



Simulation of flowering plants

Olga Petrenko

► To cite this version:

Olga Petrenko. Simulation of flowering plants. Modeling and Simulation. Université de Limoges; Universitat de Girona, 2014. English. NNT : 2014LIMO0067 . tel-01161784

HAL Id: tel-01161784

<https://theses.hal.science/tel-01161784>

Submitted on 9 Jun 2015

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

UNIVERSITE DE LIMOGES

ECOLE DOCTORALE « Sciences et Ingénierie pour l'Information »

FACULTÉ DES SCIENCES ET TECHNIQUES

Thèse

pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ DE LIMOGES

Discipline / Spécialité : Informatique et Applications

présentée et soutenue par

Olga PETRENKO

le 12/12/2014

Simulation des plantes à fleurs

Thèse dirigée par Mateu SBERT, Djamchid GHAZANFARPOUR

JURY :

Président du jury

M. Miguel CHOVER Sellés, Professeur, INIT, Université Jaume I de Castelló

Rapporteurs

M. Jean-Pierre JESSEL, Professeur, IRIT, Université Paul Sabatier

M. Xiaopeng ZHANG, Professeur, National Laboratory of Pattern Recognition,
Institute of Automation, Chinese Academy of Sciences

Examineurs

M. Jean-Pierre JESSEL, Professeur, IRIT, Université Paul Sabatier

M. Olivier TERRAZ, Maître de Conférences - HDR, XLIM, Université de Limoges

M. Miquel FEIXAS, Maître de Conférences, GILab, Université de Girona

À ma famille,

À Ivan

Remerciements

Cette thèse est le produit de nombreuses années de travail, les commentaires et contributions de nombreuses personnes autour de moi. D'abord, je tiens à remercier mes directeurs, Mateu Sbert et Djamchid Ghazanfarpour pour le soutien incomparable qui m'ont été donnée tout au long de la thèse. Deuxièmement, je tiens à remercier Olivier Terraz pour son dévouement, son aide et les commentaires. Je remercie beaucoup à Dimitri Plemenos qui m'a donné la possibilité d'effectuer cette thèse de doctorat. Il faut aussi mentionner la coopération de mes collègues du bureau : Mark, Jorge, Ruben, Mario, Ferran, Roger, Xavi ... tous ont contribué dans la réalisation de cette thèse. Je remercie également tous les autres dans le Département d'informatique et de Mathématiques appliquées - Imma Boada, Miquel Feixas. Un grand merci à Nicolas Sunyer pour son soutien moral et les bons moments passés à parler de la thèse et bien d'autres choses.

Je tiens à remercier ma famille qui m'a tout donné et aidé non seulement dans la thèse, mais aussi tout au long de ma vie. Merci à mes parents pour leur soutien constant et la confiance aveugle dans ma capacité.

Enfin, un merci tout spécial à Ivan, d'être de mon côté, pour la patience, de soutenir les bons, et surtout les moins bons moments tout au long de la thèse.

Résumé

Les plantes ont longtemps intrigué les scientifiques, qui, avec son importance vitale pour la planète, sa beauté et l'énorme quantité de formes ayant, les rend un sujet attrayant pour la recherche. Un aspect intéressant est la création d'un modèle virtuel capable de simuler de vraies plantes avec un degré élevé de précision. L'objectif de notre étude est les plantes à fleurs, qui jouent un rôle énorme dans notre vie de fins nutritives et médicales à l'embellissement de l'environnement. L'obtention d'un modèle géométrique exacte d'une fleur est très utile, car elle joue un rôle important dans la validation du modèle virtuel. Par ailleurs, la visualisation de paramètres non directement traçables dans les plantes à fleurs vivantes est d'une grande aide à l'étude de la physiologie. L'énorme biodiversité entre les différentes parties d'un spécimen et entre les différents specimens fournit une vaste zone d'objectifs qui la synthèse d'image doit contester.

Modéliser des fleurs est un sous-ensemble d'un espace de recherche beaucoup plus vaste que la modélisation de plantes. Les plantes à fleurs ont des caractéristiques structurelles qui les rendent différentes des structures d'arbres, d'arbustes ou de l'herbe. À ce jour, on ne tient pas une grande importance à essayer cette ligne de recherche d'une façon particulière et en général a été classé dans le contexte plus large de la modélisation des plantes. Nous avons choisi d'utiliser le «L-systems» pour la procédure de la modélisation, et comme base pour notre recherche. Il y a différents mécanismes de catégorisation topologie de la plante dans chacune des étapes de sa croissance. Pour construire le plan de la structure d'une plante, avec une courte grammaire, quelques lignes étaient quelque chose qui dès le premier moment a suscité l'intérêt et par la suite évolué en quelques systèmes d'interprétation géométriques pour la modélisation des plantes. Notre objectif est d'étudier les moyens efficaces de décrire la structure des plantes à fleurs en utilisant L-systems. Tout d'abord, nous proposons de représenter les formes des feuilles, pétales, étamines, carpelles, etc. Avec une extension de L-systems - un modèle basé sur trois cartes généralisées dimensions - 3Gmaps L-systèmes, qui peut être appliquée avec succès pour la modélisation des plantes à fleurs. La description de la grammaire de la structure des plantes à fleurs fournit un nombre illimité de ses interprétations géométriques. Deuxièmement, nous allons améliorer le processus d'écriture de la grammaire par l'ajout d'une nouvelle fonctionnalité de paramétrage interactif. Troisièmement, nous allons proposer une nouvelle méthode de modélisation inverse des plantes à fleurs, où l'utilisateur

peut définir de manière interactive les caractéristiques des fleurs. L'algorithme utilise cette information comme une entrée, qui est ensuite analysée et codée en tant que L -systèmes grammaire. Enfin, nous allons présenter une méthode pour créer des clairières de fleurs virtuelles à l'aide de gestes Kinect. Nous voulons faire remarquer que notre travail a été fait avec la plateforme de logiciel 3Gmaps L- système développé dans le cadre de la thèse d'intégrer toutes les techniques proposées.

Mots clefs : Synthèse d'images, rendu réaliste, phénomène naturel, phénoménologie, fleurs.

Simulation of flowering plants

Abstract

Plants have always intrigued scientists as besides of its sheer importance for the earth, their beauty and enormous variety of shapes tempt to thoroughly inquire about its nature. One of the aspects of this inquiry is the creation of the virtual model in order to mimic real plants to a high degree of accuracy. The focus of our study is the flowering plants, which play a huge role in our life from nutritive and medical purposes to beautifying the environment. Obtaining an accurate geometrical model of a flower is quite useful as it plays an important role in the validation of the virtual model. Besides, the visualization of parameters not traceable directly in living flowering plants is a stand-by in studying their physiology. A huge biological diversity both within and between individuals provides a vast area of objectives which the image synthesis must challenge.

Flower modelling constitutes a part of a larger research area, plant modelling. Flowering plants have their particular structural features which are different from the structure of trees, bushes or grass. Still not a lot of emphasis has been placed to date on this problem, as it was categorized within the modelling of plants in general. We chose a procedural modeling using L-systems as a base of our research. L-system is a very powerful method of plant simulation. It provides a means of characterizing the topology of a plant at every stage of its growth. Grasping the plant structure with just several lines of grammar attracted immediate interest and later on evolved into several powerful geometrical interpretation system used in plant modelling. Our purpose is to study efficient ways of describing the structure of flowering plants by means of L-systems. First, we will propose to represent the shapes of leafs, petals, stamens, carpels, etc. with an extension of L-systems – a model based on three dimensional generalized maps – 3Gmaps L-systems, which can be successfully applied for the modelling of flowering plants. The grammar description of the structure of the flowering plants provides an unlimited number of its geometrical interpretations. Second, we will improve the process of grammar writing by adding a new functionality of interactive parameter adjustment. Third, we will propose a new method of inverse modelling of flowering plants, where the user can interactively define the flower characteristics. The algorithm uses this information as an input, which is then analyzed and coded as L-systems

grammar. Finally, we will present a method for creating virtual glades of flowers using Kinect gestures. We want to remark that our work has been done with 3Gmaps L-system software platform developed in the scope of the thesis to integrate all the proposed techniques.

Keywords : computer graphics, realistic rendering, natural phenomena, enomenology, flower.

Droits d'auteurs



Cette création est mise à disposition selon le Contrat : « **Paternité-Pas d'Utilisation Commerciale-Pas de modification 3.0 France** » disponible en ligne :

<http://creativecommons.org/licenses/by-nc-nd/3.0/fr/>

Contents

Introduction	13
1. Flowering plants simulation	19
1.1 State of the art.....	19
1.1.1 Architectural plant modelling.....	21
1.1.1.1 L-systems and botanical structures.....	22
1.1.1.2 Pursuing L-systems in flowering plant.....	24
1.1.1.3 Architectural models.....	27
1.1.2 Image synthesis-oriented modelling.....	29
1.1.2.1 Component-based modelling.....	29
1.1.2.2 Modelling with natural interfaces.....	32
1.1.2.3 Image-based modelling.....	34
1.1.3 Hybrid methods	36
1.1.4 Inverse modelling.....	36
1.2 Comparative tables	37
1.3 Conclusions	41
2. 3Gmap L-systems application to flowers	44
2.1 Introduction	44
2.2 3Gmap L-Systems	44
2.2.1 Managing the grammar	45
2.2.2 Modules.....	47
2.2.3 Application to the modelling of flowering plants	47
2.2.4 Materials.....	51
2.3 Flower rendering.....	52
2.4 Results	54
2.5 Conclusion	55
3. Interactive modelling of flowering plants	58
3.1 Introduction	58
3.2 Inverse problem challenges in flower modelling	59
3.3 Inverse grammar generation interface	59
3.3.1 Grammar generation and assembling.....	62
3.4 Interactive parameter adjustment.....	65
3.5 Results	67
3.6 Conclusions and future work.....	68
4. Flower modelling and Kinect	74
4.1 Introduction	74
4.2 Microsoft Kinect.....	74
4.3 Flower generation using Kinect.....	75
4.3.1 System architecture	75
4.3.2 Content Generation	77
4.3.3 Exploring the parameter space using Kinect.....	78
4.4 Conclusions	81
5. Summary and conclusions	83
5.1 Key Contributions.....	83
5.2 Research Outlook	85
5.3 Conclusions	86
Bibliography	87

List of figures

Figure 1 : Flower modelling classification.....	20
Figure 2 : Lindenmayer's original L-system for modelling the growth of algae	23
Figure 3 : Botanical structure of a generalized flower. Floral diagram (Font Quer, 1938)	25
Figure 4 : Queen Anne's Lace self-similarity and symmetry	26
Figure 5 : A cross-section of a leaf blade and a petal	27
Figure 6 : Optical microscope image of the petal. Sectional view of the violet petal showing cells of the front (upper) and the back surface. The cells in both surfaces are coloured, while those inside are otherwise (Ozawa et al., 2009)	27
Figure 7 : L-studio with a L-system tab opened (Prusinkiewicz et al., 2000)	28
Figure 8 : a) Petal is denoted as a Bezier surface ; b) an apple flower is formed using a combination of a stem, six calyxes, six petals with six stamens (Peiyu et al., 2006)	29
Figure 9 : Parts of a sunflower with corresponding p-graphs (Deussen & Lintermann, 1999)	30
Figure 10 : PlantStudio main window, in which you work with the plants in a plant file and create compositions (Fernhout & Kurtz, 2014).....	31
Figure 11 : A view of the PlantFactory editor windows (E-on software, 2014)	31
Figure 12 : PlantFactory models of flowers (E-on software, 2014).....	32
Figure 13 : Lily model. The structural information is given as a floral diagram and an inflorescence. The geometry models are designed in the sketch-based editor. The user creates a flower and the entire model of a lily combining the structural information and the geometries (Ijiri et al., 2005)	33
Figure 14 : Flower modelling pipeline (Yan et al., 2014).....	35
Figure 15 : 3Gmap description. 3Gmap L-Systems operations	45
Figure 16 : Nested structure of the grammar. SEPAL, PETAL, STAMEN, CARPEL are modules, which are integrated into the initial grammar with the symbol "&"	47
Figure 17 : First whorl of the floral diagram – calyx. Growing the sepals from the side faces of the stem	48
Figure 18 : Second and the third whorl of the floral diagram: corolla and androecium. Growing petals and stamens from the side faces of the base from the side faces of the base .	49
Figure 19 : The fourth whorl of the floral diagram – gynoecium	49
Figure 20 : Compound double umbel inflorescence. The model is represented with 2 derivation steps.....	50
Figure 21 : Petal, consisting of three layers : upper and lower epidermis and mesophyll. The upper images are the results of 11 derivation steps, growing and gluing operations.....	51
Figure 22 : The equation of the curve, with parameters for linear regression model, based on the set of points. Carpel is constructed using the equation for its width and length parameters with 11 derivation steps.....	52
Figure 23 : Applying random function as parameters of the volumes	52
Figure 24 : Meadow of tulips model. The tulips are modules integrated into the initial grammar. The model is constructed with 13 derivation steps.....	53
Figure 25 : Camera and light position of the scene	53
Figure 26 : Tulip rendered without and with translucency property and subsurface scattering function.....	54
Figure 27 : Bluebell and tulip flower models.....	55
Figure 28 : Models of a) tulips , b) water lily, c) bluebells and d) poppies	56
Figure 29 : Models of flower fields and meadows.....	56

Figure 30 : Inverse grammar generation interface. The control points are represented as tabs and located on the left of the main panel. The first control point – the stem. The grammar of the stem generated according to the description	60
Figure 31 : The second control point – the leaf. Generated geometrical model of the leaf and its grammar.....	61
Figure 32 : The fourth control point – the petal. Generated geometrical model of the petal...	62
Figure 33 : The fifth control point – the stamen. Generated geometrical model of the tulip's stamen.....	63
Figure 34 : The sixth control point – the pistil. Generated geometrical model of the tulip pistil	64
Figure 35 : Left: assembled model of tulip. Right: the closer view of the tulip flower	64
Figure 36 : Separate management from topology and its embeddings	66
Figure 37 : Interactive control of flower shapes, by changing the values of parameters	67
Figure 38 : Interactive control of flower shapes, by changing the values of parameters	67
Figure 39 : The numerous florets which are crowded together with spirals were modelled manually, while the external petal, sepal and leaves are modelled automatically	68
Figure 40 : The rendered models of flower fields. Above : poppies. Below : globeflowers and dandelions.....	69
Figure 41 : Rendered models of globeflowers and bluebells	72
Figure 42 : Pipeline of our framework	76
Figure 43 : Procedural flowers rendered on Android.....	77
Figure 44 : Top view of a sparse flowerbed.....	79
Figure 45 : Bluebells and daisies in the Don Quixote game	79
Figure 46 : Controlling the flower shape with Kinect gestures.....	80
Figure 47 : Different screenshots of a daisy in which the length and rotation parameters are controlled using Kinect gestures. The red circle indicates the horizontal and vertical position of the user's hand. The movement resembles phototropism.....	80
Figure 48 : Automatically generated flowerbeds using sunflowers and grass	81

List of tables

Table 1 : A comparative analysis of the referenced techniques.	38
Table 2 : A short overview of modelling techniques	40
Table 3 : Grammar example.....	45
Table 4 : Grammar representation in gl3 and xml formats	64
Table 5 : Real flowers and modelled flowers comparison.	70
Table 6 : CGI script to interface with the flower generator	77

List of publications

1. *3Gmap L-systems grammar application to the flowering plants modelling*. 1. Studies in Computational Intelligence Volume 441, 2013, pp 1-21.
2. *Interactive modelling of flowers with 3Gmap L-Systems*. GraphiCon'2011. 21st International Conference on Computer Graphics and Vision.
3. *Modelling of Flowers with Inverse Grammar Generation Interface*. International Journal of Creative Interfaces and Computer Graphics (IJCICG).Volume 3, Issue 2. Copyright © 2012. 19 pages.
4. *Flower modelling using natural interface and 3Gmap L-systems*. 12th ACM International Conference on Virtual Reality Continuum and Its Applications in Industry (VRCAI 2013).
5. *Modelling of flowering plants*. Computer-Aided Design - Journal – Elsevier (Submitted)

Introduction

Men's habitat is a green carpet of plants covering our earth. We can find plants almost everywhere in the scenery, be it fully natural or urban. They play a huge role in our life supplying us with oxygen, being the basis of most food webs, providing habitats for animals and beautifying our environment. And the most impressive and astonishing among them are flowers. Simulation of vegetation is a wide ranging research area of computer graphics. As flowering plants have such an intricate structure consisting of numerous components which, in its turn, have an enormous variety of shapes the task of its simulating is even more challenging for computer graphics.

There are various aims of the plant simulation, such as enlarging knowledge and helping with practical applications. It can make this knowledge accessible to and usable by non-expert. The visualization of growth simulations provides us with an instructive tool useful for agronomists and foresters, as well as for educational purposes. The use of plant modelling in computer game industry is also very challenging. Flowering plant approximations are especially hard, since organic structures tend to be difficult to describe in terms that graphics cards understand.

Plant generation has received a lot of attention in computer graphics. And the main research is focused on modelling of trees rather than flowers. Although, being its part, flowering plants have their particular structural features which are different from the structure of trees, bushes or grass. Still not a lot of emphasis was placed on this problem, as it was categorized as the modelling of plants in general.

We can distinguish several approaches to plant simulation. The first one is aiming at getting a plausible model while the botanical correctness is usually disregarded. This approach is quite intuitive for a common user, but has the inconvenience of creating each sample from scratch in case of generating a variation of slightly different plants (Ijiri et al., 2005), (Kang & Quan, 2009). The other approach could be referred to as procedural modelling, which tries to provide biologically faithful and visually realistic models (Prusinkiewicz et al., 2000), (Prusinkiewicz & Federl, 1999), (Prusinkiewicz & Lindenmayer, 1990). Most of these approaches are based on a mathematical theory of plant development, namely L-systems, which can generate complicated multicellular structures from a small

number of rules. They are able to get a lot of plant samples based on a single grammar by simply changing the parameter values. Although these methods can provide impressive results, the underlying algorithms are not so intuitive for common users.

The study of these methods points to look for another approach which can combine science with art, establishing interplay of the realism of the models and clearness for the users. In this thesis, we want to face this challenge and provide solutions for a number of open problems.

Challenges

Over the years the simulation of plants has been extended and the resulting models have gained acceptance in as a research tool in biology and have led to increasingly convincing visualizations. However, in image synthesis applications the simulation-based approach has several drawbacks:

Visual realism of the models depends on the biological and physical accuracy of simulations. The modeler needs to have a good understanding of the underlying processes which makes comprehensive models complicated and results in long simulation times.

All flowers look different, even the specimens of the same type have slight variations. Using the same model in the picture would produce a striking, artificial regularity. In order to prevent this each flowering plant variation has to be modeled separately.

Using a set of rules that describe the emergence and growth of individual plant components is very efficient and provides very realistic and biologically plausible models. But the underlying tools of these methods are not so intuitive for the common user. Most of the methods assume that the user is familiar with the concepts of L-systems and turtle interpretation, as well as the elements of the C programming language.

Vegetation is not only complex in geometry; also the light interaction of leaves, grass blades or petals is highly intricate. A layered structure of the plant tissue has a profound impact on both the reflectance and translucency of leaves and petals, an integral part of the light interaction of vegetation. Moreover, no general assumptions can be made on plant tissue rendering as many leaves and petals differ not only between species but also in their light transport on the front and back, depending on the nature of the surface.

Dissertation Thesis

This work focuses on some specific parts of this huge problem set, mainly geometric modelling, which requires specialized techniques for different situations and flowering plant species. The main thesis of this work is that it is possible to combine science with art, in order to retain the realism of the models and at the same time to simplify the task of the user. This requires designing algorithms that include L-system grammar writing and interactive user interfaces helping to control the grammars.

The thesis is organized in the following manner: In Chapter 1 we present the state of the art on modelling plants and flowers. Chapter 2 describes our original method and its application in flowering plants. Interactive flower modelling method is presented and discussed in Chapter 3. Chapter 4 deals with natural interfaces applied to flower modelling using Microsoft Kinect. Conclusions, as well as a future work layout are given in Chapter 5.

Contributions

A variety of new approaches and improvements over existing techniques is presented in this thesis. They are mainly concerned with geometric modelling of flowering and herbaceous plants and interactive user interfaces helping to control the grammars.

Modelling of flowering plants. We chose a procedural modelling using L-systems as a base of our research. L-system is a very powerful method of plant simulation. It provides a means of characterizing the topology of a plant at every stage of its growth. Grasping the plant structure with just several lines of grammar attracted immediate interest and later on evolved into several powerful geometrical interpretation system used in plant modelling. During our research we have discovered that the shapes of flower components have a quite complex topology, which cannot be described by the most commonly used L-systems with one topological dimension. Some methods use predefined surfaces and 3D shapes (generalized cylinders) which are incorporated to each symbol of the grammar. But predefined surfaces and 3D shapes do not “grow”. String symbols have very little control over the integrated organs. However, we need to simulate plant development fully; therefore we have to use 3 dimensional topological structures. We propose to represent the shapes of leafs, petals, stamens, carpels, etc. with an extension of L-systems – a model based on three

dimensional generalized maps – 3Gmaps L-systems, which can be successfully applied for modelling of flowering plants . These results have been published in

- 3Gmap L-systems grammar application to the flowering plants modelling. *Intelligent Computer Graphics 2012. Studies in Computational Intelligence Volume 441*, 2013, pp 1-21.

Interactive control. The grammar description of the structure of the flowering plants provides an unlimited number of its geometrical interpretations. Yet the task of writing a grammar is not intuitive for the user. The process of adjusting parameter values could be quite cumbersome as the user has to load the grammar every time he/she needs to see the changes of the geometry. We added a functionality of interactive change of parameter values. The user can adjust the model on the fly, changing parameter values and observing the result at the same time. This way of parameters values changing is quite faster as it only takes into account the embedding part, leaving the topology part of the program untouched.

