Inverse Modeling and Animation of Growing Single-stemmed Trees at Interactive Rates

Steffen Rudnick* Lars Linsen† E. Gregory McPherson‡

* Department of Mathematics and Computer Science
Ernst-Moritz-Arndt-Universität Greifswald, Germany

† School of Engineering and Science
International University Bremen,§ Germany

‡ USDA Forest Service, Pacific Southwest Research Station
Center for Urban Forest Research, Davis, California

ABSTRACT

For city planning purposes, animations of growing trees of several species can be used to deduce which species may best fit a particular environment. The models used for the animation must conform to real measured data. We present an approach for inverse modeling to fit global growth parameters. The model comprises local production rules, which are iteratively and simultaneously applied to build a fractal branching structure, and incorporates the propensity of trees to grow towards light. The parameters of the local production rules are derived from global functions that describe the measured tree growth data over time. The production rules are influenced by the global light distribution, which is represented by the amount of light available at each position within the tree's crown. Since we want to allow the user to explore the tree's appearance interactively at any time during the animation, all modeling computations must be within a time frame that allows for interactive rendering rates. To this end, we developed a fast approximate algorithm for computing the light distribution. The rendering itself must also be fast; therefore, we sought a well-balanced compromise between photo-realism and performance. Because shadow computations play a key role for photo-realism, we developed a fast approximate shadow computation algorithm including soft shadows and self-shadowing. We applied our methods in order to model and animate the growth of seven single-stemmed tree species in an interactive setting.

Keywords
Tree Growth Modeling, Animation, Real Time Rendering.

1 Introduction

Trees produce benefits that enhance quality of life for city residents. Benefits include energy savings, air pollutant uptake, CO_2 sequestration, storm-water runoff reduction, increased property values and increased vitality in commercial areas. As trees grow larger, benefits increase as leaf surface area increases. However, the benefits can be offset when tree branches, and leaves conflict with other urban infrastructure, such as buildings, awnings, signs, traffic signals, and lighting. In dense urban environments limited space is a major constraint to tree planting. One goal of urban greening is to maximize functional benefits and minimize conflicts between trees and other infrastructure. To achieve this goal it is important to know (1) the mature size of different tree species, since both functionality and conflicts depend on tree size relative to the space that is available, and (2) the rate of growth for different trees, because this influences the length of time before conflicts occur, as well as how soon functional benefits are realized. Currently, landscape architects, city planners, and urban foresters lack tools that help them visualize how trees will grow over time, so that they can compare different species and select ones best suited to the site and their design objectives. Because trees are expensive to plant and can live to be hundreds of years old, computer-animated tree-growth modeling can help users make judicious choices that will pay large dividends over the long-term.

In computer graphics, several methods exist to describe and model computer-generated trees. The goal of these methods is to generate photo-realistic images of trees of selected species. The trees should appear as natural as possible and vary in appearance as they do

*srudnick@uni-greifswald.de
†l.linsen@iu-bremen.de
‡egmcpherson@ucdavis.edu
§Jacobs University Bremen as of Spring 2007

 ISBN 978-80-86943-98-5

in nature. Typically, the quality of the images, i.e. how natural the appearance of the trees is, is not quantitatively evaluated but just estimated by the human eye. Real measured data such as tree dimensions are not used to compare the models with nature.

For animations of trees growing over time, the complex growing process under the influence of many biological phenomena has to be considered. Some elaborate approaches exist. Our approach differs from most of these in two main ways:

1) Our tree growth is based on measured data.
2) Our entire tree growth modeling process is done on-the-fly, i.e. while users watch and interact with the animation. We achieve interactive rates for modeling and rendering.

The properties of our model allow users to interactively compare the growth of different tree species side by side. Because our model is based on real data the users confidence in the approach is increased. For a case study, we used global functions for seven single-stemmed tree species. Our growth model described in Section 3 is based on basic biological background on tree growth so that a realistic branching structure can be generated. We also used measured data for fitting the global shape properties of the respective species. Typically, a computer model for trees consists of a branching structure with position, length and orientation information for every branch. Local production rules are applied to each branch iteratively to generate the complex branching structure. We derive local production rules for our model based on the global functions given. Moreover tree growth cannot be determined by local rules only, but global phenomena like light seeking or shadow avoidance also have to be included. Thus our production rules also consider available light and its distribution.

