpellier (France), 07-11 June 2004. UMR AMAP. Montpellier.

[11] Chelle, M., and Andrieu B.: The nested radiosity model for the distribution of light within plant canopies. Ecological Modelling, Vol. 111, 1998, No. 1, pp. 75–91.

[12] Cieslak, M., Lemieux, C., Hanan, J.S., and Prusinkiewicz, P.: Quasi-Monte Carlo simulation of the light environment of plants. Functional Plant Biology, Vol. 35, 2008, No. 10, pp. 837–849.

[13] Cournède, P.-H., Mathieu, A., Houllier, F., Barthélémy, D., and de Reffye, P.: Computing competition for light in the GreenLab model of plant growth: A contribution to the study of the effects of density on resource acquisition and architectural development. Annals of Botany, Vol. 101, 2008, No. 8, pp. 1207–1219.

[14] Dauzat, J.: Radiative transfer simulation on computer models of Elaeis guineensis. Oléagineux (Paris), Vol. 49, 1994, No. 3, pp. 81–90.

[15] Dauzat, J., Franck, N., Rapidel, B., Luquet, D., and Vaast, P.: Simulation of ecophysiological processes on 3d virtual stands with the archimed simulation platform. In Plant Growth Modeling and Applications, 2006. PMA´06. Second International Symposium on, pp. 101–108, Nov. 2006.

[16] Dauzat, J., Clouvel, P., Luquet, D., and Martin, P.: Using virtual plants to analyse the light-foraging efficiency of a low-density cotton crop. Annals of Botany, Vol. 101, 2008, No. 8, pp. 1153–1166.

[17] Disney, M.I., Lewis, P., and North, P.R.J.: Monte Carlo ray tracing in optical canopy reflectance modelling. Remote Sensing Reviews, Vol. 18, 2000, No. 2-4, pp. 163–196.

[18] Feng, F.J., Li, M.J., and Ma, F.W.: The effects of bagging and debagging on external fruit quality, metabolites, and the expression of anthocyanin biosynthetic genes in 'jonagold' apple (Malus domestica Borkh.). Scientia Horticulturae, Vol. 165, 2014, pp. 123–131.

[19] Furuya, M., and Schäfer, E.: Photoperception and signalling of induction reactions by different phytochromes. Trends in Plant Science, Vol. 1, 1996, No. 9, pp. 301–307.

[20] Goins, G.D., Yorio, N.C., Sanwo, M.M., and Brown, C.S.: Photomorphogenesis, photosynthesis, and seed yield of wheat plants grown under red light-emitting diodes (LEDs) with and without supplemental blue lighting. Journal of Experimental Botany, Vol. 48, 1997, No. 7, pp. 1407–1413.

[21] Greene, N.: Voxel space automata: Modeling with stochastic growth processes in voxel space. In Proceedings of SIGGRAPH ´89 (Boston, Mass., July 31–August 4, 1989), in Computer Graphics 23, pages 175–184. ACM SIGGRAPH, New York, Aug. 1989.

[22] GroIMP Developer Group, Web site. Available on: `http://www.grogra.de/`, 2015.

[23] Hemmerling, R., Kniemeyer, O., Lanwert, D., Kurth, W., and Buck-Sorlin, G.H.: The rule-based language XL and the modelling environment GroIMP illustrated with simulated tree competition. Functional Plant Biology, Vol. 35, 2008, pp. 739–750.

[24] Illuminating Engineering Society, ANSI/IESNA LM-63-02. IES-specification, standard file format for the electronic transfer of photometric data, available on: `http://www.cn-hopu.com/upload/file/IES.pdf`, 2002.

[25] Kanamaru, N., Chiba, N., Takahashi, K., and Saito, N.: Cg simulation of natural shapes of botanical trees based on heliotropism. The Transactions of the Institute of Electronics, Information, and Communication Engineers J75-D-II, pages 76–85, 1992. In Japanese.

[26] Kasperbauer, M.J.: Far-red light reflection from green leaves and effects on phytochrome-mediated assimilate partitioning under field conditions. Plant Physiology, Vol. 85, 1987, No. 2, pp. 350–354.

[27] Karp, G.: Cell and Molecular Biology: Concepts and Experiments (7. ed.). John Wiley & Sons, 2013.

[28] Khronos Group - OpenCL, Web site. Available on: `https://www.khronos.org/opencl/`, 2015.

[29] Kniemeyer, O.: Rule-based modelling with the XL/GroIMP software. In Schaub, H., Detje, F., and Brüggemann, U., editors, The Logic of Artificial Life. Proceedings of 6th GWAL, pages 56–65. AKA Akademische Verlagsges Berlin, 2004.

[30] Kniemeyer, O.: Design and implementation of a graph grammar based language for functional-structural plant modelling. PhD thesis, BTU Cottbus, available on: `http://opus.kobv.de/btu/volltexte/2009/593/`, 2008.

[31] Kurth, W., Buck-Sorlin, G.H., and Kniemeyer, O.: Relationale Wachstumsgrammatiken: Ein Formalismus zur Spezifikation multiskalierter Struktur-Funktions-Modelle von Pflanzen. In: Modellierung pflanzlicher Systeme aus historischer und aktueller Sicht. Symposium zu Ehren von Prof. Dr. Dr. h.c. Eilhard Alfred Mitscherlich,

Schriftenreihe des Landesamtes fr Verbraucherschutz, Landwirtschaft und Flurneuordnung Brandenburg, Reihe Landwirtschaft, Band 7, pages 36–45, 2006. In German.

[32] Kurth, W.: Specification of morphological models with L-systems and relational growth grammars. Computational Visualistics and Picture Morphology (Themenheft zu IMAGE 5), Vol. 5, 2007, No. 1, pp. 50–79.

[33] Le, Y.G.: Schema Validation with Intel Streaming SIMD Extensions 4 (Intel SSE4), white paper, Dec. 16, 2008.

[34] Leroy, C., Sabatier, S., Wahyuni, N., Barczi, J.-F., Dauzat, J., Laurans, M., and Auclair, D.: Virtual trees and light capture: A method for optimizing agroforestry stand design. Agroforestry Systems, Vol. 77, 2009, No. 1, pp. 37–47.

[35] Massonnet, C.: Functional and architectural variability of vegetative system in apple tree (Malus domestica): Comparison between four cultivars by a functional-structural modelling approach. Thesis, Ecole Nationale Superieure Agronomique de Montpellier - AGRO M, Dec 2004. Available on: `https://tel.archives-ouvertes.fr/tel-00010748`.

[36] McCree, K.J.: The action spectrum, absorbance and quantum yield of photosynthesis in crop plants. Agricultural Meteorology, Vol. 9, 1971/72, pp. 191–216.

[37] Měch, R.: Modeling and simulation of the interaction of plants with the environment using L-systems and their extensions. PhD thesis, University of Calgary, Calgary, Alta., Canada, 1998.

[38] Měch, R., and Prusinkiewicz, P.: Visual models of plants interacting with their environment. In Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH´96, pages 397–410, New York, USA, 1996.

[39] Owens, J.D., Houston, M., Luebke, D., Green, S., Stone, J.E., and Phillips, J.C.: GPU computing. Proceedings of the IEEE, Vol. 96, 2008, No. 5, pp. 879–899.

[40] Perttunen, J., Sievänen, R., and Nikinmaa, E.: LIGNUM: A model combining the structure and the functioning of trees. Ecological Modelling, Vol. 108, 1998, No. 1–3, pp. 189–198.

[41] Phong, B.T.: Illumination for computer generated pictures. Commun. ACM, Vol. 18, 1975, No. 6, pp. 311–317.

[42] Pradal, C., Dufour-Kowalski, C., Boudon, F., Fournier, C., and Godin, C.: OpenAlea: A visual programming and component-based software platform for plant modelling. Functional Plant Biology, Vol. 35, 2008, No. 9/10, pp. 751–760.

[43] Sharpe, L.T., Stockman, A., Jagla, W., and Jägle, H.: A luminous efficiency function, $v*(\lambda)$, for daylight adaptation. Journal of Vision, Vol. 5, 2005, No. 11, pp. 948–968.

[44] Sievänen, R., Perttunen, J., Nikinmaa, E., and Kaitaniemi, P.: Toward extension of a single tree functional-structural model of Scots pine to stand level: effect of the canopy of randomly distributed, identical trees on development of tree structure. Functional Plant Biology, Vol. 35, No. 10, 2008, pp. 964–975.

[45] Taugourdeau, 0., Dauzat, J., Griffon, S., de Coligny, F., Sabatier, S., Caraglio, Y., and Barthélémy, D.: Retrospective analysis of fir sapling growth vs. light interception. In Theodore DeJong and David Da Silva, editors, Proceedings

of the 6th International Workshop on Functional-Structural Plant Models [FSPM10], 2010, pages 93–95, University of California Davis (USA), 12-17 Nov 2010. Etats-Unis.

[46] Thornley. J. H. M.: A model to describe the partitioning of photosynthate during vegetative plant growth. Annals of Botany, Vol. 33, 1969, pp. 419–430.

[47] van Antwerpen, D.G.: Unbiased physically based rendering on the GPU, PhD thesis, TU Delft, 2011, available on: `http://graphics.tudelft.nl/ dietger/thesis/index.htm`.

[48] Vos, J., Evers, J.B., Buck-Sorlin, G.H., Andrieu, B., Chelle, M., and de Visser, P.H.B.: Functionalstructural plant modelling: A new versatile tool in crop science. Journal of Experimental Botany, Vol. 61, 2010, No. 8, pp. 2101–2115.

[49] Wang, H.Y., Kang, M.Z., and Hua, J: Simulating plant plasticity under light environment: A source-sink approach. In et al. editors, Proceedings of the Fourth International Symposium on Plant Growth Modeling, Simulation, Visualization and Applications [PMA12], pp. 431–438, Shanghai (China), 31 Oct - 03 Nov 2012. IEEE.

Michael Henke Diploma degree in computer science from the Brandenburg University of Technology Cottbus — 2007 PhD student at the University of Göttingen, Department Ecoinformatics, Biometrics and Forest Growth — 2009 - 2010 Visiting scholar at Zhejiang University, Hangzhou, China — 2013 Researcher at French National Institute for Agricultural Research — 2014 - 2016 Researcher at Wageningen UR, The Netherlands — assistant lecturer at Brandenburg University of Technology Cottbus and University of Göttingen — GroIMP developer — research fields: functional-structural plant modelling, simulation.

Gerhard H. Buck-Sorlin Diploma degree in biology at the University of Göttingen and PhD in biology at the University of Wales in Bangor, UK (1997), subsequently junior postdoctoral scientist at the Institute of Plant Genetics and Crop Plant Research in Gatersleben, Germany — 2002 - 2007 senior scientist and lecturer at the University of Technology in Cottbus — 2005 - 2009 guest professor at the Zhejiang University in Hangzhou — 2007 - 2011 senior scientist and lecturer at Wageningen University and Research Centre — since 2011 Professor for fruit tree culture and modelling at Agrocampus Ouest, Centre d'Angers, France — research fields: ecophysiology of crop plants, functional-structural plant modelling.

COMPUTING AND INFORMATICS

Previously, COMPUTERS AND ARTIFICIAL INTELLIGENCE

EDITORIAL OFFICE
INSTITUTE OF INFORMATICS, SLOVAK ACADEMY OF SCIENCES
Dúbravská cesta 9, 845 07 Bratislava, Slovakia
Tel.: (+4212) 59411204 Fax: (+4212) 54771004 e-mail: cai.ui@savba.sk

Michael HENKE
UMR1345 Institut de Recherche
en Horticulture et Semences (IRHS)
AGROCAMPUS OUEST Centre d'Angers
INRA - Université d'Angers
France

Date: December 4, 2015

This is to acknowledge that the paper **"Using a Full Spectral Raytracer for Calculating Light Microclimate in Functional-Structural Plant Modelling"** by Michael Henke and Gerhard H. Buck-Sorlin has been accepted for publication in Computing and Informatics and will appear in a subsequent issue of the 2016 volume of the journal.

Sincerely Yours,

Assoc. Prof. Ladislav Hluchý, PhD.
Editor-in-Chief

CHAPTER 9

Discussion and Outlook

The papers presented in the previous chapters investigated and discussed most of the project activities of the common modelling process as described in Sec. 2.3. In the first paper (Henke *et al.* (2014a), Chapter 3), a simple and efficient non-destructive photometric methodology was developed and demonstrated, covering all steps from field data acquisition to binarisation and allowing for leaf contour modelling. While there are plenty of papers published dealing with one or two aspects of the modelling process only a minority of them cover the whole process from the beginning to the integration of the developed sub-models into an existing FSPM. For dynamic FSPMs it is essential to have a method to observe and measure the whole growth process of the investigated plant non-destructively. Studies on leaf area date back to the beginning of the 20th century (Gregory 1921). Measurements were done manually (Gallagher 1979) for winter wheat and spring barley, and even recently, Chen *et al.* (2009) measured leaf lengths for over 1700 hours with a ruler, which is often the common way of obtaining data. Even when technical help was used for digitisation, the image segmentation was still done manually, e.g., Neto *et al.* (2006) extracted 510 leaves in this way. In order to obtain statistically meaningful results, a large number of measurements have to be done, which requires large human and financial resources. Most of these methods are destructive; they propose cutting the leaves and

using a flatbed scanner for image acquisition, which can be a major limitation for the use in several FSPMs. This in itself shows the importance of new methods for data acquisition.

A widespread non-destructive method for data acquisition are electromagnetic digitisers (e.g., Fastrak, Polhemus, USA). This method, despite its wide distribution, however, also has its restrictions. First, it is difficult to avoid falsifying the measurements by, e.g., touching the leaves while still aiming to be as close as possible to the point to be measured in order to obtain correct data, and to avoid wind and breathing effects. Second, several points need to be recorded and repeatedly measured for each leaf (the latter if a study of morphogenetic dynamics is desired). Thus, each point needs to be repeatedly touched, which requires the application of permanent or at least durable marks on the leaf, the latter potentially having an altering influence on leaf development. Apart from that, a thigmomorphogenesis (stunted growth due to touching) effect might ensue from repeated probing by the stylus and the hand of the digitising person. However, with this method complex 3-d structures can be measured, as shown by Wiechers *et al.* (2011) who measured cucumber leaves.

The equipment presented in Henke *et al.* (2014a) (Chapter 3) is well suitable for field use and applicable to flat leaves of most species without modification. It is reaching its limits, however, when the investigated objects become too small or can not be flattened out without destroying or damaging them. A stratified process of image processing was used to extract the contours of the digitised leaves. The image processing tool developed for this purpose, implemented as ImageJ macro, is a semi-automatic tool for contour extraction of leaves of a wide range of broad-leaved plants. Several models were developed based on the data that were collected during one growth cycle. These models were adapted to black poplar and accordingly fitted, but they can also be used for other species with similar leaf shapes, with only a few small changes. Model comparison and validation showed that leaf shape could be well modelled with a small set of parameters. Finally, the models were implemented as GroIMP organ modules that can be used in different models as a component. Providing such modularised systems of predefined leaf organs represents a first step toward a more user-friendly modelling workflow by ridding the modeller from low-level programming work. It thus allows putting the focus on modelling rather than on coding.

The second paper (Henke and Sloboda (2014), Chapter 4) deals also with data acquisition. Here a method to extract tree rings completely from images of tree discs independent of their source was developed. The method uses active contours, often used in medical image processing to detect organs, in combination with an optimised image filter based on the Sobel operator to

automatically outline the tree rings. Special attention was given to the elimination of critical physical irregularities caused by branches, cracks or colourisations.

Tree rings can tell us a lot about climate variations over the last several thousand years. In dendrochronology, tree rings are used to determine the age of a piece of wood by comparison of the specific growth pattern with a pattern of known age. In most cases and for most purposes measuring the radii of the annual rings along single rays, e.g., obtained by a core sample is sufficient. To extract and analyse such samples several software systems have been developed. One commonly used, commercial representative is WinDENDRO (Guay 2012).

However, if one is interested in precise statements about stem growth, e.g., to explore how a predominating wind direction affects the stem or how a slope situation is balanced by asymmetric growth, a complete extraction of all tree rings can hardly be avoided. Therefore, digitising and processing of whole tree discs is required. Image analysis techniques have been applied already in several systems like TRESS (Conner 1999; Gopalan 2000). A watershed-based transformation in combination with other morphological operations for measuring the areas of annual tree rings was introduced by Soille and Misson (2001). Zhou *et al.* (2012) presented another method based on watershed segmentation to detect and count tree rings. Norell (2011) used image filter based methods to analyse wood quality by counting the number of annual rings.