The more complicated the flower is the more intricate and cumbersome the grammar could be. As flower complexity grows the grammar also grows leading to huge text files. As the flower consists of different components we decided to introduce modules which can substitute these components. Modules are grammars which represent petals, leaves, stamens, carpels, etc. They are included into the main grammar. The module itself can contain another module, therefore the grammar has a folded structure, which simplifies its construction and allows creating quite complicated models, which are the flower fields. This work has been published in:

- Interactive modelling of flowers with 3Gmap L-systems. *GraphiCon'2011. 21st International Conference on Computer Graphics and Vision*.

Inverse modelling. The process of writing a grammar is usually quite laborious and tedious. In order to avoid this we propose new interface functionality: the inverse modelling by automatic generation of L-systems. The user describes the flower he wants to model, by assigning the properties of its organs. The algorithm uses this information as an input, which is then analyzed and coded as L-systems grammar. This application provides an intuitive interface which permits the user to create grammars in a more comprehensive level. The user does not have to write an intricate code of the grammar, but with the help of our interface, he/she can define the flower characteristics, which are used for automatic grammar

generation. This work was published in International Journal of Creative Interfaces and Computer Graphics (IJCICG).

- Modelling of Flowers with Inverse Grammar Generation Interface. *International Journal of Creative Interfaces and Computer Graphics (IJCICG). Volume 3, Issue 2. Copyright © 2012. 19 pages.*

Modelling of large amounts of flowers using Kinect. We also proposed to create virtual glades of flowers using Kinect gestures. The user gestures are read and reinterpreted by the Kinect interface. Once the gesture is made, and a correspondence to the parameter space of the flower model is done, it is transmitted to a web server which contains a 3Gmap L-system application. The 3Gmap L-system receives the command to create or modify the flower and returns the 3D model which will be read and visualized by the Unity game engine. This work was published in ACM SIGGRAPH International Conference on Virtual-Reality Continuum and its Applications in Industry (VRCAI 2013):

- Flower modelling using natural interface and 3Gmap L-systems. *12th ACM International Conference on Virtual Reality Continuum and Its Applications in Industry (VRCAI 2013).*

Classification of flowering plants modelling methods. Flower modelling constitutes a part of a larger research area, plant modelling. Flowering plants have their particular structural features which are different from the structure of trees, bushes or grass. Still not a lot of emphasis has been placed to date on this problem, as it was categorized within the modelling of plants in general. Our review aims at evaluating the state of the art of 3D plant modelling highlighting flowering plants. After researching and analyzing the work done on flowering plants modelling we have classified it into four groups: architectural plant modelling, image-synthesis oriented modelling, hybrid methods, and inverse modelling. This work is submitted to Computer-Aided Design journal.

*A flower's appeal is in its contradictions — so delicate
in form yet strong in fragrance, so small in size yet big
in beauty, so short in life yet long on effect.*

~Terri Guillemets

Chapter 1

Flowering plants simulation

1. Flowering plants simulation

The plant nature is very complex and its simulation requires the implication of various disciplines: from botany and applied plant sciences, mathematics and statistics to theoretical computer science and computer graphics. There are various aims of the plant simulation, such as enlarging knowledge and helping with practical applications. It can make this knowledge accessible to and usable by non-expert. The visualization of growth simulations provides us with an instructive tool useful for agronomists and foresters, as well as for teaching (Fourcaud et al., 2008). The use of plant modelling in computer game industry is also very challenging. Plant approximations are especially hard, since organic structures tend to be difficult to describe in terms that graphics cards understand.

According to (Thornley & Johnson, 1990) there are so many levels of plant organization: from molecules, cells and tissues, to the whole plant and crop that the variety of the possible models is numerous. In the review of (Prusinkiewicz & Runions, 2012) modelling is represented as a wide range of notions such as: description of form, analysis of causality, analysis of self-organization, decomposition of problems, hypothesis-driven experimentation, and integrative view of development.

Our research will be focused on flowering plant morphogenesis from the geometric perspective, or description of form. Obtaining an accurate geometrical model of a flower is quite useful as it plays an important role in the validation of the virtual model. Besides, the visualization of parameters not traceable directly in living flowering plants is a stand-by in studying their physiology. A huge biological diversity both within and between individuals provides a vast area of objectives which the image synthesis must challenge.

1.1 State of the art

Modelling virtual flowering plants have been performed by several methods, most of which oriented towards the output image and based on software related motivation. Here the task of modelling is undertaken mainly by the user describing a plant structure and its components and defining the required parameters. The degree of realism depends on the users skills. However some of the methods are pursuing biological plausibility, using procedural modelling. Most of them are based on L-systems, which can generate complicated

multicellular structures from a small number of rules. They are able to get a lot of flower samples based on a single grammar by simply changing the parameter values. According to the existent methods we will divide our field of study into the following 4 groups (see Figure 1): architectural plant modelling, image-synthesis oriented modelling, and hybrid methods, inverse modelling.

These groups are not mutually exclusive (e.g. a technique can use both natural interface and L-systems production rules for the final model generation). This classification does not claim to be unique and universal, as it only reflects our general view on flowering plants modelling. Other reviews such as (Visser et al., 2002), (Deussen & Lintermann, 2005), (Prusinkiewicz & Runions, 2012) represent different visions on plant modelling thus providing their own classifications.

The review of (Visser et al., 2002) gives an overview of various types of plant models ranging from purely descriptive, focusing on graphic design, to process based, simulating growth processes using in-depth knowledge of plant physiology. The models are divided into structural, functional structural, those which are based on genetic expression and those with limited or no biological rules. Additionally the review contains a short overview of the current software tools that are commercially available for creation of artificial 3D plants.

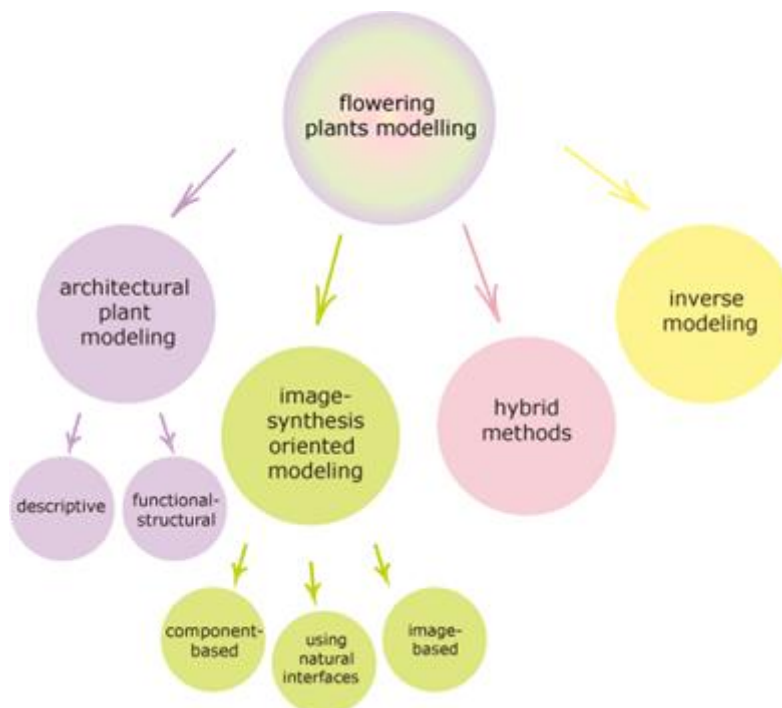


Figure 1 : Flower modelling classification

The book of (Deussen & Lintermann, 2005) provides a very extensive research on computer generated plants and organics. The book gives general information on botanical description of plants and represents plants as mathematical objects, including geometrical and topological models, branching structures, fractals, phyllotaxis, etc. Such methods as procedural modelling, rule-based modelling and rule-based object production are presented for the generation of individual plants. Additionally, the modelling of terrain and its plant communities along with the rendering of synthetic landscapes are examined in the book of (Deussen & Lintermann, 2005).

(Prusinkiewicz & Runions, 2012) article surveys the modelling techniques and selected models that are designed to elucidate plant development in mechanistic terms. The review provides the history of mathematical and computational approaches to developmental plant biology, followed by the key objectives and methodological aspects of model construction. The diverse mathematical and computational methods related to plant modelling are reviewed; and the essence of two classes of models, which approach plant morphogenesis from the geometric and molecular perspectives, is presented. In the geometric domain, they review models of cell division patterns, phyllotaxis, the form and vascular patterns of leaves, and branching patterns. In the molecular-level domain, the focus is put on the role of auxin in plant morphogenesis. The review is addressed to both biologists and computational modellers.

1.1.1 Architectural plant modelling

Architectural modelling considers a plant to be a set of relatively independent spatially arranged modules. From the different levels of abstraction it can be plant organs like leaves, petals, etc., or plant cellules. Different types of models can be provided using this kind of modelling. There are descriptive models and functional-structural models, or virtual plants. The first ones refer to the models which structure and development is characterized geometrically. The second type of models also takes into account the physiological processes involved into plant growth.

Plants were the object of intensive study and as flowering plants are among them we will have a look at diverse methods of plants modelling. Plants are living organisms with the structure and developmental processes obeying to some internal rules, which the scientists are trying to reveal since always. One of such attempts was made by Aristid Lindermayer who proposed a formal description of plant development as a string rewriting mechanism, known

as L-system, which has a recursive nature and leads to a self-similarity in plants. Since then it has been expanded into a very efficient mechanism, which is applied in modelling of growth processes of plant development and also in modelling of morphology of a variety of organisms (Prusinkiewicz & Lindenmayer, 1990), (Prusinkiewicz & Federl, 1999), (Prusinkiewicz et al., 2000).

1.1.1.1 L-systems and botanical structures

Trying to explore and understand the nature scientists were looking for the rules that lie underneath its external forms. As we have mentioned above, Aristid Lindenmayer proposed the multicellular organisms' description model, known as L-systems. Grasping a plant structure with just several lines of grammar attracted immediate interest and later on evolved into several powerful geometrical interpretation systems used in plant modelling. Let us have a look on L-system traces in nature and on how it is ingrained in flowering plant shapes.

We can consider L-systems as a string-rewriting mechanism, which allows individual symbols in a string to be replaced by strings of new symbols. A symbol can be used to denote a cell or organ in a plant and the replacement of the symbol by a successor sub-string may represent cell division or growth of plant structure. L-systems provide a means of characterizing the topology of a plant at every stage of its growth. The rewriting process starts from an initial string called the axiom. As plant grows, new organs appear, resulting in more complex plant structure. At every step, L-systems generate new sequences of symbols by applying various productions or rewriting rules to a string. The preceding string (the predecessor) may be the axiom or a descendant string resulting from the previous application of productions. The newly generated string (the successor), in turn, is passed back to the set of productions at the next step (see Figure 2).

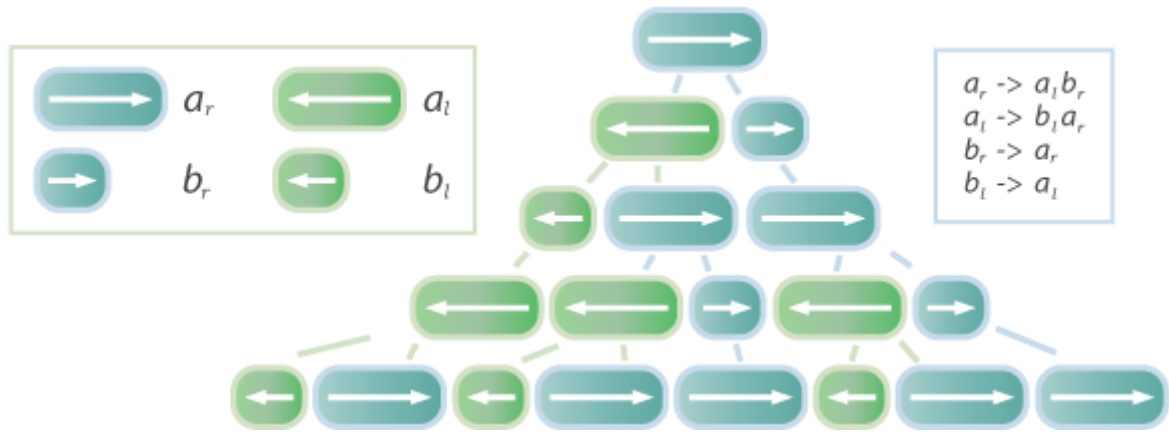


Figure 2 : Lindenmayer's original L-system for modelling the growth of algae

Since its advent L-Systems have been branched out into various extensions starting from the simplest, which is deterministic and context free (DOL-systems) and ending with more elaborated context sensitive parametric L-Systems. The theory of L-systems is reviewed in many survey papers (Prusinkiewicz et al., 1996), (Prusinkiewicz, 2004), (Prusinkiewicz et al., 1995) and books (Prusinkiewicz & Lindenmayer, 1990). Context-sensitive parametric L-systems is a specifically favorable technique for modelling such biological forms as flowering plants.

Simple DOL-Systems are not able to easily reproduce continuous phenomena, since it uses a technique of discretizing continuous values, thus requiring hundreds of symbols and productions. In order to resolve this obstacle Lindenmayer proposed to explicitly associate the numerical parameters with L-systems symbols. This was then formulated into parametric L-systems by (Prusinkiewicz et al., 1996).

Parametric L-systems follow the same principles of rewriting mechanism but operate on parametric words, which are strings of modules consisting of letters with associated parameters. A simplified definition is presented below, while the more detailed information on parametric L-systems can be found in (Prusinkiewicz et al., 1996), (Prusinkiewicz & Lindenmayer, 1990).

A parametric L-system is defined as an ordered quadruplet $G = (V, \Sigma, \omega, P)$, where

- V is the alphabet of the system,
- Σ is the set of formal parameters,

- ω is a nonempty parametric word called the axiom,
- P is a finite set of productions.

The production rule is defined as : predecessor : condition \rightarrow successor. For example, $A(x, y) : y > 3 \rightarrow B(x)A(x/y, 0)$.

Productions in the L-systems discussed above are context-free, which means that they can be applied, without taking into account the context in which the predecessor appears. But while modelling a complex flowering plant, we need an information exchange between its parts. For example, the order in which the flower organs grow and the interaction between them is important to sustain the botanical correctness.

Context-sensitive L-systems reckon on the context, using productions of the form $a_l < a > a_r \rightarrow \chi$, where the letter a (called the strict predecessor) can produce word χ if and only if a is preceded by letter a_l and followed by a_r (Prusinkiewicz et al., 1996). Therefore, counting on the described properties we can enter upon the modelling of flowering plants.

1.1.1.2 Pursuing L-systems in flowering plant

Before starting to explore the regular occurrences in flower shapes, let us have a look at its botanical structure. The flower basis is a stem which serves as a framework for the whole plant. The stem is enveloped with the leaves, which are attached directly to it. On the top of the stem there is a flower which is made of four concentric rings or whorls (Coen & Meyerowitz, 1991). There is an outer ring of modified leaves called sepals. They protect the flower before it opens and are usually green. This ring is known as calyx. There is another ring of modified leaves inside the calyx, the ring of petals (corolla) which are often brightly coloured.

Within the corolla there are one or more stamens - the male reproductive structures - forming the third ring called androecium. In the very centre of the flower are the female reproductive organs, called carpels, which is the fourth whorl - gynoecium. The structure of the flower is represented in a floral diagram which depicts the floral morphology plan of each species. It is a projection of the cross-section of each ring, which parts are arranged at the same relative position as they are in the flower (see Figure 3).

Some flowers are organized in groups, called inflorescence, and are placed on the stem in different manners. Queen Anne's Lace flower has a fractal nature. Each of its blossoms produces smaller iterative blooms (see Figure 4). Many flowers have radial symmetry and are called regular flowers, e.g. Queen Anne's Lace or trillium. But some of them are irregular, which, while bisected, form only one line that produces symmetrical halves, e.g. snapdragon or most orchids.

Underneath this structure lies an overwhelming variety of different species, and we can hardly find two flowers with the same shape. How can we capture the essence of all this variety and express it elegantly using L-systems? It is all about rewriting. Recursively applying the appropriate production rules on the initial element can result in complex structures. Such aspects of the flowering plants as symmetry, self-similarity, repeatability and also inflorescences and floral diagrams can be easily represented by L-systems. Botanical application of L-systems in different kinds of inflorescences and regular arrangements of lateral organs are thoroughly studied in (Prusinkiewicz & Lindenmayer, 1990).

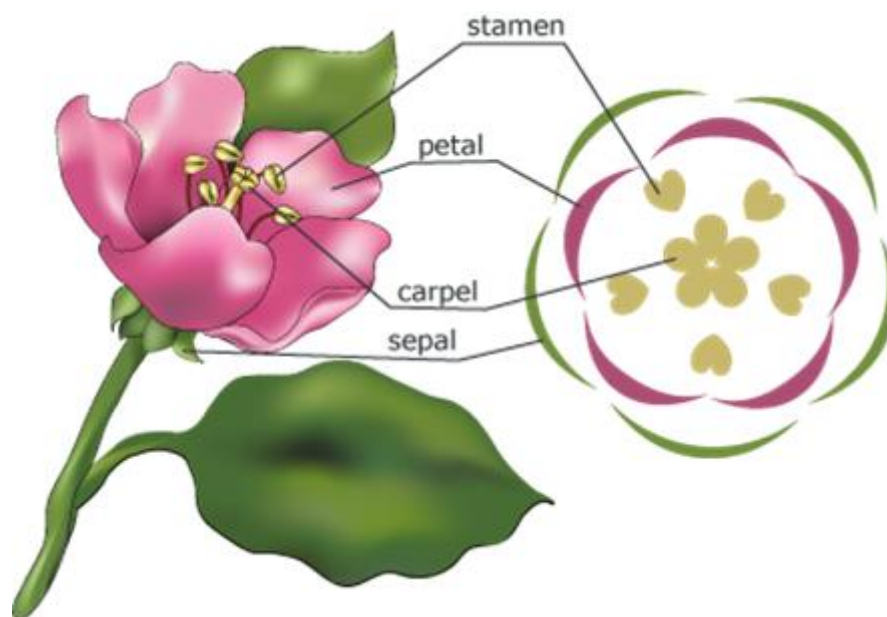


Figure 3 : Botanical structure of a generalized flower. Floral diagram (Font Quer, 1938)

If we go further into analysis of flower, we can find that the internal structure of its organs is not homogeneous. Leaves and petals tissues consist of different layers of cells, which have their own properties defining the variety of surfaces and colorations. Plant tissue has a very complex interaction with light as its inner structure is not homogeneous. Let's have a look at the cross-section of a leaf blade and a petal (See Figure 5). We can see a

comparatively thick photosynthetic region, known as the mesophyll, bounded by thin, protective layers of epidermal by a waxy coating (cuticle), responsible for specular reflection. The epidermis layers are transparent and allow light to pass through the mesophyll, which is composed of two layers. Beneath the upper epidermis is a layer of elongated, highly-diffusing palisade cells, responsible for much of the scattering of light that enters a leaf. Just above the lower epidermis lies a spongy layer consisting largely of air space. Petals are thought to be modified leaves with a simplified internal structure, having only one vascular bundle compares with the several normally found in leaves and sepals. It was found out that the structures like pigments having the colour are included only in the surface cells. The cells in the bulk have no observable colour (Ozawa et al., 2009) (see Figure 6).

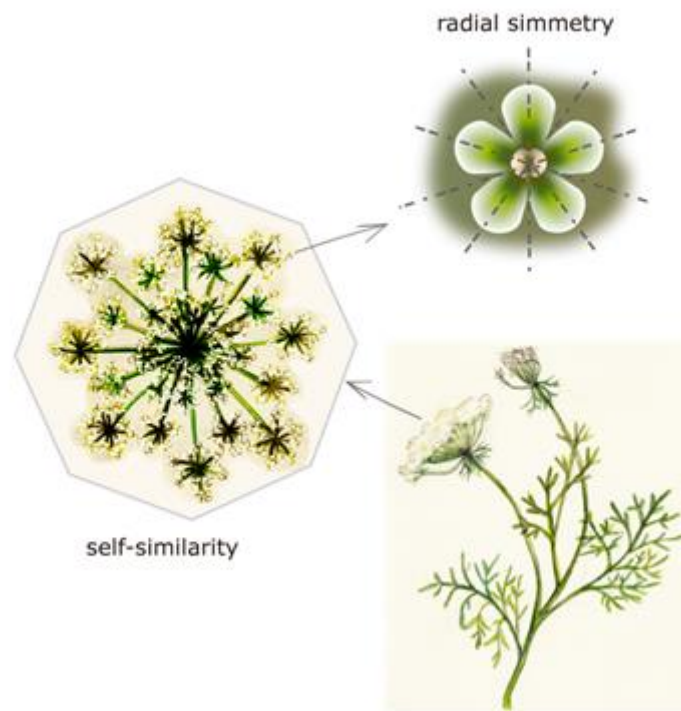


Figure 4 : Queen Anne's Lace self-similarity and symmetry

A layered structure of the plant tissue has a profound impact on both the reflectance and translucency of leaves and petals, an integral part of the light interaction of vegetation, which shows the need of using topological 3D dimensions in order to model flowering plants.