For tree rendering, several photo-realistic approaches exist including sophisticated techniques such as the creation of soft shadows by multiple subsurface scattering. However, these approaches do not meet real-time requirements especially when dealing with steadily changing geometry. Thus we had to find a compromise between photo-realistic rendering and fast rendering. We observed that shadows are of high importance in this context. We developed a simple and fast method to approximate and render soft shadows. In addition, the effect of self-shadowing is included by drawing leaves with a luminance depending on the light distribution. Details of our rendering methods are described in Section 4.

2 Related Work

The modeling, simulation, and rendering of trees and plants is a well-studied topic in computer graphics. The formal description of trees is usually based on local production rules. Starting with a trunk and iteratively applying local production rules generates a complex branching structure.

The Lindenmayer systems (L-systems), introduced by the theoretical biologist Aristid Lindenmayer [Lin68], are the most common approach for a formal description of plants. In their book *The Algorithmic Beauty of Plants* [PL90], Prusinkiewicz and Lindenmayer describes the general concept of L-systems and their application to modeling different plants with different structures. They also introduced the concept of parametric L-systems in which the local production rules depend on parameters that are locally stored and updated. This concept has been used and extended in several algorithms [AK85, Blo85, LD99]. A survey of existing L-system approaches is given in [PHMH95].

Natural phenomena such as growth, death, reproduction, and information flow in growing plants can be modeled via L-systems. Prusinkiewicz et al. [PHHM97] explained how the L-system model applies to nature. The influence of the environment on the growth of plants is also considered.

For computer graphics applications, the main objective of modeling plants is to generate a highly realistic scene. Therefore stochastic tree models have been introduced to simulate the variety within one species. The individual, realistic-looking plants differ from each other and can be organized to render forests or fields [CSHD03] and even entire ecosystems [DCSD02]. The goal of the previous modeling systems differs from ours in that they weren't limited to interactive frame rates.

Approaches using inverse modeling of trees have been developed by Galbraith et al. [GMW04] and Prusinkiewicz et al. [PMKL01]. They used relative positions inside the crown to describe the local properties of the plant organs to generate trees that fit a given shape. Inverse modeling in this sense means, for example, that the relative position of a branch along the trunk (height of the starting point of the branch divided by the crown height) and the knowledge of the crown shape and dimension determine the length of the branch. This approach allows the use of real measured tree data such as height or crown shape.

Linsen et al. [LKMH05] presented a visualization method for tree growth at interactive frame rates using global functions to model the shape and size of the trees. They used some global parameters to determine the local production rules but their approach was too simple to produce realistic branch lengths. In their method, several biological phenomena were not considered including the fact that growth is periodical in years, or that plants try to avoid growing leaves in shadows.

Several approaches account for the light available for the growth of plants [Ben96, HB03, HdFBR04, SSBD03]. In these approaches a light source is modeled and the amount of light a leaf or a branch receives is computed. The illumination computation is very intense due to the complex and changing geometry of growing trees. The computations of the precise illumination inside a tree crown are further complicated by the reflected and transmitted light. To allow for in-

teractive modeling, our light computation needs to be much faster. We propose a fast approximate computation of the light distribution that is sufficiently precise for growth modeling and for realistic shadow computations.

3 Modeling

3.1 Global Functions

Our tree growth model is based on global functions of time that describe global properties of the tree during growth. For modeling we use functions for the diameter at breast height f_{DBH}, tree height f_H, crown diameter f_{CD} and the crown height f_{CH}. These functions are derived from real measured data for seven single-stemmed tree species, the London planetree (*Platanus x acerifolia*), silver maple (*Acer saccharinum*), Modesto ash (*Fraxinus excelsior*), goldenrain tree (*Koelreutaria paniculata*), Southern magnolia (*Magnolia grandiflora*), Chinese pistache (*Pistacia chinensis*), and hackberry (*Celtis occidentalis*).

3.2 Biological Background

We briefly summarize the biological characteristics of tree growth for single-stemmed trees that are important to our approach; for a more detailed explanation, we refer the reader to the literature [SS02, Mat91, Nul68].

