Flower Modelling Using Natural Interface And 3Gmap L-Systems

Olga Petrenko
Universitat de Girona
Girona, Spain
lelya.fleur@gmail.com

Rubén Jesús García Hernández
Universitat de Girona
Girona, Spain
rgarcia@ima.udg.edu

Mateu Sbert
Universitat de Girona
Girona, Spain
mateu@ima.udg.edu

Olivier Terraz
Université de Limoges
Limoges, France
olivier.terraz@unilim.fr

Djamchid Ghazanfarpour
Université de Limoges
Limoges, France
djamchid.ghazanfarpour@xlim.fr

Abstract

In this paper we propose 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. 3Gmap L-systems (an extension of L-Systems) are based on three-dimensional generalized maps, and have been successfully applied to the modeling of flowering plants. The 3Gmap L-system receives the command to create or modify the flower and returns the 3D model which will be read and visualised by the Unity game engine.

CR Categories: I.3.3 [Computer Graphics]: Three-Dimensional Graphics and Realism—Display Algorithms

Keywords: 3Gmaps L-System, Kinect, flower modeling, grammar generation, natural phenomena

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 parametrized using a customizable grammar with different tunable 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. Figure 8 provides an example. In order to create a flower we are using a 3GmapLsystem [Terraz et al. 2009]. Here the L-systems operate with subdivision of volumes, namely 3Gmaps [Frijters and Lindenmayer 1976], [Lienhardt 1994]. The used L-systems grammars have a nested structure allowing combining several grammars which represent the different flower organs. Although this technique can provide impressive results, the underlying algorithms are not so intuitive for common users. In order to lighten the task of the user we combine 3Gmap L-system with the natural user interface by means of Microsoft Kinect. Analyzing 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.

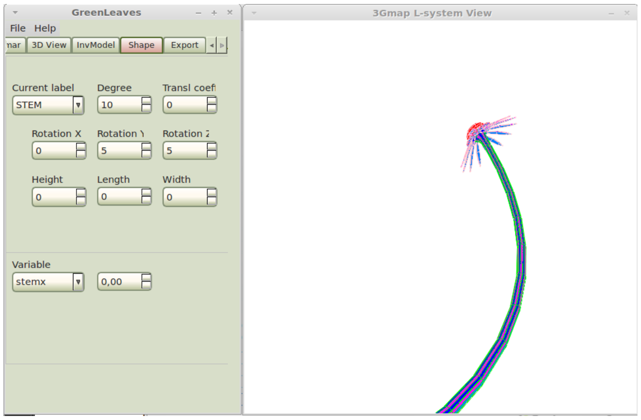
---

2 Previous work

The origins of plant modeling and flowers in particular are traced back to 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 modeling of growth processes of plant development [Prusinkiewicz and Lindenmayer 1990], [Federl and Prusinkiewicz 1999], [Peiyu et al. 2006].

Although string L-systems are quite efficient and are applied to model a wide variety of plants [Prusinkiewicz 2004], [Prusinkiewicz et al. 1995], they are of one-topological dimension, even if 3D geometrical features are incorporated into a model. 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 and Lindenmayer 1990] map L-systems and cellwork L-systems which were mainly used for modeling 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 [Lienhardt 1994], 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.

Still, the L-systems specification is not so intuitive for the common user, so other methods have also found its niche in the plant



Current label: STEM
Degree: 10
Transl coefi: 0
Rotation X: 0
Rotation Y: 5
Rotation Z: 5
Height: 0
Length: 0
Width: 0
Variable: stemx 0,00

Figure 1: *Traditional interface for flower modelling based on mouse interaction*

modeling area. Sketch-Based modeling techniques allow a user to easily create a rough model from several strokes. The 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. Here the task of the user 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.

Taking into account the benefits of the previous approaches some methods were proposed, such as [Onishi et al. 2006], [Power et al. 1999], [McCormack 1993]. Here the L-systems are mixed with interactive methods, such as Sketch-based or 3D gesture modeling or simply interactive control of parameter values. In [Petrenko et al. 2011]] and [Petrenko et al. 2012] 3Gmap Lsystem was combined with interactive tools, where the user could adjust the flower shape by interactively changing the parameter values (see figure 1).

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 color 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 analyzing 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. 2013], to track plant leaves [IRII 2011], and to segment them [Marcus et al. 2011].

3 Flower generation using Kinect

In the following sections, we describe the architecture of our system, with special enphasis on the flower generation module, the content generation module and the natural interface module.