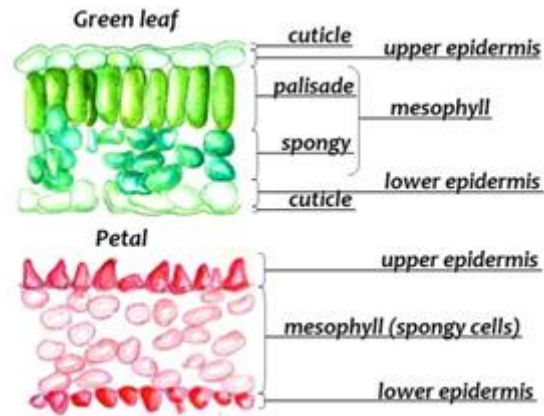


Figure 5 : A cross-section of a leaf blade and a petal

1.1.1.3 Architectural models

A very ample research on L-systems is performed in (Prusinkiewicz & Lindenmayer, 1990), (Prusinkiewicz, 2004), (Prusinkiewicz et al., 1995). The studies resulted in the modelling software L-studio (for Windows platforms) (see Figure 7) (Prusinkiewicz et al., 2000) and the Virtual Laboratory (for Linux platforms) (Prusinkiewicz & Federl, 1999). These tools enable to specify the architecture of various modular organisms, from filamentous bacteria and algae to herbaceous plants, trees, and plant ecosystems. Yet the shapes of individual plant organs, represented mostly as predefined surfaces or generalized cylinders (Fuhrer et al., 2006), are specified by the user and then are incorporated into a plant model.

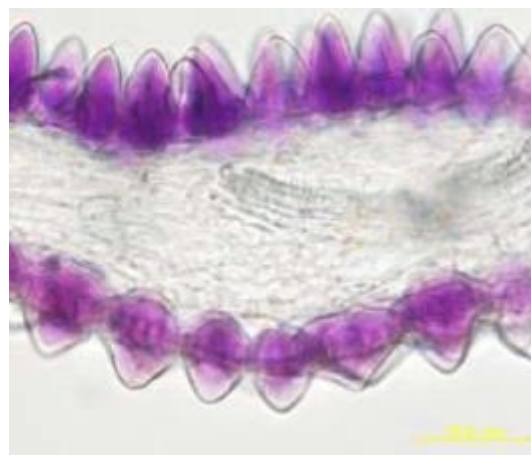


Figure 6 : Optical microscope image of the petal. Sectional view of the violet petal showing cells of the front (upper) and the back surface. The cells in both surfaces are coloured, while those inside are otherwise (Ozawa et al., 2009)

A huge research on architectural plant modelling is presented in the reviews (Prusinkiewicz, 1998) and (Prusinkiewicz & Runions, 2012), classifying the resulting models in empirical (descriptive) and causal (functional-structural), and emphasizing on the use of L-systems as a unifying framework for spatial model construction. In (Deussen & Lintermann, 2005) a detailed review on computer generated plants is presented including its botanical description and considering plants as mathematical objects.

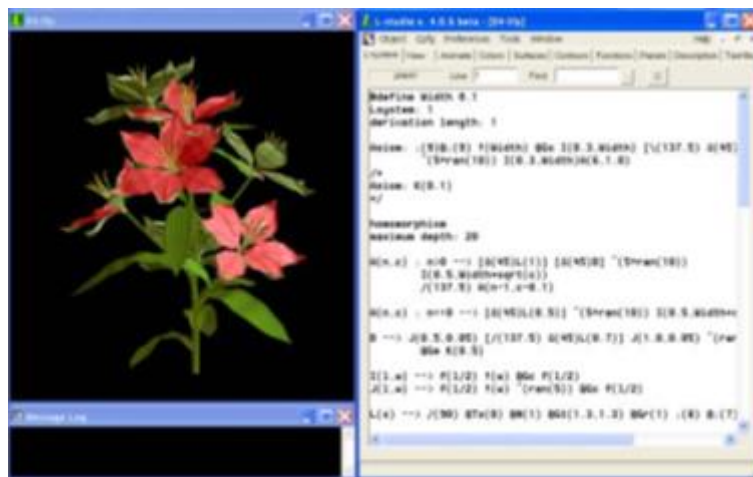


Figure 7 : L-studio with a L-system tab opened (Prusinkiewicz et al., 2000)

In (Frijters & Lindenmayer, 1974), (Frijters & Lindenmayer, 1976) flowers were described as configurations of modules in space. In (Prusinkiewicz & Lindenmayer, 1990), (Fowler et al., 1992) they were modelled using phyllotaxis - the regular arrangement of lateral organs. Examples such as sunflower head, zinnias, water lily and roses were presented by following phyllotactic patterns and associating different surfaces. In (Peiyu et al., 2006) it was proposed a flower model using the L- system and Bezier surfaces. L- systems were used to represent the topologic information of plant flower, while Bezier surfaces excelled at depicting geometric information of flower (See Figure 8).

Other methods, where the L-systems are not explicitly used but following the basic principles of architectural plant modelling, are available. Plant modelling tools such as AMAP (Reffye et al., 1997) and LIGNUM (Perttunen et al., 1996) provide a wide range of functional-structural models and introduce physiological concepts in order to simulate the dynamic functioning of trees. The model is presented as a network of parallel pipes or units which correspond to the organs of the plant. The plant is considered as a hydraulic structure, transporting water from the roots to the leaves, and producing assimilates via photosynthesis.

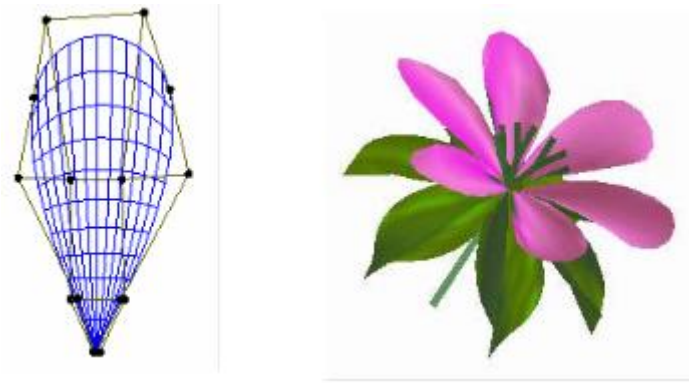


Figure 8 : a) Petal is denoted as a Bezier surface ; b) an apple flower is formed using a combination of a stem, six calyxes, six petals with six stamens (Peiyu et al., 2006)

Architectural plant modelling techniques provides very realistic and biologically plausible models. Most of the methods assume that the user is familiar with the concepts of L-systems and turtle interpretation, as well as the elements of the C programming language. The underlying tools of these methods are not so intuitive for the common user, which narrows the field of use of the system.

1.1.2 Image synthesis-oriented modelling

The main objective of the image-synthesis applications is the visual presentation of the models. Therefore the tools provided by these methods are user-oriented fostering the intuitive interfaces where the user can easily specify and interactively shape the models. There is a huge list of practical uses where the resulting models can be applied to, such as computer animations and games, virtual reality installations, multimedia presentations for educational purposes, computer-assisted landscape and garden designs, visual- impact analysis for forest harvesting, etc. A comparative table of plant software tools is presented in (Discoe, 2013). We next review the most representatives ones.

1.1.2.1 Component-based modelling

A system, proposed by (Deussen & Lintermann, 1999) is a modelling method that allows easy generation of many types of objects that have branching structures, including flowers, bushes, trees, and even some non-botanical objects. In this approach, components encapsulate data and algorithms to generate plant elements. All components have a set of parameters to control their behaviour. To establish the complete plant description the component prototypes are connected in a directed graph called the prototype graph, or p-graph. When the system traverses the p-graph, it builds a temporary tree of component

instances, which is then used to generate the geometry. In Figure 9 a model of the sunflower is represented. By editing splines, users can choose the appropriate curvature and scale of the components responsible for the stem and the leaves to create a typical outline of a small leaf. Next the system iterates the leaves as branches describing the plant's stalk. The top of the stalk is opened to form the head of the flower. Then the responsible components construct the blossom of the sunflower – for arranging the petals and the seeds. Finally, everything is connected to the full p-graph.



Figure 9 : Parts of a sunflower with corresponding p-graphs (Deussen & Lintermann, 1999)

This research was converted into a commercial system Xfrog (Deussen & Lintermann, 2014), (Deussen & Lintermann, 2005). It combines the advantages of the rule-based and parameterized-based modelling schemes, yielding highly realistic plant models. It provides an intuitive interface to manipulate the plant components and design key frames of a plant animation process. Xfrog system is oriented on creating elegant models instead of true-to-nature development modelling. However some of the components have parameters that interact with a plant's response to its environment.

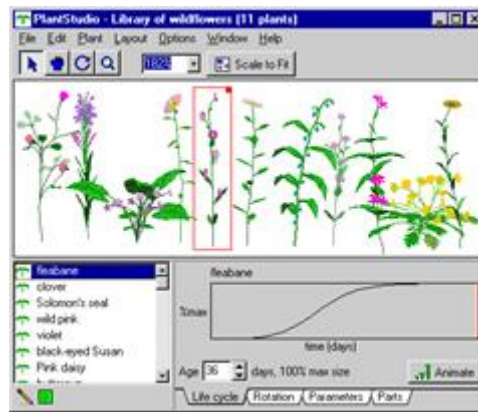


Figure 10 : PlantStudio main window, in which you work with the plants in a plant file and create compositions (Fernhout & Kurtz, 2014)

PlantStudio (Fernhout & Kurtz, 2014) is another component based system which uses a simulation model with over 200 parameters to encompass a wide range of herbaceous (non-woody) plants (see Figure 10). Objects are introduced to model the major plant parts. No interaction with the environment is computed and growth and development only depend on the time elapsed since emergence.

Plant Factory (E-on software, 2014) is a commercial 3D vegetation modelling, animation and rendering software, dedicated to the CG, SFX, Architecture and Gaming communities. Using the graph of nodes, the user builds the plants from simple geometry nodes. Each geometry node features multiple parameters that are adjusted and combined together to achieve the desired look (See Figure 11 and Figure 12).



Figure 11 : A view of the PlantFactory editor windows (E-on software, 2014)

(Lu et al., 2000) proposed a perceptually realistic flower generation. The system focuses on the simulation of flower petals through all transient stages from a bud to the fully developed flower. Lu et al introduced smooth surface model with a bicubic patch for the

petal. The biological factors in length, width and depth represent the principle growth for the petal.

1.1.2.2 Modelling with natural interfaces

Procedural modelling explicitly using mathematical formula or grammar specification is very powerful and efficient, but to a novice user they can also be cumbersome and daunting. A complex model requires considerable expertise and effort. More intuitive and natural interfaces also found its niche in the plant modelling area, such as sketch-based interfaces and 3d gestures.

The main idea is to use sketches - rapidly executed freehand drawings – in the process of modelling instead of directly editing polygons. The system is then automatically interprets a sketch and creates a 3D model (Zelevnik et al., 2006). With 3d gestures the user can mold and manipulate the model due to the system that turns gestures into computer commands.



Figure 12 : PlantFactory models of flowers (E-on software, 2014)

(Ijiri et al., 2005) presented a system for modelling flowers in three dimensions quickly and easily while preserving correct botanical structures. They use floral diagrams and inflorescences, which were developed by botanists to concisely describe structural information of flowers (see Figure 13). Floral diagrams represent the layout of floral components on a single flower, while inflorescences are arrangements of multiple flowers. According to this a simple user interface that is specially tailored to flower editing, is created. To define the geometries of flower components sketching interfaces are provided. The user first defines the flower's structure in the floral diagram editor by editing the layout of the floral components. Then he models the shapes of the floral receptacle and floral components

by inputting its outlines and drawing the modifying strokes in the geometry editor. After that, the user associates geometries of floral components with corresponding elements in the floral diagrams. The system automatically places geometric objects on the receptacle model. After designing individual flowers, the user models inflorescence by defining its structure. Although this is a system which allows an efficient modelling of flowers with correct botanical structures there are some limitations such as: shape restriction (i.e., petal-like shapes that do not have an elliptical outline), inflorescence editor is not able to support the creation of a gradual progression of developmental flower stages.

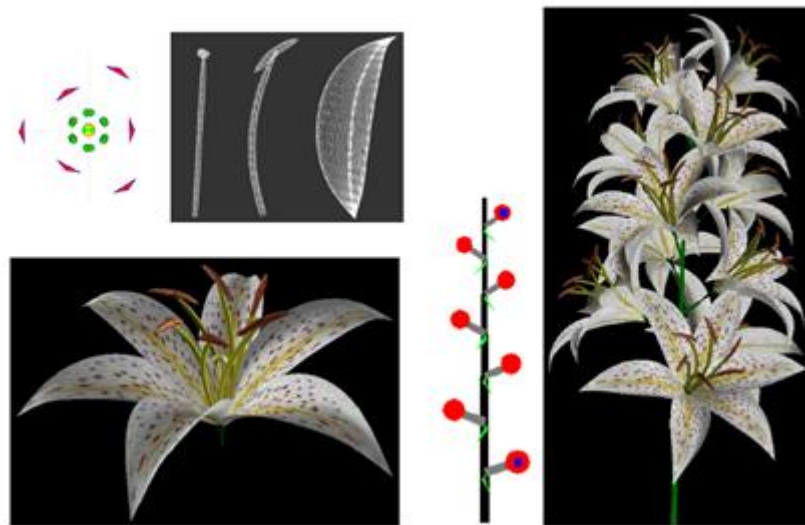


Figure 13 : Lily model. The structural information is given as a floral diagram and an inflorescence. The geometry models are designed in the sketch-based editor. The user creates a flower and the entire model of a lily combining the structural information and the geometries (Ijiri et al., 2005)

The method of (Ding et al., 2008) is based on (Ijiri et al., 2005). It separates individual flower modelling and inflorescence modelling procedures into structure and geometry modelling. The interactive editing gestures are incorporated to allow the user to edit structure parameters freely onto structure diagram. Free-hand sketching techniques are used to allow users to create and edit 3D geometrical elements freely and easily. The final step is to automatically merge all independent 3D geometrical elements into a single waterproof mesh. The final model can be printed onto real flower toy or decoration directly.

The next work of (Ijiri et al., 2006) is an interactive modelling system for flower composition that supports seamless transformation from an initial sketch to a detailed 3D model. To begin with, the user quickly sketches the overall appearance of the desired model as a collection of 2D strokes on hierarchical billboards. Then the user iteratively replaces the

coarse sketch with a detailed 3D model referring to the initial sketch as a guide. Since a flower model consists of many repetitive components, the system helps the user to reuse 3D components to facilitate the modelling process. The global view of the entire model is always shown in a separate window to visualize how local modifications affect the global appearance. The system helps the user make appropriate design decisions to keep the model consistent with the initial design, which is difficult in traditional bottom-up plant modelling systems in which the global view only emerges after all of the details are specified.

The task of the user while modelling using natural interfaces is quite easier and takes less time, but still we cannot reckon on creating the models with quite complicated structures with botanical correctness, neither cannot consider the obtained model as a sample for creating the huge diversity of individuals.

1.1.2.3 Image-based modelling

The traditional approach of computer graphics has been to create a geometric model in 3D and to try to reproject it onto a two-dimensional image. Computer vision, conversely, is mostly focused on detecting, grouping, and extracting features (edges, faces, etc.) present in a given picture and then trying to interpret them as three-dimensional clouds of points. Instead of specifying the plant model by the user image-based approaches use images to help generate 3D models. They vary from single image and shape priors use (Han & Zhu, 2003), to multiple images (Sakaguchi, 1998), (Shlyakhter et al., 2001), (Reche et al., 2004). A popular approach is to use the visual hull to aid the modelling process (Sakaguchi, 1998). However, the models generated by these approaches are only approximate and have limited realism. (Reche et al., 2004), on the other hand, compute a volumetric representation with variable opacity. While realism is achieved, their models cannot be edited or animated easily.

A method of (Neubert et al., 2007) produces 3D tree models from several photographs based on limited user interaction. Their system is a combination of image-based and sketch-based modelling.

(Quan et al., 2006), (Kang & Quan, 2009) proposed a semi-automatic technique for modelling plants. Using this approach as an example we can obtain the idea of the image-based modelling of plants. This is an interactive system which automatically recovers the shape relying on the user to provide simple hints on segmentation. The system is divided into three parts: image acquisition and structure from motion, leaf segmentation and recovery, and interactive branch recovery. A hand-held camera is used to capture images of the plant at

different views. A standard structure from motion technique is applied to recover the camera parameters and a 3D point cloud. Then, the 3D data points and 2D images are segmented into individual leaves. To make this process easier, a simple interface is designed that allows the user to define the segmentation together with 3D data points and 2D images. The data to be partitioned is implemented as a 3D undirected weighted graph that gets updated on-the-fly. To model a plant, the user first segments out a leaf; this is used as a deformable generic model. This generic leaf model is subsequently used to fit the other segmented data to model all the other visible leaves. The reconstruction of the branches is carried out interactively by the user.

(Yan et al., 2014) proposed a semi-automatic method for reconstructing flower models from a single photograph. The technique assumes that the flower head typically consists of petals embedded in 3D space that share similar shapes and form certain level of regular structure (see Figure 14). Based on these assumptions the method first fits a cone and subsequently a surface of revolution to the flower structure and then computes individual petal shapes from their projection in the photo. Flowers with multiple layers of petals are handled through processing different layers separately. Occlusions are dealt with both within and between petal layers. Being an initial attempt of modelling flowers from single images, the proposed approach concentrates on rather simple flowers. There are many other types of flowers that include more complex arrangements of petals; prominent examples are roses or moms.

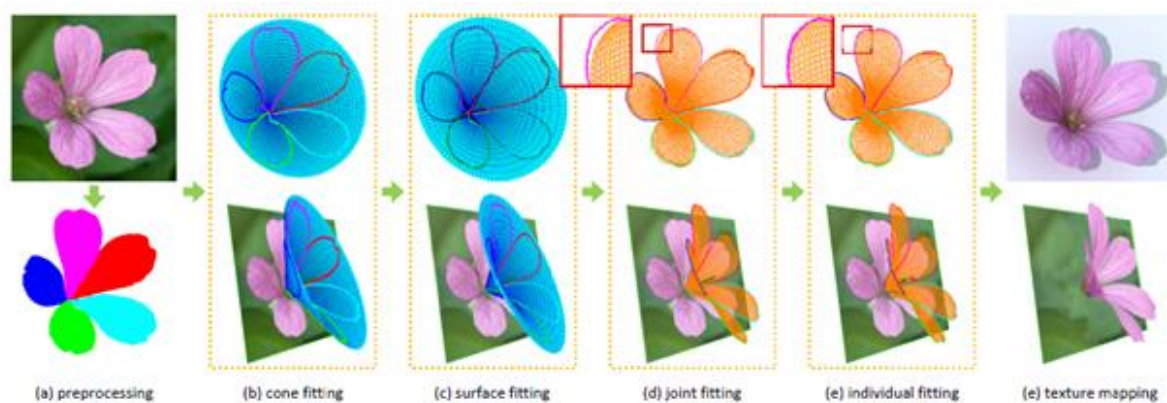


Figure 14 : Flower modelling pipeline (Yan et al, 2014)

Image-based techniques provide realistic results since they are based on images of real plants. However the user interaction could not be completely avoided as the computer vision techniques have to be customized in order to work well. Besides they can obtain

models from only preexisting plants rather than create new and visually-plausible plant models.

1.1.3 Hybrid methods

The groups described above are not mutually exclusive. Taking into account the benefits of the previous approaches these methods combine science with art, establishing interplay of the realism of the models and clearness for the users. In (McCormack, 1993), (Ijiri et al., 2006), (Onishi et al., 2006) the L-systems are mixed with interactive methods, such as Sketch-based or 3D gesture modelling or simply interactive control of parameter values. In (Power et al., 1999) the plant structures can be interactively manipulated using inverse-kinematics optimization technique. Here such kinds of manipulation are provided: bending and pruning branches and arranging and clipping leaves and flowers. (Anastacio et al., 2008) proposed a combination between sketch-based modelling and L-System, where construction lines are employed to parameterize global features of L-System models. This means that the user sketches the construction lines that define the overall structure of the plant. An interpretation of construction lines is then used to automatically derive a set of positional B-spline functions, which are employed as parameters for productions in predefined L-System templates, representing phyllotactic patterns (Fowler et al., 1992) for positioning lateral organ surfaces such as leaves and petals.

1.1.4 Inverse modelling

Up to now we were describing the conventional modelling, which requires the user, in order to obtain the model, to anticipate and define feature constraints, relations, and dependencies. The object of the inverse modelling is just the contrary; it is to determine unknown causes based on observation of their effects. In other words taking a flower model or a photo as input the systems estimates the parameters of procedural model or obtains the grammar of the model so that it produces flower similar to the input. We will not consider in this thesis an image-based modelling as an inverse-modelling. Although it allows the use of a single or multiple two-dimensional images in order to generate directly a 3D model, still the obtained result is just a geometrical model without any botanical principles beneath it. The creating of a new flowering plants having the models of the real flowering plants, or simply editing or animating the obtained models using the image-based modelling is likely to be a non-trivial task.

Inverse modelling is not an easy task for computer graphics. In recent work several methods were trying to create procedural rules from user input. In (Aliaga et al., 2007) the user draws simple building blocks and using the proposed system he/she can automatically complete the building “in the style of” other buildings using view-dependent texture mapping or non-photorealistic rendering techniques. The system supports an arbitrary number of building grammars created from user subdivided building models and captured photographs. (Štáva et al., 2010) proposed an algorithm that takes an input vector image with objects formed as groups of line segments or Bézier curves and produces an L-system that generates the given input 2D model. In (Bokeloh et al., 2010) it is addressed the problem of inverse procedural modelling of 3D geometry. This approach consists of semi-or fully automatic creation of 3D models that are similar to a piece of example geometry. It is able to compute shape grammars for general 3D surfaces from example geometry without any user interaction (the similarity radius r is the only parameter). However, the proposed formal framework requires models that have perfect partial symmetries, which is currently still a main limitation of this approach.

In (Shlyakhter et al., 2001) a hybrid approach is presented in order to solve the inverse problem. The input of the system is a set of images of a tree usually numbering between 4 and 15 and uniformly covering at least 135 degrees around the tree. The approach involves first segmenting the images and constructing a plausible skeleton of the tree (trunk and major branches, and growing the rest of the tree with an L-System.