A tree is a plant that grows over years, building up a large branching structure. The growth of a branch segment is divided into two phases, primary growth and secondary growth. Primary growth is the elongation of a bud from last year's growth to a shoot terminated by an apical bud. During primary growth, leaves and lateral buds are produced. After the first year's growth, branch segments start the secondary growth phase, which is characterized by steady thickening while elongation stops. The apical bud always produces the longest shoot and elongates the branch, while the lateral buds produce shorter shoots, a phenomenon known as apical dominance. The lateral buds do not always produce a shoot. The probability that they will produce a shoot is mainly dependent on the available light. A schematic example of the growth of one branch is shown in Figure 1.

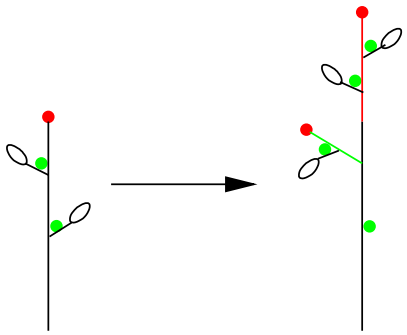

Figure 1: Annual growth of a branch. The apical bud (red) on the left produces an apical shoot (red) on the right with a new apical bud (red) and lateral buds (green). One of the lateral buds on the left produces a lateral shoot (green).

The leaves growing along the shoots produce energy by the process of photosynthesis. Thus trees tend to grow leaves in positions with good light. The different responses of plants to light and shadow are explained and discussed in *Photomorphogenesis in Plants* by Kendrick and Kronenberg [KK86]. Branches receiving enough light try to produce many leaves to produce energy. Thus they tend to bifurcate more often. Branches exposed to less light do not bifurcate as much; they tend to grow faster and to produce longer branches in order to reach for more light.

3.3 Tree Growth

A branching structure consists of bifurcations and branches. To generate complex branching structures, computer models typically use iteratively and simultaneously applied local production rules. In order to generate branching structures that consider global properties like the global functions derived in Section 3.1 or the available light distribution, we have to extend these local production rules to rules that couple local reproducibility with global growth properties.

In our model, a branch consists of branch segments representing the annual growth. A branch segment is defined by its length, diameter, starting point, and direction. A bifurcation is characterized by the bifurcation angle, the divergence angle, and the ratios of length and diameter between the parent branch and the child branches. The bifurcation angle describes the angle between the new and old branches and the divergence or twisting angle describes the change in orientation because not all branches lie in one plane. In addition to branches and bifurcations, we also model leaf growth. Flowers and fruits can be handled in the same way but are not included in our model, since our goal is the visualization of growth over several years while seasonal changes are omitted. The leaves spiral around the branch equidistantly. The leaves only grow on young and small branches that are exposed to sufficient light.

Tree growth is modeled in discrete steps each representing $\frac{1}{n}$-th of a year where n is exchangeable and can accommodate existing graphics hardware equipment. In our implementation we use $n = 20$. In each step, the production rules are applied to all branch segments. Using a pseudocode description, the recursive function for growing branches is given by:

```
grow (branch)
   if (primary growth)
        compute new length
   if (end of primary growth)
        produce apical branch segment
```

 ISBN 978-80-86943-98-5

```
      produce lateral branch segment
if (secondary growth)
      compute radius
      grow(apical branch)
      if (lateral branch exists)
           grow(lateral branch)
```

The result of "produce lateral branch segment" is a new lateral branch segment but only if the global parameters allow its generation. Otherwise nothing is done. A branch segment is in the phase of primary growth for one year. Afterwards, the secondary growth starts.

Every branch has an order representing its depth in the branching structure (Weibull ordering). The trunk has order 0, the branches connected to the trunk have order 1 and so on.

3.3.1 Trunk length

The trunk consists of two parts, one below the crown and one surrounded by the crown. The lengths of both are controlled by the global functions f_H and f_{CH}. The length of the part below the crown is calculated as the difference between the measured tree height and the measured height of the crown $l(t) = f_H(t) - f_{CH}(t)$ where t is the age of the tree in years. The length of the part surrounded by the crown is given by the measured height of the crown, $f_{CH}(t)$. The whole trunk consists of many trunk segments each representing annual growth and the last one is the only one in the primary growth phase. Thus the growth of the last segment is given by the difference between old and new crown height, $f_{CH}(t) - f_{CH}(t - \Delta t)$ for time steps of size Δt.