With our system, we provide a tool for semiautomatic tree ring segmentation of whole tree rings of a stem disc. The key criterion to conduct successful, accurate and reliable measurements of whole tree rings with this system is mainly the quality of input images. The quality of extraction is proportional to the visibility of each ring within the image. Consequently, species in which early- and latewood are clearly distinguishable are preferred. For our purpose, we can conclude that the technique of active contours produces reproducible and reasonable results. The system has been applied to several conifer species.

In Henke and Sloboda (2014), an extended Sobel edge detector (Sobel and Feldman 1968), optimised for the detection of tree rings, was developed. The basic idea of our filter optimisation is to consider the angles of the detected edges and to emphasise only those edges which are orthogonal to a ray through the marrow of the tree disc. This condition is based on the property that tree rings expand in a more or less circular way around the pith and thus have tangents orthogonal to a ray through the centre. In his master thesis, Conner (1999) used a modified

Canny edge detector (Canny 1986) where tree ring orientation within a region of interest is used to suppress inter-ring connections.

The reliability of our system was evaluated by comparison with manual measurements performed on 324 measuring points. The overall accuracy of measurement of our system might be low compared with touchstones applied in dendrochronology. While systems used in dendrochronology normally use light microscopes and consider only radial ring-width measurements, our system is designed to extract whole tree rings, which can compensate the errors to a certain extent. For the deduction of growth behaviour, the difference has no significant consequences.

Our system reaches its limits when the contrast between early- and latewood is too low, and as a consequence, no edge can be detected. The same applies when a gradient instead of a sharp switch in colour is given. Very thin rings, low ring width and large image noise, can lead to more user interaction and thereby increase the time for extraction.

In Henke *et al.* (2017) (Chapter 5) two well-known approaches for modelling virtual vegetation, the grammar-based methods used in L-systems and the Xfrog (Sec. A.17) method (modelling with instantiation), are unified in the framework of RGG and implemented in GroIMP. The possibility and the synergistic benefits of the combination of both methods were demonstrated at the example of simple plant models.

Modelling with instantiation components is a powerful technique to obtain quick results. Although this modelling technique has no botanical basis as it does not capture the growth process, the results are visually satisfying. It is hard to translate data taken from nature into the parameters of the component. A relation between reality and model can only subsequently be produced by measurements at the archetype and comparison with the model. This rarely yields botanically correct models. The degree of realism lies completely in the responsibility of the modeller. The generated graphical results are nevertheless more than only pretty pictures. With them, new possibilities arise, for example, in the visualisation of ecological data. In landscape planning, the results of interventions in ecological systems can be represented using the Arrange component. Using the combination of instantiation with generative relational growth grammars, some of the restrictions of pure instantiation-based modelling can be compensated. It is now possible to model significant causal aspects of processes of growth, their control being realised by a botanically-tested growth grammar.

The Xfrog multiplier nodes are another concept for object instantiation. The Xfrog approach enables the interactive editing of a graph made up of component prototypes, the so-called p-graph (Deussen 2003). Among the nodes of this graph are not only shape nodes representing graphical primitives, but also various sorts of multipliers which have the semantics of copying the structures encoded by their descendant nodes and placing them in specified positions. The interactive access to the p-graph requires a medium level of abstraction and allows a quick feedback from the resulting rendered model to the editing process, thus enabling quite intuitive working. The portfolio of components (node types), however, is restricted, and there is no natural way to simulate processes of growth and development in this framework – let alone to include biologically-inspired process-based models (e.g., of plant hormonal effects controlling flowering), which is relatively easy in L-systems. However, the Xfrog approach demonstrates an interesting technique for the generation of structural plant models by providing a new method for object instantiation.

In Henke *et al.* (2016) (Chapter 6) we developed a more efficient method for model development in order to have an answer to the increasing demand for FSPM in crop and tree production. This includes the consideration of model reuse (by modularisation), combination and comparison, and the enhancement of existing models. The introduced FSPM-Prototype can be seen as a first step towards establishing an efficient and general, i.e. not species-specific FSPM with standardised modules, processes and communication structure, which enables a clear model design, is easy to parameterise, understand and extend. The model structure is hierarchical and object-oriented, with plant organs being the base-level objects and plant individual and canopy the higher-level objects. Modules for the majority of physiological processes are incorporated, more than in other platforms that have a similar aim (e.g., photosynthesis, organ formation and growth). In Sec. 9.1 and Sec. 9.2 the advantages and importance of modularisation and prototyping are discussed in detail.

Simulation runs with several general parameter sets adopted from the literature show that the present prototype was able to reproduce a plausible output range for different crops (rapeseed, barley, etc.) in terms of both the dynamics and final values (at harvest time) of model state variables such as assimilate production, organ biomass, leaf area and architecture. To monitor and document the dynamics of growth and developmental processes, a variety of charts have been implemented, e.g., dynamics of organ dry weight and length. Even without a proper parameterisation for a certain crop species, the model already exhibited general patterns similar to those found in plants, with respect to the phenology of growth stages or stem extension dynamics. This systematic approach provides all the necessary infrastructure and documenta-

tion to develop efficient FSPMs based on their measurements for different target groups (with or without knowledge of programming or modelling) and could also be useful for professional FSPM developers as a basic framework.

In Henke *et al.* (2014b) (Chapter 7) the FSPM-P was used as base model to implement a root/-soil model. Root plasticity is a key process affecting the root system foraging capacity while itself being affected by the nutrient availability around the root environment. Root system architecture is determined by three types of plastic responses: chemotropism, spacing of lateral roots, and hierarchy between laterals and their mother root. With this model, we attempt a systematic comparison of the effect of each mechanism on the whole root plasticity when the root is grown under four distinct nutrient distribution scenarios using a functional-structural root model. Nutrient distributions included i) a completely random distribution, ii) a layered distribution, iii) a patch distribution, and iv) a gradient distribution. To our knowledge, the incorporation of all these mechanisms in one root model in combination with a spatially explicit representation of the nutrient distribution has not been previously attempted.

The model presented in this paper could be potentially relevant to address questions in domains like plant ecology or crop breeding: Representing basic mechanisms of root plasticity in response to nutrient resource availability, the present model or its extensions could be useful for investigating mechanisms of plant competition in monocultures or intercropping systems. 3-d model-based design of ideotypes with respect to certain relevant traits of the aerial architecture of a crop (e.g., light interception, stability), has recently been proposed as a valuable new method in horticultural crop breeding (e.g., Sarlikioti *et al.* (2011)). Finding an optimal root architecture with respect to minimisation of nitrogen loss or maximisation of N uptake in a given layered soil might be worthwhile applications of our model.

With the model presented in this paper, we have demonstrated the potential of the FSPM-P to be extended and applied to a root model in combination with a new soil model. For an extended list of models based on the FSPM-P see Tab. 9.2.

With the sixth and last paper of this thesis, Henke and Buck-Sorlin (accepted) (Chapter 8), a method for accurate computation of the light budget and the light microclimate was introduced. Such an extended method is a fundamental part of every light interception model, not only at the level of the single plant individual but also, or even more so, at the canopy level. Light being the single-most important parameter for photosynthesis, it is varying strongly in structurally heterogeneous canopies, especially during clear days. A number of common raytracers already

allow the spatially explicit calculation of the fate of light beams in a 3-d scene and thus offer significant advantages over models that are based on horizontal layers and in which the vertical light distribution is described by a Beer-Lambert law. Using these raytracers made it possible to consider shading (including self-shading), reflected and transmitted light. New findings on the effects of light quality on cellular and growth processes recently created the interest to extend this modelling paradigm, allowing the representation of detailed spectra instead of monochromatic or white light and to extend existing FSPM platforms accordingly.

In this study, we demonstrated how at present commonly available computer hardware can be used to drastically reduce computation time for calculation of light fate in a 3-d scene using inverted Monte Carlo raytracing. Furthermore, not only performance can be increased with this implementation but also the quality of light computation can be improved by including information about the light spectrum. The simulation of natural light, including spectral information, opens up new possibilities for application in the domain of FSPM, such as realistic simulations of additional light sources, e.g., in greenhouses, climate chambers, or indoor farming, the latter in the context of urban horticulture, which is already playing an important role for local food production in some urban areas.

Furthermore, the present approach renders possible the modelling of an arbitrary diversity of specific light sources, e.g., High Pressure Sodium, Mercury Vapor, and LED lamps, in combination with commercially available reflectors and panels, e.g., double-ended wing, or raptor. The effect of certain wavebands, in general light quality, on physiological processes and the combination and concentrations of nutrients, vitamins and antioxidants are another domain in which the present model could be used: processes affecting plant growth or fruit quality traits are often affected by the strength of certain light signals; the most common example is the ratio of the red to far-red waveband which is instantly translated in the plant into a ratio of phytochromes. On a more practical note, differences and changes in light quality within a canopy can be influential for product quality, notably homogeneity or yield stability, e.g., the red colour of apple fruit which is due to sunlight induced anthocyan formation in the skin of the apple (Feng *et al.* 2014). In crops that grow in height or in different layers the lower layers receive much less light, due to shading by upper structures. The lower parts benefit from a higher percentage of diffuse light, as this allows light rays to penetrate deeper into the canopy than direct light. However, the process of light scattering also goes along with a shift in the spectrum which might have an effect on the lower leaves yet which can not be considered in models that neglect spectral information.

9.1. Contribution of Modularisation

In biology, modularity can be described as the concept that organisms or metabolic pathways are composed of modules. Similar to this definition, in software design, modularity refers to a logical partitioning of the 'software design'. Basically, this follows the strategy of solving problems by the process of 'divide and conquer', which means breaking down a problem into sub-problems until these become simple enough to be solved (conquered) directly. This type of modularity originally applied to a special type of recursive algorithm but can be upscaled to software systems. A model, if it is algorithmically implemented in a programming language, is nothing else than a piece of software. This concept allows complex software to become manageable for the purpose of implementation, reuse and maintenance - the main aspects of successful software engineering. The logic of partitioning may differ according to the logical order of functions, implementation considerations, data links, or other criteria. The degree to which a system is broken down into smaller parts (e.g., components and modules) is called granularity, i.e. level of decomposition. In software engineering, a component is a reusable software element with a specification used for the composition of a model, while a module is a closed, functional, independent, interchangeable unit of software. This includes reoccurring calculations or data processing as required in the implementation of processes (e.g., growth functions and the calculation of photosynthesis rates) within a component.

With the FSPM-P (Chapter 6), we made a first step towards a general FSPM prototype model that follows several of the common concepts used in software engineering. The model is subdivided into several files representing the different aspects of the model: a main file for model initiation and control; a file for defining objects (such as plant organs) and their properties; a library of photosynthesis rate models to be coupled with plant organs; global parameter definition; a file containing auxiliary tools and functions like charts. For the definition of plant organs, a strictly object-oriented approach was used. The hierarchical scale at which the model is implemented is the same as that of the organ, but processes can also be aggregated at the plant individual or some intermediate scales (shoot, branch). Plant-related processes and other functionality needed for the overall infrastructure of a model (e.g., model initiation, in- and output routines) are encapsulated in single modules.

Adhering to a rigorous modularisation of the model code itself has a great impact on a large number of modelling aspects (Tab. 9.1). A clear modularisation, for example, makes it easier for a model user (i.e. a person not identical with the model developer) to understand how the

model works and how the contained sub-models interact with each other. As a consequence, this makes it more straightforward to fix bugs, to extend or adapt the model to new tasks. In software engineering, these aspects are referred to as software reuse and maintenance.

Table 9.1.: Summary of advantages of modularisation in plant modelling.

- acceleration of model development workflow
- models easier to understand
- models become comparable (since they use the same standardised modules)
- enhance flexibility in FSPM: faster to adapt, to change, and to extend
- modelling and implementation tasks more closely combined
- easier model validation
- enhanced model quality
- improved chance of a model being applied as decision-support tool
- easier model maintenance
- facilitated reuse of models and sub-models
- good modelling practices
- facilitated selection of an appropriate model structure → software design patterns
- easier incorporation of new sub-models → 'plug-and-play'
- facilitated interfacing with existing models (often treated as black boxes)

A derived benefit of modularisation is that it typically pays off already very soon after the implementation of the first few models. It generates a base of reusable - predefined - models ready to be re-assembled to furnish the base for new models. This point already has great potential and provides benefit with respect to several aspects, e.g., reducing time spent on model development from scratch, facilitating validation, extension and maintenance of models - just to mention a few of them. It also leads to an enhanced model quality, since the re-used modules are already validated and have been used in previous models. Furthermore, this helps to make models more comparable. This topic is closely related to the recurrently expressed demand for a 'library of FSPM modules' that is raised at nearly every one of the past FSPM-related meetings and conferences.

Transferring some of the knowledge in current software engineering techniques to the modelling process of FSPM means to get all the benefits made in software engineering over several decades, and can therefore not be overrated with respect to its beneficial impact.

The analysis of the designs of these modularised models will indicate reoccurring parts, e.g., some parts responsible for data in- and output, model initiation, a main function containing the main (growth and development) loop, and parts for the plant definitions. This results lead us to develop and implement a general prototype for plant modelling, as described in Chapter 6, and in the following sub-section. This is closely related to the topic of 'design-pattern' in software engineering. Design-patterns are implementation schemata for common, reoccurring problems (Gamma *et al.* 1995). This directly leads to a set of appropriate model structures (i.e. a combination of modules for standard problems) that helps to make the model structure more clear and therefore traceable.

As described above, modularisation can help to make models comparable and also to combine models. However, this only holds true as long as the involved models have been implemented in the same, or at least incompatible, programming languages. As soon as different modelling/programming languages are involved things become complicated. Approaches using wrapper code often generate a noticeable overhead in the model code, with all the ensuing new problems. This intermediate step of translation by the wrapper, e.g., data conversion, typically also leads to a significant increase in computation time.

A related approach, OpenAlea (Pradal *et al.* (2008), Sec. A.13), is a distributed collaborative effort to develop Python libraries and tools that address the needs of current and future works in plant architecture modelling. OpenAlea includes modules to analyse, visualise and model the functioning and growth of plant architecture. However, while OpenAlea essentially links different programmes (possibly written in different languages and exhibiting different compilation states: runtime libraries, source code, ...), our approach is a core FSPM that runs a priori, and in which the main functional elements (light interception, photosynthesis, ...) are already implemented in the same programming environment and language (GroIMP and XL).

The GreenLab approach (Hu *et al.* (2003), Sec. A.7) is comparable to the present model as it provides a fully runnable model that can be parameterised for different species. However, its source function being based on radiation use efficiency and lacking internal transport, it falls short of the generality which we consider necessary for an extensible FSPM. In this respect, it is closer related to CANON (McMaster and Hargreaves 2009), in which a composite design pattern was implemented at the phytomer (phyton) level for use in a FSPM.

9.2. Importance of a Prototype Approach

The growing recognition of the FSPM approach as a logical continuation of the crop modelling tradition (Vos *et al.* 2010) (see also the other articles in that special issue on FSPM) necessitates the provision of possibilities for efficient model development as well as for maintenance, support and enhancement. An FSPM, like any other computer programme, can draw substantial benefit and advantage from computer science techniques, mainly software engineering, e.g., object-oriented programming, modularisation, design patterns, software re-usability and basic programming standards (cf. Wilson (2006)). This can enhance both the models themselves and the development process, turning it more structured, efficient, and clearer.

By applying such good practices, models will become easier to understand and better comparable, sub-models can be replaced more easily. Development, combination, implementation, calibration and validation of models should equally benefit. The establishment of best practices in FSPM is a solution for recurring problems and would rationalise work and enhance productivity as it reduces the time for coding, testing and documentation. A predefined, consistent solution like a prototype can also provide standards for testing of parts or the whole model.

Finally, a prototype like the one presented here will facilitate communication between modeller, programmer and experimentator, which can be mutually beneficial and help to establish FSPM as a tool for research, development and education in the plant and crop sciences.

9.3. Need for Standards for FSPM

The large diversity of approaches developed for FSPM have led to an even larger number of languages and software systems (see Sec. 2.2 and Sec. A). This circumstance, of course, causes some problems and restrictions as soon as the modelling done by individual persons attains a level of exchange and cooperation, as is the case in, e.g., interdisciplinary projects or international cooperations. To illustrate these, the following problems are given.

Different models use various **base units** for their plant (models). Depending on the level of abstraction the following base units are frequently used: compartment, metamer, growth unit, organ. While a conversion from a higher physical resolution to a low resolution should be

possible in most cases, a conversion in the opposite direction, which involves adding resolution, is only possible under additional assumptions.