In (Stava et al., 2014) inverse procedural modelling of trees is represented as a framework based on a novel parametric model for tree generation which uses Monte Carlo Markov Chains to find the optimal set of parameters. The approach consists on taking polygonal tree models as input and estimating the parameters of a procedural model so that it produces trees similar to the input.

1.2 Comparative tables

In this section we overview most of the techniques previously referenced in this survey using tables for comparison. In Table 1 we compare the methods using such criteria as: botanical plausibility (whether the resulted model follows botanical rules), interface (if it is intuitive for the user) and final image (how realistic is the final result). We are based on (Prusinkiewicz, 2000) review in order to determine “botanical plausibility” which includes geometric characterization of plant structure and development and physiological mechanisms

controlling plant development (endogenous interaction – information transfer between adjacent elements of the plant structure, for example water, minerals, hormones transported by plant tissues; and exogenous interaction – information transfer through the physical space in which plants grow, for example competition for space and light or the influence of moisture or temperature). By the term “interface” we mean the presence of two factors such as : interactivity – whether the user can interactively manipulate the models and change its parameters and learnability – the easiness for the common user (who is not familiar with the concepts of L-systems or programming languages) to accomplish basic tasks. The “final image” is represented by visual coincidence with the prototype form and realistic rendering of the model (using translucency and global illumination). The methods are grouped by colors in order to distinguish to which group of our classification it belongs.

Table 1 : A comparative analysis of the referenced techniques.

	architectural plant modelling
	image-synthesis oriented modelling
	hybrid methods
	inverse modelling

Plant type	
T	Trees
H	Herbaceous plants
F	Flowers

botanical plausibility	geometric characterization of plant structure and development	A
	endogenous interaction	B
	exogenous interaction	C
interface	interactivity	A
	learnability	B
final image	visual coincidence with the prototype form	A
	realistic rendering	B

Author	Technique	Botanical plausibility	Interface	Final image	Type of plant
(Fowler et al., 1992)	collision-based model	A+B	B	A	F
(Prusinkiewicz et al., 2000)	L-systems, generalized cylinders	A+B+C	A	A	T+H+F
(Prusinkiewicz & Federl, 1999)	L-systems, generalized	A+B+C	A	A	T+H+F

	cylinders				
(Frijters & Lindenmayer, 1974)	L-Systems	A	×	×	F
(Peiyu et al., 2006)	L-systems, Bezier curves	A	A	A	F
(Peyrat et al., 2008)	2Gmap L-systems	A	N/A	A+B	H
(Terraz et al., 2009)	3Gmap L-systems	A	N/A	A	T
(Reffye et al., 1997)	AMAP	A+B+C	A	A	T
(Perttunen et al., 1996)	LIGNUM	A+B+C	×	×	T
(Deussen & Lintermann, 1999)	component-based	A	A+B	A+B	T+H+F
(Deussen & Lintermann, 2014)	XFROG	A+C	A+B	A+B	T+H+F
(E-on software, 2014)	component-based	A	A+B	A+B	T+H+F
(Fernhout & Kurtz, 2014)	component-based, 3D Object-oriented model	A	A	A	H+F
(Lu et al., 2000)	plant growth function	×	A	A	F
(Ijiri et al., 2005)	sketch-based	A	A+B	A	H+F
(Ding et al., 2008)	sketch-based	A	A+B	A	F
(Ijiri et al., 2006)	sketch-based, seamless transformation	×	A+B	A	F
(Reche et al., 2004)	image-based	×	A+B	A	T
(Neubert et al., 2007)	image-based	×	A+B	A	T
(Quan et al., 2006)	image-based, a semi-automatic technique	×	A+B	A	T+F
(Kang & Quan, 2009)	image-based	×	A+B	A+B	T+F
(Yan et al., 2014)	image-based	×	A+B	A+B	F
(Ijiri et al., 2006)	sketch-based, L-Systems	A	A+B	A+B	T
(Power et al., 1999)	inverse-kinematics, L-Systems	A	A+B	A	T+H
(Anastacio et al., 2008)	Sketch-based, L-Systems, construction lines	A	A+B	A	T+H
(Shlyakhter et al., 2001)	inverse problem, image segmentation, L-System	A	A	A	T
(Stava et al., 2014)	procedural modelling, Monte Carlo Markov Chains,	A	A+B	A	T

Table 2 provides a short overview of the classified groups, where a short description, advantages and disadvantages are mentioned. Each group has a list of representative techniques previously referenced in this survey.

Table 2 : A short overview of modelling techniques

Modelling group		Representative systems	Description
Architectural plant modelling	Descriptive	(Fowler et al., 1992) (Peiyu et al., 2006) L-Studio and Virtual Laboratory (Prusinkiewicz & Lindenmayer, 1990)	It considers a plant to be a set of relatively independent spatially arranged modules. Descriptive type refers to the models which structure and development is characterized geometrically. Functional-structural type of models also takes into account the physiological processes involved into plant growth. L-systems are widely involved in this group of modelling. The techniques provide very realistic and biologically plausible models. But the underlying tools are not so intuitive for the common user, which narrows the field of use of the system.
	Functional-structural	L-Studio and Virtual Laboratory (Prusinkiewicz, 2004)] (Harder & Prusinkiewicz, 2013)	
Image-synthesis oriented modelling	Component based modelling	(Deussen & Lintermann, 1999) XFROG (Deussen & Lintermann, 2014), Plant Studio (Fernhout & Kurtz, 2014), The Plant Factory (E-on software, 2014),	The main objective is the visual presentation of the models. The provided tools are user-oriented fostering the intuitive interfaces where the user can easily specify and interactively shape the models. These methods provide impressive results, however they don't take into account the biological rules or their application is quite limited.
	Based on natural interfaces	(Ijiri et al., 2005), (Ijiri et al., 2006), The Plant Factory (E-on software, 2014), (Ding et al., 2008)	
	Image-based modelling	(Yan et al., 2014) (Kang & Quan, 2009)	
Hybrid methods		(Power et al., 1999) (Anastacio et al., 2008)	A combination of architectural plant modelling with image-synthesis oriented modelling. This group establishes interplay of the realism of the models and clearness for the users.
Inverse modelling		(Stava et al., 2014) (Shlyakhter et al., 2001)	The object is to determine unknown causes based on observation of their effects. Taking a flower model or a photo

		as input the systems estimates the parameters of procedural model or obtains the grammar of the model so that it produces flower similar to the input. This is a relatively new problem and the provided techniques are limited.
--	--	--

1.3 Conclusions

Flowering plants comprise about 90 percent of earth plants. A huge biological diversity both within and between individuals provides a vast area of objectives which the image synthesis must challenge. Having all the features that define botanical structures, such as self-similarity, symmetry, branching arrangement, etc., flowers constitute a part of a large scope of study, which is modelling of plants.

There are various methods of plant modelling. Some of them are aiming at getting a plausible model while the botanical correctness is usually disregarded. Here the task of modelling is undertaken mainly by the user describing a plant structure and its components and defining the required parameters. As flower has quite a complicated structure, the degree of realism will depend on the user skills. These approaches are quite intuitive for a common user, but have an inconvenience of creating each sample from scratch in case of generating a variation of slightly different flowers.

Other methods could be referred to as procedural modelling, which tries to provide biologically faithful and visually realistic models. Most of these methods are based on a mathematical theory of plant development, namely L-systems, which can generate complicated multicellular structures from a small number of rules. They are able to get a lot of flower samples based on a single grammar. Although these methods can provide impressive results, the underlying algorithms are not so intuitive for common users.

The study of these methods points to look for another approach which can combine science with art, establishing interplay of the realism of the models and clearness for the users. Pursuing this goal we propose an application of the 3Gmap L-systems: flower modelling by growth simulation. Our approach combines L-systems grammar writing with interactive control of parameter settings. Here the L-systems operate with subdivision of volumes, namely 3Gmaps. The used L-systems grammars have a nested structure allowing combining several grammars which represent the different flower organs. In order to avoid the laborious task of grammar writing we propose a new interface function: the inverse modelling by automatic generation of L-systems. The user describes the flower he wants to

model, by mentioning the properties of its organs. The algorithm uses this information as an input, which is then analyzed and coded as L-systems grammar. The user can control the final result by interactively setting the parameters of the grammar. These contributions make the task of a user more obvious and intuitive which in turn enables to create more accurate models. In addition, the way the model is built allows us to take into account its internal structure. As the flower tissue is non-homogeneous, this can be quite useful to render more accurate subsurface scattering.

The artist is the confidant of nature, flowers carry on dialogues with him through the graceful bending of their stems and the harmoniously tinted nuances of their blossoms. Every flower has a cordial word which nature directs towards him.

~Auguste Rodin

Chapter 2

3Gmap L-systems application to flowers

2. 3Gmap L-systems application to flowers

2.1 Introduction

String L-systems are quite efficient and are applied to model a wide variety of plants. Although they are of one-topological dimension, even if 3D geometrical features are incorporated into a model. However, many shapes in nature can only be described by two or three topological dimensions (for example, leaves, petals, pistils and stamens). Thus, Prusinkiewicz and Lindenmayer described in (Prusinkiewicz & Lindenmayer, 1990) map L-systems and cellwork L-systems which were mainly used for modelling cellular layers. A map is a finite set of regions, surrounded by a boundary consisting of edges. A map L-system is a parallel rewriting system that operates on maps and does not allow for interaction between regions. Cellwork L-system is an extension of map-L-systems which was proposed in order to capture the three-dimensional aspect of cellular layers. These methods provide quite realistic results. However, it is quite difficult to specify L-system grammar and there is not enough control over the generated topologies.

In (Peyrat et al., 2008), (Terraz et al., 2009) 2Gmap and 3Gmap L-systems address the limitations of previously described methods. These approaches are applied to model realistic leaves and wood. 2Gmap and 3Gmap L-Systems are based on two and three-dimensional generalized maps, which could be controlled by the operations associated with production rules. The direct use of high level operations on surfaces and volumes simplifies model specification and the use of adjacency relations between volumes allows context-dependent behaviors. 3Gmap L-systems can be successfully applied for the modelling of flowering plants.

2.2 3Gmap L-Systems

3Gmap is an ordered topological model that allows representing the topology of subdivisions of orientable or non-orientable 3D spaces, with or without boundary. It is close to facet-edge data structure (Brisson, 1993) or cell tuple (Dobkin & Laszlo, 1987). A subdivision of a topological space is a partition of this space into cells with dimensions 0, 1, 2, 3, i.e. into vertices, edges, faces, volumes. This model is based on the use of an unique basic element - a volume - on which four operators act. These operators are used to represent

adjacency relations between edges, faces and volumes. A combination of these basic elements allows representing the topology of an object, which corresponds to an unlimited number of embeddings of this structure in three-dimensional space. As we have mentioned above 3Gmap L-systems are operated with volumes, which are mostly regular prisms. In order to control each volume we use a label, associated to it, which is a word in capital letters. A prism GERM of order n is denoted as GERM (n). Each face of a volume also has a label which is defined as GERMO, GERME and GERMC1, GERMC2, ..., GERMC n for the base, the end and the side faces of the prism STEM respectively (see Figure 15 **Error! Reference source not found.**). A flower model is created by growing, splitting and gluing its building blocks, following the production rules of a grammar. The grammar is a description of a flowering plant, containing the information of its building blocks and the instructions used in the development of its final shape.

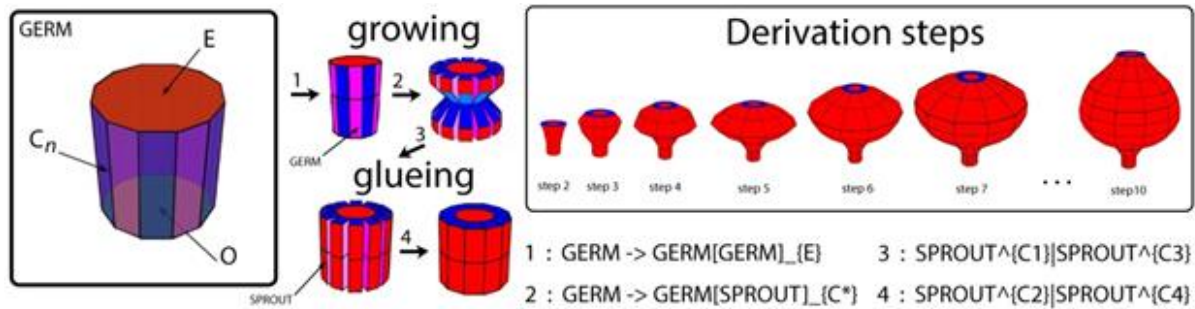


Figure 15 : 3Gmap description. 3Gmap L-Systems operations

2.2.1 Managing the grammar

The grammar structure is represented in Table 3. It consists of a volume description, a variable and function definition, an axiom and a set of production rules (growing, glueing, and splitting) (see Figure 15).

Table 3 : Grammar example

```
@volume description@
#define STEM(21,1,0,0,0,1,1,6)
#define RECEPTACLE(21,1,0,0,0,3,1,1,6)
#define PETAL(4,1,0,0,0,0,5,0.5,1,4)

@variable and function definition@
#define rd=7

#axiom: STEM
```

@production rules@

```
p00 STEM{ } {step<4} { }->STEM[STEM(,,,,,4.1,5,1.5,)]_ {E}
p00 STEM{ } {step>=4} { }->STEM[STEM(,,,,,,)]_ {E}
p00 STEM{ } {step>=finalStep-10} { }->STEM[RECEPTACLE(,,,,,,)]_ {E}
p00 RECEPTACLE->RECEPTACLE[PETAL(4,1,0,2,0,1,1,1,4)]_ {C5}
p00 RECEPTACLE->RECEPTACLE[PETAL(4,1,0,-3,0,1,1,1,4)]_ {C9}
p00 RECEPTACLE->RECEPTACLE[PETAL( , , <math.random(0,rd)>,,,,)]_ {C13}
p00 RECEPTACLE->RECEPTACLE[PETAL( , , <math.random(0,rd)>,,,,)]_ {C17}
p00 PETAL->PETAL[&PTULIP(16,1,0,0,3,0.1,0.1,4)]_ {E}
```

All the operations depend on the adjacency of the objects. In 3Gmap model the dependence on the context is expressed through the relations of adjacencies between various topological objects. In the grammar the following formalism of context dependency is used:

predecessor : block1 cond block2 \rightarrow successor,

where

- *cond* : a boolean condition that guards production application;
- *block1* : optional code lines always executed;
- *block2* : optional code lines executed only if *cond* is true.

Here is a short description of topological operations on volumes.

Growing. Creating a new volume and gluing it on one of the faces of predecessor, called a “support face”. This operation is denoted as: $VOLA \rightarrow VOLA[VOLB(n)]F$, where F is the support face of the volume $VOLA$ and n is the degree of the new volume $VOLB$.

Gluing. This operation glues two adjacent faces and is denoted as follows: $VOL : VOL_{F1} < VOL_{F3} \rightarrow VOL_{F1} | VOL_{F3}$, where the faces $F1$ and $F3$ of volumes labelled VOL are glued if they are adjacent.

Splitting. This operation splits a volume into two parts and is denoted $VOLA \xrightarrow{F} VOLB VOLC$, where F is a face of volume $VOLA$. Here all faces adjacent to F are split and the volumes, obtained with a face are closed.

The described operations are traditional and formally defined in the corresponding literature (for more details, readers can refer to (Lienhardt, 1994)).

3GMaps are topological objects without inherent geometric properties. In order to visualize them, we assign a set of parameters to each of the volume. These parameters refer to the topological properties (order of the prism N), material (m) and geometrical attributes (angles of the turtle: atx , aty , atz , dimensions of volume) of the volume. The geometrical interpretation of the model is obtained using these parameters. More precisely, the order of

them is as follows VOL(N, atx, aty, atz, H, L, W, m). The values of parameters can be specified directly in the rewriting rules or can be defined using instructions “#define”.

2.2.2 Modules

Some models can be very elaborated with a huge amount of detail. This can result into writing an intricate grammar, containing many lines of code and a lot of parameters difficult to control. Some parts of the grammar (for instance, lateral organs) can be extruded and serve as independent grammars. We call them - modules. These are the grammars that can be integrated into another grammar (see Figure 16).

Therefore we reduce the size of the grammar, by introducing a nested structure. This constitutes an advantage, as we can create very complex models and still maintain the conciseness and clearness of the grammar.

2.2.3 Application to the modelling of flowering plants

```
#define STEM
#define STEMS
#define BASE
#define PETAL
#define STAMEN
#define BASEC
#define CARPEL

#axiom : STEM

(p1) STEM -> STEM[STEMS(18)]E
(p2) STEMS -> STEMS[&SEPAL(4)]C1,C4,C7,C10,C13,C16
(p3) STEMS -> STEMS[BASE(18)]E
(p4) BASE -> BASE[&PETAL(4)]C2,C5,C8,C11,C14,C17
(p5) BASE -> BASE[&STAMEN(4)]C3,C6,C9,C12,C15,C18
(p6) BASE -> BASE[BASEC(18)]E
(p7) BASEC -> BASE[&CARPEL(6)]E
```

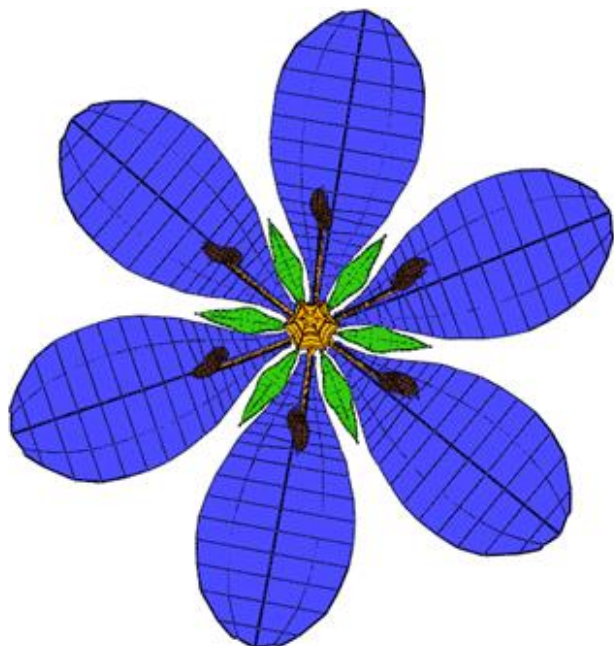


Figure 16 : Nested structure of the grammar. SEPAL, PETAL, STAMEN, CARPEL are modules, which are integrated into the initial grammar with the symbol “&”

While applying 3Gmaps L-systems to the modelling of flowering plants we have to follow natural laws, lying underneath their botanical structure. In section 1.1.1.2 we have outlined how the flower is arranged and we can mark out the main characteristics which we have to take into account while modelling. Symmetry, self -similarity, repeatability, the way the flowers are arranged on the stem are the essentials of most of the flowering plants.

In order to depict the arrangement of the lateral organs, according to the floral diagram we use the initial element as a short stem, the side faces of which serve as basis for growing the sepals (see Figure 17).

On the end face there is another building block which is the basis for growing petals and stamens on its side faces (see Figure 18).

Finally, on its end face there is a volume, on the top of each grows carpel. If there is more than one carpel, they will grow on the side faces of this block (see Figure 19).

In the grammar in Table 3 we use a growing operation, attaching the modules of lateral organs. Controlling the way the lateral organs are attached to the side faces of the basic building blocks we model a regular (with radial symmetry) or irregular flowers.

The inflorescences are also represented using the appropriate sequence of side faces of the stem, from which the whole flowers are growing (see Figure 20).

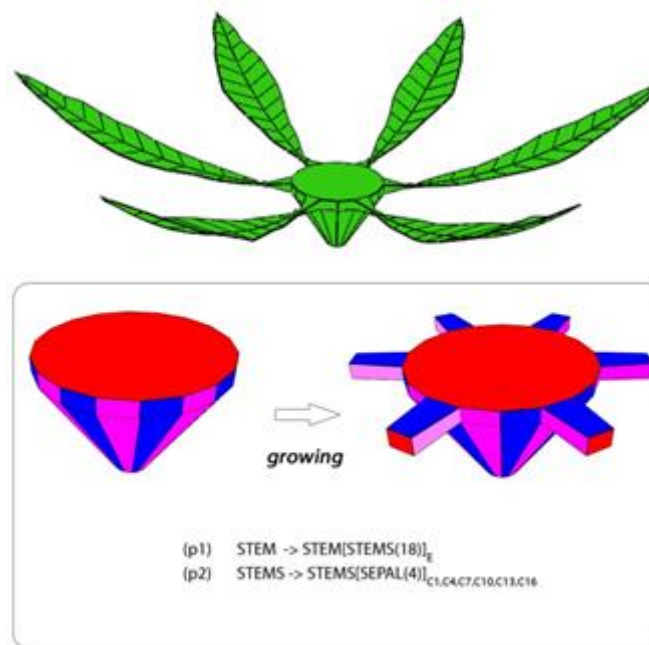


Figure 17 : First whorl of the floral diagram – calyx. Growing the sepals from the side faces of the stem

As we can see from the flower structure in Figure 3, the shapes of flower components have a more complex topology, which cannot be described by the most commonly used L-systems with one topological dimension. In some methods like (Prusinkiewicz & Lindenmayer, 1990) predefined surfaces and 3D shapes (generalized cylinders) are incorporated to each symbol, during the final representation. But predefined surfaces and 3D shapes do not “grow”. String symbols have very little control over the integrated organs. However, we need to simulate plant development fully; therefore we have to use topological

3d dimension structures. We use 3Gmap L-systems to describe them. They are integrated into the grammar as modules, which are grammars too. Thus the entire grammar has a nested structure, making it more intuitive to control.