3.1 System architecture

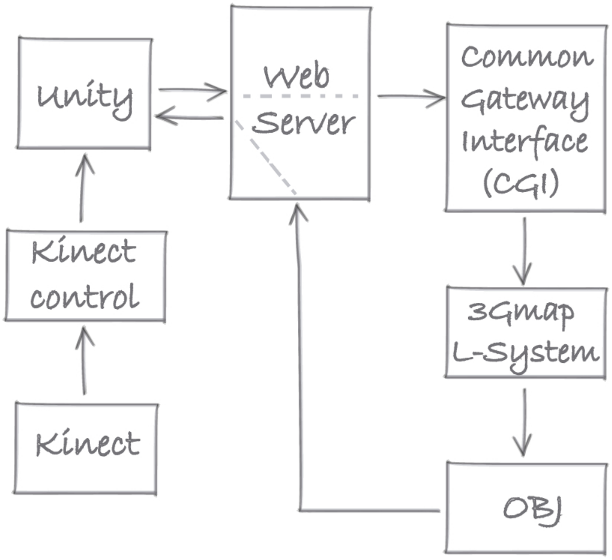


Figure 2: *Pipeline of our framework.*

The architecture of our system is as follows (Figure 2):

- 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 [Petrenko et al. 2012].
- A webserver using Common Gateway Interface (CGI) provides the interface between the flower generator and the clients requiring the flowers (figure 3). 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.
- 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 (figures 8 and 9). The flowerbeds

```
#!/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
```

Figure 3: *CGI script to interface with the flower generator.*

can be parametrized 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 chosen by the user.

- We have build a library of gestures on top of the standard OpenNI unity sdk [Zigfu 2013] and the NITE middleware, which is described in [Rodríguez 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 4 shows the rendering of different flowers in an android device. Additionally, the web server provides authentification, authorization and encryption, a possibly useful feature in DRM schemes, and can hide the grammars from the final users (if needed).

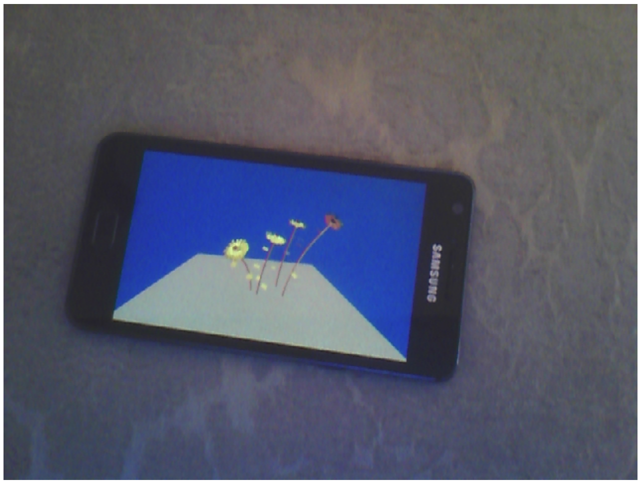
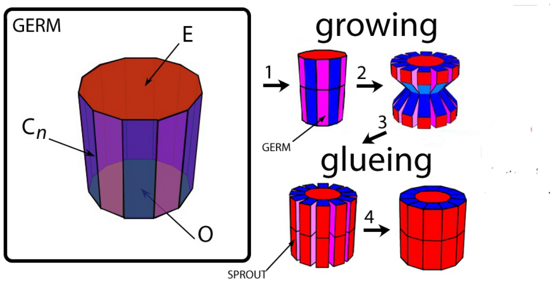


Figure 4: *Procedural flowers rendered on Android.*

3.2 Flower generation module

Flowers are modeled using an extension of L-Systems - a model based on three-dimensional generalized maps. 3Gmap is an ordered topological model that allows to represent the topology of subdivisions of orientable or non-orientable 3D spaces, with or without boundary. It is close to facet-edge data structure [Lienhardt 1994] or cell tuple [D.Dobkin and Laszlo 1987], [Brisson 1989] . 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 a unique basic element - a dart - 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 to represent the topology of an object, which corresponds to an unlimited number of embeddings of this structure in three-dimensional space.

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 $GERM_O, GERM\_\{E\}, GERM_{C_1}, GERM_{C_{2,...,GERM_N}}$ for the base, the end and the side faces of the prism GERM respectively. A flower shape is created by operating on prisms. Following production rules the volumes can grow, be glued and split. A short description of topological operations on volumes is represented in Figure 5.



1 : GERM -> GERM[GERM]_{E}

2 : GERM -> GERM[SPROUT]_{C*}

3 : SPROUT^{C1}|SPROUT^{C3}

4 : SPROUT^{C2}|SPROUT^{C4}

Figure 5: *3Gmap L-System description.*

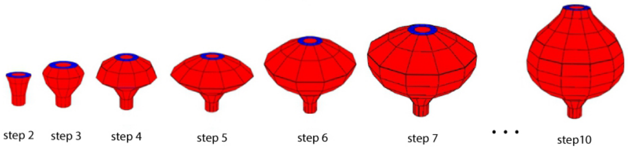
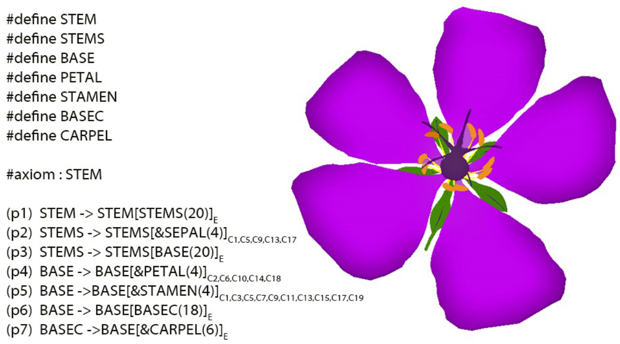