Different definitions of **coordinate systems** lead to different positions in the 3-d space. While in a global coordinate system coordinates are relative to the origin of the coordinate system, in a local coordinate system coordinates are relative to the location of the previous object. The conversion between both coordinate systems is a typical source of errors. For example, L-systems usually use a local coordinate system. The generated 3-d structure is the result of rule application to an initial axiom. Between neighbouring organs, transformations like rotations can take place. This affects the position and orientation of the whole sub-structure originating from these objects (organs).

Different **import and export formats** for structures and data is another critical point. Common problems for structures are the already mentioned differences of the used base unit and the underlying coordinate systems. Furthermore, the data structure used to describe the 3-d structure, e.g., lists, graphs, or tables, can cause problems during conversion. For data exchange problems related to the different definitions of the structures are typical. Therefore, data exchange and model cooperation often turn out to become very difficult and time-consuming bottlenecks. Often plausible dummy values are needed during data transformation to replace missing values.

Differences in the **implementations**, e.g., of sub-functions like the calculation of photosynthesis or light calculation on a higher level of modelling, can occur. While some models approximate light distribution using the Beer-Lambert law other models use sophisticated ray-tracing techniques to calculate exact light distributions at the level of the single organ. Using fundamentally different techniques will naturally lead to a divergence in results. Concerning the implementation of sub-functions other problems can be observed. While different models claim to use the same algorithm based on the same paper, the actual (re-)implementations typically differ in details.

Different **programming languages and platforms** commonly cause incompatibilities between models. A special (problematic) case are 'standalone' implementations, e.g., in C++, that can be handled only as 'black box' without any information about what is going on inside. The direct communication between models implemented in different programming languages, if possible at all, requires cumbersome wrapper code. A typical workaround are external (text)

files written by one model and read in by the second model to become processed before written again to an output file that can, in turn, be imported by the first model.

If two models differ in one or more of these points, model comparison becomes rather complicated if not impossible. To overcome some of the described problems, it would already be helpful to introduce basic standards for data exchange and commonly used libraries providing elementary functions.

To facilitate communication between models, the definition of uniform interfaces is required. This would not only enable data exchange but also allow model comparison.

By introducing and using a community-based collection of sub-models, e.g., in the form of a library, one would make sure that the same implementations of basic functions are used within each model. Freely accessible databases of input values and verified results for common base functions are needed to develop and test models for more reliable, robust simulations. For instance, a set of measured radiation data together with corresponding light response data of a leaf could be used to calibrate different photosynthesis models.

Free access to all parts of a model, including the used database, the model code and the documentation is essential for scientific work, and in case of FSPM, for model verification and comparison.

Besides these more technical aspects of modelling standards, standards for a consequent modelling workflow are needed. In Sec. 2.3 a first draft was made to develop basic modelling workflows, model designs, and documentations. The topic of 'good modelling practice' for FSPM has not been discussed a lot in literature. One of the very few authors who dealt with this topic is Cournède *et al.* (2013), who derives from the increasing number of FSPM a need for the 'characterisation of different steps or a methodology in the process of modelling in terms of good modelling practice'. From a more mathematical point of view, Cournède identifies model analysis, model identification, and model evaluation as main steps for 'good modelling practice' for the PYGMALION (Sec. A.3.2) platform. For ABMs, Grimm and Railsback (2005) introduced the POM approach for designing, fitting and validating of models.

With respect to model design, one could think of model design solutions for common situations, e.g., of model components and the ways they communicate. Such design patterns could

be inspired by the well-known design patterns used in software engineering (Gamma *et al.* 1995).

For crop models (PBM), van Ittersum and Donatelli (2003) identified that 'model conceptualisation remains the heart of the matter'. They further explained the need to couple principles of systems analysis with new software engineering techniques. Adam (2010) investigated how the modelling task can be made more flexible using techniques taken from software engineering.

Concerning standardisation in model description and documentation for individual-based and agent-based models Grimm and Railsback (2005) first introduced the PSPC+3 (purpose, structure, process, concepts) protocol before it was revised to the ODD (overview, design concepts, and details) protocol as described in Grimm *et al.* (2006, 2010).

9.4. Scientific Impact of the Presented Work

For most of the presented publications (see Chapter 3 - 8) it is still too early to evaluate their scientific impact on plant modelling since they have just been published or in the case of two of them have only been accepted for publication. However, with respect to the main paper (see Chapter 6), in which the FSPM-P was presented, a significant influence can already be reported here. Tab. 9.2 lists models which use either parts of the FSPM-P or are totally based on it.

It is the essence of a prototype to provide all necessary elements and tools to start directly with a new implementation based on this prototype. This is also the case of the FSPM-Prototype (Chapter 6). In order to provide a working FSPM including a wide range of functions and at the same time being as general as possible a lot of code had to be written, which made the model quite complex (FSPM-P v0.4: nearly 6950 lines of code in 20 files). The complexity of the FSPM-P is essentially due to the minimal requirement to have a complete yet general functional-structural plant model. Even if about one-third of the code is actually occupied by the implementation of a set of nine photosynthesis models (that do not necessarily need to be consulted by the user) the remaining part can become rather challenging for beginners. A short introduction usually is sufficient to overcome this problem. shortcoming

Table 9.2.: Existing and envisaged models based on the FSPM-Prototype.

Cotton model	Buck-Sorlin, GH and Zhang, LZ, 2010-11, Beijing, PR China
Temperature sensitive rapeseed model	Tian, T *et al.* (submitted), Hangzhou, PR China
Tomato model	de Visser *et al.* (2014), Wageningen, The Netherlands
Cut-rose	Buck-Sorlin *et al.* (2011), Wageningen, The Netherlands
Rice model	Wu, LT, Buck-Sorlin, GH and Henke, M, 2010-11, Hangzhou, PR China
Soil/Root model	Henke *et al.* (2014b), Angers, France
Cucumber model	Chen *et al.* (2015), Hannover, Germany
Extension to tree model	Rimmele, S, 2016, master thesis, Göttingen, Germany
Maize/soy intercropping model	Munz, S, work in progress, Hohenheim, Germany

9.5. Outlook and Conclusion

Functional-structural plant modelling is a complex and time-consuming task with several activities and sub-activities. In the present work, we discussed most of these activities and explained them in detail. We have shown that using the prototype approach is an efficient way to develop new models, especially for beginners who are not yet too familiar with programming. As illustrated in the previous sections, a large number of open questions remain, which in turn harbours a large potential for improvements in FSPM. FSPM is a complex, interdisciplinary task that cannot be solved by a scientist working in isolation but that instead requires the collaboration of at least two researchers, ideally with complementary skills and experiences. The complexity of many FSPMs is hardly tangible hence barely transparent which makes such models scientific tools in their own right. Furthermore, these models should serve to enhancing transparency and encouraging communication and make newly generated knowledge accessible. It almost goes without saying communication and exchange on many levels are necessary in order to extend our knowledge and understanding of plants.

Therefore, we predict that among the main future challenges that the plant modelling community will be facing will be the introduction of *standards* and of *community-based libraries* of standard functions and data. Buck-Sorlin and Delaire (2013) listed 'design models as tools of

information exchange between the different types of experts and practitioners...' as one of the three principal future challenges for FSPM.

More powerful computers makes it increasingly unnecessary for FSPM to be restricted to the individual plant and small canopy scale for logistic reasons. Instead, larger and complex plant communities can now be modelled at high resolution, and keeping upright parameterisation at the organ scale. First steps in this direction are models of mixed-stands or inter-cropping models, e.g., Barillot *et al.* (2014), Gu *et al.* (2014), and Zhu (2015). In the future, this trend will surely be continued and probably be extended. Closely associated with intercropping models are root models and consequently soil models. Both are challenges by themselves and in various aspects stiff problems, e.g., with respect to data acquisition for root architecture or to the continuous modelling of soil with the varying and naturally dynamic distribution of mineral nutrients and organic matter.

Extending FSPM with (quantitative) genetic data and mechanisms to turn them into 3-d genotype-phenotype models is another major application area. Some promising first steps have been done in this domain by Kang *et al.* (2008) and Letort *et al.* (2008). Such genotype-phenotype models can be used for instance for virtual breeding (Xu *et al.* 2011) or to link (eco)physiological processes with identified QTLs. Genome-wide association studies are often coupled with high-throughput phenotyping - mass screening techniques - to reveal genetic variations in drought-stress resistance, heat or salinity tolerance of crop plants. These techniques are known to produce huge amounts of (raw) data, typically images or point clouds, often with a high temporal resolution. Here novel techniques of data preparation (e.g., image segmentation combined with photometric measurements) and management are required. These new domains of phenotyping, genomics, 'big data' and parameter estimation will in the future benefit from integration in FSPM.

Current FSPMs mainly focused on basic processes related to primary production of biomass and basic transport processes - increasing the yield was and still is the key focus. In future, further functional aspects will step in focus of interest of modelling. In order to increase product quality, secondary metabolism and related processes need to be taken into account. For instance, fruit quality relies on several different types of sugar, acids and ethylene, ... The content of vitamins, antioxidants, sugar as well as some secondary phytochemicals can be influenced by light (Brazaityte *et al.* 2010; Olle and Viršile 2013). In order to model such processes, advanced light modelling techniques are required to simulate light quality, mainly the simulation of spectral light. But beyond light, other aspects of the microclimate like air and organ temper-

ature, humidity, leaf surface wetness, or carbon dioxide concentrations needs to be simulated dynamically on a high temporal and physical resolution. Here further efforts are required.

Further promising applications of FSPMs are plant/pest interactions (de Vries, unpublished) and simulating the spreading and dynamics of pathosystem, e.g., fungal epidemics (Garin *et al.* 2014). Coupling FSPMs with so-called physics engines, e.g., to simulate rainfall or the distribution of sprayed pesticides (Dufour-Kowalski *et al.* 2007) is still mostly neglected and needs to be improved further.

This thesis is only a first step towards new techniques and concepts in FSPM – there is still a lot to be done.

APPENDIX A

FSPM Resources and Tools

As diverse as the different schools for plant modelling (see Sec. 2.2) that have emerged over the past fifty years are the software systems and platforms that have been developed by them and for them. A short collection of links to current and former plant modelling projects including the most common tools are mentioned in the following selection below. No claim is made that this is a complete listing, but it does show that there is currently very intensive activity in the field of FSPM. List in alphabetical order.

A.1. Algorithmic botany

The Biological Modeling and Visualization research group (http://algorithmicbotany.org) at the Department of Computer Science, University of Calgary, Canada.

A.1.1. L-Studio/cpfg

L-Studio (http://algorithmicbotany.org/lstudio/index.html), is one widely used L-system software tool, with numerous applications in functional-structural plant modelling, Prusinkiewicz *et al.* (2000a,b).

A.1.2. VLAB

VLAB (http://algorithmicbotany.org/vlab/index.html), the Virtual Laboratory.

A.2. AMAP

AMAP (http://amap.cirad.fr/en/index.php) joint research unit (French acronym UMR) is an interdisciplinary laboratory that conducts basic research on plants and plant communities with the aim of predicting ecosystems responses to environmental forcing, in terms of the distribution/conservation of species and biodiversity, crop production, carbon storage in plant biomass, protection of the environment and the provision of ecosystem services.

A.2.1. AMAPstudio

AMAPstudio (http://amapstudio.cirad.fr) is a software suite for plant architecture modelling. It contains applications and models to rebuild, explore, analyse and study the growth of plants from an architectural point of view. Main components are XPlo and Simeo, Griffon and de Coligny (2012).

A.2.2. AMAPsim

A simulator for structural growth of plants with a precise botanical basis based on the concept of physiologial age by Jean-Francois Barczi, Barczi *et al.* (2008).

A.3. Digiplante

Since 2002, the Digiplante team (http://digiplante.mas.ecp.fr) has been focussing on the mathematical modelling of plant growth. Digiplante develops mathematical and statistical methods as well as software to design, evaluate and apply plant growth models, Cournède *et al.* (2006).

A.3.1. Digiplante and DGP Suite

These tools are dedicated to the simulation of the GreenLab model and its parametric identification on real experimental data.

A.3.2. PyGMAlion

PyGMAlion (https://raweb.inria.fr/rapportsactivite/RA2011/digiplante/uid35.html) stands for Plant Growth Modelling Analysis, Identification and Optimization. It is an internal research framework which embeds some routines for modelling, simulation, parameter estimation, and analysing the sensibility of discrete dynamic systems. MAS (now MICS), Ecole Centrale Paris, France, Cournède *et al.* (2013).

A.4. ECOPHYS

ECOPHYS is an ecophysiological growth process model of juvenile poplar clones by Rauscher *et al.* (1990), USA.

A.5. FLORADIG

FLORADIG (http://pais.cirad.fr/tools/floradig.html) is a software to automate measurements of plant architecture (uses sonic and magnetic (Polhemus) digitisers), Hanan and Room (2002).

A.6. GRAAL

GRAAL (http://library.wur.nl/ojs/index.php/frontis/article/view/1380) is an integrated functional-structural model to simulate and analyse the interactions between morphogenetic processes and assimilate partitioning during the vegetative development of individual plants. INRA, Avignon, France, Drouet and Pagès (2007).

A.7. GreenLab AMAP, Modelling Plant Development & Growth

The GreenLab AMAP (http://greenlab.cirad.fr) research team which was created in January 2001 stands for a long history of modelling plant growth architecture and functioning: AMAP's 1980's pioneering work, the Sino-French Cplant/GreenLab project, and the Digiplante CIRAD-ECP-INRIA EPI.

A.7.1. GreenLab

GreenLab (http://greenlab.cirad.fr) is a dynamical model of plant growth for environmental applications, by Philippe de Reffye, France, Hu *et al.* (2003).

Meanwhile, various GreenLab (re-)implementations and extensions under different environments are available:

GreenScilab (http://greenlab.cirad.fr/GLUVED/html/P3_Tools/Tool_GreenScilab), Chinese Academy of Sciences, CASIA, Kang *et al.* (2006).

QingYuan (http://www.cybernature.com.cn/cPlant/software.html), Chinese Academy of Sciences, CASIA, Kang *et al.* (2010).

Digiplante see Sec. A.3

GreenLab-XL In GroIMP implemented version of GreenLab, Department Ecoinformatics, Biometrics and Forest Growth, University of Götingen, Smoleňová *et al.* (2012).

A.8. GROGRA

GROGRA – **Gro**wth **Gra**mmar Interpreter (http://wwwuser.gwdg.de/~groimp/grogra.de/software/grogra.html) is a software for the interpretation of sensitive growth grammars which represent an extended variant of L-systems, developed by Winfried Kurth. Their creation was guided by the fact that the pure L-systems formalism cannot cope properly with the representation of the great variety of plant architectures and growth behaviours. Göttingen, Germany, Kurth (1994a,c).

A.9. GroIMP

GroIMP – **Gro**wth-grammar related **I**nteractive **M**odelling **P**latform (http://grogra.de, Tab. A.1), developed by Ole Kniemeyer *et al.* 2008, is a 3-d-modelling platform which supports features like interactivity, a rich set of 3-d objects and data interfaces. Its speciality is the integrated modelling language XL. Department Ecoinformatics, Biometrics and Forest Growth at the University of Göttingen, Germany, Hemmerling *et al.* (2008) and Kniemeyer (2008). For a list of contributors see App. B.

A.9.1. XL

The programming language extended L-system language (XL) is an implementation of relational growth grammars. XL is built as a super-set of the programming language Java: This

Table A.1.: GroIMP - profile.

Paradigm(s)	multi-paradigm: imperative, object-oriented, rule-based, chemical
Appeared in	2003
Designed by	Ole Kniemeyer
Stable release	1.5 (08.11.2016)
Typing discipline	static, strong, safe
Influenced by	GROGRA, ...
Operating System	Cross-platform, (multi-platform - java virtual machine (JVM))
License	GPL - GNU General Public License V.3
File name extensions	gsz, rgg, xl
Available at	sourceforge.net
WWW	grogra.de

combines the advantages of the rule-based paradigm with the power of Java, including the rich set of existing Java libraries, Kniemeyer and Kurth (2008).

A.10. LIGNUM

LIGNUM (http://www.metla.fi/metinfo/kasvu/lignum/index-en.htm) model for structural dynamics of trees. The tree in LIGNUM consists of components that are similar to real tree parts. Both metabolism (e.g., photosynthesis, respiration etc.) and crown architecture are accounted for. Developed by Jari Perttunen at Metla (now LUKE), Vantaa, Finland, Perttunen *et al.* (1998, 1996).

A.11. LParser

LParser (http://web.archive.org/web/20080226045625/home.wanadoo.nl/laurens.lapre/lparser.html), was developed by Laurens Lapré. Parts of the origi-

nal code can be found in the following L-system software packages: FLEA, Lmuse, L-neuron (Ascoli and Krichmar 2000), Lsystem4, L-breeder and AM Lsystem.