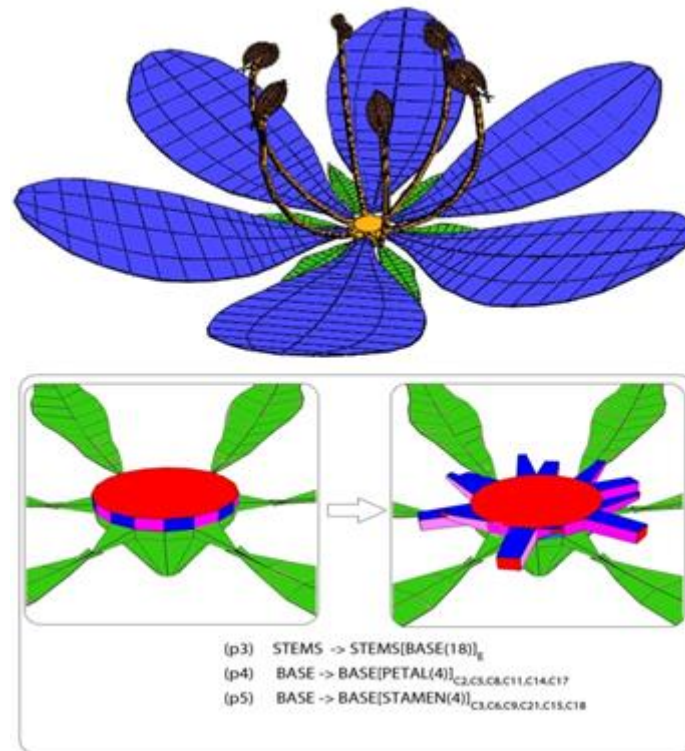


Figure 18 : Second and the third whorl of the floral diagram: corolla and androecium. Growing petals and stamens from the side faces of the base from the side faces of the base

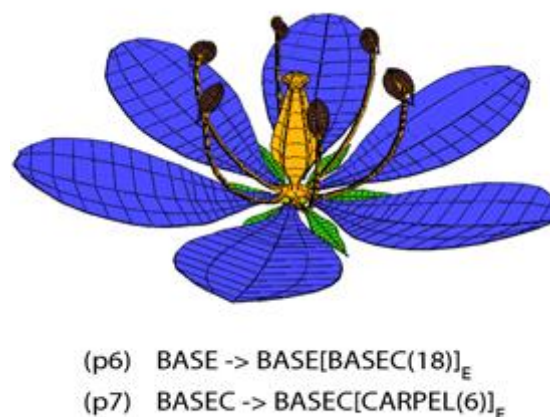


Figure 19 : The fourth whorl of the floral diagram – gynoecium

Lateral organs are mostly volumetric. Even if leaves, sepals and petals seem to be flat and can be represented with predefined surfaces, we consider them as having an internal structure consisting of various layers, thus the third dimension, the thickness, is also taken

into account. For example, in order to model a petal, we use its central vein as an initial building block. On the side faces of the vein volume grow the right and the left lobes of the petal, forming 3 layers: upper epidermis, mesophyll and lower epidermis (see Figure 5). After various derivation steps, the number of building blocks has increased and such faces as the sides of the lobes are glued (see Figure 21). Finally the topology of the petal is defined, that is the neighbourhood relations between 3Gmaps are established, passing the baton to the geometrical interpretation.

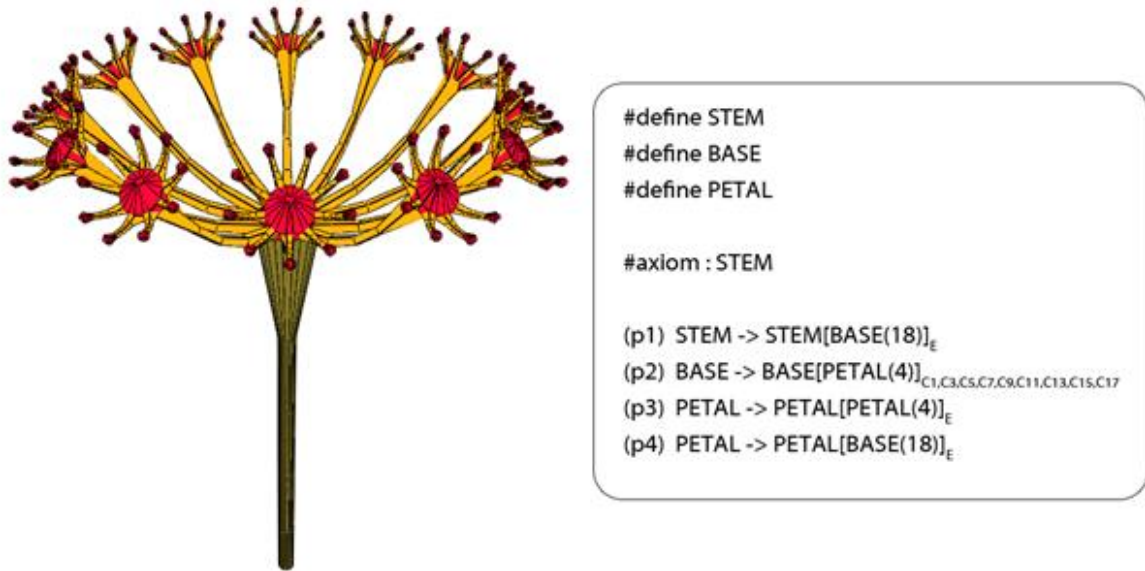


Figure 20 : Compound double umbel inflorescence. The model is represented with 2 derivation steps

In order to shape the lateral organs into appropriate forms we use parameters. The curvy shapes are achieved by using equations of linear regression model, where the independent variable is a derivation step. In Figure 22 we can see how, the parameters for linear regression model are determined, based on the set of points. The obtained equation (1) is then used as a parameter in the grammar, where x is a derivation step, y represent the width and height of the volume and a, b, c, d, f, g, h are parameters of linear regression.

$$y = a + bx + cx^2 + dx^3 + fx^4 + gx^5 + hx^6 \quad (1)$$

Using random functions as parameters we can add some irregularities into the final shape thus creating more realistic models of flowers (see Figure 23).

It is also possible to create the models of a huge amount of flowers, like meadows or fields, due to the nested structure of grammar. In Figure 24 a small meadow is created, where some of its building blocks are represented as “seeds” from which the flowers will grow. The

entire structure of the flower is stored in a module and is integrated into the principal grammar, by using the grow operation onto the “seed” building block.

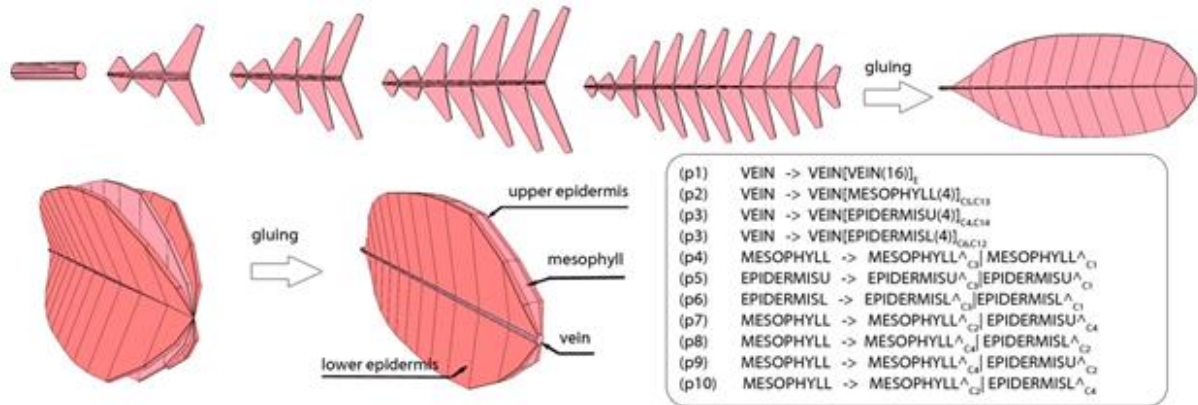


Figure 21 : Petal, consisting of three layers : upper and lower epidermis and mesophyll. The upper images are the results of 11 derivation steps, growing and gluing operations

2.2.4 Materials

Flowers are not homogeneous structures and consist of lots of different components. Each one of them has its own properties and not only distinct shapes but also distinct materials. While writing a grammar we try to compose the volumes in such a manner that the main components of the flower can be quite distinguished. So one or another group of volumes can represent a petal or a leaf, etc. And of course they differ in colour and illumination properties, as well as texture, etc. Thereby we decided to add a new parameter in a volume definition of grammar. It represents a material of a flower organ and is denoted as integer, which is a number of materials listed in material.mtl file, containing definition of its various properties. This function is also useful for the models having an internal structure. Flower tissue is not homogeneous and consists of several layers: upper epidermis, lower epidermis, veins, and mesophyll. If we create a model of a petal with all these layers, we assign to each one of them its proper material. Using this contribution the rendered results look much more realistic, as each component of a flower has its own colour, illumination properties, etc.

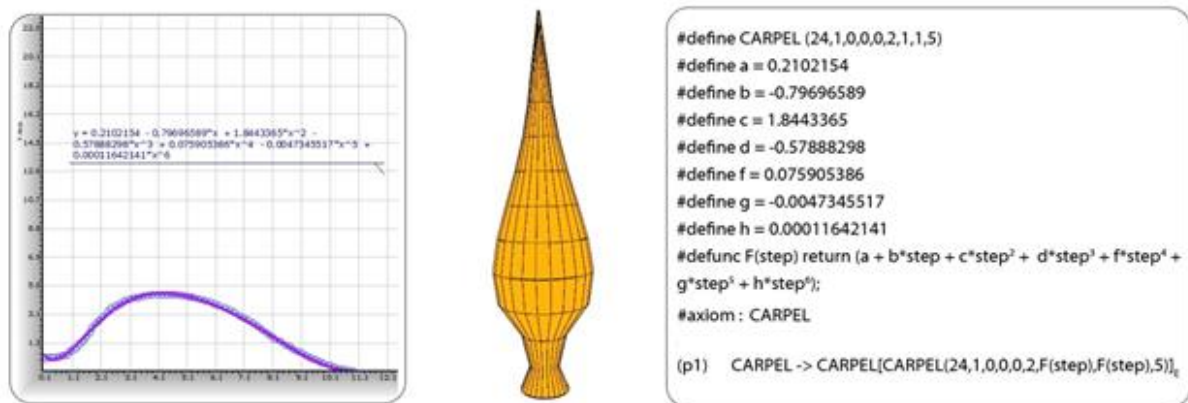


Figure 22 : The equation of the curve, with parameters for linear regression model, based on the set of points. Carpel is constructed using the equation for its width and length parameters with 11 derivation steps

2.3 Flower rendering

The models were rendered with Blender 2.49b. We used 2 spot lights and a lamp. One of the spot lights was placed behind the model (see). For each model we define a set of materials with Lambert diffuse shader, Cook-Torrance specular shader, translucency property and subsurface scattering function.

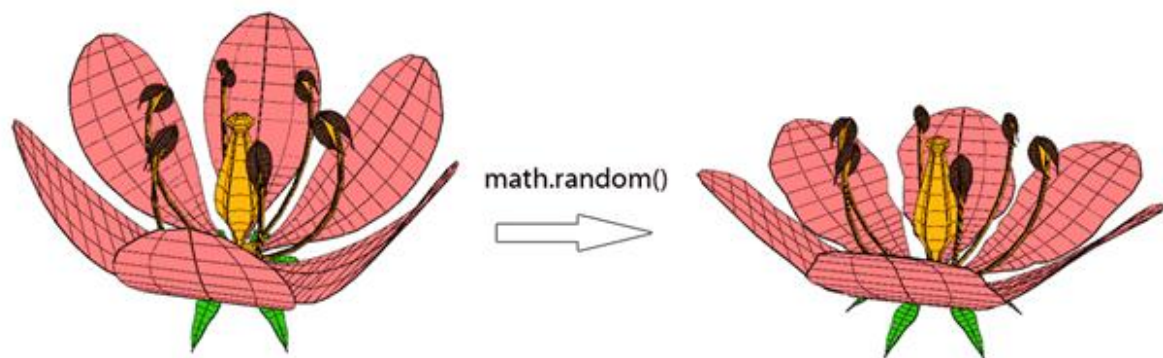


Figure 23 : Applying random function as parameters of the volumes

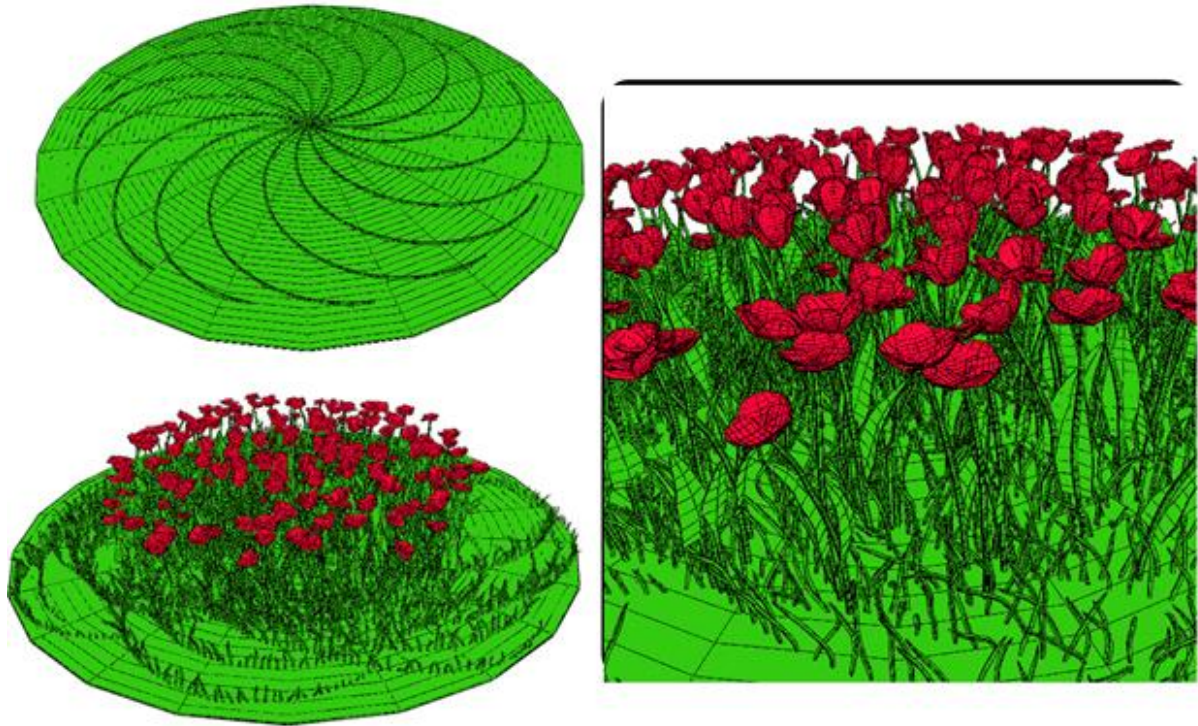


Figure 24 : Meadow of tulips model. The tulips are modules integrated into the initial grammar. The model is constructed with 13 derivation steps

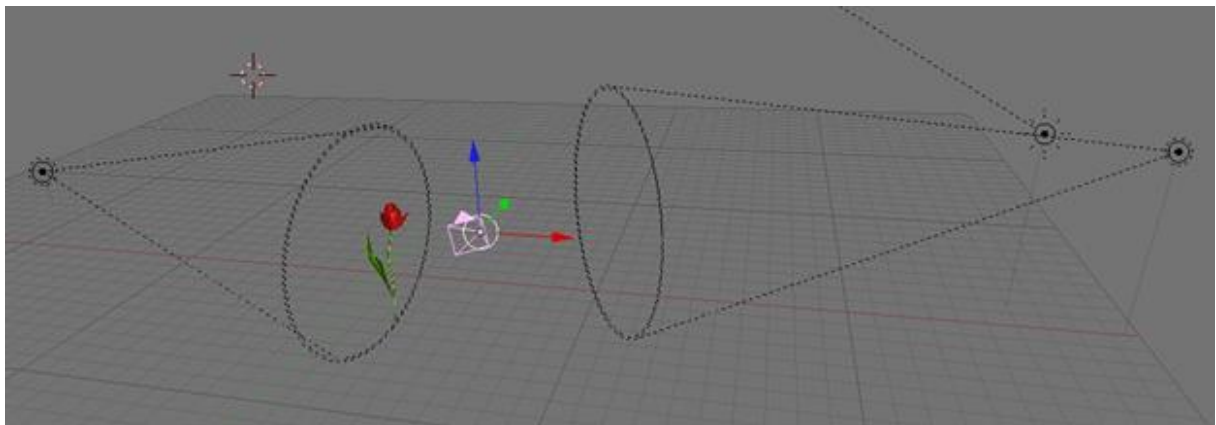


Figure 25 : Camera and light position of the scene

In Figure 26 we can see the difference between the results rendered without and with translucency and subsurface scattering.

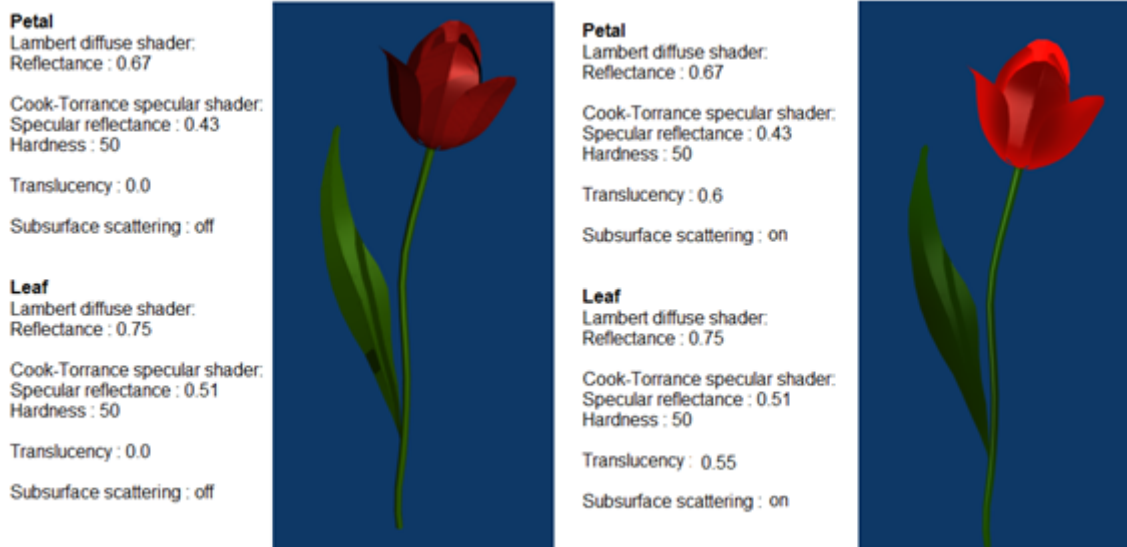


Figure 26 : Tulip rendered without and with translucency property and subsurface scattering function

2.4 Results

Our method allows getting a mesh of huge variety of flowers, as well as flower compositions and terrains. The software was developed under Linux Ubuntu , in c++ using cross platform library Qt,. The output is exported to obj format with the information of material assigned to each volume composing flower geometry.

In Figure 27 we can see a bluebell and tulip flower models. Here we can see that all the petals slightly differ one from another with their shapes and angles. To obtain this we used random values of parameters for the petal grammar. To make a grammar more intuitive we also used a nested structure adding modules for each component.

The geometric models are then rendered using Blender version 2.49. In Figure 28 several species of flowers are presented. Here using interactivity function we can easily adjust the shape of all flower components to get more realistic models.

Our method also allows creating realistic terrains of flowers just using one grammar. We construct a terrain containing volumes and add modules which represent different flowers using random parameter values. In Figure 29 we can see several terrains of flowers, where each flower is distinct from another due to passing random values of the parameters of the modules. The times of generation of the models, represented in Figures 28a, 28b, 28c and 28d is 37:5 s, 120:4 s, 56:3 s, 63:4 s respectively. This possibility of automatic creation of a huge amount of flowers in a quite short period of time is a great advantage in the area of plant

modelling. Besides the entire flower samples are different one from each other even if the species of the flower are the same.



Figure 27 : Bluebell and tulip flower models

2.5 Conclusion

We introduced a method of flower generation based on 3Gmap L-system, allowing to create a grammar in order to construct flower models. Once written one grammar we can get a great variety of flowers by simply changing the values of its parameters. In that way we have a possibility to obtain complicated scenes with a large number of different flowers with a minimum amount of work on the grammar. We also took into account the needs of the user to have an intuitive modelling tool. Thus the shape of the flower can be modified interactively and the grammar has an intuitive structure allowing to use the modules. Due to volumetric model we can construct the internal structure of flower tissue, which consists of several layers. Assigning a material with special properties to each layer, we would be able to get more realistic results while rendering.