Figure 6: *3Gmap L-System derivation steps.*

- Growing: Creating a new volume and gluing it on one of the faces of predecessor, called a support face. This operation is denoted as: $GERM \rightarrow GERM[GERM(n)]\_\{E\}$, where E is the support face of the volume GERM and n is the degree of the new volume GERM.
- Gluing:This operation glues two adjacent faces and is denoted as follows: $SPROUT \rightarrow SPROUTC1|SPROUTC3$, where the faces C1 and C3 of volumes labeled SPROUT are glued if they are adjacent.
- Splitting: This operation splits a volume into two parts and is denoted $SPROUT \rightarrow ESPROUTASPROUTB$, where E is a face of volume SPROUTA. Here all faces adjacent to E are split and the volumes, obtained with a face are closed.

All the information of the flower building blocks and the instructions used in the development of its final shape is stored in the grammar. Using a grammar, we are operating recursively with different prisms and thus forming an appropriate shape.(Figure 6). The grammar serves as a ”gene” which has an unlimited number of potential embodiments. Once written a grammar we can change its parameter values and get a diversity of flowers. The detailed description of 3Gmap L-system grammars is presented in [Petrenko et al. 2012]. While applying 3Gmaps L-systems to the modeling of flowering plants we have to follow natural laws, lying underneath their botanical structure. 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. On the end face there is another building block which is the basis for growing petals and stamens on its side faces.And finally on its end face there is a volume, on the top of each grows carpel, or if there are more than one carpel, they will grow on the side faces of this block (see Figure 7).



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

#axiom : STEM

(p1) STEM -> STEM[STEMS(20)]$_{E}$
(p2) STEMS -> STEMS[&SEPAL(4)]$_{C1,C5,C9,C13,C17}$
(p3) STEMS -> STEMS[BASE(20)]$_{E}$
(p4) BASE -> BASE[&PETAL(4)]$_{C2,C6,C10,C14,C18}$
(p5) BASE ->BASE[&STAMEN(4)]$_{C1,C3,C5,C7,C9,C11,C13,C15,C17,C19}$
(p6) BASE -> BASE[BASEC(18)]$_{E}$
(p7) BASEC ->BASE[&CARPEL(6)]$_{E}$

Figure 7: *Floral diagram grammar.*

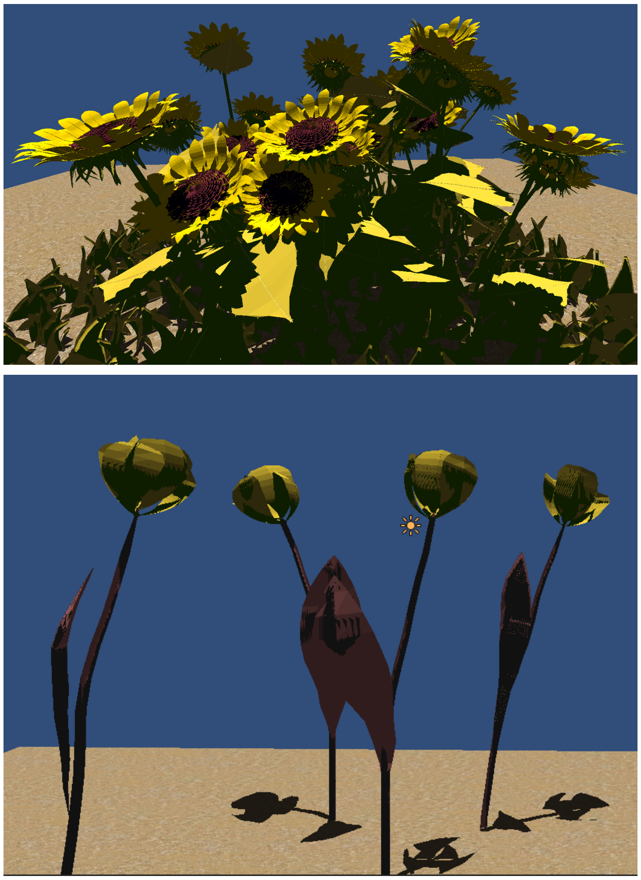


Figure 8: *Automatically generated flowerbeds. Top: using sunflowers and grass. Bottom: using tulips.*