A.12. MAppleT

A simulation of apple tree architecture development by Evelyn Costes *et al.* (2008), INRA, France.

A.13. OpenAlea Modelling Framework

OpenAlea (http://openalea.gforge.inria.fr/dokuwiki/doku.php) is an open source project, by the 'Virtual plant' team, primarily aimed at the plant research community. It is a distributed collaborative effort to develop Python libraries and tools that address the needs of current and future works in plant architecture modelling. OpenAlea includes modules to analyse, visualise and model the functioning and growth of plant architecture. Developed at CIRAD, INRA and INRIA, France, Pradal *et al.* (2008).

A.13.1. L-Py

L-Py (http://openalea.gforge.inria.fr/wiki/doku.php?id=packages:vplants:lpy:main) is a Python version of L-systems published under GPL-license. It is based on the specification of L-Studio/cpfg-lpfg, Boudon *et al.* (2010) and Boudon *et al.* (2012).

A.13.2. OpenAleaLab

OpenAleaLab is a multi-paradigm modelling environment for plants. It will permit to divide the modeller's work into multiple tasks. Each task can be viewed as a virtual experiment. OpenAleaLab is based on OpenAlea Library. Developed by Christophe Pradal, Pradal *et al.* (2008).

A.13.3. PlantGL

PlantGL (http://openalea.gforge.inria.fr/wiki/doku.php?id=packages:visualization:plantgl:plantgl) is an open-source graphic toolkit for the creation, simulation and analysis of 3-d virtual plants developed by Boudon, Pradal, and Nouguier, Pradal *et al.* (2009).

A.14. PlantStudio

PlantStudio (http://www.kurtz-fernhout.com/summary_plantstudio.html) is a tool for creating 3-d plant models and 2-d illustrations. PlantStudio simulates herbaceous (non-woody) plants like wildflowers and cut flowers, vegetables, weeds, grasses, and herbs using a parameter-driven simulation of plant growth and structure.

A.15. VICA

VICA (http://library.wur.nl/ojs/index.php/frontis/article/view/1371) is a generic functional-structural plant model specified for barley. Developed by Wernecke at the Martin-Luther-University of Halle-Wittenberg, Germany, Wernecke *et al.* (2007).

A.16. Virtual Plants

Virtual Plants (https://team.inria.fr/virtualplants/) is a joint research team between INRIA, CIRAD and INRA. Its aim is to develop computational models of plant development to understand the physical and biological principles that drive the development of plant branching systems and organs. OpenAlea is one of its modelling projects.

A.17. Xfrog

Xfrog (http://xfrog.com) is a procedural organic 3-d modeller that allows the creation and animation of 3-d trees, flowers, nature based special effects or architectural forms. Xfrog is available as a Mac OS, Windows, and Linux plug-in for Maya, Cinema 4D and as a standalone application. Developed by Oliver Deussen and Bernd Lintermann, Germany, Deussen and Lintermann (2005) and Greenworks (2017).

APPENDIX B

GroIMP Developer Team

GroIMP was developed as open source project by Ole Kniemeyer. Thanks to the dedicated work and help of several contributors during the past decade, GroIMP could grow to what we have today. List in alphabetical order:

- Dietger van Antwerpen
- Udo Bischof
- Jan Dérer
- Cong Ding
- Octave Etard
- Birka Fonkeng
- Konni Hartmann
- Reinhard Hemmerling
- Michael Henke
- Thomas Huwe
- Ole Kniemeyer
- Ralf Kopsch
- Qinqin Long
- Wanchun Luo
- Uwe Mannl
- Paul Masters
- Johannes Merklein
- Yongzhi Ong
- Stephan Rogge
- Sören Schneider
- Hagen Steidelmüller
- Katarína Streit
- Michael Tauer
- Shining Wu
- Dexu Zhao

Glossary

absorption spectrum

An absorption spectrum is a plot of the intensity of light absorbed by plant organs relative to its wavelengths. It follows directly the content and combination of pigments in the plant organ.

action spectrum

The action spectrum indicates the relative efficiency with which light of various wavelengths is able to promote photosynthesis in the photosynthetically active plant organs. It can be estimated by measuring the O_2 production of, e.g., leaves following exposure to various wavelengths.

active contour

Active contours, also called snakes, are a mathematical concept, used in image processing to detect an object outline in a 2-d image. Snakes use parametric curves (e.g., deformable splines) to describe the object contour. Based on so-called internal and external forces, after an initialisation, the shape of the contour is calculated, in which the sum of all forces reaches a minimum.

actual growth rate

Actual increment in length, area or dry weight of an organ per unit time, usually observed under conditions of limited assimilate supply, shortage of nutrients, light and water, and/or competition/damage.

actual photochemical efficiency

Actual photochemical efficiency is calculated from the mean values of the minimal (F) and maximal ($F_m{}'$) fluorescence signals: $\phi_P = (F_m{}' - F)/F'_m$. See potential photochemical efficiency.

architecture

Is the term referring to the topology (connection with each other) and geometry (arrangement in space), and size distribution (biometry and shape as a function of position) of plant organs. Plant architecture thus comprises the overall appearance of an individual plant as a result of organogenesis and morphogenesis.

Beer-Lambert law

Describes the attenuation of light due to the properties of a homogeneous medium through which the light is travelling. The absorbance of light is directly proportional to the thickness of the media through which the light is being transmitted multiplied by the concentration of absorbing chromophore, (Beer 1852).

biomass

The dry or fresh matter of plants, produced by photosynthesis and growth.

biometry

Also referred to as biostatistics, biometry is the application of statistics to biology. This comprises the design of biological experiments (e.g., in horticulture or agriculture); data

acquisition and statistical analysis; and lastly its interpretation of the findings, often using linear or non-linear models.

chaos game

In mathematics, the term chaos game referred to a method of creating fractals, using a polygon. In an iterative process, starting with an initial random point inside the polygon, new points are created by applying one of the functions chosen at random from the function system to transform the point in order to obtain a next point. The new point is placed as a fraction of the distance between the previous point and one of the vertices of the polygon. At the end of this process, depending on the used fraction, a fractal shape can be produced.

compartment

In the context of plant biology, a compartment refers to a distinct space within a plant or its different organs which is characterised by physical or physiological boundaries to other compartments or the environment, and by uniform conditions for metabolic reactions.

computer-aided design

Is the use of computer techniques in designing, drafting and/or modelling of products or structures in 2-d and 3-d, especially involving the use of computer graphics. CAD software is used in art, architecture, engineering and manufacturing to assist in precision drawing.

curve fitting

Curve fitting is the determination of a curve, or mathematical function, that fits a set of data points best. Points are interpolated when the resulting curve fits exactly to the data otherwise the data points are approximated by a smooth curve.

degree-day

Unit of time used to describe the rate of development of a plant: it is based on the mean temperature on a given day (average of maximum and minimum temperature, or using a more refined method like hourly temperature), minus a base temperature below which no development takes place (the latter must be determined experimentally by growing the crop plant at a wide range of different temperatures). The developmental age can thus be indicated in 'heat units' or 'cumulated temperature', which is more accurate than using the chronological age of a plant in days.

functional-structural plant model/modelling

Refers to a paradigm for the description of a plant by creating a (usually object-oriented) computer model of its structure and selected physiological and physical processes, at different hierarchical levels: organ, plant individual, canopy (a stand of plants), and in which the processes are modulated by the local environment. Structure comprises the explicit topology (connection between organs) and geometry (orientation, inclination, and shape) of the organs and the plant. At the individual level, this is also referred to as plant architecture. An FSPM may consider a change in organ and plant structure in time, thereby simulating growth, extension, and branching processes of a given plant. This type of FSPM is referred to as dynamic. A static FSPM, on the other hand, only considers an unchanging structure, which is used as a model input in order to explain spatial heterogeneity in physiological processes (Buck-Sorlin 2013).

generative

Is the (later) developmental phase of the plant in which flowers and fruits are formed and after which the plant dies off if it is not perennial.

graph

In graph theory, a graph is an abstract structure representing a set of objects and their relations. Formally, a graph is a tuple (V, E), comprising a set of vertices V and a set of edges E. Depending on the definition of E several types of graphs can be distinguished, e.g., directed or undirected. A tree is a special case of connected graph with no cycles.

growth function

Any mathematical function that describes cell, organ, individual, or population growth, i.e. the dynamics of a state variable associated with growth (biomass, length, width, area) in time. Most often sigmoid functions, like the logistic, Gompertz, Pearl or Richards curve are used.

growth respiration

Amount of assimilates needed for production of new structural biomass, usually an organ-specific constant depending on the composition of the newly formed biomass (proteins, carbohydrates, fats, minerals).

growth unit

Set of metamers developed during one growth cycle.

harvest index

In agronomy, the harvest index defines a measurement of crop yield, defined by the weight of the harvested products as a percentage of the total plant weight of a crop. It is a typical output of PBMs.

Huber value

Refers to the transverse xylem area per gram dry weight of supplied leaves, (Huber 1928).

inflorescence

Branched or unbranched structure on the plant which bears the flowers.

iterated function system

In mathematics, an iterated function system (IFS) is a method of constructing self-similar fractals. They were conceived in their present form by John E. Hutchinson (1981).

leaf area index

It is defined as the one-sided green leaf area per unit ground surface area [leaf area / ground area]. LAI can be used to characterise plant canopies and to predict photosynthetic primary production (PP). LAI is a standard reference tool in process-based models.

light model

In computer graphics a light model refers to a tool (ideally) based on physical laws able to simulate light within a 3-d scene of objects. A shading model is used to compute the local illumination of a surface by one or more light sources. The application of this tool is called light modelling or light computation. Light modelling should not be confused with rendering.

light use efficiency

Is the ratio between dry mass and the absorbed irradiance of photosynthetically active radiation (PAR) per square meter per day. It differs from radiation use efficiency (RUE) in which total irradiance is considered. See also: PAR.

maintenance respiration

Process that maintains existing structural biomass by respiration of sugars; usually the amount of sugars corresponds to 1% of the structural biomass per day, (Goudriaan and van Laar 1994).

meristem

From Greek merisein, 'divide', and stemma, 'crown'. Plant tissue consisting of undifferentiated stem cells (meristematic cells) usually located at shoot tips; meristems form new phytomers (phytons). Meristems can be classified into three classes: primary meristem, secondary meristem, and tertiary meristem.

metamer

See phytomer (phyton).

morphogenesis

Morphogenesis comprises all processes of organ extension in length and width, leading to the final dimensions of an organ. Growth processes in the form of biomass allocation to extending organs accompany these processes but are not strictly linked to them.

morphology

Derived from Greek roots: *morphe* which means form and/or shape and *logos*, meaning discourse or investigation, together it can be translated as 'study of shape'. In biology, morphology is dealing with the study of the form and structure of organisms and their specific structural features. Plant morphology can be divided into external morphology (eidonomy) and internal morphology (anatomy).

ontogeny

In Biology, ontogeny, also referred to as ontogenesis, describes the life cycle of a single organism. Essentially it is the process of an individual organism growing organically, or the biological unfolding of events involved in an organism changing gradually from a simple to a more complex level.

organogenesis

Is the formation of new organs, as primordia in a shoot apical meristem (SAM), due to controlled cell division and differentiation in certain tissue regions of the SAM.

phenology

Appearance of a plant or canopy during its development, usually defined by characteristic events, such as germination, appearance of the nth leaf, tillering/branching, flowering, fruiting, etc., as controlled by climatic conditions (temperature sum, day length, etc.).

photogrammetry

Photogrammetry comprises all of the methods and apparatus of measurement, processing and evaluation of remote sensing to determine from photographs and exact measuring images of an object's spatial position or three-dimensional shape. Normally, the images are recorded with dedicated measuring cameras. In the two-dimensional case flatbed scanners are typical.

photosynthesis

Photosynthesis is the active production of carbohydrates from CO_2 and water in plant organs, mainly leaves, using energy from sunlight, more precisely, from photosynthetically active radiation. The energy of sunlight is converted into chemical energy.

photosynthesis model

A mathematical model to calculate either the instantaneous photosynthesis rate or the integral of assimilate production over a day, using a number of driving variables, most often PAR and temperature. One widely used biochemical model was developed by Farquhar *et al.* (1980). Ball *et al.* (1987) introduced quantitative models for stomatal conductance, leaf energy balance, and transpiration. This model can be linked to models of leaf carbon metabolism and the environment to predict fluxes of CO_2, H_2O and energy.

photosynthetic (active) photon flux density

Current density of photons in the photosynthetically active solar spectrum (400 - 700 nm). PPFD is a typical input for most photosynthesis models since photosynthesis is a quantum process and the chemical reactions of photosynthesis are more dependent on the number of photons than the energy contained in the photons (given by the PAR-value). It is usually given in µmol photons m-2 s-1.

photosynthetically active radiation

The radiation (range of wavelengths) that plants can use for photosynthesis, roughly the visible spectrum of sunlight (400 - 700 nm). Since it depends on the concentration of pigments (mainly chlorophyll, alpha- and beta-carrotine) inside plant organs it is species-

specific and even varying within an individual. Measured in W m-2. When PAR is quantified by the number of photons in the active range received by a surface for a specified amount of time the photosynthetic (active) photon flux density-value can be obtained.

phyllochron

Time in degree-days between emergence of successive leaves.

physical light distribution

Describes the distribution of light within a scene, as caused by light sources and objects which reflect, absorb and transmit light.

physiological age

Relates to the degree of differentiation of the structures produced by a meristem. It may be estimated a posteriori by a non-limitative series of qualitative and quantitative criteria, (Barthélémy and Caraglio 2007).

physiology

Plant physiology is a subdiscipline of botany concerned with the functioning, or physiology, of plants. It comprises the study of all the internal activities of plants-those chemical and physical processes associated with life as they occur in plants. This includes study at many levels of scale of size and time. From Wikipedia: Physiology, modified.

phytomer (phyton)

A phytomer (also called a metamer), from Greek phyton, 'plant', and meris, 'part', thus literally 'plant part', is a structural unit produced by a shoot apical meristem (SAM), continuously or rhythmically, throughout a plant's vegetative life-cycle. A typical phytomer is a repeated constructional unit, consisting of an internode, a node to which a leaf is attached, and a lateral bud (containing another SAM) in the axil of the leaf. As a variation to this general theme, one or more of the aforementioned organs may be lacking, or else an organ may be occurring in more than one copy, as is the case of axillary buds in some tropical trees.

Initially, a young plant will produce a main stem due to the activity of the terminal SAM. At some later stage, first and higher-order branches are formed when some of the lateral buds break (e.g., start to grow out after a rest period).

plastochron

Refers to the rate of initiation of undifferentiated primordia. The plastochron index and the leaf plastochron index are ways of measuring the age of a plant dependent on morphological traits rather than chronological.

potential growth rate

Maximum increment in length, area or dry weight of an (actively growing) organ per unit time under unlimiting conditions, i.e. if there is a surplus of assimilates and no shortage of nutrients, light and water. Usually described as the derivative of a logistic or beta function.

potential photochemical efficiency

Potential photochemical efficiency is calculated from the mean values of the minimal (F_0) and maximal (F_m) fluorescence: $\phi_{P_0} = (F_m - F_0)/F_m$. See actual photochemical efficiency.

primordia

See primordium.

primordium

Embryonic stage of an organ, after its initiation in the shoot apical meristem (SAM).

process-based model

Qualifies a mathematical or computational model that expresses the plant/crop production dynamics on the basis of processes like photosynthesis.

radiation use efficiency

Is the ratio between dry mass and the irradiance per square meter per day. It differs from light use efficiency (LUE) in which absorbed irradiance of PAR is considered. See also: PAR.

raytracing

In computer graphics, ray tracing is a rendering method for generating an image of a virtual three-dimensional scene on a computer by following a light path from a virtual camera through pixels in an image plane. It simulates light reflections, refractions and shadows. Although computationally extremely intensive, ray tracing provides the most realistic shadows, reflections and refractions.

regression analysis

Is a collection of statistical techniques for modelling and analysing several variables, describing the relationship between a dependent variable and one or more independent variables.

relational growth grammar

Special type of parallel graph rewriting system incorporating rule-based, procedural and object-oriented concepts (Kniemeyer *et al.* 2004). The first programming language that implements the concept of RGG is called XL, (Kniemeyer 2004, 2008), and was made available for plant modelling as part of the modelling platform GroIMP, (Kniemeyer *et al.* 2007).

rendering

In computer graphics, rendering is a method for generating an image of a virtual 3-dimensional scene on a computer. It simulates light reflections, refractions and shadows. The most realistic results can be obtained by so-called raytracing methods.

scene graph

A specialised data structure used to represent 3-d geometries.

senescence

Last developmental stage of an organ characterised by processes such as cessation of, or decrease in photosynthesis, active reallocation of assimilates, and finally dehydration and decay.

sensitivity analysis

Study of how the uncertainty in the output of a mathematical model or system (numerical or otherwise) can be apportioned to different sources of uncertainty in its inputs, (Saltelli 2002).

shoot apical meristem

The shoot apical meristem is a region of undifferentiated stem cell tissue. It is located either within a bud at the tip of a shoot or in the axil of a leaf, and is responsible for primary plant growth by rhythmic formation of phytomers.

sink strength

Assimilates are transported via the phloem from sources (leaves, green internodes, or green fruits) to sinks (growing organs). Sink strength is defined as the rate of an organ to use these assimilates, e.g., for growth. A good proxy for sink strength is thus the instantaneous potential growth rate of an organ, but a mature organ can still exhibit a certain sink strength due to maintenance respiration.

specific internode length

The ratio between the internode length and the internode dry weight SIL = len/DW; measured in meter per kg.

specific internode mass

The ratio between the internode dry weight and the internode volume SIM = DW/volume; measured in kg per cubic meter.

specific leaf area

Ratio between leaf area and dry mass. Its value is the leaf area [m-2] that weighs exactly one gram; used as a conversion coefficient to compute leaf area from leaf dry mass.

specific leaf weight

The ratio between the leaf dry weight and the leaf surface. Specific leaf weight (SLW), of a single leaf = 1/specific leaf area; measured in kg per meter squared.

spectral power distribution

Describes the power per unit area per unit wavelength of an illumination.

string rewriting system

In mathematical logic and theoretical computer science, a string rewriting system, historically called a semi-Thue system, is a rule based system for the manipulation of strings.

surface triangulation

Triangulation of a surface involves the production of a net of triangles, which covers a given surface partly or totally.