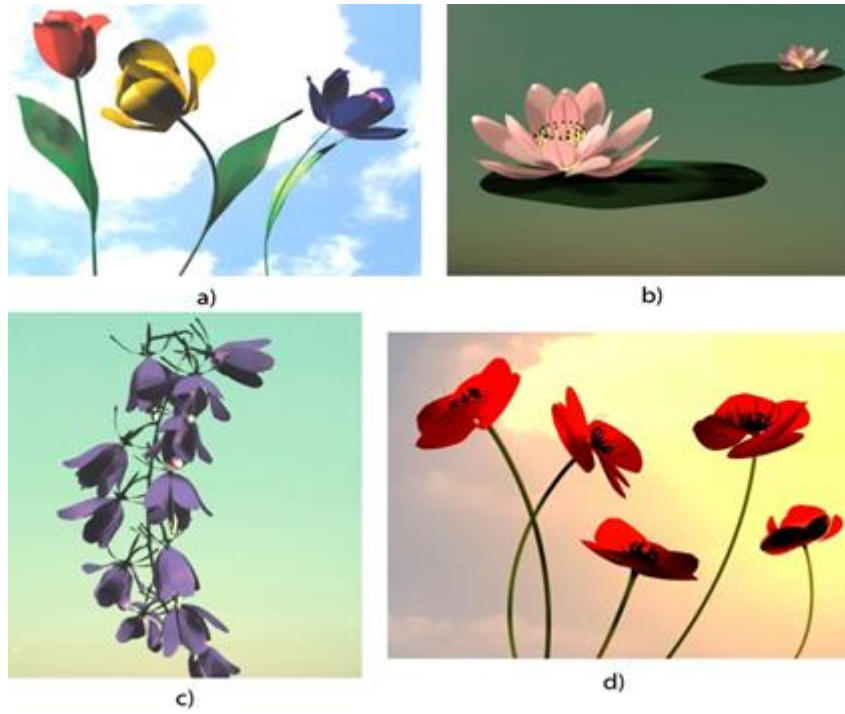


Figure 28 : Models of a) tulips , b) water lily, c) bluebells and d) poppies

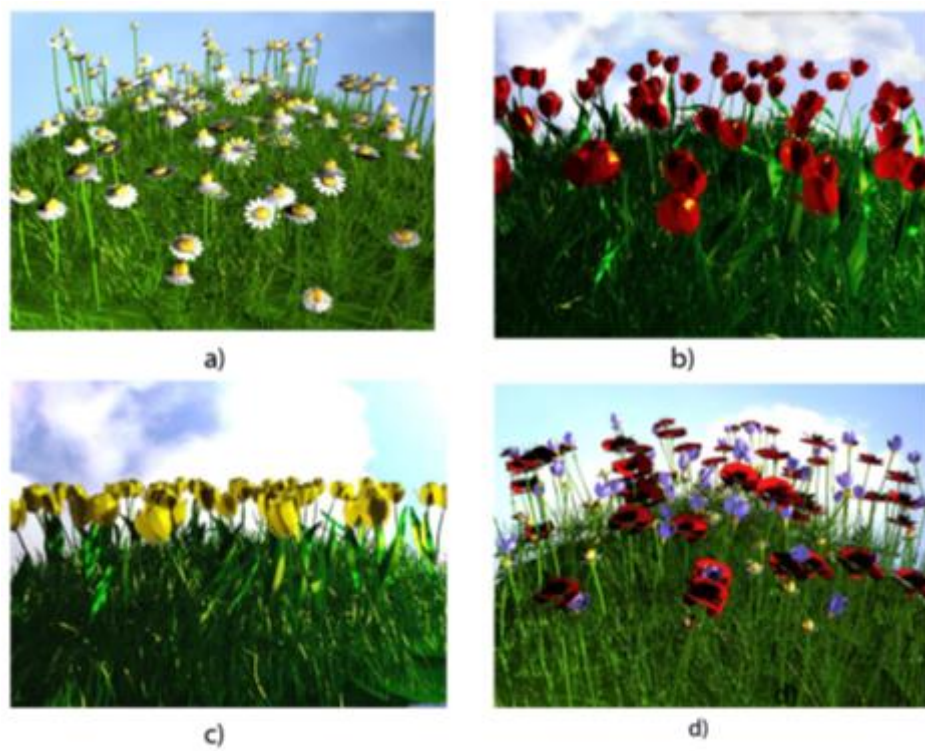


Figure 29 : Models of flower fields and meadows

Chapter 3

Interactive modelling of flowering plants

3. Interactive modelling of flowering plants

3.1 Introduction

As we have mentioned before, we can distinguish two different approaches to flower simulation. The first one is aiming at getting a plausible model while the botanical correctness is usually disregarded. Here the task of modelling is undertaken mainly by the user describing a plant structure and its components and defining the required parameters. The degree of realism depends on the users skills. This approach is quite intuitive for a common user, but has the inconvenience of creating each sample from scratch in case of generating a variation of slightly different flowers (Ijiri et al., 2005), (Quan et al., 2006).

The other approach could be referred to as procedural modelling, which tries to provide biologically faithful and visually realistic models (Prusinkiewicz & Lindenmayer, 1990). Most of these approaches are based on L-Systems, which can generate complicated multicellular structures from a small number of rules. They are able to get a lot of flower samples based on a single grammar by simply changing the parameter values. Although these methods can provide impressive results, the underlying algorithms are not so intuitive for common users.

An analysis of previous work points to look for some kind of symbiosis between these two groups of approaches in order to simplify the task of the user and at the same time to retain the realism of the models. Pursuing this goal we propose to combine L-systems grammar writing with interactive control of parameter settings. In order to avoid the laborious task of grammar writing we propose a new interface function: the inverse modelling by automatic generation of L-systems. The user describes the flower he wants to model, by mentioning the properties of its organs. The algorithm uses this information as an input, which is then analysed and coded as L-systems grammar. The user can control the final result by interactively setting the parameters of the grammar. These contributions make the task of a user more obvious and intuitive which in turn enables to create more accurate models. In addition, the way the model is built allows us to take into account its internal structure. As the flower tissue is non-homogeneous, this can be quite useful to render more accurate subsurface scattering.

3.2 Inverse problem challenges in flower modelling

The methods of inverse modelling group attempt to obtain feature constraints, relations, and dependencies based on the observation of the given model. Automatic generation of procedural rules has been an open problem for a long period of time. Its achievement would constitute a great advantage as the rules allow generation of classes of similar models, the internal structure of the model could be modified, the model could be represented by its generative rules (compressed), the syntactic analysis of the rules could be used for image analysis, and so forth (Stava et al., 2014). This is not a trivial task and the obtained models up to now have limited functions, which require a further research.

Procedural models have an ability to generate a wide range of complex structures from a small set of specified parameters. However precisely because of the nature of this ability, the procedural models are hard to control as a small perturbation in the parameter space results in a significant change in the resulting structure. The obtained grammar with a current set of parameters can correspond to the current model of the flower, but it does not guaranty that with the small change of parameter values the result would correspond to a botanically plausible flower model. Besides, because of the huge diversity of flowering plants, the gaps between the problem and solution domain and between concrete and abstract are rather vast. Perhaps this is the reason that few work has been done on the inverse procedural models.

We propose a hybrid approach to solving the inverse problem of modelling of flowering plants. Our method involves first defining the structure of the abstract flower, and then applying 3Gmaps L-Systems.

3.3 Inverse grammar generation interface

The whole process of writing a grammar is not intuitive and tedious. But still the grammar itself represents a valuable piece of information, as it has in its code an unlimited variety of potential flower models. In addition, storing geometries in a grammar format is much more compact than storing it in a conventional way. Taking into account all these advantages we added a new functionality of inverse procedural modelling. Here we have to face up to the inverse problem that is given as input the description of flower we have to find a grammar which contains the approximate structure of this input.

Our interface provides an option to describe the flower we want, instead of editing a grammar. By choosing an appropriate organ shape, its size, its growth nature, etc. the user

defines a framework for generating a grammar. We are following the biological rules of flowering plant structure and growth, managing an interface in a way that the user can take into account every organ of the plant, resulting in a botanically correct flower.

We are trying to create the models of botanically correct structures but of course we cannot pretend to develop a tool which could be able to describe all the existing types of flowers. There is an enormous variety some of which have not even been studied yet.

Let us have a look at an example to see more clearly what we are dealing with. Let us suppose the user wants to generate a grammar of the tulip flower. In order to do this he/she has to describe the flower structure, which we have divided into 6 control points: stem, leaves, and the four flower rings (calyx, corolla, androecium and gynoecium). Figure 30 depicts the interface containing all control points of the plant, represented with the tab menu bar.

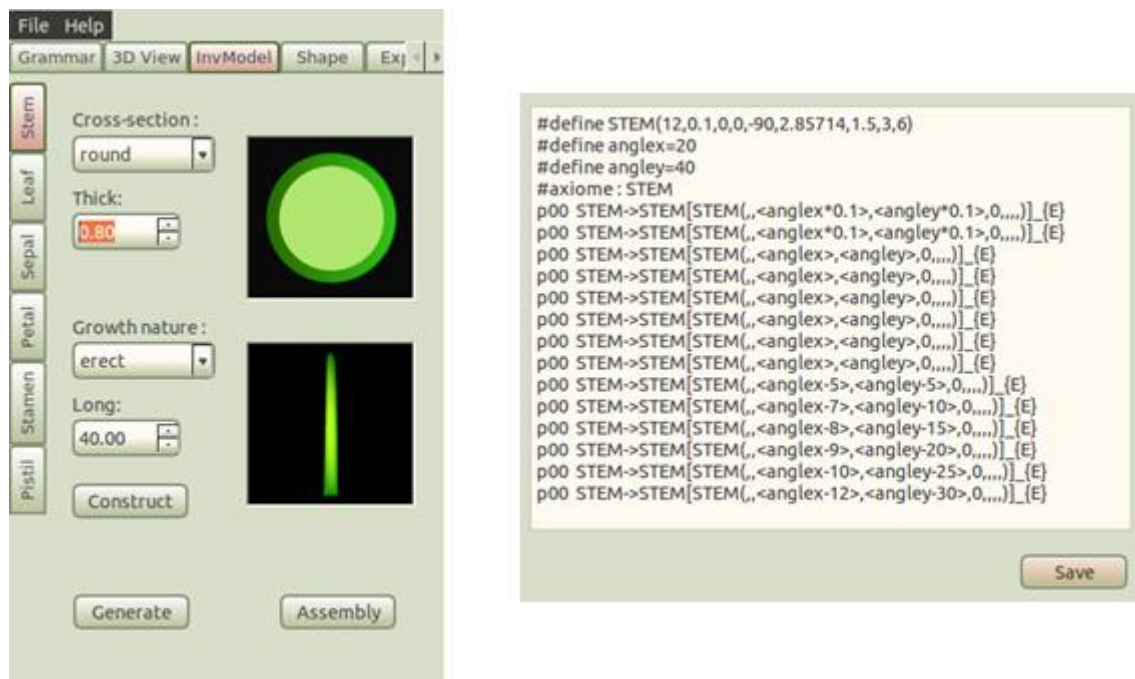


Figure 30 : Inverse grammar generation interface. The control points are represented as tabs and located on the left of the main panel. The first control point – the stem. The grammar of the stem generated according to the description

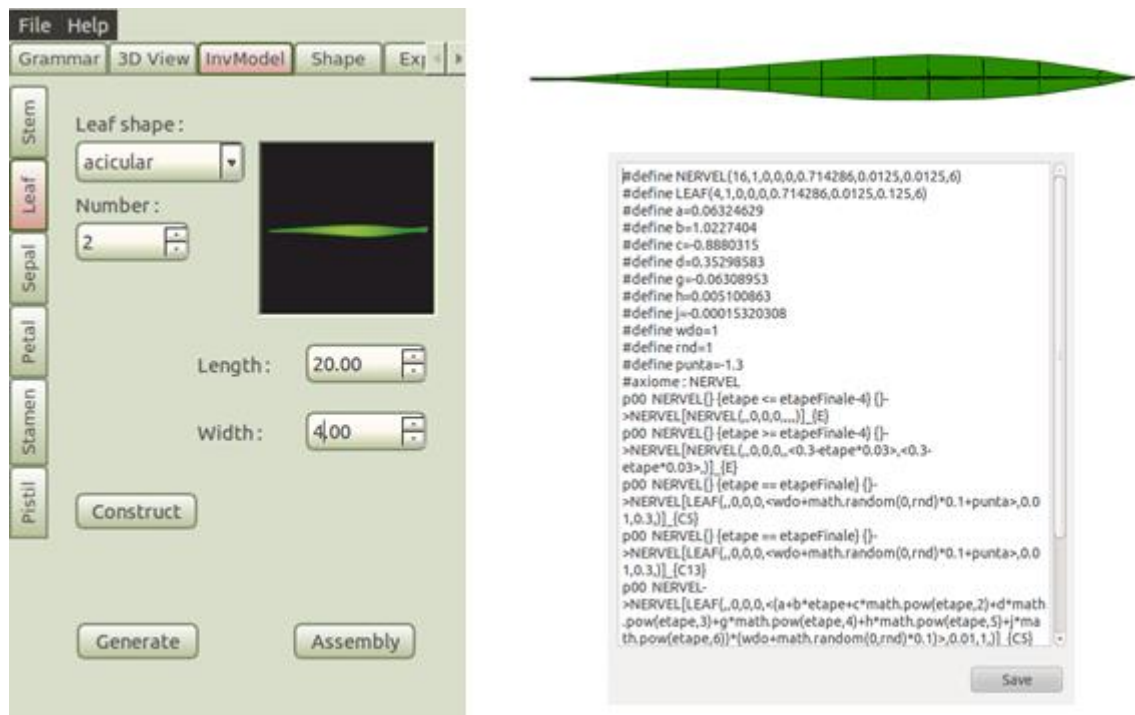


Figure 31 : The second control point – the leaf. Generated geometrical model of the leaf and its grammar

Starting with the first control point, we describe the stem of the tulip which has a round form of cross-section and erect growth nature. We also can mention its length and radius size or leave it for the default values to be automatically assigned. The grammar is then generated and stored in a file.

The second control point is a leaf (see Figure 31), which has several characteristics to be determined. We chose the value of contour form as apical, length, width and thickness of the leaf, and the quantity of leaves on the stem.

Checking the absence of sepals on the third control point, we are passing directly to the fourth control point which is petal (see Figure 32). In order to define a petal we distinguished several properties, such as: its size (length, width and thickness), shape, material and quantity.

Choosing a rounded shape, the size and quantity of 6 petals we are moving to the next control points. The fifth and the sixth control points are the descriptions of flower reproductive organs: stamens and carpels.

The stamen (see Figure 33) is that part of a flower that looks like a thin hair with a follicle on top. Usually there are several stamens surrounding the pistil(s). The hair is called the filament and the follicle is the anther where pollen is produced. The filament and anther together make up the stamen. Tulip has 6 stamens, whose anthers are attached at its base to

the filament. Choosing the number of stamens as 6, with basifixed attachment, we are passing to the last ring of the flower (see Figure 34).

The carpel or pistil is in the very centre of the flower and contains the female organs. There may be one or more pistils. This part of a flower resembles a bowling pin in shape with the rounded lower base being the ovary. Coming up from the ovary the pistil narrows into a neck called the style, and the knob at the top of the neck is the stigma. Tulip has multiple connate fused carpels. Therefore we chose a syncarpous property.

3.3.1 Grammar generation and assembling

Our inverse modelling interface permits creating the grammars of the separate flower organs, as well as the whole flower grammar.

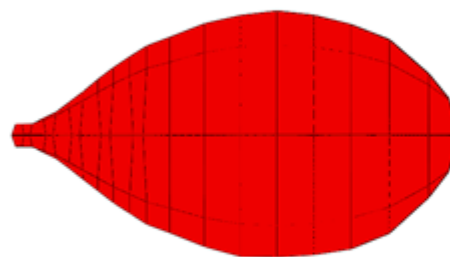
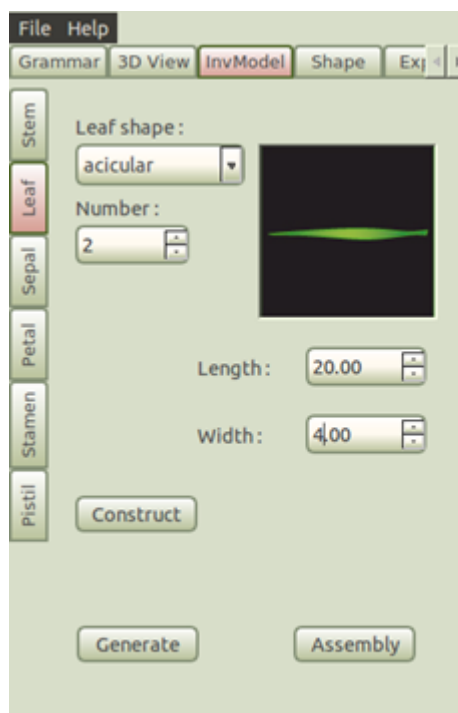


Figure 32 : The fourth control point – the petal. Generated geometrical model of the petal

If the user wants to obtain the grammar of some flower organ, he/she has to indicate all the properties that this organ must have. As it is explained in the previous section the user goes through the control points and chooses the properties he wants. For example, defining the leaf for a tulip flower, the user chooses acicular leaf shape in the corresponding control point. The program connects with a database, searching for the acicular value in a leaf shape table, and then extracts the necessary parameters, which are the equation coefficients (see

Figure 22). The extracted parameters and its values are then stored in a vector and will be used by the program later.

The method of grammar generation is based on templates grammars, which are predefined grammars corresponding to each control point. Every template grammar has a particular list of volumes, parameters and rules describing a generic organ structure. We have 4 templates corresponding to stem, petal-like units (leaf, sepal, petal), stamen and pistil. We store the template in XML format, arranging its content as a tree of tags and its default values. The grammar structure is represented in Table 4.

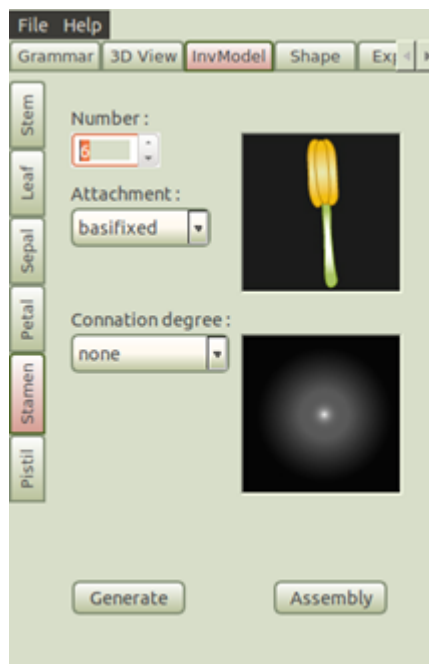


Figure 33 : The fifth control point – the stamen. Generated geometrical model of the tulip's stamen

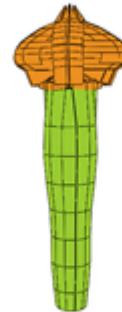
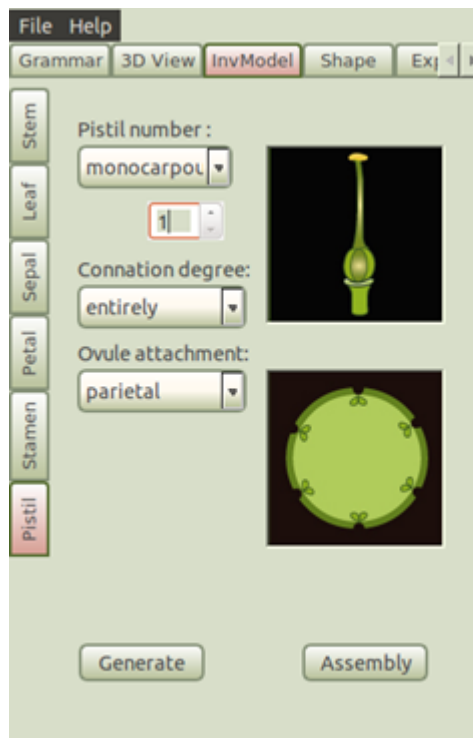


Figure 34 : The sixth control point – the pistil. Generated geometrical model of the tulip pistil

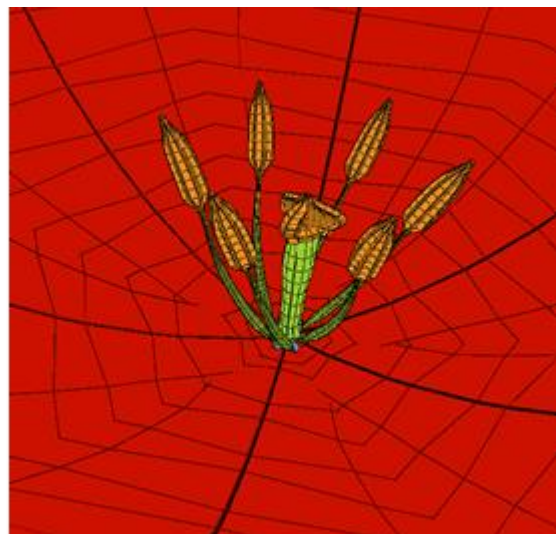


Figure 35 : Left: assembled model of tulip. Right: the closer view of the tulip flower

Table 4 : Grammar representation in gl3 and xml formats

gl3	xml
<i>volume description</i>	
#define STEM(8,0,0,0,0,1,1,2,1)	<pre> <volume category="#define"> <name>STEM</name><degree>8</degree> <transl_coef>0</transl_coef><x_rotation>0</x_rotation> <y_rotation>0</y_rotation><z_rotation>0</z_rotation> <height>1</height><length>1</length><width>2</width> <material>1</material> </volume> </pre>

variable definition	
#define thickness=7	<variable category="#define"> <thickness>7</thickness> </variable>
axiom	
#axiome: STEM	<axiom prefix="#axiome : ">STEM</axiom>
set of production rules	
p00 STEM->STEM[STEM(8,0.1,0,0,0,1,1,1,1)]_ {E}	<rule> <predecessor init="p00 "> <volume category="predcsr"> <name>STEM</name><degree/> <transl_coef/><x_rotation/><y_rotation/><z_rotation/> <height/><length/><width/><material/> </volume> <block1/> <condition/> <block2/> <element>-></element> </predecessor> <successor> <volume category="root"> <name>STEM</name><degree/> <transl_coef/><x_rotation/><y_rotation/><z_rotation/> <height/><length/><width/><material/> </volume> <element>[</element> <volume category="successor"> <name>STEM</name><degree/> <transl_coef/><x_rotation>0</x_rotation> <y_rotation>0</y_rotation><z_rotation>0</z_rotation> <height/><length/><width/><material/> </volume> <element>]_</element> <face>{E}</face> </successor> </rule>

Once the user defines all the properties for a particular control point (a leaf for a tulip flower in our example), the program then opens a template for a petal-like unit for parsing. The program will now use the stored vector of parameters. The DOM parser goes through the template, putting the necessary parameter values in its places and thus generating the final grammar.