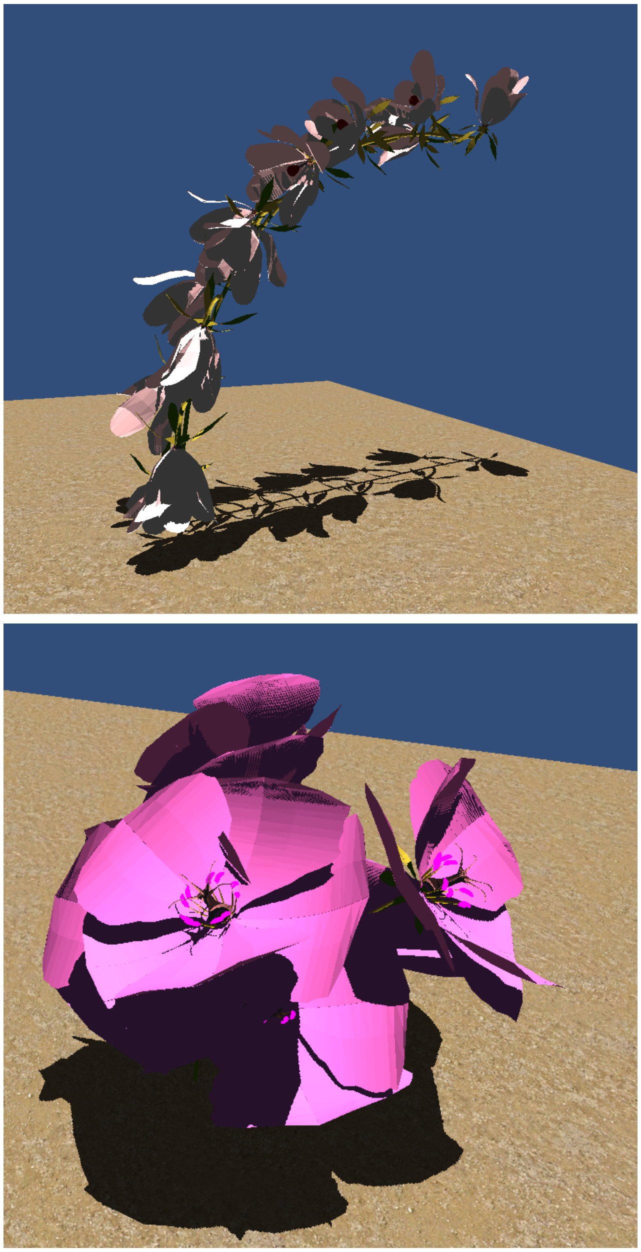


Figure 9: *Top: A single bluebell. Bottom: geraniums*

3.3 Content Generation

The content generation routines have been implemented as scripts running in the Unity game engine, to provide support for the different architectures it supports (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).

The flowerbed generator code fills a terrain by choosing flower locations using stratified sampling (figure 10; 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 (figure 8). Additionally, different flowers can be interespaced in the same flowerbed by choosing randomly among a predefined set of flowers. Figure 11 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]. An example of a real-world flowerbed can be seen in figure 12 for comparison purposes; we can see the tangling of the flowers is well preserved in our dense flowerbeds.

Figure 10: *Top view of a sparse flowerbed.*

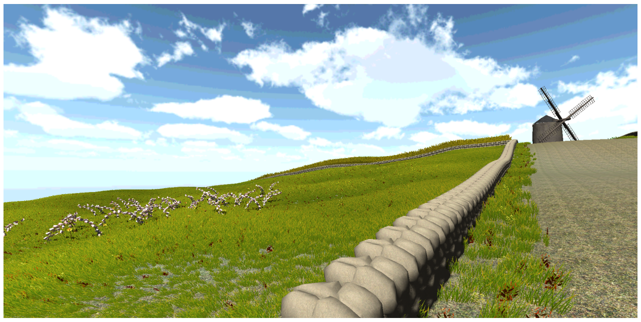


Figure 11: *Bluebells and daisies in the Don Quixote game.*

Figure 12: *Photograph of a natural flowerbed.*

3.4 Exploring the parameter space using Kinect

Figure 13: *Photograph of a real flower*

Flowers are complex objects (figure 13), and procedural modelling of them requires attention to many parameters. Classical interfaces based on keyboard and mouse 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.

Figure 14: *Controling the flower shape with Kinect gestures.*

Specific gestures can be designed and mapped to different parameters of the flower grammar, obtaining very intuitive modelling gestures (Figure 14). 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 and Hangarter 2006] (Figure 15). 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 ranges 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 axes is linked to the angle parameter by transforming $[0, 1]$ to $[-5, 5]$.

We can also use Kinect to delete flowers if the flowerbed is too dense. The Kinect cursor can select a flower for deletion by pushing the hand forward (figure 16).

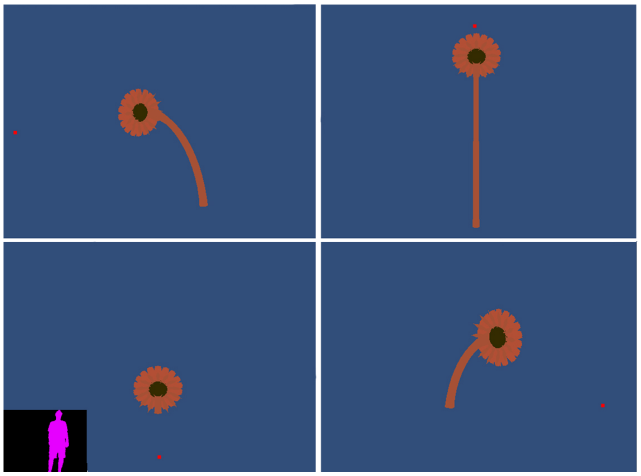


Figure 15: *Different screenshots of a daisy in which the length and rotation parameters are controlled using Kinect gestures. The red circle indicates the horizontal and vertical position of the user's hand. The movement resembles phototropism.*

4 Conclusions and Future work

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 [Rodríguez 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 Drozdz's Objloader [Drozdz 2010]). To alleviate this bottleneck, we will search for efficient implementations of 3D model loaders for Unity and integrate them in our framework.