Turing complete

In computability theory, a system of data-manipulation rules (e.g., a computer programming language) is said to be 'Turing complete' or 'computationally universal' if it can be used to simulate any single-taped Turing machine. In other words, it is possible to write

a program, in that language, that can compute anything that a (single-taped) Turing machine could compute.

Turing machine

A Turing machine is a very simplified computer that can be analysed mathematically and therefore an important abstract machine in theoretical computer science.

uncertainty analysis

The study of the uncertain aspects of a model and of their influence on the (uncertainty of the) model output, (Grasman and van Straten 2012).

vegetative

Is the (early) developmental phase of the plant in which leaves and stems are formed.

yield photon flux

Photosynthesis is fundamentally driven by photon flux rather than energy flux, but not all absorbed photons yield equal amounts of photosynthesis. This results in two measures for photosynthetically active radiation: photosynthetic (active) photon flux density and yield photon flux (YPF). The yield photon flux is a weighted photon range according to plant photosynthetic response.

Acronyms

ABM	agent-based model
APE	actual photochemical efficiency
CAD	computer-aided design
CER	carbon exchange rate
CIRAD	Centre de coopération internationale en recherche agronomique pour le développement
CPU	central processing unit
CSG	constructive solid geometry
CSV	comma separated values
DoY	day of year
DW	dry weight
FSPM	functional-structural plant model/modelling
GM	geometrical model
GPL	GNU General Public License
GPU	graphics processing unit

GroIMP	**Gro**wth-grammar related **I**nteractive **M**odelling **P**latform
IDE	integrated development environment
IFS	iterated function system
JVM	java virtual machine
LAD	leaf angle distribution
LAI	leaf area index
LiDAR	light detection and ranging
LM	light model
LPI	leaf position index
LSC	leaf specific conductivity
LUE	light use efficiency
MTG	multiscale tree graph
NURBS	non-uniform rational B-Spline
ODE	ordinary differential equation
PA	physiological age
PAR	photosynthetically active radiation
PBM	process-based model
PCA	principal component analysis
PLD	physical light distribution
POM	pattern-oriented modelling
PP	primary production
PPFD	photosynthetic (active) photon flux density
PS	photosynthesis
PSM	photosynthesis model

QTL	quantitative trait locus
RGG	relational growth grammar
RLE	rate of leaf emergence
RT	raytracing
RUE	radiation use efficiency
SAM	shoot apical meristem
SPD	spectral power distribution
SRS	string rewriting system
VRML	virtual reality modelling language
XL	extended L-system language
YPF	yield photon flux

Bibliography

Abelson, H. and diSessa, A. (1981). Turtle Geometry: The Computer as a Medium for Exploring Mathematics. MIT Press Series in Artificial Intelligence. MIT Press (cit. on p. 20).

Adam, M. (2010). A framework to introduce flexibility in crop modelling: From conceptual modelling to software engineering and back. PhD thesis. Wageningen University (cit. on p. 218).

Aksoy, E. E., Abramov, A., Wörgötter, F., Scharr, H., Fischbach, A., and Dellen, B. (2015). Modeling leaf growth of rosette plants using infrared stereo image sequences. In: *Computers and Electronics in Agriculture* 110: pp. 78–90. DOI: 10.1016/j.compag.2014.10.020 (cit. on p. 8).

Aono, M. and Kunii, T. L. (1984). Botanical tree image generation. In: *Computer Graphics and Applications, IEEE* 4(5): pp. 10–34. DOI: 10.1109/MCG.1984.276141 (cit. on p. 14).

Arvo, J. and Kirk, D. (1988). Modeling plants with environment-sensitive automata. In: *Proceedings of Ausgraph '88*: pp. 27–33 (cit. on p. 39).

Ascoli, G. A. and Krichmar, J. L. (2000). L-neuron: A modeling tool for the efficient generation and parsimonious description of dendritic morphology. In: *Neurocomputing* 32-33: pp. 1003–1011. DOI: 10.1016/S0925-2312(00)00272-1 (cit. on p. 229).

Backus, J. (1998). The history of fortran I, II, and III. In: *IEEE Annals of the History of Computing* 20(4): pp. 68–78. DOI: 10.1109/85.728232 (cit. on p. 13).

Ball, J., Woodrow, I. E., and Berry, J. A. (1987). A model predicting stomatal conductance and its contribution to the control of photosynthesis under different environmental conditions. In: *Proceedings of the 7th International Congress on Photosynthesis Providence*. Ed. by Biggins,

J. Vol. 4. Progress in Photosynthesis Research. Rhode Island, USA: Springer Netherlands: pp. 221–224. DOI: 10.1007/978-94-017-0519-6_48 (cit. on p. 242).

Balzert, H. (2009). Lehrbuch der Softwaretechnik: Basiskonzepte und Requirements Engineering. Spektrum Akademischer Verlag (cit. on p. 72).

Barczi, J.-F., Rey, H., Caraglio, Y., de Reffye, P. H., Barthélémy, D., Dong, Q. X., and Fourcaud, T. (2008). AmapSim: A structural hole-plant simulator based on botanical knowledge and designed to host external functional models. In: *Annals of Botany* 101(8): pp. 1125–1138. DOI: 10.1093/aob/mcm194 (cit. on p. 224).

Barillot, R., Escobar-Gutiérrez, A. J., Fournier, C., Huynh, P., and Combes, D. (2014). Assessing the effects of architectural variations on light partitioning within virtual wheat–pea mixtures. In: *Annals of Botany* 114(4): p. 725. DOI: 10.1093/aob/mcu099 (cit. on p. 220).

Barnsley, M. F. and Demko, S. (1985). Iterated function systems and the global construction of fractals. In: *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*. Vol. 399. 1817. School of mathematics, Georgia Institute of Technology. Royal Society, Atlanta, Georgia: The Royal Society: pp. 243–275. DOI: 10.1098/rspa.1985.0057 (cit. on p. 48).

Barnsley, M. F. (1988). Fractals Everywhere. San Diego, CA, USA: Academic Press Professional, Inc. (cit. on pp. 46, 48, 49).

Barnsley, M. F., Elton, J. H., and Hardin, D. P. (1989). Recurrent iterated function systems. In: *Constructive Approximation* 5(1): pp. 3–31. DOI: 10.1007/BF01889596 (cit. on p. 49).

Barthélémy, D. and Caraglio, Y. (2007). Plant architecture: A dynamic, multilevel and comprehensive approach to plant form, structure and ontogeny. In: *Annals of Botany* 99(3): pp. 375–407. DOI: 10.1093/aob/mcl260 (cit. on p. 243).

Beer, A. (1852). Bestimmung der Absorption des rothen Lichts in farbigen Flüssigkeiten. In: *Annalen der Physik und Chemie* 86: pp. 78–88 (cit. on p. 236).

Bell, A., Roberts, D., and Smith, A. (1979). Branching patterns: The simulation of plant architecture. In: *Journal of Theoretical Biology* 81: pp. 351–375 (cit. on p. 15).

Borchert, R. and Honda, H. (1984). Control of development in the bifurcating branch system of tabebuia rosea: A computer simulation. In: *Botanical Gazette* 145(2): pp. 184–195. DOI: 10.1086/337445 (cit. on p. 15).

Boudon, F., Cokelaer, T., Pradal, C., and Godin, C. (2010). L-Py, an open L-systems framework in Python. In: *Proceedings of the 6th International Workshop on Functional-Structural Plant Models* (cit. on p. 229).

Boudon, F., Pradal, C., Cokelaer, T., Prusinkiewicz, P., and Godin, C. (2012). L-Py: An L-system simulation framework for modeling plant development based on a dynamic language. In: *Frontiers in Plant Science* 3(76). DOI: 10.3389/fpls.2012.00076 (cit. on p. 229).

Brazaityte, A., Duchovskis, P., Urbonaviciute, A., Samuoliene, G., Jankauskiene, J., Sakalauskaite, J., Sabajeviene, G., Sirtautas, R., and Novickovas, A. (2010). The effect of light-emitting diodes lighting on the growth of tomato transplants. In: *Zemdirbyste-Agriculture* 97(2): pp. 89–98 (cit. on p. 220).

Buck-Sorlin, G. H. (2013). Functional-Structural Plant Modeling. In: *Encyclopedia of Systems Biology*. Ed. by Dubitzky, W., Wolkenhauer, O., Cho, K.-H., and Yokota, H. Springer New York: pp. 778–781. DOI: 10.1007/978-1-4419-9863-7_1479 (cit. on pp. 59, 238).

Buck-Sorlin, G. H., Burema, B., Evers, J. B., van der Heijden, G., Heuvelink, E., Marcelis, L. F. M., Struik, P. C., de Visser, P. H. B., Damen, T., and Vos, J. (2007a). Virtual rose: A new tool to optimize plant architecture in glasshouse rose production systems. In: *Proceedings of the 5th International Workshop on Functional-Structural Plant Models*. Ed. by Prusinkiewicz, P., Hanan, J., and Lane, B. Napier, New Zealand: p. 48 (cit. on p. 65).

Buck-Sorlin, G. H., de Visser, P. H. B., Burema, B. S., Heuvelink, E., Marcelis, L. F. M., van der Heijden, G. W. A. M., and Vos, J. (2010). SIMPLER: An FSPM coupling shoot production, human interaction with the structure, morphogenesis, photosynthesis and light environment in cut-Rose. In: *Proceedings of the 6th International Workshop on Functional-Structural Plant Models*. University of California, Davis, CA, USA (cit. on p. 82).

Buck-Sorlin, G. H., de Visser, P. H. B., Henke, M., Sarlikioti, V., van der Heijden, G. W. A. M., F. M., M. L. F., and Vos, J. (2011). Towards a functional–structural plant model of cut-rose: Simulation of light environment, light absorption, photosynthesis and interference with the plant structure. In: *Annals of Botany* 108(6): pp. 1121–1134. DOI: 10.1093/aob/mcr190 (cit. on pp. 65, 82, 219).

Buck-Sorlin, G. H. and Delaire, M. (2013). Meeting present and future challenges in sustainable horticulture using virtual plants. In: *Frontiers in Plant Science* 4(443). DOI: 10.3389/fpls.2013.00443 (cit. on pp. 1, 3, 219).

Buck-Sorlin, G. H., Guillermin, P., Delaire, M., Sané, F., and le-Morvan, C. (2012). Towards a multi-scaled functional-structural model of apple, linking ecophysiology at the fruit and branch scales. In: *4th International Symposium on Plant Growth Modeling, Simulation, Visualization and Applications (PMA12), 2012*: pp. 66–69. DOI: 10.1109/PMA.2012.6524814 (cit. on p. 65).

Buck-Sorlin, G. H., Kniemeyer, O., and Kurth, W. (2007b). A grammar-based model of barley including virtual breeding, genetic control and a hormonal metabolic networks. In: *Functional-Structural Plant Modelling in Crop Production*. Ed. by Vos, J., Marcelis, L. F. M., de Visser, P. H. B., Struik, P. C., and Evers, J. B. NewYork: Springer, Dordrecht: pp. 243–252 (cit. on p. 65).

Buck-Sorlin, G. H., Kniemeyer, O., and Kurth, W. (2005). Barley morphology, genetics and hormonal regulation of internode elongation modelled by a relational growth grammar. In: *New Phytologist* 166(3): pp. 859–867. DOI: 10.1111/j.1469-8137.2005.01324.x (cit. on p. 65).

Buck-Sorlin, G. H., Kniemeyer, O., and Kurth, W. (2008). A model of poplar (*Populus sp.*) physiology and morphology based on relational growth grammars. In: *Mathematical Modeling of Biological Systems, Volume II*. Ed. by Deutsch, A., Parra, R. B. d. l., Boer, R. J. d., Diekmann, O., Jagers, P., Kisdi, E., Kretzschmar, M., Lansky, P., and Metz, H. Vol. 2. Modeling and Simulation in Science, Engineering and Technology. Birkhäuser Boston: pp. 313–322. DOI: 10.1007/978-0-8176-4556-4_28 (cit. on pp. 65, 82).

Canny, J. (1986). A computational approach to edge detection. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 8(6): pp. 679–698 (cit. on p. 208).

Carson, E. R. and Cobelli, C. (2001). Modelling methodology for physiology and medicine. Elsevier Insights. Academic Press, San Diego (USA) (cit. on p. 2).

Cartenì, F., Giannino, F., Schweingruber, F. H., and Mazzoleni, S. (2014). Modelling the development and arrangement of the primary vascular structure in plants. In: *Annals of Botany* 114(4): pp. 619–627. DOI: 10.1093/aob/mcu074 (cit. on p. 9).

Chen, C.-C., Chen, H., and Chen, Y.-R. (2009). A new method to measure leaf age: Leaf measuring-interval index. In: *American Journal of Botany* 96(7): pp. 1313–1318. DOI: 10.3732/ajb.0800303 (cit. on p. 205).

Chen, T.-W., Henke, M., de Visser, P. H. B., Buck-Sorlin, G. H., Wiechers, D., Kahlen, K., and Stützel, H. (2014). What is the most prominent factor limiting photosynthesis in different layers of a greenhouse cucumber canopy? In: *Annals of Botany* 114(4): pp. 677–688. DOI: 10.1093/aob/mcu100 (cit. on pp. 65, 82).

Chen, T.-W., Nguyen, T. M. N., Kahlen, K., and Stützel, H. (2015). High temperature and vapor pressure deficit aggravate architectural effects but ameliorate non-architectural effects of salinity on dry mass production of tomato. In: *Frontiers in Plant Science* 6(887). DOI: 10.3389/fpls.2015.00887 (cit. on p. 219).

Chomsky, N. (1956). Three models for the description of language. In: *IRE Transactions on Information Theory* 2(3): pp. 113–124. DOI: 10.1109/TIT.1956.1056813 (cit. on pp. 27, 28).

Cohen, D. (1967). Computer simulation of biological pattern generation processes. In: *Nature* (216): pp. 246–248. DOI: 10.1038/216246a0 (cit. on pp. 7, 13).

Conner, W. S. (1999). A computer vision based tree ring analysis and dating system. MA thesis. The University of Arizona (cit. on p. 207).

Costes, E., Smith, C., Renton, M., Guédon, Y., Prusinkiewicz, P., and Godin, C. (2008). MAppleT: Simulation of apple tree development using mixed stochastic and biomechanical models. In: *Functional Plant Biology* 35: pp. 936–950. DOI: 10.1071/FP08081 (cit. on p. 229).

Côté, J.-F., Widlowski, J.-L., Fournier, R. A., and Verstraete, M. M. (2009). The structural and radiative consistency of three-dimensional tree reconstructions from terrestrial lidar. In: *Remote Sensing of Environment* 113(5): pp. 1067–1081. DOI: 10.1016/j.rse.2009.01.017 (cit. on p. 63).