In order to get the whole flower we have to assemble all control points. Here the system collects all the input information and unifies all the grammars of flower organs into one flower structure grammar (see Figure 35).

3.4 Interactive parameter adjustment

After obtaining the grammar of the organ or flower, the user may want to modify the resulting geometry. He/she would have to come back to the grammar, modify the values of appropriate parameters and reload the grammar once and before he sees the final result. Our system allows us avoid this laborious work, providing a functionality of interactively

changing L-systems parameters. We do not have to recreate the model, but, reusing its topology, reload the embedding of the model (see Figure 36). This constitutes a great advantage because separate management from the topology and from its embedding simplifies the algorithms allowing us to easily create lots of model variations. This way of changing parameter values is quite faster as it only takes into account the embedding part, leaving the topology part of the program untouched.

Let us come back to our example of tulip flower. For the moment, the model is very static. The user can now interactively manipulate with every organ. In Figure 37 we can see how different components of the flower are changing. A user just selects an appropriate volume and, by clicking on a box of parameter values, changes a model shape on the fly, observing the results at the same time. Groups of different volumes can represent flower organs. In order to control them we used variables, which represent different flower component measures. The variable values can also be interactively changed, thus making the flower shape managing even more intuitive and faster.

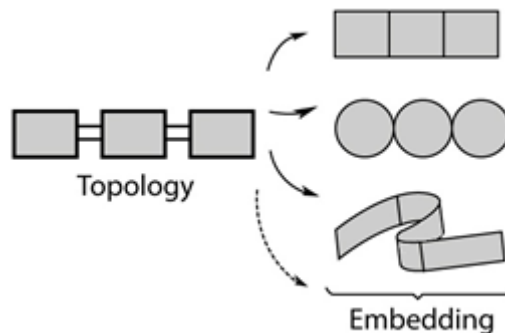


Figure 36 : Separate management from topology and its embeddings

We can also change the topology of volumes interactively which simplifies the creation of models with different number of petals, leaves, etc. In this case the order of the prism is changed and we have to rerun the grammar from the very beginning which is a much slower process than the previous one. Nevertheless, this makes the process of modelling more intuitive for the user.

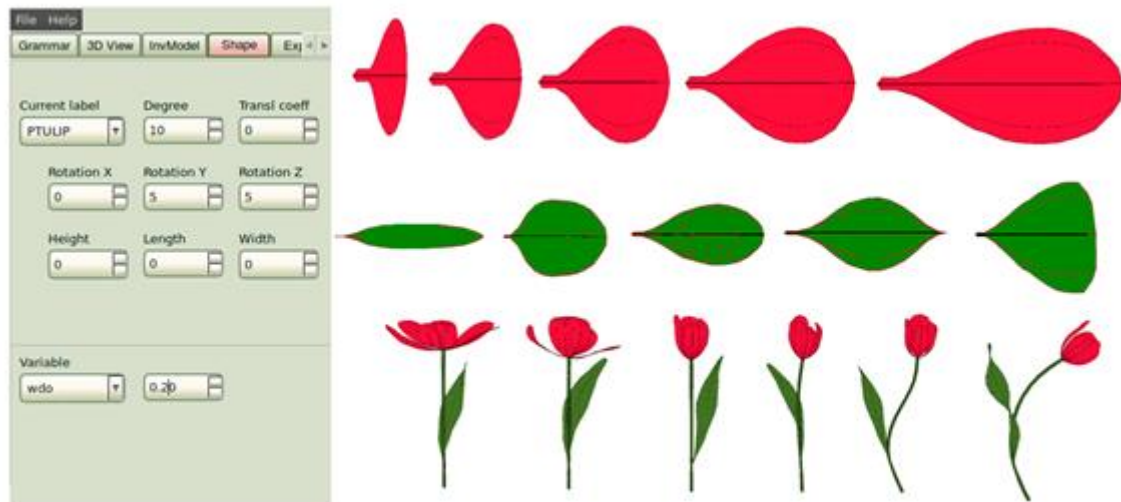


Figure 37 : Interactive control of flower shapes, by changing the values of parameters

3.5 Results

Here are some examples of using 3Gmap L system interface functionality of inverse modelling.



Figure 38 : Interactive control of flower shapes, by changing the values of parameters

The globeflower was modelled by describing the control points and then assembling the organs together. After that the model was adjusted using an interactive control interface (see Figure 38).

We can also mix the inverse grammar modelling with the manual grammar writing. Some flowers have very complicated structure and could be modelled only with individual grammars, which the user must write himself. Nevertheless he/she can combine the

complicated parts with those which can be modelled with the inverse grammar generation tool. The model of a sunflower was obtained by writing the grammar of its complex headflower and combining it with the grammars of organs (such as petals, sepals and leaves) obtained by inverse grammar generation tool (see Figure 39).



Figure 39 : The numerous florets which are crowded together with spirals were modelled manually, while the external petal, sepal and leaves are modelled automatically

All the geometrical models were rendered with 3Ds Max 2012, using mental ray (see Figure 41). The advanced user can use the existing grammars to create fields of flowers (see Figure 40). In Chapter 3 the folded structure of the grammar was explained and some examples of constructing the flower fields were presented. Using this method we unify all the flowers we need into one grammar. Random values of parameters provide slightly different geometries using the same grammar, thus making the results more realistic.

The comparison of real flowers and rendered models of poppies, daisies, globe-flowers, dandelions, bluebells and sunflowers are presented in Table 5.

3.6 Conclusions and future work

In this chapter we have presented intent to combine the two modelling groups, in order to ease the task of the user while preserving the plausibility and efficiency of procedural methods. This application provides an intuitive interface which permits the user to create grammars in a more comprehensive level. The user doesn't have to write an intricate code of

the grammar, but with the help of our interface, he/she can define the flower characteristics, which are used for automatic grammar generation.



Figure 40 : The rendered models of flower fields. Above : poppies. Below : globeflowers and dandelions

Table 5 : Real flowers and modelled flowers comparison.

Real flowers	Modeled flowers
Poppies	
	
Daisies	
	
Globeflowers	
	

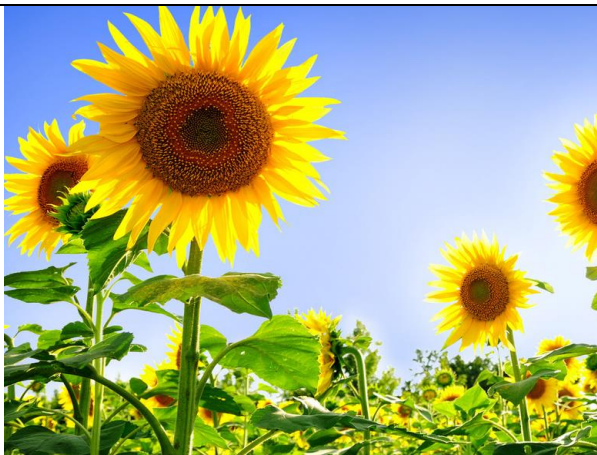
Dandelions



Bluebells



Sunflowers



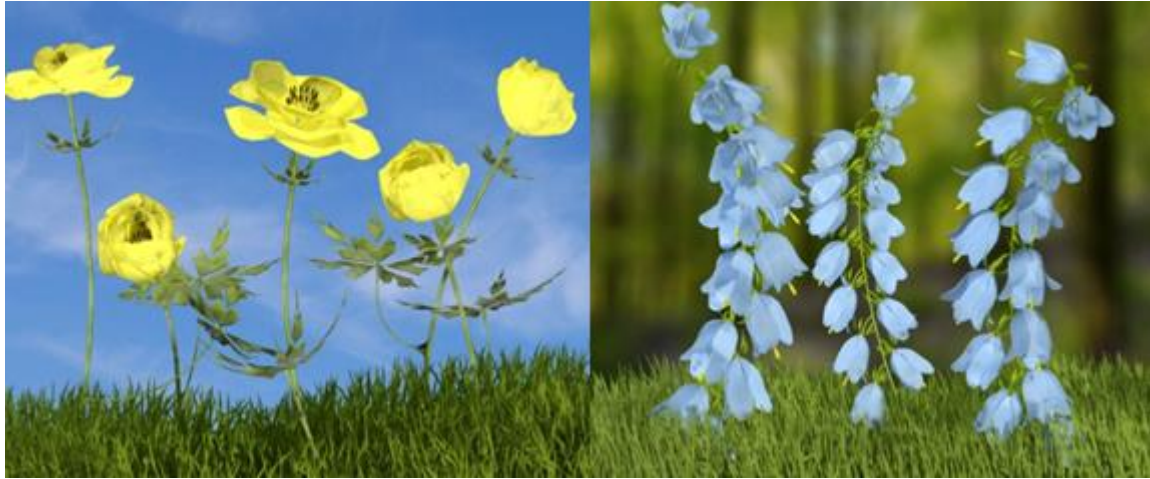


Figure 41 : Rendered models of globeflowers and bluebells

There are several straightforward improvements on our current implementation. The assembling of the flower model is manual, which requires the user to collaborate partly in grammar writing. A needed extension of the system is an automatic assembling of the flower organs. This includes an implementation of the parser collecting the grammars of the flower organs and plugging them into the base flower grammar. Another direction of the future work is enlarging the data base of the template grammars, in order to describe more precisely the flower we need.

Chapter 4

Interactive modelling of flowering plants

4. Flower modelling and Kinect

4.1 Introduction

Virtual worlds in videogames have grown in size and complexity as computers increased in performance. The creation of these worlds requires many hours by expensive modellers to provide a believable experience to the final user. As technology advanced, more tools have been provided to modellers to ease their work. For example, interactive terrain editors allow the modeller to paint over a terrain texture, and the colours of this texture are used by the game engine to add trees or other vegetation, using a predefined set of models. While the results are quite good for casual walkthroughs through the terrain, closer inspection shows that only a discrete number of models are being used to populate the terrain. In our framework, the models are located in the terrain using similar techniques, but each model instance is requested to a flower server. Since the flower models are parameterized using a customizable grammar with different tuneable parameters, we can guarantee that all the flowers will be different, providing the final user with a much more believable world even when performing close inspections of the models. The grammar model ensures that all the flowers are physically plausible. In order to lighten the task of the user we combine 3Gmap L-system with the natural user interface by means of Microsoft Kinect. Analysing the gestures of the user, Kinect provides basic interactions, which are reinterpreted as signals for changing parameter values of the grammar, and which in its turn returns a modified geometrical model of the flower. Using simple gestures the user can create new flowers or interactively modify its shape, such as the curvature, the length and the width of each of its organs.

4.2 Microsoft Kinect

To hide the complexity of the L-systems from the final users, we can map the different parameters to different gestures using a Microsoft Kinect (Microsoft, 2012). A Kinect, originally intended for the Xbox 360 game, is a webcam-style add-on peripheral designed to support the most natural ways of communication with the computer: gesture recognition or spoken commands (often referred to as natural user interface). It is both equipped with a colour camera and an infrared projector extended with a sensor providing depth information, turning the Kinect into a low-cost, real-time full body 3D motion capture device. According

to the documentation, two skeletons, and up to 6 people within its field of view can be detected. For a single skeleton, 20 joints can be used in standing posture and 10 joints while sitting. By analysing the gestures and poses of the user, Kinect can provide basic interactions similar to mouse, keyboard and touch interactions (i.e. selecting buttons, zooming and panning around a surface). Kinect control has been successfully applied to many different areas, such as computer games and entertainment, education or healthcare. The creation of intuitive gestures allows the user to control a large quantity of parameters seamlessly. In addition, Kinects have been used to directly measure vegetation structure (Azzari et al., 2012), to track plant leaves (IRI, 2011), and to segment them (Wallenberg et al., 2011).

4.3 Flower generation using Kinect

In the following sections, we describe the architecture of our system, the content generation module and the natural interface module.

4.3.1 System architecture

The architecture of our system is as follows (see Figure 42):

- A command line flower generator reads the grammar and applies the requested transformations to create a unique flower, and generates an OBJ file using the libraries developed to support the system presented in Chapter 2.
- A webserver using Common Gateway Interface (CGI) provides the interface between the flower generator and the clients requiring the flowers (Table 6). A unique url provides the information of the base grammar, the depth and the different values of the parameter space. Accessing the url produces the corresponding OBJ file.

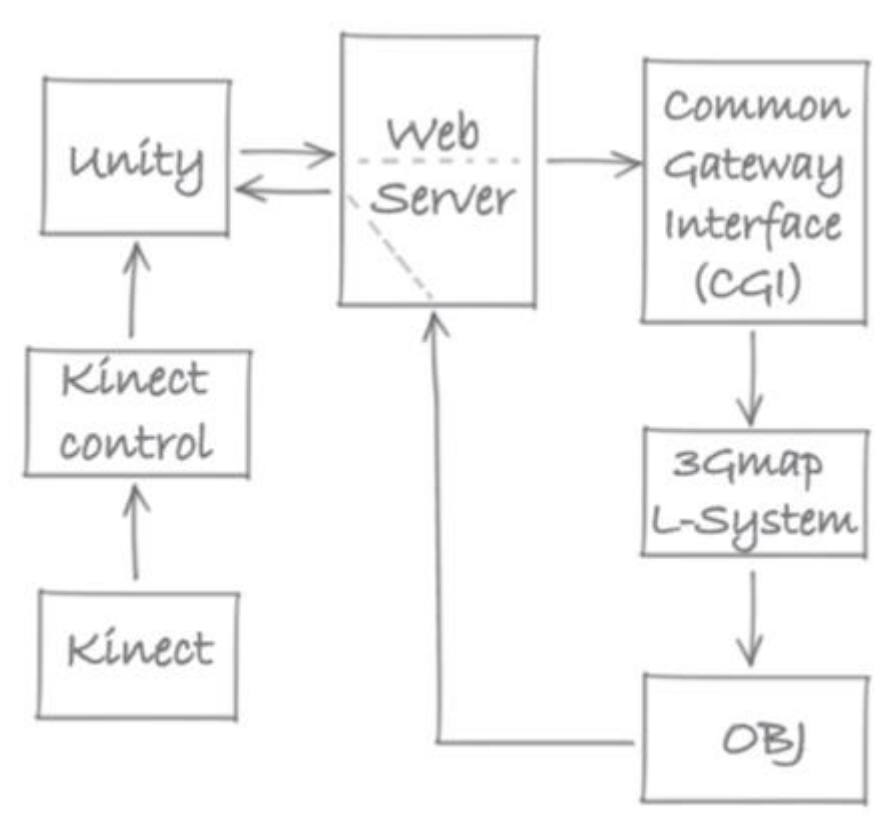


Figure 42 : Pipeline of our framework

- A library of routines running in the Unity game engine (Unity_Technologies, 2013) can be used to load either a specific flower or to generate flowerbeds (see Figure 48). The flowerbeds can be parameterized setting the minimum and maximum values of the different flower parameters, and each flower is generated by sampling uniformly in the desired parameter space. The url for the flower is used to retrieve the OBJ file with the flower model. The number of flowers and their density can also be chosen by the user.
- We have used a library of gestures on top of the standard OpenNI unity sdk (Zigfu, 2013) and the NITE middleware, which is described in (Rodriguez et al., 2013). The intensity of these gestures has been mapped into the available range of the corresponding flower parameter.

This architecture allows us to separate the flower generation from the rendering, and is less demanding for low-power devices such as mobile phones, since the flower generation is run on a separate server. As an example, Figure 43 shows the rendering of different flowers in an android device. Additionally, the web server provides authentication, authorization and encryption, a possibly useful feature in DRM schemes, and can hide the grammars from the final users (if needed).

Table 6 : CGI script to interface with the flower generator

```
#!/bin/bash
echo Content-type: text/plain
echo
saveIFS=$IFS
IFS='&'
parm=($QUERY_STRING)
IFS=$saveIFS
./CL3Gmap ${parm[0]} ${parm[1]} ${parm[2]}${parm[3]} ${parm[4]} ${parm[5]}
${parm[6]}
${parm[7]} ${parm[8]} ${parm[9]}${parm[10]} ${parm[11]} ${parm[12]}${parm[13]}
${parm[14]} ${parm[15]}${parm[16]} ${parm[17]} \
2>&1 >/dev/null ||
echo "Error running CL3Gmap"
cat export/out.obj
rm -f export/out.obj
```

4.3.2 Content Generation

The content generation routines have been implemented as scripts running in the Unity game engine, which provides support for different architectures (Microsoft Windows, Mac OSX, Android, iOS and Flash). The routines are portable, and are only constrained by the processing power and memory of the device (very realistic flowers contain on the order of tens of thousands of triangles, and a flowerbed contains many flowers).

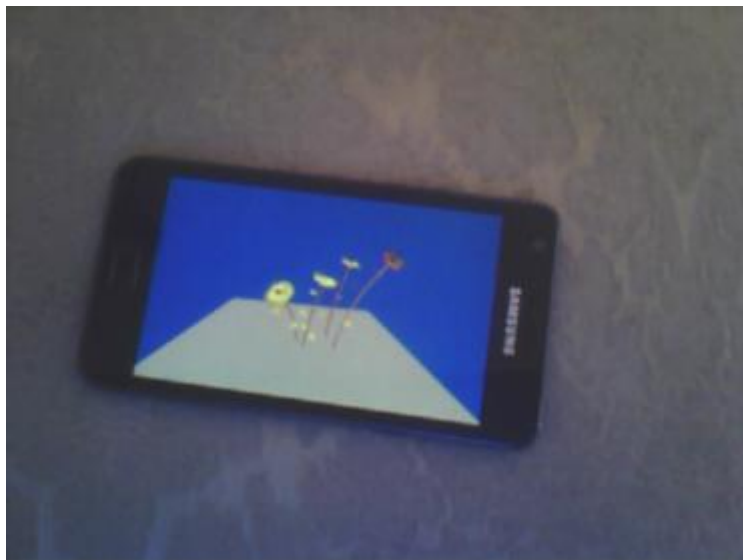


Figure 43 : Procedural flowers rendered on Android

The flowerbed generator code fills a terrain by choosing flower locations using stratified sampling (Figure 44; flowers are more sparse to highlight the sampling). The

number of flowers, their species and density are parameters to the script. In addition, a range of possible values for the grammar parameters can be given (or a default will be used). The script samples stochastically the parameter space, generates a url, and requests each flower, until the flowerbed is filled. Different flowerbed scripts can be used to interspace different species of flowers or to add grass. Additionally, different flowers can be interspaced in the same flowerbed by choosing randomly among a predefined set of flowers. Figure 45 shows a flowerbed with interspaced bluebells and daisies integrated in a game being developed at the group based on the Windmill adventure of Don Quixote (de Cervantes, 1605).

4.3.3 Exploring the parameter space using Kinect

Flowers are complex objects, and procedural modelling of them requires attention to many parameters. Classical interfaces based on keyboard and mice require the display of a multitude of parameters and nested menus. However, newer, camera-based input devices such as Microsoft Kinect allow the user to use a much richer collection of gestures using different parts of the body to indicate their wishes. Specific gestures can be designed and mapped to different parameters of the flower grammar, obtaining very intuitive modelling gestures (see Figure 46).

As an example, we have modelled the horizontal and vertical movement of the right hand to the rotation and length of the stem of the flower, respectively. When we display a circle indicating the position of the current parameters, we observe an interesting emerging behaviour: the flower grows and bends towards the circle, in a manner reminiscent of the known biological concept of phototropism (Whippo & Hangarter, 2006) (see Figure 47). We believe that this emerging behaviour provides an intuitive control for modellers and biologists. Formally, the Kinect gesture for hand position provides two axes (vertical and horizontal position of the hand), which range in values between 0 and 1. By contrast, the rotation parameter of the flower is in degrees (which range for realistic flowers between -5° and $+5^{\circ}$) and the length parameter ranges between 0 and 100. Two linear transforms connect the values of the Kinect axes to the flower parameters. The vertical axis is linked to the length of the flower by transforming the interval $[0; 1]$ to $[100; 0]$ as the screen coordinates grow down. The horizontal axis is linked to the angle parameter by transforming $[0; 1]$ to $[-5; 5]$.