3.3.2 Branch length

For calculating the length of each branch, its order, its relative position, $p_r \in [0,1]$, along the parent branch, and the length of the parent branch, l_p, are used. The length calculation of the branches is done with the help of a crown shape function, $c : [0,1] \to \mathbb{R}$, which is represented by a geometric function that returns the relative length of a branch at a relative position, p_r, along the trunk to fit the crown shape (see Figure 2). The crown shape function is applied recursively for the entire branching structure, i.e., the length of each branch is determined with respect to the crown shape function applied to its parent branch.

Multiplying the relative length of a branch, $c(p_r)$, by the length of the parent branch, l_p, by an order-depending factor, and by a length scaling factor, s_l, gives the new length

$$l \quad = \quad s_l \cdot \frac{c(p_r) \cdot l_p}{0.5 + 0.5 \cdot order} \; .$$

This scaling factor $s_l \in \mathbb{R}_+$ randomly assigned to each branch during its initialization is uniformly distributed over $[0.6, 1]$. To allow for a few branches to break out

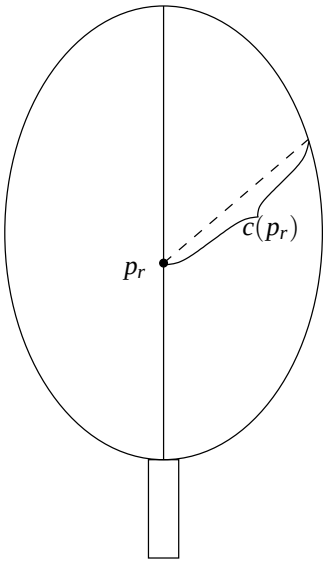

Figure 2: Example for elliptical crown shape function: p_r is the relative position of the starting point of a branch and $c(p_r)$ the computed relative branch length.

of the given shape and grow towards empty regions, 5% of the branches of order ≥ 2 are assigned a factor uniformly distributed over $[1.2, 1.5]$. Again the growth is only added to the length of the last branch segment, i.e., the only segment of the branch in primary growth phase.

Since branches grow towards light, we adjust our model so that light influences the scaling factor s_l. When there is only a little light at the branch apex, the scaling factor increases such that the branch reaches out of the shadow. When there is plenty of light at the apex of the branch, the scaling factor does not increase.

3.3.3 Radius

The radius of each branch segment is computed using the diameter at breast height function f_{DBH}. Similar to the method of scaling the length with factor s_l, we apply a scaling factor $s_r \in [0,1]$ to the radius. The new radius for each branch segment or trunk segment is given by

$$r(t) = s_r \cdot f_{DBH}(t) \; ,$$

where t, again, denotes the age of the branch segment in years. The scaling factor s_r for a branch segment is computed during initialization depending on the scaling factor $s_{r,f}$ of the previous branch segment as

$$s_r = s_{r,f} \cdot (1 - s_{r,f} \cdot 0.7) \; .$$

3.3.4 Bifurcation

Once the primary growth of a branch segment stops, the production rules for new branches have to be applied. As mentioned in Section 3.2, the probability

that a bud near the apex will produce a shoot is much larger than for the other buds. In our model, the apical bud always produces a new shoot, which elongates the branch and becomes a new segment. One lateral bud can produce a new shoot which starts a new branch of one order higher. The probability is determined by the local tree density, the available light at this position, and a random factor. Basically, plenty of light leads to many bifurcations and the absence of light leads to no further bifurcations.

For these new shoots, the scaling factors s_l and s_r are generated, and the angles characterizing the bifurcation are determined. The angles used are inspired by the ones that can be seen in nature in the respective tree species. In particular, branches tend to spiral around the parent branch.

3.4 Light

For the purpose of interactive modeling and visualization light calculations must be fast. We use a method to approximate the available light at each position inside the crown of the tree. We divide an axes-aligned box that covers the full-grown tree into discrete volumes. We send light rays simulating daylight through the scene that lose intensity as they pass through the cells. The loss of intensity is proportional to the percentage of the cell's space covered by branches and leaves.