Cournède, P.-H., Chen, Y., Wu, Q., Baey, C., and Bayol, B. (2013). Development and evaluation of plant growth models: Methodology and implementation in the PYGMALION platform. In: *Mathematical Modelling of Natural Phenomena* 8(4): pp. 112–130 (cit. on pp. 2, 217, 225).

Cournède, P.-H., Kang, M. Z., Mathieu, A., Barczi, J.-F., Yan, H. P., Hu, B. G., and de Reffye, P. H. (2006). Structural factorization of plants to compute their functional and architectural growth. In: *Simulation, Transactions of the Society for Modelling and Simulation International* 82(7): pp. 427–438. DOI: 10.1177/0037549706069341 (cit. on p. 225).

Crick, F. H. C. (1971). The scale of pattern formation. In: *The Symposia of the Society for Experimental Biology 25*: pp. 429–438 (cit. on p. 9).

Da Vinci, L. (1508). Notebook ('The Codex Arundel') (cit. on pp. 41, 43).

De Reffye, P. (1979). Modélisation de l'architecture des arbres par des processus stochastiques: simulation spatiale des modèles tropicaux sous l'effet de la pesanteur ; application au Coffea robusta. PhD thesis. Université de Paris-Sud, France (cit. on p. 15).

De Reffye, P. H., Edelin, C., Françon, J., Jaeger, M., and Puech, C. (1988). Plant models faithful to botanical structure and development. In: *Proceedings of the 15th Annual Conference on Computer Graphics and Interactive Techniques*. Ed. by Dill, J. Vol. 22. SIGGRAPH '88. New York, NY, USA: ACM: pp. 151–158. DOI: 10.1145/54852.378505 (cit. on pp. 15–17, 61).

De Reffye, P. H., Heuvelink, E., Barthélémy, D., and Cournède, P.-H. (2008). Plant Growth Models. In: *Encyclopedia of Ecology*. Ed. by Jørgensen, S. E. and Fath, B. D. Oxford: Academic Press: pp. 2824–2837. DOI: 10.1016/B978-008045405-4.00217-2 (cit. on pp. 56, 57).

De Swaef, T., Hanssens, J., Cornelis, A., and Steppe, K. (2013). Non-destructive estimation of root pressure using sap flow, stem diameter measurements and mechanistic modelling. In: *Annals of Botany* 111(2): pp. 271–282. DOI: 10.1093/aob/mcs249 (cit. on p. 8).

De Visser, P. H. B., van der Heijden, G., and Buck-Sorlin, G. H. (2014). Optimizing illumination in the greenhouse using a 3D model of tomato and a ray tracer. In: *Frontiers in Plant Science* 5(48). DOI: 10.3389/fpls.2014.00048 (cit. on pp. 65, 75, 219).

De Wit, C. T. (1958). Transpiration and crop yields. Vol. 64. Institute for biological and chemical research on field crops and herbage 6. Wageningen, The Netherlands: Verslagen van Landbouwkundige Onderzoekingen (cit. on p. 58).

Deussen, O. (2003). Computergenerierte Pflanzen: Technik und Design digitaler Pflanzenwelten. German. Springer Berlin. DOI: 10.1007/978-3-642-55822-1 (cit. on p. 209).

Deussen, O. and Lintermann, B. (1997). A modelling method and user interface for creating plants. In: *Proceedings of the Conference on Graphics Interface '97*. Kelowna, British Columbia, Canada: Canadian Information Processing Society: pp. 189–197 (cit. on p. 64).

Deussen, O. and Lintermann, B. (2005). Digital Design of Nature - Computer Generated Plants and Organics. Springer, Berlin (cit. on p. 231).

Deussen, O. and Strothotte, T. (2000). Computer-generated pen-and-ink illustration of trees. In: *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '00. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co.: pp. 13–18. DOI: 10.1145/344779.344792 (cit. on pp. 56, 62).

Digiuni, S., Schellmann, S., Geier, F., Greese, B., Pesch, M., Wester, K., Dartan, B., Mach, V., Srinivas, B. P., Timmer, J., Fleck, C., and Hulskamp, M. (2008). A competitive complex formation mechanism underlies trichome patterning on Arabidopsis leaves. In: *Molecular Systems Biology* 4(1): p. 217. DOI: 10.1038/msb.2008.54 (cit. on p. 9).

Disney, M., Lewis, P., and Saich, P. (2006). 3D modelling of forest canopy structure for remote sensing simulations in the optical and microwave domains. In: *Remote Sensing of Environment* 100(1): pp. 114–132. DOI: 10.1016/j.rse.2005.10.003 (cit. on p. 63).

Drouet, J. L. and Pagès, L. (2007). GRAAL: Growth, architecture, allocation: A functional-structural model to analyse the interactions between growth and assimilates allocation integrating processes from organ to whole plant. In: ed. by Vos, J., Marcelis, L. F. M., de Visser, P. H. B., Struik, P. C., and Evers, J. B. Functional-Structural Plant Modelling in Crop Production. Wageningen UR Frontis Series. Chap. 14: pp. 165–174 (cit. on p. 226).

Dufour-Kowalski, S., Bassette, C., and Bussière, F. (2007). A software for the simulation of rainfall distribution on 3-d plant architecture: PyDrop. In: *Proceedings of the 5th International Workshop on Functional-Structural Plant Models*. Ed. by Prusinkiewicz, P., Hanan, J., and Lane, B. Napier, New Zealand: p. 48 (cit. on p. 221).

Duncan, W., Loomis, R., Williams, W., and Hanau, R. (1967). A model for simulating photosynthesis in plant communities. In: *Hilgardia* 38(4): pp. 181–205. DOI: 10.3733/hilg.v38n04p181 (cit. on p. 59).

Eden, M. (1961). A two-dimensional growth process. In: *Proceedings of the 4th Berkeley Symposium on Mathematical Statistics and Probability, Volume 4: Contributions to Biology and Problems of Medicine*. Berkeley, California: University of California Press: pp. 223–239 (cit. on p. 9).

Eloy, C. (2011). Leonardo's rule, self-similarity, and wind-induced stresses in trees. In: *Physical Review Letters* 107 (25): p. 258101. DOI: 10.1103/PhysRevLett.107.258101 (cit. on p. 44).

Evers, J. B., Vos, J., Yin, X., Romero, P., van der Putten, P. E. L., and Struik, P. C. (2010). Simulation of wheat growth and development based on organ-level photosynthesis and assimilate allocation. In: *Journal of Experimental Botany* 61(8): pp. 2203–2216. DOI: 10.1093/jxb/erq025 (cit. on pp. 65, 82).

Farquhar, G. D., Caemmerer, S., and Berry, J. A. (1980). A biochemical model of photosynthetic CO2 assimilation in leaves of C3 species. In: *Planta* 149(1): pp. 78–90. DOI: 10.1007/BF00386231 (cit. on p. 242).

Feliciangeli, H. and Herman, G. T. (1973). Algorithms for producing grammars from sample derivations: a common problem of formal language theory and developmental biology. In: *Journal of Computer and System Sciences* 7(1): pp. 97–118. DOI: 10.1016/S0022-0000(73)80051-0 (cit. on p. 34).

Feng, F. J., Li, M. J., and Ma, F. W. (2014). The effects of bagging and debagging on external fruit quality, metabolites, and the expression of anthocyanin biosynthetic genes in 'Jonagold' apple (*Malus domestica* Borkh.) In: *Scientia Horticulturae* 165: pp. 123–131 (cit. on p. 211).

Fernandez, R., Das, P., Mirabet, V., Moscardi, E., Traas, J., Verdeil, J.-L., Malandain, G., and Godin, C. (2010). Imaging plant growth in 4D: Robust tissue reconstruction and lineaging at cell resolution. In: *Nature Methods* 7(7): pp. 547–553. DOI: 10.1038/nmeth.1472 (cit. on p. 8).

Ferraro, P., Godin, C., and Prusinkiewicz, P. (2005). Toward a quantification of self-similarity in plants. In: *Fractals* 13(02): pp. 91–109. DOI: 10.1142/S0218348X05002805 (cit. on p. 18).

Fisher, J. B. and Honda, H. (1977). Computer simulation of branching pattern and geometry in Terminalia (Combretaceae) a tropical tree. In: *Botanical Gazette* 138(4): pp. 377–384 (cit. on p. 14).

Fisher, J. B. and Honda, H. (1979). Branch geometry and effective leaf area: A study of Terminalia-branching pattern. 1 Theoretical ideas. In: *American Journal of Botany* 66(6): pp. 633–644 (cit. on p. 14).

Fishman, S. and Génard, M. (1998). A biophysical model of fruit growth: Simulation of seasonal and diurnal dynamics of mass. In: *Plant, Cell and Environment* 21(8): pp. 739–752. DOI: 10.1046/j.1365-3040.1998.00322.x (cit. on p. 63).

Fourcaud, T., Zhang, X., Stokes, A., Lambers, H., and Körner, C. (2008). Plant growth modelling and applications: The increasing importance of plant architecture in growth models. In: *Annals of Botany* 101(8): pp. 1053–1063. DOI: 10.1093/aob/mcn050 (cit. on pp. 3, 59, 82).

Fournier, C. and Andrieu, B. (1998). A 3D architectural and process-based model of maize development. In: *Annals of Botany* 81(2): pp. 233–250. DOI: 10.1006/anbo.1997.0549 (cit. on p. 32).

Frijters, D. and Lindenmayer, A. (1974). A model for the growth and flowering of *Aster novae-angliae* on the basis of table (1,0) L-systems. In: *L-systems, Lecture Notes in Computer Science 15*. Ed. by Rozenberg, G. and Salomaa, A. Springer-Verlag, Berlin: pp. 24–52 (cit. on p. 20).

Gács, P. (2001). Reliable cellular automata with self-organization. In: *Journal of Statistical Physics* 103(1/2): pp. 45–267. DOI: 10.1023/A:1004823720305 (cit. on p. 13).

Gallagher, J. N. (1979). Field studies of cereal leaf growth. In: *Journal of Experimental Botany* 30(4): pp. 625–636. DOI: 10.1093/jxb/30.4.625 (cit. on p. 205).

Gamma, E., Helm, R., Johnson, R., and Vlissides, J. (1995). Design Patterns. Elements of Reusable Object-Oriented Software. Addison-Wesley (cit. on pp. 72, 214, 218).

Gardner, M. (1970). Mathematical games - The fantastic combinations of John Conway's new solitaire game "life". In: *Scientific American* 223(4): pp. 120–123 (cit. on p. 13).

Gardner, M. (1983). The Game of Life, Parts I-III. In: *Wheels, Life, and Other Mathematical Amusements*. Ed. by Freeman, W. H. New York, NY, USA: pp. 20–22 (cit. on p. 13).

Garin, G., Fournier, C., Andrieu, B., Houlès, V., Robert, C., and Pradal, C. (2014). A modelling framework to simulate foliar fungal epidemics using functional–structural plant models. In: *Annals of Botany* 114: pp. 795–812. DOI: 10.1093/aob/mcu101 (cit. on p. 221).

Gervautz, M. and Traxler, C. (1996). Representation and realistic rendering of natural phenomena with cyclic CSG graphs. In: *The Visual Computer* 12(2): pp. 62–74. DOI: 10.1007/BF01782105 (cit. on p. 53).

Gierer, A. and Meinhardt, H. (1972). A theory of biological pattern formation. In: *Kybernetik* 12(1): pp. 30–39. DOI: 10.1007/BF00289234 (cit. on p. 9).

Godin, C. (2000). Representing and encoding plant architecture: A review. In: *Annals of Forest Science* 57: pp. 413–438. DOI: 10.1051/forest:2000132 (cit. on p. 81).

Godin, C. and Sinoquet, H. (2005). Functional-structural plant modelling. In: *New Phytologist* 166(3): pp. 705–708. DOI: 10.1111/j.1469-8137.2005.01445.x (cit. on p. 1).

Gopalan, G. (2000). An interactive image analysis sytem for dendrochronology. MA thesis. The University of Arizona (cit. on p. 207).

Goudriaan, J. and van Laar, H. H. (1994). Modelling Potential Crop Growth Processes. Kluwer Academic Publishers, Dordrecht, The Netherlands (cit. on p. 240).

Grasman, J. and van Straten, G., eds. (2012). Predictability and Nonlinear Modelling in Natural Sciences and Economics. Springer Science & Business Media, B. V. (cit. on p. 248).

Greene, N. (1989). Voxel space automata: Modeling with stochastic growth processes in voxel space. In: *Proceedings of the 16th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '89. New York, NY, USA: ACM: pp. 175–184. DOI: 10.1145/74333.74351 (cit. on p. 50).

Greenworks (2017). Xfrog modelling software (cit. on pp. 62, 231).

Gregory, F. G. (1921). Studies in the energy relations of plants. I. The increase in area of leaves and leaf surface of *Cucumis sativus*. In: *Annals of Botany* 35(1): pp. 93–123 (cit. on p. 205).

Gribbin, J. (2005). Deep Simplicity: Bringing Order to Chaos and Complexity. Random House, Inc. NY (cit. on p. 9).

Grieneisen, V. A. and Scheres, B. (2009). Back to the future: Evolution of computational models in plant morphogenesis. In: *Current Opinion in Plant Biology* 12(5). Cell signalling and gene regulation – Edited by Lohmann, J. and Nemhauser, J.: pp. 606–614. DOI: 10.1016/j.pbi.2009.07.008 (cit. on p. 8).

Griffon, S. and de Coligny, F. (2012). AMAPstudio: A software suite for plants architecture modelling. In: *4th International Symposium on Plant Growth Modeling, Simulation, Visualization and Applications (PMA12), 2012*: pp. 141–147 (cit. on p. 224).

Grimm, V. and Railsback, S. F. (2005). Individual-based modeling and ecology. Princeton series in theoretical and computational biology. Princeton University Press (cit. on pp. 2, 66, 217, 218).

Grimm, V., Berger, U., Bastiansen, F., Eliassen, S., Ginot, V., Giske, J., Goss-Custard, J., Grand, T., Heinz, S. K., Huse, G., Huth, A., Jepsen, J. U., Jørgensen, C., Mooij, W. M., Müller, B., Pe'er, G., Piou, C., Railsback, S. F., Robbins, A. M., Robbins, M. M., Rossmanith, E., Rüger, N., Strand, E., Souissi, S., Stillman, R. A., Vabø, R., Visser, U., and DeAngelis, D. L. (2006). A standard protocol for describing individual-based and agent-based models. In: *Ecological Modelling* 198(1-2): pp. 115–126. DOI: 10.1016/j.ecolmodel.2006.04.023 (cit. on p. 218).

Grimm, V., Berger, U., DeAngelis, D. L., Polhill, J. G., Giske, J., and Railsback, S. F. (2010). The {ODD} protocol: A review and first update. In: *Ecological Modelling* 221(23): pp. 2760–2768. DOI: 10.1016/j.ecolmodel.2010.08.019 (cit. on p. 218).

Groer, C. (2006). Dynamisches 3D-Modell der Rapspflanze (*Brassica napus* L.) zur Bestimmung optimaler Ertragskomponenten bei unterschiedlicher Stickstoffdüngung. MA thesis. University of Technology at Cottbus, Institut für Informatik, Informations- und Medientechnik (cit. on p. 65).

Groer, C., Kniemeyer, O., Hemmerling, R., Kurth, W., Becker, H., and Buck-Sorlin, G. H. (2007). A dynamic 3D model of rape (*Brassica napus* L.) computing yield components under variable nitrogen fertilization regimes. In: *Proceedings of the 5th International Workshop on Functional-Structural Plant Models*. Ed. by Prusinkiewicz, P., Hanan, J., and Lane, B. Napier, New Zealand: pp. 4.1–4.3 (cit. on p. 65).

Gu, S. H., Evers, J. B., Zhan, L. Z., Mao, L. L., Zhang, S. P., Zhao, X. H., Liu, S. D., van der Werf, W., and Li, Z. H. (2014). Modelling the structural response of cotton plants to mepiquat chloride and population density. In: *Annals of Botany* 114(4): pp. 877–887. DOI: 10.1093/aob/mct309 (cit. on p. 220).

Guay, R. (2012). WinDENDRO 2012 User's Guide, Regent Instruments, Inc., Quebec, Canada (cit. on p. 207).

Hallé, F. (1971). Architecture and growth of tropical trees exemplified by the Euphorbiaceae. In: *Biotropica* 3: pp. 56–62 (cit. on p. 60).

Hallé, F., Oldemann, R. A. A., and Tomlinson, P. B. (1978). Tropical trees and forests: An architectural analysis. Springer-Verlag, Berlin, Heidelberg, New York: p. 441 (cit. on pp. 44, 60, 61).