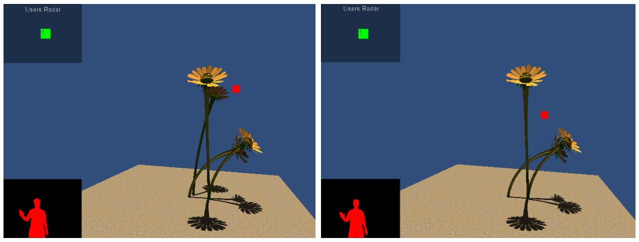


Figure 16: *Deletion of flowers using a Kinect push gesture.*

As further future work, we plan to add more realistic materials to the flower models and to perform a user study to compare the traditional mouse interface for flower modelling to our Kinect-based gesture interface.

Acknowledgements

This work has been funded in part by grant number TIN2010-21089-C03-01 of Spanish Government and grant number 2009-SGR-643 of Generalitat de Catalunya (Catalan Government). We would also like to thank the anonymous reviewers for their comments.

References

Azzari, G., Goulden, M. L., and Rusu, R. B. 2013. Rapid characterization of vegetation structure with a microsoft kinect sensor. *Sensors 13*, 2, 2384–2398.

Brisson, E. 1989. Representing geometric structures in d dimensions: topology and order. In *5th A.C.M. Symposium on Computational Geometry, Saarbrcken*, 218–227.

D.Dobkin, and Laszlo, M. 1987. Primitives for the manipulation of three-dimensional subdivisions. In *SCG '87:Proceedings of the third annual symposium on Computational geometry*, 86–99.

de Cervantes, M. 1605. *Don Quijote de la Mancha*. Ed. Juan de la Cuesta.

Drozdz, B., 2010. Loading 3d models at runtime in Unity3d - Everyday 3D. "`http://www.everyday3d.com/blog/index.php/2010/05/24/loading-3d-models-runtime-unity3d/`".

Federl, P., and Prusinkiewicz, P. 1999. Virtual laboratory. an interactive software environment for computer graphics. In *Computer Graphics International*, 93–100.

Frijters, D., and Lindenmayer, A. 1976. Developmental descriptions of branching patterns with paracladial relationships. *Automata, Languages and Development*, 57–73.

Ijiri, T., Owada, S., Okabe, M., and Igarashi, T. 2006. Floral diagrams and inflorescences: Interactive flower modeling using botanical structural constraints. In *Proceedings of ACM SIGGRAPH, 24(3)*, ACM Press / ACM SIGGRAPH, Computer Graphics Proceedings, Annual Conference Series, ACM, 720–726.

IRII, 2011. Plant leaf tracking with a Kinect camera. "`http://www.youtube.com/watch?v=pZ-W4eiy9vk`".

Lienhardt, P. 1994. N-dimensional generalized combinatorial maps and cellular quasi-manifolds. *International Journal of Computational Geometry and Applications 4*, 3, 275–324.

Marcus, W., Michael, F., and Per-Erik, F. 2011. Leaf segmentation using the kinect. In *Proceedings of SSBA 2011 Symposium on Image Analysis*, SSBA 2011, Linköping 14-18 March 2011.

McCormack, J. 1993. Interactive evolution of l-system grammars for computer graphics modelling. In *Complex Systems: From Biology to Computation*, ISO Press, Amsterdam, D. Green and T. Bossomaier, Eds., 118–130.

MICROSOFT, 2012. Introducing Kinect for Xbox 360. "`http://www.xbox.com/en-US/KINECT`". Accessed 26 December 2012.

ONISHI, K., MURAKAMI, N., KITAMURA, Y., AND KISHINO, F. 2006. Modeling of trees with interactive l-system and 3d gestures. In *Biologically Inspired Approaches to Advanced Information Technology (BioADIT'06)*, 222–235.

PEIYU, Q., CHUANBO, C., AND ZEHUA, L. 2006. Simulation model of flower using the interaction of l-systems with bezier surfaces. *Computer Engineering and application*, 16, 6–8.

PETRENKO, O., TERRAZ, O., SBERT, M., AND GHAZANFARPOUR, D. 2011. Interactive modeling of flowers with 3Gmap L-Systems. *21st International Conference on Computer Graphics and Vision (GraphiCon'2011)*, 20–24.

PETRENKO, O., TERRAZ, O., SBERT, M., AND GHAZANFARPOUR, D. 2012. 3Gmap L-Systems grammar application to the flowering plants modeling. In *Intelligent Computer Graphics 2012. Studies in Computational Intelligence*, no. 441, 1–21.