After each growth step, the light distribution within the crown has to be recomputed. Therefore, we define equidistant planes parallel to the ground in the crown space. Thus, the bounding box of the tree is partitioned into slabs. Each branch is projected into the lower plane in each slab using parallel projection while information on its radius and number of leaves is stored.

Each plane is divided into squares forming a regular grid. The size of the grid is adapted to the size of the crown. For each square S, a density, $D \in [0,1]$, is computed that measures the percentage of the square not covered by branches or leaves

$$D = \prod_{i \in S} (1 - A(i)) \cdot (1 - l(i))$$

where $A(i)$ is the area of the part of the projected branch i that intersects S, and $l(i)$ is the area of the leaves of i whose projection intersects S. Density D can be interpreted as the density of the cuboid above the square and it approximates the percentage of the incoming light that passes through this cuboid. Note that branches that are projected to the same position in the square are both fully considered as we are computing the density of the cuboid.

With this information we need to simulate daylight to compute the illumination within the trees crown. The direction of the incoming sunlight varies over the day. Thus we cannot assume a steady light source with parallel rays, but need to consider directional light. Benes [Ben96] discussed the effect of daylight and its intensity and direction. The light enters the scene with the greatest intensity from the top and decreases towards the sides. Therefore we send many rays from the top and fewer rays from the sides as shown in Figure 3. For interactive modeling, the efficiency of the

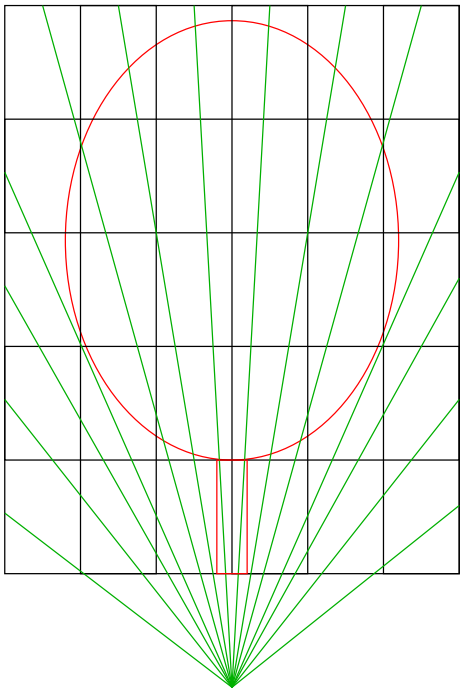

Figure 3: Light rays (green) from the boundary faces directed underneath the tree are used to simulate daylight.

ray traversal computations is important. We adopt the idea of Bresenham's line-drawing algorithm [Bre65] and generalize it to 3D to quickly compute the sequence of cuboids that are traversed by each ray. The amount of light available for each cuboid accumulates with each ray that traverses it. If a ray with intensity I_r hits a cuboid with previous intensity I_c, I_c gets updated using the accumulation rule

$$I_c = 1 - ((1 - I_c) \cdot (1 - I_r)) = I_c + I_r - I_c I_r \ .$$

If the cuboid has density D, then the intensity of the traversing ray I_r is reduced by multiplying it by D. After all ray traversals we have a good approximation of the light available for each cuboid.

4 Rendering

In our implementation, all geometry is rendered as textured polygons. As a texture for the bark, we used photographs of the trees' bark, and as a texture for the leaves, we scanned a typical leaf or took photographs.

 ISBN 978-80-86943-98-5

4.1 Branches

For each branch a closed mesh of generalized cylinders is generated. For branches of higher orders, a new mesh is generated that starts inside the parent branch. Hence, we do not have to compute a special bifurcation structure but instead generate the impression of branches coming out of the parent branch.

4.2 Leaves

The leaves surrounding a branch are rendered as equidistant triangles twisted around the branch. The position of each leaf has to be computed after every growth step as it depends on the new position of the branch and on the new diameter of the branch. For each leaf, the coordinates of a triangle are computed in such a way that the triangles are tangent to the branch. Therefore, no leaf stalk starts inside the branch, and each leaf is connected to the branch. These triangles are rendered using a partially transparent leaf texture.