Hallé, F. (1974). Architecture of trees in the rain forest of morobe district, New Guinea. In: *Biotropica* 6(1): pp. 43–50. DOI: 10.2307/2989696 (cit. on p. 60).

Hallé, F. and Oldeman, R. A. A. (1970). Essai sur l'architecture et la dynamique de croissance des arbres tropicaux. Masson (cit. on p. 60).

Hanan, J. and Room, P. (2002). Floradig User Manual. CPAI, University of Queensland, Brisbane (cit. on p. 226).

Harrison, L. G. (1994). Kinetic theory of living pattern. In: *Endeavour* 18(4): pp. 130–136. DOI: 10.1016/0160-9327(95)90520-5 (cit. on p. 9).

Hemmerling, R., Kniemeyer, O., Lanwert, D., Kurth, W., and Buck-Sorlin, G. H. (2008). The rule-based language XL and the modelling environment GroIMP illustrated with simulated tree competition. In: *Functional Plant Biology* 35(9/10): pp. 739–750. DOI: 10.1071/FP08052 (cit. on pp. 65, 227).

Henke, M. and Buck-Sorlin, G. H. (accepted). Using a full spectral raytracer for the modelling of light microclimate in a functional-structural plant model. In: *Computing and Informatics* (cit. on pp. 6, 210).

Henke, M., Huckemann, S., Kurth, W., and Sloboda, B. (2014a). Reconstructing leaf growth based on non-destructive digitizing and low-parametric shape evolution for plant modelling over a growth cycle. In: *Silva Fennica* 48(2). DOI: 10.14214/sf.1019 (cit. on pp. 5, 78, 205, 206).

Henke, M., Kniemeyer, O., and Kurth, W. (2017). Realization and extension of the Xfrog approach for plant modelling in the graph-grammar based language XL. In: *Computing and Informatics* 36(1): pp. 33–54. DOI: 10.4149/cai_2017_1_33 (cit. on pp. 5, 208).

Henke, M., Sarlikioti, V., Kurth, W., Buck-Sorlin, G., and Pagès, L. (2014b). Exploring root developmental plasticity to nitrogen with a three-dimensional architectural model. In: *Plant and Soil* 385(1-2): pp. 49–62. DOI: 10.1007/s11104-014-2221-7 (cit. on pp. 6, 210, 219).

Henke, M. and Sloboda, B. (2014). Semiautomatic tree ring segmentation using Active Contours and an optimised gradient operator. In: *Forestry Journal* 60(3): pp. 185–190 (cit. on pp. 5, 206, 207).

Henke, M., Kurth, W., and Buck-Sorlin, G. H. (2016). FSPM-P: Towards a general functional-structural plant model for robust and comprehensive model development. In: *Frontiers of*

Computer Science 10(6): pp. 1103–1117. DOI: 10.1007/s11704-015-4472-8 (cit. on pp. 6, 65, 74, 76, 209).

Hogeweg, P. and Hesper, B. (1974). A model study on biomorphological description. In: *Pattern Recognition* 6(3): pp. 165–179. DOI: 10.1016/0031-3203(74)90019-3 (cit. on p. 20).

Holton, M. (1994). Strands, gravity and botanical tree imagery. In: *Computer Graphics Forum* 13(1): pp. 57–67. DOI: 10.1111/1467-8659.1310057 (cit. on p. 44).

Honda, H., Tomlinson, P. B., and Fisher, J. B. (1982). Two geometrical models of branching of botanical trees. In: *Annals of Botany* 49(1): pp. 1–11 (cit. on p. 14).

Honda, H. (1971). Description of the form of trees by the parameters of the tree-like body: Effects of the branching angle and the branch length on the shape of the tree-like body. In: *Journal of Theoretical Biology* 31(2): pp. 331–338. DOI: 10.1016/0022-5193(71)90191-3 (cit. on pp. 8, 14, 40).

Honda, H. and Hatta, H. (2004). Branching models consisting of two principles: phyllotaxis and effect of gravity. In: *Forma* 19(3): pp. 183–196 (cit. on p. 15).

Honda, H., Tomlinson, P. B., and Fisher, J. B. (1981). Computer simulation of branch interaction and regulation by unequal flow rates in botanical trees. In: *American Journal of Botany* 68(4): pp. 569–585 (cit. on p. 14).

Hu, B. G., de Reffye, P., Zhao, X., Yan, H. P., and Kang, M. Z. (2003). GreenLab: A new methodology towards plant functional-structural model - Structural aspect. In: *International Symposium on Plant Growth Modeling, Simulation, Visualization and Applications (PMA03), 2003*. Ed. by Hu, B. G. and Jaeger, M. Beijing, PR China: Tsinghua University Press and Springer: pp. 21–35 (cit. on pp. 214, 226).

Huber, B. (1928). Weitere quantitative Untersuchungen über das Wasserleitungssystem der Pflanzen. Jahrbücher für wissenschaftliche Botanik (cit. on pp. 45, 46, 239).

Hutchinson, J. E. (1981). Fractals and self similarity. In: *Indiana University Mathematical Journal* 30(5): pp. 713–747 (cit. on pp. 46, 239).

Huwe, T. and Hemmerling, R. (2010). Stochastic path tracing on consumer graphics cards. In: *Proceedings of the 24th Spring Conference on Computer Graphics (SCCG08), 2008*. Budmerice, Slovakia: ACM: pp. 105–111. DOI: 10.1145/1921264.1921287 (cit. on p. 82).

Iovan, C., Cournède, P. H., Guyard, T., Bayol, B., Boldo, D., and Cord, M. (2014). Model-based analysis 2013; Synthesis for realistic tree reconstruction and growth simulation. In: *IEEE Transactions on Geoscience and Remote Sensing* 52(2): pp. 1438–1450. DOI: 10.1109/TGRS.2013.2251467 (cit. on p. 63).

Jones, J. W., Hesketh, J. D., Kamprath, E. J., and Bowen, H. D. (1974). Development of a nitrogen balance for cotton growth models: A first approximation. In: *Crop Science* 14(5): pp. 541–546. DOI: 10.2135/cropsci1974.0011183X001400040014x (cit. on p. 59).

Jönsson, H. and Krupinski, P. (2010). Modeling plant growth and pattern formation. In: *Current Opinion in Plant Biology* 13(1): pp. 5–11. DOI: 10.1016/j.pbi.2009.10.002 (cit. on p. 8).

Kang, M. Z., Cournède, P. H., Mathieu, A., Letort, V., Qi, R., and Zhan, Z. G. (2008). A functional-structural plant model-theory and applications in agronomy. In: *International Symposium on Crop Modeling and Decision Support: ISCMDS 2008*. Nanjing, China (cit. on p. 220).

Kang, M. Z., Hua, J., Hu, B. G., and de Reffye, P. (2010). QingYuan–a GreenLab based plant simulator and solver. In: *Proceedings of the 6th International Workshop on Functional-Structural Plant Models*. University of California, Davis, CA, USA (cit. on p. 227).

Kang, M. Z., Qi, R., de Reffye, P., and Hu, B. G. (2006). GreenScilab: A toolbox simulating plant growth in the Scilab environment. In: *8th Middle Eastern Multiconference on Simulation and Modelling (MESM06)*. Alexandria, Egypt: pp. 174–178 (cit. on p. 226).

Kang, S. B. and Quan, L. (2009). Image-Based Modeling of Plants and Trees. Morgan & Claypool. DOI: 10.2200/S00205ED1V01Y200911COV001 (cit. on p. 54).

Kawaguchi, Y. (1982). A morphological study of the form of nature. In: *Proceedings of the 9th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '82. Boston, Massachusetts, USA: ACM: pp. 223–232. DOI: 10.1145/800064.801284 (cit. on p. 15).

Kniemeyer, O. (2004). Rule-based modelling with the XL/GroIMP software. In: *The Logic of Artificial Life. Proceedings of 6th GWAL*. Ed. by Schaub, H., Detje, F., and Brüggemann, U. Bamberg, Germany: AKA Akademische Verlagsges Berlin: pp. 56–65 (cit. on pp. 36, 37, 245).

Kniemeyer, O. (2008). Design and implementation of a graph grammar based language for functional-structural plant modelling. PhD thesis. BTU Cottbus (cit. on pp. 33, 36, 37, 227, 245).

Kniemeyer, O., Barczik, G., Hemmerling, R., and Kurth, W. (2008). Relational growth grammars - A parallel graph transformation approach with applications in biology and architecture. In: ed. by Schürr, A., Nagl, M., and Zündorf, A. Kassel, Germany: Springer Berlin Heidelberg. Chap. Applications of Graph Transformations with Industrial Relevance: Third International Symposium, AGTIVE'07, Revised Selected and Invited Papers: pp. 152–167. DOI: 10.1007/978-3-540-89020-1_12 (cit. on p. 82).

Kniemeyer, O., Buck-Sorlin, G. H., and Kurth, W. (2004). A graph-grammar approach to Artificial Life. In: *Artificial Life* 10: pp. 413–431 (cit. on pp. 36, 245).

Kniemeyer, O., Buck-Sorlin, G. H., and Kurth, W. (2007). GroIMP as a platform for functional-stuctural modelling of plants. In: *Functional-Structural Plant Modelling in Crop Production. Proceedings of a workshop held in Wageningen (NL), 5.-8. 3. 2006*. Ed. by Vos, J., Marcelis, L. F. M., de Visser, P. H. B., Struik, P. C., and Evers, J. B. Springer, Dordrecht: pp. 43–52 (cit. on pp. 36, 245).

Kniemeyer, O. and Kurth, W. (2008). The modelling platform GroIMP and the programming language XL. In: ed. by Schürr, A., Nagl, M., and Zündorf, A. Kassel, Germany: Springer Berlin Heidelberg. Chap. Applications of Graph Transformations with Industrial Relevance: Third International Symposium, AGTIVE'07, Revised Selected and Invited Papers: pp. 570–572. DOI: 10.1007/978-3-540-89020-1_39 (cit. on pp. 227, 228).

Koch, H. (1906). Une méthode géométrique élémentaire pour l'étude de certaines questions de la théorie des courbes planes. In: *Acta Mathematica* 30: pp. 145–174 (cit. on p. 22).

Kolb, A., Latta, L., and Rezk-Salama, C. (2004). Hardware-based simulation and collision detection for large particle systems. In: *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS Conference on Graphics Hardware*. HWWS '04. Grenoble, France: ACM: pp. 123–131. DOI: 10.1145/1058129.1058147 (cit. on p. 40).

Krieger, K. (2011). Leonardo's formula explains why trees don't splinter. In: *ScienceNOW* (cit. on p. 43).

Kruger, J., Kipfer, P., Konclratieva, P., and Westermann, R. (2005). A particle system for interactive visualization of 3D flows. In: *IEEE Transactions on Visualization and Computer Graphics; Joint EUROGRAPHICS-IEEE TCVG Symposium on Visualization* 11(6): pp. 744–756. DOI: 10.1109/TVCG.2005.87 (cit. on p. 40).

Kurth, W. (1994a). Growth Grammar Interpreter GROGRA 2.4 : A software tool for the 3-dimensional interpretation of stochastic, sensitive growth grammars in the context of plant modelling. Introduction and Reference Manual. Berichte des Forschungszentrums Waldökosysteme der Universität Göttingen Ser. B, 38. Göttingen (cit. on pp. 31, 32, 34, 227).

Kurth, W. (1994b). Morphological models of plant growth: Possibilities and ecological relevance. In: *ISEM's 8th International Conference on the State-of-the-Art in Ecological Modelling*. Vol. 75/76: pp. 299–308. DOI: 10.1016/0304-3800(94)90027-2 (cit. on pp. 55, 56).

Kurth, W. (1994c). Web page of GROGRA (cit. on pp. 34, 227).

Kurth, W. (1999). Die Simulation der Baumarchitektur mit Wachstumsgrammatiken: stochastische, sensitive L-Systeme als formale Basis für dynamische, morphologische Modelle der Verzweigungsstruktur von Gehölzen. Wissenschaftlicher Verlag Berlin: p. 327 (cit. on pp. 33, 56).

Kurth, W., Kniemeyer, O., and Buck-Sorlin, G. H. (2004). Relational growth grammars - A graph rewriting approach to dynamical systems with a dynamical structure. In: *Unconventional Programming Paradigms*. Ed. by Banâtre, J.-P., Fradet, P., Giavitto, J.-L., and Michel, O. Vol. 3566. Lecture Notes in Computer Science. Springer-Verlag Berlin Heidelberg: pp. 56–72. DOI: 10.1007/11527800_5 (cit. on pp. 36, 37).

Lanwert, D. (2007). Funktions-/Strukturorientierte Pflanzenmodellierung in E-Learning-Szenarien. PhD thesis. University of Göttingen (cit. on p. 82).

Letort, V., Mahe, P., Cournède, P. H., de Reffye, P., and Courtois, B. (2008). Quantitative genetics and functional–structural plant growth models: simulation of guantitative trait loci detection for model parameters and application to potential yield optimization. In: *Annals of Botany* 101: pp. 1243–1254. DOI: 10.1093/aob/mcm197 (cit. on p. 220).

Lindenmayer, A. (1968). Mathematical models for cellular interactions in development I. Filaments with one-sided inputs. In: *Journal of Theoretical Biology* 18(3): pp. 280–299. DOI: 10.1016/0022-5193(68)90079-9 (cit. on p. 18).

Lintermann, B. and Deussen, O. (1999). Interactive modeling of plants. In: *IEEE Computer Graphics and Applications* 19(1): pp. 56–65. DOI: 10.1109/38.736469 (cit. on p. 64).

Mandelbrot, B. B. (1977). Fractals: Form, Chance and Dimension. San Francisco: W. H. Freeman & Co Ltd. (cit. on p. 40).

Mandelbrot, B. B. (1982). The Fractal Geometry of Nature. W. H. Freeman & Co Ltd. (cit. on p. 40).

McMaster, G. S. and Hargreaves, J. N. G. (2009). CANON in D(esign): Composing scales of plant canopies from phytomers to whole-plants using the composite design pattern. In: *NJAS - Wageningen Journal of Life Sciences* 57(1). Recent Advances in Crop Growth Modelling: pp. 39–51. DOI: 10.1016/j.njas.2009.07.008 (cit. on p. 214).

Meinhardt, H. (1982). Models of Biological Pattern Formation. Academic Press, London, UK (cit. on p. 9).

Meinhardt, H. (2009). The Algorithmic Beauty of Sea Shells. 4th ed. Springer, Heidelberg, New York (cit. on p. 9).

Minamino, R. and Tateno, M. (2014). Tree branching: Leonardo da Vinci's rule versus biomechanical models. In: *PLoS ONE* 9(4). DOI: 10.1371/journal.pone.0093535 (cit. on p. 44).

Murray, C. D. (1927). A relationship between circumference and weight in trees and its bearing on branching angles. In: *The Journal of General Physiology* 10(5): pp. 725–729. DOI: 10.1085/jgp.10.5.725 (cit. on pp. 42, 43, 45).

Murray, C. D. (1926). The physiological principle of minimum work applied to the angle of branching of arteries. In: *The Journal of General Physiology* 9(6): pp. 835–841. DOI: 10.1085/jgp.9.6.835 (cit. on p. 43).

Nagel, J., Duda, H., and Hansen, J. (2006). Forest simulator BWINPro7. In: *Forst und Holz* 61: pp. 427–429 (cit. on p. 63).

Neto, J. C., Meyer, G. E., Jones, D. D., and Samal, A. K. (2006). Plant species identification using elliptic Fourier leaf shape analysis. In: *Computers and Electronics in Agriculture* 50(2): pp. 121–134. DOI: 10.1016/j.compag.2005.09.004 (cit. on p. 205).

Norell, K. (2011). Automatic counting of annual rings on *Pinus sylvestris* end faces in sawmill industry. In: *Computers and Electronics in Agriculture* 75(2): pp. 231–237. DOI: 10.1016/j.compag.2010.11.005 (cit. on p. 207).

Olle, M. and Viršile, A. (2013). The effects of lIght-emitting diode lighting on greenhouse plant growth and quality. In: *Agricultural and Food Science* 22 (cit. on p. 220).

Ong, Y., Streit, K., Henke, M., and Kurth, W. (2014). An approach to multiscale modelling with graph grammars. In: *Annals of Botany* 114(4): pp. 813–827. DOI: 10.1093/aob/mcu155 (cit. on p. 38).

Oppenheimer, P. E. (1986). Real time design and animation of fractal plants and trees. In: *Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '86. New York, NY, USA: ACM: pp. 55–64. DOI: 10.1145/15922.15892 (cit. on pp. 40, 42).