PEYRAT, A., TERRAZ, O., MERILLOU, S., AND GALI, E. 2008. Generating vast varieties of realistic leaves with parametric 2GMap L-Systems. *The Visual Computer 24*, 7-9 (Jul), 807–816.

POWER, J. L., BRUSH, A. J. B., PRUSINKIEWICZ, P., AND SALESIN, D. H. 1999. Interactive arrangement of botanical l-system models. In *Proceedings of the 1999 symposium on Interactive 3D graphics*, I3D.

PRUSINKIEWICZ, P., AND LINDENMAYER, A. 1990. *The Algorithmic Beauty of Plants*. Springer - Verlag.

PRUSINKIEWICZ, P., HAMMEL, M., MECH, R., AND HANAN, J. 1995. The artificial life of plants. In *In SIGGRAPH 95 Course Notes*.

PRUSINKIEWICZ, P. 2004. Modeling plant growth and development. *Current Opinion in Plant Biology 7*, 1 (Feb), 79–83.

RODRÍGUEZ, A., GARCÍA, R. J., GARCÍA, J. M., MAGDICS, M., AND SBERT, M. 2013. Implementation of a videogame: Legends of Girona. In *Actas del Primer Simposio Español de Entretenimiento Digital*, P. González Calero and M. Gómez Martín, Eds., 96–107.

TERRAZ, O., GUIMBERTEAU, G., MERILLOU, S., PLEMENOS, D., AND GHAZANFARPOUR, D. 2009. 3gmap lsystems: an application to the modeling of wood. *The visual Computer 25*, 2, 165–180.

UNITY TECHNOLOGIES, 2013. Unity.
http://unity3d.com.

WHIPPO, C. W., AND HANGARTER, R. P. 2006. Phototropism: Bending towards enlightenment. *The Plant Cell 18*, 5 (May), 1110–1119.

ZIGFU, 2013. ZDK for Unity3D — OpenNI. "`http://www.openni.org/files/zdk-for-unity3d`".

Appendices

The following appendices contain the grammars used to generate the daisies, sunflowers and herbs shown in section 3.

The grammar of the grass consists of a one building block HB, to which a growing operation is applied sequentially and its final shape is sculptured with parameters modifying the grass rotation and curvature and the the height and the width of the leaf. The daisy and the sunflower are constructed according to the floral diagram. We use a building block STEM both for the daisy and for the sunflower grammars. The STEM building blocks of the sunflower stem are interfered with STEML, STEMLL and STEMLLL building blocks. These are the bases for the sunflower leafs, which will grow from the side faces.
$STEML \rightarrow STEML[LEAF(,0,0,,0,,,,)]_{C1}$
$STEMLL \rightarrow STEMLL[LEAF(,0,0,,0,,,,)]_{C5}$
$STEMLLL \rightarrow STEMLLL[LEAF(,0,0,,0,,,,)]_{C9}$
On the last building block of the stem we grow the receptacle of the daisy and the sunflower using a RECEPT building block. It is colocated on the side faces of the top of the stem and grows the sepals. Daisies and sunflowers have circular flowerheads which are represented in the grammars as CENTRE building block for the daisy and SPIRAL module foe the sunflower. Petals grow from the side faces of the top of the stem using the modules PETAL and PTSF for daisy and sunflower accordingly.

A Daisy grammar

$\#define\ STEM(14,2,0,5,5,2,2,2,4)$
$\#define\ CENTRE(14,1,0,0,0,1,1,1,3)$
$\#define\ COL(14,1,0,0,0,1,1,1,3)$
$\#define\ UREC(18,1,0,0,0,1,1,1,4)$
$\#define\ LREC(14,1,0,0,0,1,1,0.008,4)$
$\#define\ SPONGY(4,1,0,0,0,1,1,1,3)$
$\#define\ PALISADE(4,1,0,0,0,1,1,1,3)$
$\#define\ CUTICLE(4,1,0,0,0,1,1,1,3)$