Figure 44 : Top view of a sparse flowerbed



Figure 45 : Bluebells and daisies in the Don Quixote game



Figure 46 : Controlling the flower shape with Kinect gestures

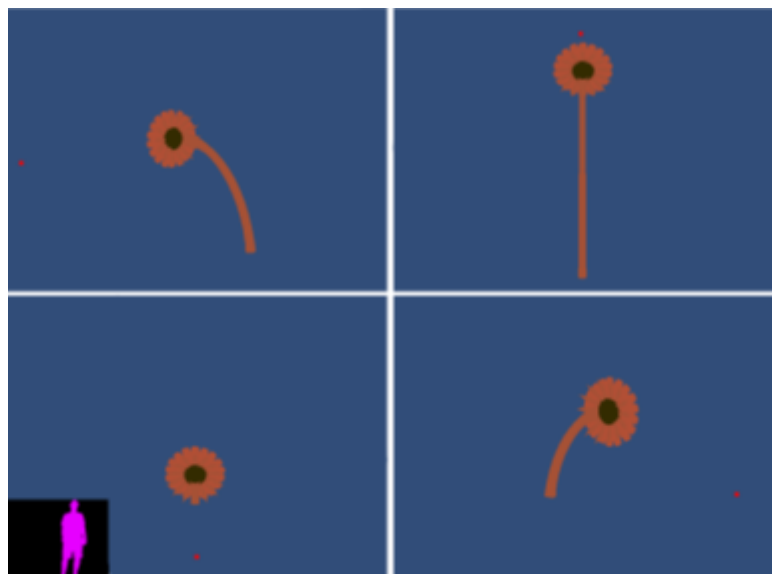


Figure 47 : Different screenshots of a daisy in which the length and rotation parameters are controlled using Kinect gestures. The red circle indicates the horizontal and vertical position of the user's hand. The movement resembles phototropism



Figure 48 : Automatically generated flowerbeds using sunflowers and grass

4.4 Conclusions

We have shown how unique, realistic grass and flowers can be generated by L-Systems, and described a framework to model them using a natural interface (Microsoft Kinect), and to generate flowerbeds in the Unity game engine. These flowerbeds can be integrated in videogames very easily. We plan to integrate the procedural flowerbed generator in the Legends of Girona game (Rodriguez et al., 2013) to provide more realistic rendering of the fields outside of the city, and to validate the software in real-world scenarios. The current bottleneck of the system is the load of the flower models, which is sent using the OBJ format (we are currently loading the models using a library based on Bartek Drozd's Objloader (Drozd, 2010)). To alleviate this bottleneck, we will search for efficient implementations of 3D model loaders for Unity and integrate them in our framework.

Chapter 5

Summary and conclusions

5. Summary and conclusions

5.1 Key Contributions

Modelling of vegetation is a huge area of investigation where computer graphics scientists have been exploring since decades. Although the resulting models have gained acceptance in as a research tool in biology and have led to increasingly convincing visualizations, still a lot of areas are left unexplored. One of such areas is the flowering plants simulation which received less attention since the main research is focused on modelling of trees rather than flowers. Although, being part of vegetation, flowering plants have their particular structural features which are different from the structure of trees, bushes or grass. Another area is the interchange between biological plausibility, efficiency and visualisation is still very weak. A 3D artist can use commercial tools, spend more than several hours for creation a model from scratch, but obtain a visually impressive result. While an L-Systems specialist can create an unlimited number of visually less impressive, but biologically plausible models generating them from one grammar in a short period of time. The underlying tools of L-system methods assume that the user is familiar with the concepts of L-systems and turtle interpretation, as well as the elements of the C programming language. In this thesis, the following contributions are made to these areas:

Modelling of flowering plants

Flowers have quite an intricate structure consisting of numerous components which, in turn, have an enormous variety of shapes. A number of very different approaches have been proposed, depending on the grass properties as well as the modelling purposes. We chose a procedural modelling using L-systems as a base of our research. We propose to represent the shapes of leafs, petals, stamens, carpels, etc. with an extension of L-Systems – a model based on three dimensional generalized maps – 3Gmaps L-systems, which can be successfully applied for modelling of flowering plants . The grammar description of the structure of the flowering plants provides an unlimited number of its geometrical interpretations. Moreover the way the model is built allows us to take into account its internal structure. As the flower tissue is non-homogeneous, the possibility of obtaining its internal composition could be quite useful for rendering, allowing for instance to render more accurate subsurface scattering.

Interactive control the model

Procedural modelling of flowering plants is very efficient and provides impressive results but the underlying tools are not intuitive for the common user. The process of adjusting parameter values of the grammar could be quite cumbersome as the user has to load the grammar every time he/she needs to see the changes of the geometry. We added a functionality of interactive change of parameter values. The user can adjust the model on the fly, changing parameter values and observing the result at the same time. This way of parameters values changing is quite faster as it only takes into account the embedding part, leaving the topology part of the program untouched.

Inverse modelling

The process of writing a grammar is usually quite laborious and tedious. In order to avoid this we propose new interface functionality: the inverse modelling by automatic generation of L-systems. The user describes the flower he wants to model, by assigning the properties of its organs. The algorithm uses this information as an input, which is then analysed and coded as L-systems grammar.

This application provides an intuitive interface which permits the user to create grammars in a more comprehensive level. A big advantage is that the user does not have to write an intricate code of the grammar, but with the help of our interface, he/she can define the flower characteristics, which are used for automatic grammar generation.

Modelling of large amounts of flowers

The creation of virtual terrains requires many hours by expensive modellers to provide a believable experience to the final user. Interactive terrain editors allow the modeller to paint over a terrain texture, and the colours of this texture are used by the game engine to add trees or other vegetation, using a predefined set of models. While the results are quite good for casual walkthroughs through the terrain, closer inspection shows that only a discrete number of models are being used to populate the terrain. In our framework, the models are located in the terrain using similar techniques, but each model instance is requested to a flower server. Since the flower models are parameterized using a customizable grammar with different tuneable parameters, we can guarantee that all the flowers will be different, providing the final user with a much more believable world even when performing close inspections of the models. The grammar model ensures that all the flowers are physically plausible.

Modelling plants using gesture capture

In order to lighten the task of the user we combine 3Gmap L-system with the natural user interface by means of Microsoft Kinect. Analysing the gestures of the user, Kinect provides basic interactions, which are reinterpreted as signals for changing parameter values of the grammar, and which in its turn returns a modified geometrical model of the flower. Using simple gestures the user can create new flowers or interactively modify its shape, such as the curvature, the length and the width of each of its organs.

5.2 Research Outlook

Vegetation is not only complex in geometry; also the light interaction of leaves or grass blades is highly intricate. A leaf for example usually consists of different layers and is strongly structured, which has a profound impact on both the reflectance and translucency of leaves, an integral part of the light interaction of vegetation. Rendering vegetation is substantially different from rendering geometry with less geometric complexity such as houses, manufactured products or other objects consisting of largely connected surfaces. Modelling each flower individually in a landscape would require a huge amount of geometry, making a naive geometric approach impractical for interactive rendering. In order to solve this problem, new approaches should be proposed. By using modern GPU capabilities to full extend could considerably optimize the process of visualisation. Considering a lawn of flowers for example, sending the grammar directly to the GPU provides model rendering entirely on the GPU without having to send the geometry to the CPU. Additionally due to the parallelism of the GPU it is possible to generate and visualise thousands of slightly different flower instances using only one grammar and a set of parameter values.

Our application provides grammar generating of a limited number of flower types. We are planning to expand our database with more properties, permitting to describe the flower in a more complex way thus obtaining more intricate grammars. Another future research work will be directed on generating the grammars of flower models, given one or several images.

Using subsurface scattering for rendering by taking into account the internal structure of the tissue is a future task yet to be implemented. Another improvement to be attained is texture generation according to the flower organs.

5.3 Conclusions

Men's habitat is a green carpet of plants covering our earth. And the most impressive among them are flowers. Flowering plants play a huge role in our life from nutritive and medical purposes to beautifying the environment and therefore form an essential part of computer graphics. As the technology progresses, new possibilities will be available to increase the quality of all aspects, but specialized techniques still will be necessary to optimize the calculations.

Significant advances have been made, but a fully photo-realistic real-time display of plants down to the last detail is still far ahead. Nonetheless, state-of-the-art methods as presented in this thesis can provide the visual complexity in appearance needed to render many forms of flowering plants in a convincing and faithful way.

Bibliography

Aliaga, D., Rosen, A. & Bekins, R., 2007. Style Grammars for Interactive Visualization of Architecture. *IEEE Trans. Vis. Comput. Graph*, 13(4), pp.786-97.

Anastacio, F., Prusinkiwicz, P. & Souza, C., 2008. Sketch-based parameterization of L-systems using illustration-inspired construction lines. In *Proceedings of the EUROGRAPHICS Workshop on Sketch-Based Interfaces and Modeling.*, 2008.

Azzari, G., Goulden, & Rusu , , 2012. Rapid Characterization of Vegetation Structure with a Microsoft Kinect Sensor. *Sensors*, 13(2), pp. 2384-2398.

Baranoski, G. & Rokne, J., 1997. An Algorithmic Reflectance and Transmittance Model for Plant Tissue. *Computer Graphics Forum*, 16(3), pp.141–50.

Baranoski, G.V.G. & Rokne, , 2001. Efficiently simulating scattering of light by leaves. *The Visual Computer*, 17(8), pp.491-505.

Bokeloh, M., Wand, M. & Seidel, H.-P., 2010. A connection between partial symmetry and inverse procedural modeling. In *SIGGRAPH '10 ACM SIGGRAPH 2010 papers*. NY, 2010. ACM New York.

Brakke, T., 1994. Specular and diffuse components of radiation scattered by leaves. *Agricultural and Forest Meteorology*, 71(3-4), pp.283–95.

Brisson, E., 1993. Representing geometric structures ind dimensions: Topology and order. *Discrete & Computational Geometry*, 9(1), pp.387-426.

Coen, E.S. & Meyerowitz, E., 1991. The war of the whorls: genetic interactions controlling flower. *Nature* , 353, pp.31 - 37.

de Cervantes, , 1605. *Don Quijote de la Mancha*. Punto de Lectura (October 1, 2008).

Deussen, O. & Lintermann, B., 1999. Interactive modeling of plants. *Computer Graphics and Applications, IEEE*, 19 (1), pp.56 - 65.

Deussen, & Lintermann, , 2005. *Digital Design of Nature: Computer Generated Plants and Organics*. Springer.

Deussen, & Lintermann, B., 2014. *Xfrog*. [Online] Available at: "<http://xfrog.com/>" <http://xfrog.com/> [Accessed 1996].

Ding, Z. et al., 2008. Flower solid modeling based on sketches. *Journal of Zhejiang University SCIENCE A*, 9(4), pp.481-88.

Discoe, B., 2013. *Virtual Terrain Project*. [Online] Available at: <http://vterrain.org/Plants/plantsw.html> [Accessed 19 November 1997].

Dobkin, D. & Laszlo, , 1987. Primitives for the manipulation of three-dimensional subdivisions. In *SCG '87 Proceedings of the third annual symposium on Computational geometry*. NY, 1987. ACM New York.

Drozdz, B., 2010. *Loading 3d models at runtime in Unity3d*. [Online] Available at: <http://www.everyday3d.com/blog/index.php/2010/05/24/loading-3d-models-runtime-unity3d/comment-page-2/>.

E-on software, 2014. *The Plant Factory*. [Online] Available at: <http://www.plantfactory-tech.com/overview/> [Accessed 1997].

Fernhout, P. & Kurtz, C., 2014. *Kurtz-Fernhout Software*. [Online] Available at: http://www.kurtz-fernhout.com/summary_plantstudio.html, http://www.kurtz-fernhout.com/summary_plantstudio.html [Accessed 1992].

Font Quer, P., 1938. *Iniciació a la botànica*. Barcelona: FONTALBA.

Fourcaud, T. et al., 2008. Plant Growth Modelling and Applications: The Increasing Importance of Plant Architecture in Growth Models. *Ann Bot*, 101(8), pp.1053–63.

Fourcaud, T. et al., 2008. Plant growth modeling and applications: the increasing importance of plant architecture of growth models. *Annals of botany*, 101(8), pp.1053-63.

Fowler, D., Prusinkiewicz, P. & Battjes, J., 1992. A collision-based model of spiral phyllotaxis. In *SIGGRAPH '92 Proceedings of the 19th annual conference on Computer graphics and interactive techniques*. NY, USA , 1992. ACM New York.

Franzke, O. & Deussen, O., 2003. Accurate graphical representation of plant leaves. In Hu, B.G. & Jaeger, M., eds. *Plant growth modelling and its applications*. Beijing, 2003. Springer-Verlag.

Frijters, D. & Lindenmayer, A., 1974. A Model for the Growth and Flowering of *Aster Novae-Angliae* on the Basis of Table $\langle 1, 0 \rangle$ -L-Systems. In *Proceeding L Systems*. Aarhus, Denmark, 1974. Springer-Verlag London.

Frijters, D. & Lindenmayer, A., 1976. Developmental descriptions of branching patterns with paracladial relationships. In A. Lindenmayer, G.R., ed. *Automata, languages, development*. North-Holland, 1976.

Fuhrer, M., Jensen, H. & Prusinkiewicz, P., 2006. Modeling hairy plants. *Graphical Models - Special issue on PG2004*, 68(4), pp.333 - 342.

Han, F. & Zhu, S.-C., 2003. Bayesian Reconstruction of 3D Shapes and Scenes From A Single Image. In *HLK '03 Proceedings of the First IEEE International Workshop on Higher-Level Knowledge in 3D Modeling and Motion Analysis*. Washington, 2003. IEEE Computer Society Washington.

Harder, L. & Prusinkiewicz, P., 2013. The interplay between inflorescence development and function as the crucible of architectural diversity. *Annals of Botany*, 112(8), pp.1477-93.

Ijiri, T., Owada, S. & Igarashi, T., 2006. Seamless Integration of Initial Sketching and Subsequent Detail Editing in Flower Modeling. In *Computer Graphics Forum. Eurographics 2006.*, 2006.

Ijiri, T., Owada, S. & Igarashi, T., 2006. The sketch lsystem: Global control of tree modeling using free-form strokes. In *In Smart Graphics.*, 2006.

Ijiri, T., Owada, S., Okabe, M. & Igarashi, T., 2005. Floral Diagrams and Inflorescences: flower modeling interface using botanical structural constraints. In *ACM Transactions on Graphics. SIGGRAPH 2005.*, 2005.

IRI, 2011. *Plant leaf tracking with a Kinect camera*. [Online] IRI Available at: <http://www.youtube.com/watch?v=pZ-W4eiy9vk>, <http://www.youtube.com/watch?v=pZ-W4eiy9vk>.

Kang, S. & Quan, L., 2009. *Image-based Modeling of Plants and Trees*. Morgan & Claypool Publishers.

Lienhardt, P., 1994. N-Dimensional Generalized Combinatorial Maps and Cellular Quasi-Manifolds. *International Journal of Computational Geometry & Applications*, 4(3), pp.275-324.

Lu, Z., Willis, C. & Paddon, D., 2000. Perceptually Realistic Flower Generation. In *8th International Conference in Central Europe on Computer Graphics, Visualization and Interactive Digital Media.*, 2000.

McCormack, J., 1993. Interactive Evolution of L-System Grammars for Computer Graphics Modelling. In D. Green & Bossomaier, eds. *Complex Systems: From Biology to Computation*. Amsterdam: ISO Press. pp.118-30.

Microsoft, 2012. *Introducing Kinect for Xbox 360*. [Online] Microsoft Available at: <http://www.xbox.com/en-US/KINECT>, <http://www.xbox.com/en-US/KINECT> [Accessed 26 December 2012].

Neubert, B., Franken, T. & Deussen, O., 2007. Approximate image-based tree-modeling using particle flows. In *Proceeding SIGGRAPH '07 ACM SIGGRAPH*. NY, 2007. ACM New York.

Onishi, K., Murakami, , Kitamura, Y. & Kishino, F., 2006. Modeling of Trees with Interactive L-System and 3D Gestures. In *Biologically Inspired Approaches to Advanced Information Technology. Lecture Notes in Computer Science*. Springer Berlin Heidelberg. pp. 222-235.

Ozawa, A., Uehara, , Sekiguchi, F. & Imai, , 2009. Spectral Analysis of Scattered Light from Flowers' Petals. *Optical Review*, 16(4), pp.458-60.

Peiyu, , Chuanbo, & Zehua, , 2006. Simulation Model of Flower Using the Interaction of L-systems with Bezier Surfaces. *COMPUTER ENGINEERING AND APPLICATIONS*, (16), pp.6-8.

Perttunen, J. et al., 1996. LIGNUM: A Tree Model Based on Simple Structural Units. *Ann Bot*, 77(1), pp.87-98.

Petrenko, O. et al., 2013. Flower modelling using natural interface and 3Gmap L-systems. In *12th ACM International Conference on Virtual Reality Continuum and Its Applications in Industry (VRCAI 2013)*., 2013. ACM.

Petrenko, , Sbert, , Terraz, O. & Ghazanfarpour, D., 2012. 3Gmap L-systems grammar application to the flowering plants modeling. In *Intelligent Computer Graphics 2012. Studies in Computational Intelligence*. Springer Berlin Heidelberg. pp.1-21.

Petrenko, , Sbert , M., Terraz, O. & Ghazanfarpour, D., 2012. Modeling of Flowers with Inverse Grammar Generation Interface. *International Journal of Creative Interfaces and Computer Graphics*, 3(2), p.19.

Petrenko, O., Sbert , M., Terraz, & Ghazanfarpour, D., 2012. Modeling of Flowers with Inverse Grammar Generation Interface. *International Journal of Creative Interfaces and Computer Graphics*, 3(2), pp.23-41.

Petrenko, O., Terraz, O., Sbert, M. & Ghazanfarpour, D., 2011. Interactive modeling of flowers with 3Gmap L-Systems. In *21st International Conference on Computer Graphics and Vision (GraphiCon'2011)*. Moscow, 2011.

Peyrat, A., Terraz, O., Merillou, & Galin, E., 2008. Generating vast varieties of realistic leaves with parametric 2Gmap L-systems. *The Visual Computer*, 24(7-9), pp.807-16.

Power, J.L., Brush, B., Prusinkiewicz, P. & Salesin, D., 1999. Interactive Arrangement of Botanical L-System Models. In *Proceedings of the 1999 Symposium on Interactive 3D Graphics*., 1999.

Prusinkiewicz, P. & Federl, P., 1999. Virtual Laboratory: an interactive software environment for computer graphics. In *Proceedings of Computer Graphics International*., 1999.

Prusinkiewicz, P., Hammel, M., Hanan, J. & Mech, R., 1996. L-systems: from the theory to visual models of plants. In Michalewicz, M.T., ed. *Proceedings of the 2nd CSIRO Symposium on Computational Challenges in Life Sciences*., 1996. CSIRO Publishing.

Prusinkiewicz, P., Hammel, M., Mech, R. & Hanan, J., 1995. The Artificial Life of Plants. In *Artificial life for graphics, animation, and virtual reality. SIGGRAPH Course Notes.*, 1995.

Prusinkiewicz, P., Karwowski, R., Měch, & Hanan, J., 2000. L-Studio/cpfg: A Software System for Modeling Plants. In *Applications of Graph Transformations with Industrial Relevance.* pp.457-64.

Prusinkiewicz, P., & Lindenmayer, A., 1990. *The algorithmic beauty of plants.* 1st ed. New York: Springer New York.

Prusinkiewicz, P. & Runions, A., 2012. Computational models of plant development and form. *New Phytologist*, 193(3), pp.549–69.

Prusinkiewicz, P., 1998. Modeling of spatial structure and development of plants: a review. *Scientia Horticulturae*, 74(1-2), pp.113–49.

Prusinkiewicz, P., 2000. Simulation modeling of plants and plant ecosystems. *Communications*, 43(7), pp.84-93.

Prusinkiewicz, P., 2004. Art and science for life: Designing and growing virtual plants with L-system. *Acta Horticulturae*, 630, pp.15-2.

Prusinkiewicz, P., 2004. Modeling plant growth and development. *Curr Opin Plant Biol.*, pp.79-83.

Quan, L. et al., 2006. Image-based plant modeling. In *SIGGRAPH '06 ACM SIGGRAPH 2006.* NY, 2006. ACM New York.

Reche, A., Martin, I. & Drettakis, G., 2004. Volumetric Reconstruction and Interactive Rendering of Trees from Photographs. *ACM Transactions on Graphics*, 23(3), pp.720-27.

Reffye, P. et al., 1997. A functional model of tree growth and tree architecture. *Silva Fennica*, 31(3), pp.297–311.

Rodriguez, A. et al., 2013. Implementation of a videogame: Legends. In *Simposio Español de Entretenimiento Digital, SEED 2013 / Congreso Español de Informática, CEDI.*, 2013.

Sakaguchi, T., 1998. Botanical tree structure modeling based on real image set. In *SIGGRAPH '98 ACM SIGGRAPH 98 Conference abstracts and applications.* NY, 1998. ACM New York.

Shlyakhter, I., Rozenoer, M., Dorsey, J. & Teller, S., 2001. Reconstructing 3D tree models from instrumented photographs. *IEEE Computer Graphics and Applications*, 21(3), pp.53 - 61.

Šťáva, et al., 2010. Inverse Procedural Modeling by Automatic Generation of L-systems. In *Computer Graphics Forum (Impact Factor: 1.64).*, 2010.

Stava, O. et al., 2014. Inverse Procedural Modeling of Trees. *Computer Graphics Forum.*

Terraz, O. et al., 2009. 3Gmap L-systems: an application to the modelling of wood. *The Visual Computer*, 25(2), pp.165-80.

Thornley, J.H.M. & Johnson, R., 1990. *Plant and crop modelling: a mathematical approach to plant and crop physiology*. Oxford: Clarendon Press.

Unity_Technologies, 2013. *Unity*. [Online] Available at: <http://unity3d.com/unity>, <http://unity3d.com/unity>.

Visser, d. et al., 2002. *3D modelling of plants: a review, report of the virtual plant network Wageningen*. Report. Wageningen : Plant Research International.

Wallenberg, , Felsberg, M., Forssén, P.-E. & Dellen , B., 2011. Leaf Segmentation using the Kinect. In *Proceedings*

Whippo, W. & Hangarter, R., 2006. Phototropism: Bending towards Enlightenment. *The Plant Cell May*, 18(5), pp.1110-19.

Yan, F. et al., 2014. Flower Reconstruction from a Single Photo. In *Computer Graphics Forum (Proceedings of Eurographics 2014)*., 2014.

Zelevnik, R., Herndon, P. & Hughes, F., 2006. SKETCH: an interface for sketching 3D scenes. In *Proceeding SIGGRAPH '06 ACM SIGGRAPH 2006 Courses*. NY, 2006. ACM New York.

Zigfu, 2013. ZDK for Unity3D — OpenNI. [Online] Available at: <http://www.openni.org/files/zdk-for-unity3d>, <http://www.openni.org/files/zdk-for-unity3d>.