Perttunen, J., Sievänen, R., and Nikinmaa, E. (1998). LIGNUM: A model combining the structure and the functioning of trees. In: *Ecological Modelling* 108(1-3): pp. 189–198. DOI: 10.1016/S0304-3800(98)00028-3 (cit. on p. 228).

Perttunen, J., Sievänen, R., Nikinmaa, E., Salminen, H., Saarenmaa, H., and Väkevä, J. (1996). LIGNUM: A tree model based on simple structural units. In: *Annals of Botany* 77(1): pp. 87–98. DOI: 10.1006/anbo.1996.0011 (cit. on p. 228).

Pradal, C., Boudon, F., Nouguier, C., Chopard, J., and Godin, C. (2009). PlantGL: A Python-based geometric library for 3D plant modelling at different scales. In: *Graphical Models* 71(1): pp. 1–21. DOI: 10.1016/j.gmod.2008.10.001 (cit. on pp. 33, 230).

Pradal, C., Dufour-Kowalski, S., Boudon, F., Fournier, C., and Godin, C. (2008). OpenAlea: A visual programming and component-based software platform for plant modeling. In: *Functional Plant Biology* 35: pp. 751–760 (cit. on pp. 214, 229).

Pretzsch, H. (2001). Modellierung des Waldwachstums. Blackwell Verlag Berlin (cit. on p. 63).

Prusinkiewicz, P. (1986). Graphical applications of L-systems. In: *Proceedings on Graphics Interface '86/Vision Interface '86*. Vancouver, British Columbia, Canada: Canadian Information Processing Society: pp. 247–253 (cit. on p. 33).

Prusinkiewicz, P. and Lindenmayer, A. (1990). The Algorithmic Beauty of Plants. http://algorithmicbotany.org/papers/abop/abop.pdf. Springer, New York (cit. on pp. 18, 19, 21, 26–32, 62).

Prusinkiewicz, P. and Remphrey, W. (2000). Characterization of architectural tree models using L-systems and Petri nets. In: *L'arbre - The Tree 2000: Papers presented at the 4th International Symposium on the Tree*. Ed. by Labrecque, M.: pp. 177–186 (cit. on p. 61).

Prusinkiewicz, P. (1987). Applications of L-systems to computer imagery. In: *Graph-Grammars and Their Application to Computer Science*. Ed. by Ehrig, H., Nagl, M., Rozenberg, G., and

Rosenfeld, A. Vol. 291. Lecture Notes in Computer Science. Springer Berlin Heidelberg: pp. 534–548. DOI: 10.1007/3-540-18771-5_74 (cit. on p. 24).

Prusinkiewicz, P. (1993). Modeling and visualization of biological structures. In: *In Proceeding of Graphics Interface*: pp. 128–137 (cit. on pp. 63, 64).

Prusinkiewicz, P. (2004). Modeling plant growth and development. In: *Current Opinion in Plant Biology* 7(1): pp. 79–83. DOI: 10.1016/j.pbi.2003.11.007 (cit. on pp. 8, 18).

Prusinkiewicz, P., Hanan, J., and Měch, R. (2000a). An L-system-based plant modeling language. In: *Proceedings of the international Workshop on Applications of Graph Transformations with Industrial Relevance: AGTIVE'99*. Ed. by Nagl, M., Schürr, A., and Münch, M. Vol. 1779. Lecture Notes in Computer Science. Kerkrade, The Netherlands: Springer Berlin, Heidelberg: pp. 395–410. DOI: 10.1007/3-540-45104-8_31 (cit. on pp. 32, 33, 224).

Prusinkiewicz, P., James, M., and Mech, R. (1994). Synthetic topiary. In: *Proceedings of the 21st annual conference on Computer graphics and interactive techniques SIGGRAPH '94*. Orlando, Florida, USA: ACM: pp. 351–358 (cit. on p. 64).

Prusinkiewicz, P., Karwowski, R., Měch, R., and Hanan, J. (2000b). L-Studio/cpfg: A software system for modeling plants. In: *Proceedings of the international Workshop on Applications of Graph Transformations with Industrial Relevance: AGTIVE'99*. Ed. by Nagl, M., Schürr, A., and Münch, M. Vol. 1779. Lecture Notes in Computer Science. Kerkrade, The Netherlands: Springer Berlin, Heidelberg: pp. 457–464. DOI: 10.1007/3-540-45104-8_38 (cit. on p. 224).

Quan, L., Tan, P., Zeng, G., Yuan, L., Wang, J., and Kang, S. B. (2006). Image-based plant modeling. In: *ACM Transactions on Graphics (TOG) - Proceedings of ACM SIGGRAPH 2006*. SIGGRAPH '06. Boston, Massachusetts: ACM: pp. 599–604. DOI: 10.1145/1179352.1141929 (cit. on p. 54).

Rauscher, H. M., Isebrands, J. G., Host, G. E., Dickson, R. E., Dickmann, D. I., Crow, T. R., and Michael, D. A. (1990). ECOPHYS: An ecophysiological growth process model for juvenile poplar. In: *Tree Physiology* 7(1-2-3-4): pp. 255–281. DOI: 10.1093/treephys/7.1-2-3-4.255 (cit. on p. 225).

Reeves, W. T. (1983). Particle systems: A technique for modeling a class of fuzzy objects. In: *Transactions on Graphics* 2(2): pp. 91–108. DOI: 10.1145/357318.357320 (cit. on p. 38).

Reeves, W. T. and Blau, R. (1985). Approximate and probabilistic algorithms for shading and rendering structured particle systems. In: *Proceedings of the 12th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '85. New York, NY, USA: ACM: pp. 313–322. DOI: 10.1145/325334.325250 (cit. on p. 38).

Reuter, L. H. (1987). Rendering and magnification of fractals using iterated function systems. PhD thesis. Georgia Institute of Technology, USA (cit. on p. 49).

Room, P., Hanan, J., and Prusinkiewicz, P. (1996). Virtual plants: new perspectives for ecologists, pathologists and agricultural scientists. In: *Trends in Plant Science* 1(1): pp. 33–38. DOI: 10.1016/S1360-1385(96)80021-5 (cit. on pp. 3, 8, 62).

Runions, A. (2008). Modeling biological patterns using the space colonization algorithm. MA thesis. University of Calgary, Canada (cit. on pp. 40, 41).

Runions, A., Lane, B., and Prusinkiewicz, P. (2007). Modeling trees with a space colonization algorithm. In: *Proceedings of the Third Eurographics Conference on Natural Phenomena*. NPH'07. Prague, Czech Republic: Eurographics Association: pp. 63–70. DOI: 10.2312/NPH/NPH07/063-070 (cit. on p. 40).

Sakaguchi, T. and Ohya, J. (1999). Modeling and animation of botanical trees for interactive virtual environments. In: *Proceedings of the ACM Symposium on Virtual Reality Software and Technology*. VRST '99. London, United Kingdom: ACM: pp. 139–146. DOI: 10.1145/323663.323685 (cit. on p. 54).

Saltelli, A. (2002). Sensitivity analysis for importance assessment. In: *Risk Analysis* 22(3): pp. 579–590. DOI: 10.1111/0272-4332.00040 (cit. on p. 246).

Sarlikioti, V., de Visser, P. H. B., Buck-Sorlin, G. H., and Marcelis, L. F. M. (2011). How plant architecture affects light absorption and photosynthesis in tomato: towards an ideotype for plant architecture using a functional–structural plant model. In: *Annals of Botany* 108(6): pp. 1065–1073. DOI: 10.1093/aob/mcr221 (cit. on pp. 80, 210).

Savage, N. S., Walker, T., Wieckowski, Y., Schiefelbein, J., Dolan, L., and Monk, N. A. M. (2008). A mutual support mechanism through intercellular movement of CAPRICE and GLABRA3 can pattern the *Arabidopsis* root epidermis. In: *PLOS (Public Library of Science) Biology* 6(9): pp. 1899–1909. DOI: 10.1371/journal.pbio.0060235 (cit. on p. 9).

Shebell, M. (1986). Modelling branching plants using attribute L-systems. MA thesis. Worcester Polytechnic Institute, USA (cit. on p. 61).

Shinozaki, K., Yoda, K., Hozumi, K., and Kira, T. (1964). A quantitative analysis of plant form - The pipe model theory I. Basic analyses. In: *Japanese Journal of Ecology* 14(3): pp. 97–105 (cit. on p. 44).

Shlyakhter, I., Rozenoer, M., Dorsey, J., and Teller, S. (2001). Reconstructing 3D tree models from instrumented photographs. In: *IEEE Computer Graphics and Applications* 21(3): pp. 53–61. DOI: 10.1109/38.920627 (cit. on p. 54).

Sierpiński, W. (1915). Sur une courbe dont tout point est un point de ramification. In: *Comptes rendus de l'Académie des sciences, Paris* 160: pp. 302–305 (cit. on p. 47).

Sievänen, R., Godin, C., DeJong, T. M., and Nikinmaa, E. (2014). Functional-structural plant models: A growing paradigm for plant studies. In: *Annals of Botany* 114(4): pp. 599–603. DOI: 10.1093/aob/mcu175 (cit. on p. 1).

Smith, A. R. (1984). Plants, fractals, and formal languages. In: *Proceedings of the 11th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '84. New York, NY, USA: ACM: pp. 1–10. DOI: 10.1145/800031.808571 (cit. on p. 18).

Smoleňová, K., Hemmerling, R., and Kurth, W. (2010). Towards the reconstruction of historical gardens and parks using the techniques of FSPM. In: *Proceedings of the 6th International Workshop on Functional-Structural Plant Models (FSPM10), 2010*. Ed. by DeJong, T. and Da Silva, D.: p. 273 (cit. on p. 82).

Smoleňová, K., Henke, M., and Kurth, W. (2012). Rule-Based Integration of GreenLab into GroIMP with GUI Aided Parameter Input. In: *IEEE 4th International Symposium on Plant Growth Modeling, Simulation, Visualization and Applications (PMA12), 2012*. Ed. by Kang, M., Dumont, Y., and Y., G. Shanghai (China): IEEE: pp. 347–354 (cit. on p. 227).

Sobel, I. and Feldman, G. (1968). Isotropic 3x3 image gradient operator. In: *Stanford Artificial Intelligence Project (SAIL)* (cit. on p. 207).

Soille, P. and Misson, L. (2001). Tree ring area measurements using morphological image analysis. In: *Canadian Journal of Forest Research* 31(6): pp. 1074–1083. DOI: 10.1139/x01-025 (cit. on p. 207).

Tan, P., Zeng, G., Wang, J., Kang, S. B., and Quan, L. (2007). Image-based tree modeling. In: *ACM Transactions on Graphics (TOG) - Proceedings of ACM SIGGRAPH 2007*. SIGGRAPH '07. San Diego, California: ACM. DOI: 10.1145/1275808.1276486 (cit. on p. 54).

Thompson, D. W. (1917). On growth and form. Cambridge University Press (cit. on p. 7).

Thornley, J., Hand, D., and Wilson, J. (1992). Modelling light absorption and canopy net photosynthesis of glasshouse row crops and application to cucumber. In: *Journal of Experimental Botany* 43(3): pp. 383–391. DOI: 10.1093/jxb/43.3.383 (cit. on p. 57).

Thue, A. (1914). Probleme über Veränderungen von Zeichenreihen nach gegebenen Regeln. In: *Skrifter utgit av Videnskapsselskapet i Kristiania, I Mathemastisk-naturvidenskablig klasse* 34(10) (cit. on p. 18).

Turing, A. M. (1952). The chemical basis of morphogenesis. In: *Philosophical Transactions of the Royal Society of London B: Biological Sciences* 237(641): pp. 37–72. DOI: 10.1098/rstb.1952.0012 (cit. on p. 9).

Ulam, S. (1962). On some mathematical properties connected with patterns of growth of figures. In: *Proceedings of Symposia on Applied Mathematics*. Vol. 14: pp. 215–224 (cit. on p. 40).

Ulam, S. (1966). Module, Proportion, Symmetry, Rhythm. (Vision and Value series). In: ed. by Kepes, G. George Braziller, New York. Chap. Pattern of growth of figures: Mathematical aspects (cit. on pp. 7, 10, 12).

Van Ittersum, M. K. and Donatelli, M. (2003). Modelling cropping systems – highlights of the symposium and preface to the special issues. In: *European Journal of Agronomy* 18(3-4).

Modelling Cropping Systems: Science, Software and Applications: pp. 187–197. DOI: 10.1016/S1161-0301(02)00095-3 (cit. on p. 218).

Van Waveren, R., Groot, S., Scholten, H., van Geer, F., Wosten, H., Koeze, R., and Noort, J. (1999). Good modelling practice handbook. STOWA report 99-05. Utrecht, The Netherlands: Dutch Deptment of Public Works, Institute for Inland Water Management and Waste Water Treatment, report 99.036 (cit. on p. 2).

Von Neumann, J. (1966). Theory of Self-Reproducing Automata. Ed. by Burks, A. W. Champaign, IL, USA: University of Illinois Press (cit. on p. 10).

Vos, J., Evers, J. B., Buck-Sorlin, G. H., Andrieu, B., Chelle, M., and de Visser, P. H. B. (2010). Functional–structural plant modelling: A new versatile tool in crop science. In: *Journal of Experimental Botany* 61(8): pp. 2101–2115. DOI: 10.1093/jxb/erp345 (cit. on pp. 1, 3, 59, 82, 215).

Vos, J., Marcelis, L. F. M., de Visser, P. H. B., Struik, P. C., and Evers, J. B., eds. (2007). Functional-Structural Plant Modelling in Crop Production. Vol. 22. Wageningen UR Frontis Series. Springer Netherlands (cit. on pp. 1, 3, 59, 82).

Wernecke, P., Müller, J., Dornbusch, T., Wernecke, A., and Diepenbrock, W. (2007). The virtual crop-modelling system 'VICA' specified for barley. In: ed. by Vos, J., Marcelis, L. F. M., de Visser, P. H. B., Struik, P. C., and Evers, J. B. Functional-Structural Plant Modelling in Crop Production. Wageningen UR Frontis Series. Chap. 5: pp. 53–64 (cit. on p. 230).

Wiechers, D., Kahlen, K., and Stützel, H. (2011). Evaluation of a radiosity based light model for greenhouse cucumber canopies. In: *Agricultural and Forest Meteorology* 151(7): pp. 906–915. DOI: 10.1016/j.agrformet.2011.02.016 (cit. on p. 206).

Wilson, G. V. (2006). Where's the real bottleneck in scientific computing? In: *American Scientist* 94(1): pp. 5–6. DOI: 10.1511/2006.1.5 (cit. on p. 215).

Wolfram, S. (1983). Statistical Mechanics of Cellular Automata. In: *Reviews of Modern Physics* 55: pp. 601–644 (cit. on pp. 10, 11).

Wolfram, S. (1984). Cellular automata as models of complexity. In: *Nature* 311: pp. 419–424. DOI: 10.1038/311419a0 (cit. on p. 11).

Wolfram, S. (2002). A New Kind of Science. Wolfram Media (cit. on pp. 10–12).

Xu, L. F., Henke, M., Zhu, J., Kurth, W., and Buck-Sorlin, G. H. (2011). A functional-structural model of rice linking quantitative genetic information with morphological development and physiological processes. In: *Annals of Botany* 107(5): pp. 817–828. DOI: 10.1093/aob/mcq264 (cit. on pp. 65, 220).

Xu, L. F., Henke, M., Zhu, J., Kurth, W., and Buck-Sorlin, G. H. (2010). A rule-based functional-structural model of rice considering source and sink functions. In: *Third International Symposium on Plant Growth Modeling, Simulation, Visualization and Applications (PMA09), 2009.*

Ed. by Li, B., Jaeger, M., and Guo, Y. Los Alamitos 2010. Beijing, China: pp. 245–252. DOI: 10.1109/PMA.2009.36 (cit. on p. 65).

Yun, C.-H., Metzler, W., and M., B. (2008). Image compression predicated on recurrent iterated function systems. In: *2nd International Conference on Mathematics and Statistics*. Athens, Greece (cit. on p. 49).

Zhou, H., Feng, R., Huang, H. H., Lin, E. P., and Yu, J. L. (2012). Method of tree-ring image analysis for dendrochronology. In: *Optical Engineering* 51(7): pp. 1–7. DOI: 10.1117/1.OE.51.7.077202 (cit. on p. 207).

Zhu, J. (2015). Plant plasticity in Intercropping: mechanisms and consequences. PhD thesis. Wageningen University (cit. on p. 220).

Zimmermann, M. H. (1978). Hydraulic architecture of some diffuse-porous trees. In: *Canadian Journal of Botany* 56(18): pp. 2286–2295. DOI: 10.1139/b78-274 (cit. on p. 46).

total references: 224