$\#axiome: STEM$

$STEM \rightarrow STEM[STEM(,,,,,,,,)]\_\{E\}$
$STEM \rightarrow STEM[STEM(,,,,,,,,)]\_\{E\}$
$STEM \rightarrow STEM[STEM(,,,,,,,,)]\_\{E\}$
$STEM \rightarrow STEM[STEM(,,,,,,,,)]\_\{E\}$
$STEM \rightarrow STEM[STEM(,,,,,,,,)]\_\{E\}$
$STEM \rightarrow STEM[STEM(,,,,,,,,)]\_\{E\}$
$STEM \rightarrow STEM[STEM(,,,,,,,,)]\_\{E\}$
$STEM \rightarrow STEM[STEM(,,,,,,,,)]\_\{E\}$
$STEM \rightarrow STEM[STEM(,,,,,,,,)]\_\{E\}$
$STEM \rightarrow STEM[STEM(,,,,,,,,)]\_\{E\}$
$STEM \rightarrow STEM[STEM(,,,,,,,,)]\_\{E\}$
$STEM \rightarrow STEM[STEM(,,,,,,,,)]\_\{E\}$
$STEM \rightarrow STEM[STEM(,,,,,,,,)]\_\{E\}$
$STEM \rightarrow STEM[STEM(,,,,,,,,)]\_\{E\}$
$STEM \rightarrow STEM[LREC(,,,,,,,,)]\_\{E\}$
$LREC \rightarrow LREC[UREC(,,,,5,,)]\_\{E\}$
$UREC \rightarrow UREC[CENTRE(,,,,5,,,)]\_\{E\}$
$CENTRE \rightarrow CENTRE[CENTRE(,,,,5,,,)]\_\{E\}$
$CENTRE \rightarrow CENTRE[CENTRE(,,,,5,,,)]\_\{E\}$
$CENTRE \rightarrow CENTRE[COL(,,,,,,,,)]\_\{E\}$
$CENTRE \rightarrow CENTRE[SPONGY(,,,,,,,,)]_{C*}$
$LREC \rightarrow LREC[\&RECEPT(4,0.1,0,0,0,1,0.1,0.002,4)]\_\{C*\}$
$UREC \rightarrow UREC[\&PETALDAISYY(4,0.1,0,0,0,1,0.1,$
$0.2,9)]\_\{C*\}$
$SPONGY \rightarrow SPONGY\tilde{}C2|SPONGY\tilde{}C4$
$SPONGY \rightarrow SPONGY\tilde{}C1|SPONGY\tilde{}C3$
$PALISADE \rightarrow PALISADE\tilde{}C2|PALISADE\tilde{}C4$
$PALISADE \rightarrow PALISADE\tilde{}C1|PALISADE\tilde{}C3$
$CUTICLE \rightarrow CUTICLE\tilde{}C2|CUTICLE\tilde{}C4$
$CUTICLE \rightarrow CUTICLE\tilde{}C1|CUTICLE\tilde{}C3$

B Sunflower grammar

$\#define\ STEM(16,0,0.1,0,0,3,1.1,1.1,4)$
$\#define\ STEMS(16,0,0.1,0,0,3,1.1,1.1,4)$
$\#define\ STEMM(16,0,0,0,0,3,1.1,1.1,4)$
$\#define\ STEML(16,0,0.1,0,0,3,1.1,1.1,4)$
$\#define\ STEMLL(16,0,0.1,0,0,3,1.1,1.1,4)$
$\#define\ STEMLLL(16,0,0.1,0,0,3,1.1,1.1,4)$
$\#define\ CENTER(8,0,0,0,0,2,0.5,0.5,2)$
$\#define\ RECEPT(8,0,0.1,0,0,0.1,1.1,1.1,4)$
$\#define\ RECEPTSF(8,0,0.1,0,0,0.1,1.1,1.1,4)$
$\#define\ SFR(8,0,0.1,0,0,0.1,1.1,1.1,3)$
$\#define\ RSF(8,0,0.1,0,0,0.1,1.1,1.1,3)$
$\#define\ LEAF(4,1,0,0,0,0.2,0.1,0.1,4)$
$\#define\ SEPAL(4,1,0,0,0,0.2,0.1,0.1,4)$
$\#define\ PT(4,1,0,0,0,0.2,0.1,0.1,3)$

$\#define\ x = 0.006$
$\#define\ ko = 16$
$\#define\ s = 0.8$
$\#define\ angle = 0.984$
$\#define\ ang = 13.02$