4.3 Shadows

Shadows are very important for natural-looking scenes. For realistic shadows, we need soft shadowing, which is too time-consuming to compute precisely. Instead, we approximately compute hard shadows and render them using textures with higher opacity in the center and decreasing opacity to the boundary. We can render such polygons in real-time and generate soft shadows.

The positions of the shadows of the branches and the attached leaves are computed using the results of the light computation described in Section 3.4. As a result of the light computation, we obtain planes with projected coordinates of all branch segments. In addition, we store information about their diameters and the leaves at these branch segments. For each projected branch segment, the coordinates of a quadrilateral are computed. For the shadows of the leaves the coordinates of small squares around the respective projected branch are computed. The soft-shadow textures of all branches and leaves are accumulated in a texture buffer, which is rendered at each frame.

Real shadowing includes not only the shadows underneath the tree on the ground but also self-shadowing within the tree. We apply self-shadowing to the leaves with the help of the available light information obtained from the computations described in Section 3.4. The leaves are divided into 10 luminance classes depending on the amount of incoming light. The leaves of each class are rendered with the respective luminance so that leaves appear darker or brighter due to the light that arrives at the small cuboid in which they are located. All leaves are rendered with the same texture.

4.4 Implementation

For every branch segment, the orientation is stored as a rotation relative to the orientation of the previous branch segment. These rotations are not stored as matrices but as quaternions to avoid matrix computations. The branch segments are stored in a binary tree with the first trunk segment being the root. While traversing the tree, the start position of a branch segment is passed as a parameter of the parent branch segment. This position can be used directly, since only the ancestors affect this new position. With this position and the orientation, the coordinates of the generalized cylinders are computed, again with the help of quaternions. Even the leaf positions are computed this way.

The simulation can be stopped at any time for a closer look at the tree of this age. To speed up the performance, we save all the necessary coordinates, normals and texture coordinates in each step and render them at once. Therefore no new geometry computations have to be performed when the simulation is interrupted.

5 Results and Discussion

We animated the growth of prototypical trees from 4 to 50 years. We did not start the animation earlier, as we had no data available for trees younger than 4 years. The animation can be halted at any time to have a closer look at a tree of a certain age. The frame rates we achieved allow for changing the viewpoint interactively even while the animation continues, and growth modeling processes are executed in addition to the rendering.

We were able to animate trees of realistic natural appearance. Moreover, the trees fit the global measured functions of tree height, crown height and width, trunk length, and radius. One can also observe common natural phenomena of trees within our computer-generated models. For example, the interior of the crown exhibits fewer branches and leaves than along the outside due to the light distribution. This shows that using light information for modeling growth really helps improve the model, getting closer to a real tree. The branching structure of the trees looks very realistic. It is not as regular as it would be when applying simple L-systems, and it follows natural laws of tree growth. For the local production rules, we use several stochastic components to achieve a higher diversity within our trees and also among our trees.

In Figure 4, some screen-shots of growing trees taken from the animation can be seen. The trees all have different densities, crown shapes, and overall appearances. The last screen-shot of the two Southern magnolia trees shows the effect randomness has on the trees. Both trees use the same light and growth parameters, but have a distinct appearance. In addition, we provide a video showing the animated growth at `www.math-inf.uni-greifswald.de/informatikJP/rudnick/video`.