$\#axiome : STEM$

$STEM \rightarrow STEM[STEM(,,,,rnd(1,5),,,)]\_\{E\}$
$STEM \rightarrow STEM[STEM(,,,,rnd(1,5),,,)]\_\{E\}$
$STEM \rightarrow STEM[STEM(,,,,rnd(1,5),,,)]\_\{E\}$
$STEM \rightarrow STEM[STEML(,,,,rnd(1,5),,,)]\_\{E\}$
$STEML \rightarrow STEML[STEM(,,,,rnd(1,5),,,)]\_\{E\}$
$STEM \rightarrow STEM[STEM(,,,,rnd(1,5),,,)]\_\{E\}$
$STEM \rightarrow STEM[STEM(,,,,rnd(1,5),,,)]\_\{E\}$
$STEM \rightarrow STEM[STEMLL(,,,,rnd(1,5),,,)]\_\{E\}$
$STEMLL \rightarrow STEMLL[STEM(,,,,,,,,)]\_\{E\}$
$STEM \rightarrow STEM[STEM(,,,,,,,,)]\_\{E\}$
$STEM \rightarrow STEM[STEM(,0,0,,0,,,,)]\_\{E\}$
$STEM \rightarrow STEM[STEMLLL(,0,0,,0,,,,)]\_\{E\}$
$STEMLLL \rightarrow STEMLLL[STEMS(,0,0,,0,,,,)]\_\{E\}$
$STEMS \rightarrow STEMS[STEMS(,0,0,,0,,,,)]\_\{E\}$
$STEMS \rightarrow STEMS[STEMS(,0,0,0,0,0.3,,,)]\_\{E\}$
$STEMS \rightarrow STEMS[STEMM(16,,,0,0.5,8.8,4)]\_\{E\}$
$STEMM \rightarrow STEMM[STEMS(16,,,0,0.5,7,7,4)]\_\{E\}$
$STEMS \rightarrow STEMS[RECEPT(26,,,,0.5,5,5,4)]\_\{E\}$
$RECEPT \rightarrow RECEPT[RECEPT(16,,,,0.1,4.5,4.5,4)]\_\{E\}$
$RECEPT \rightarrow RECEPT[RECEPTSF(16,,,,0.1,6,6,4)]\_\{E\}$
$RECEPTSF \rightarrow RECEPTSF[RECEPTSF(26,,,,0.1,5,5,4)]\_\{E\}$
$RECEPTSF \rightarrow RECEPTSF[SFR(26,,,,0.1,5,5,3)]\_\{E\}$
$SFR \rightarrow SFR[RSF(16,,,,0.9,4.5,4.5,3)]\_\{E\}$
$RSF \rightarrow RSF[STEMS(12,,,0,0.2,1.5,1.5,3)]\_\{E\}$
$STEMS \rightarrow STEMS[STEMS(8,,,,0.2,0.5,0.5,3)]\_\{E\}$
$STEMS \rightarrow STEMS[STEMS(8,,,,0.2,0.5,0.5,3)]\_\{E\}$
$STEMS \rightarrow STEMS[\&SPIRAL(8,0,0,90,0,math.sqrt(pow((s+etape)*x*(1+ko*etape),2)+pow((s+etape)*x*(2+ko*etape),2)-2*(s+etape)*x*(1+ko*etape)*(s+etape)*x*(2+ko*etape)*angle),0,0,8)]\_\{E\}$
$RECEPT \rightarrow RECEPT[SEPAL(4,1,,,0.2,0.1,0.1,4)]_{C*}$
$SFR \rightarrow SFR[PT(4,1,,,0.9,0.1,0.1,3)]_{C*}$
$RSF \rightarrow RSF[PT(4,1,,rnd(3,9),rnd(3,9),2,0.1,0.1,3)]_{C*}$
$STEML \rightarrow STEML[LEAF(,0,0,,0,,,,)]_{C1}$
$STEMLL \rightarrow STEMLL[LEAF(,0,0,,0,,,,)]_{C5}$
$STEMLLL \rightarrow STEMLLL[LEAF(,0,0,,0,,,,)]_{C9}$
$LEAF \rightarrow LEAF[LEAF(4,1,,0,0.5,0.1,0.1,4)]\_\{E\}$
$LEAF \rightarrow LEAF[LEAF(4,1,,,0.5,0.1,0.1,4)]\_\{E\}$
$LEAF \rightarrow LEAF[LEAF(4,1,,,0.5,0.1,0.1,4)]\_\{E\}$
$LEAF \rightarrow LEAF[\&LEAFSF(16,1,,0,1,0.1,0.1,13)]\_\{E\}$
$SEPAL \rightarrow SEPAL[\&SEPALSF(16,1,,0,0.5,0.1,0.1,13)]\_\{E\}$
$PT \rightarrow PT[\&PTSF(16,1,,0,1,0.1,0.1,3)]\_\{E\}$
$SEPAL \rightarrow (SEPAL)$

C Herb grammar

$\#define\ HERB(16,1,0,0,0,0.5,2.5,0.2,4)$

$\#axiome$: HERB

$p01\ HERB \rightarrow HERB[HERB(16,1,,\langle math.random(-18,15)\rangle,\langle math.random(-18,15)\rangle,3,2,0.2,)]\_\{E\}$ $p01\ HERB \rightarrow HERB[HERB(16,1,,\langle math.random(-15,15)\rangle,\langle math.random(-18,15)\rangle,3,1.8,0.2,)]\_\{E\}$ $p01\ HERB \rightarrow HERB[HERB(16,1,,\langle math.random(-15,15)\rangle,\langle math.random(-18,15)\rangle,3,1.5,0.2,)]\_\{E\}$ $p01\ HERB \rightarrow HERB[HERB(16,1,,\langle math.random(-18,15)\rangle,\langle math.random(-18,15)\rangle,3,1.3,0.2,)]\_\{E\}$ $p01\ HERB \rightarrow HERB[HERB(16,1,,\langle math.random(-15,15)\rangle,\langle math.random(-18,15)\rangle,3,1,0.2,)]\_\{E\}$ $p01\ HERB \rightarrow HERB[HERB(16,1,,\langle math.random(-15,15)\rangle,\langle math.random(-18,15)\rangle,3,0.8,0.2,)]\_\{E\}$ $p01\ HERB \rightarrow HERB[HERB(16,1,,\langle math.random(-15,15)\rangle,\langle math.random(-18,15)\rangle,3,0.6,0.2,)]\_\{E\}$ $p01\ HERB \rightarrow HERB[HERB(16,1,,\langle math.random(-15,15)\rangle,\langle math.random(-18,15)\rangle,3,0.4,0.2,)]\_\{E\}$ $p01\ HERB \rightarrow HERB[HERB(16,1,,\langle math.random(-15,15)\rangle,\langle math.random(-18,15)\rangle,2,0.2,0.2,)]\_\{E\}$