Depending on the computer system used, the interac-

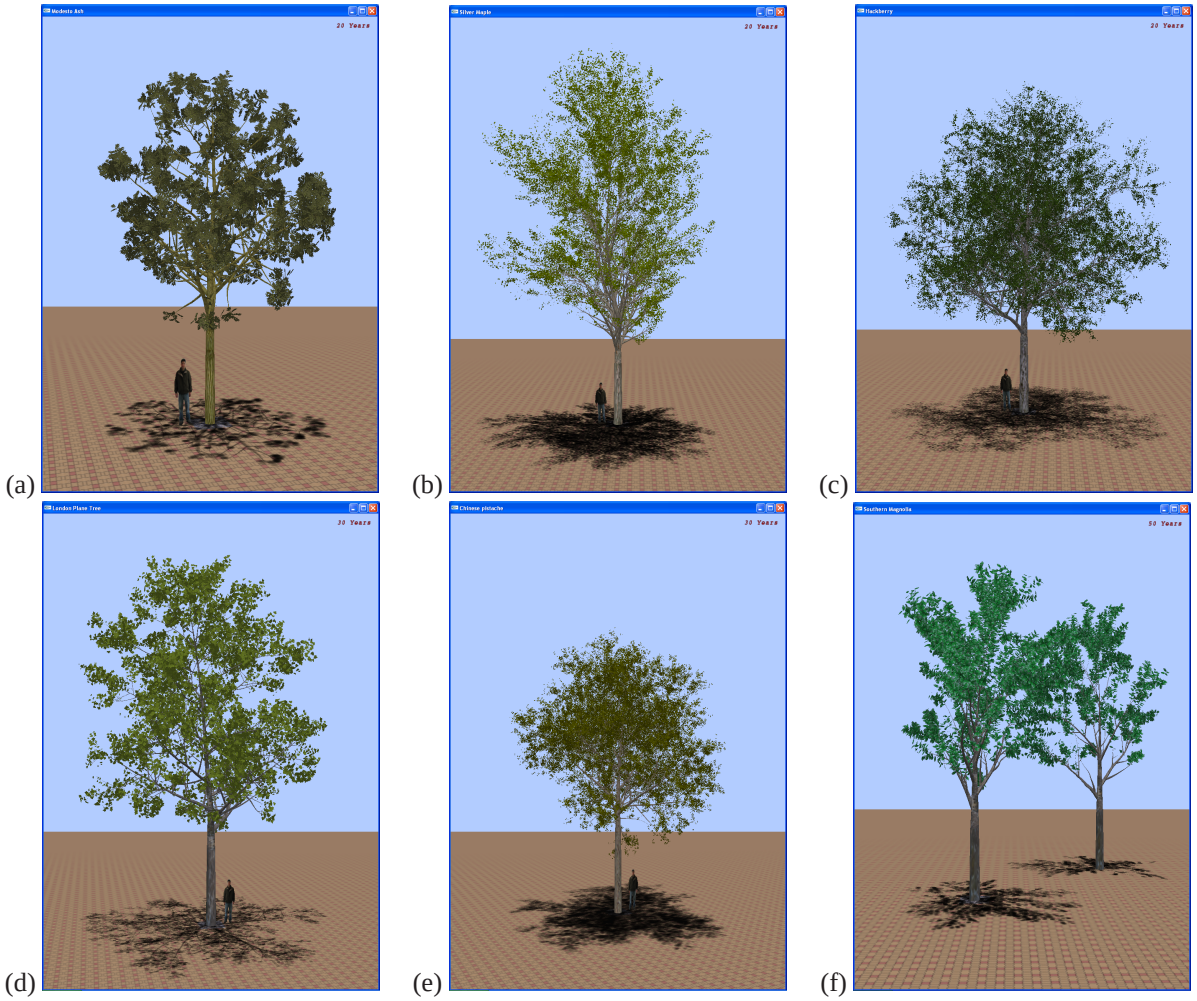

Figure 4: Screen-shots of different tree species taken from our animations: Modesto ash at 20 years (a), silver maple at 20 years (b), hackberry at 20 years (c), London planetree at 30 years (d), Chinese pistache at 30 years (e), and two Southern magnolias at 50 years (f).

tive animation may slow down a little for older trees due to the large number of branch segments. Obviously the time for all calculations depends on the number of branch segments. In every growth step, the local production rules have to be applied to all segments, the light information for every position inside the crown has to be computed, and every branch has to store this information for its own position. For rendering all polygons of branches, leaves, and shadows, the respective geometry has to be computed. When the growth simulation is halted, the animation achieves a better frame rate because no growth steps have to be performed and only the rendering computations are needed. In Table 1, the computation time in milliseconds for the individual modeling computations and the rendering is shown dependent on the number of branch segments. We enlist the computation time for the growth modeling (applying production rules), for updating the light distribution, and for generating the mesh geometry. The Southern magnolia, for example, with 30,000 branch segments is rendered with nearly half a million polygons. The times were measured on a PC equipped with a 3.06GHz processor and an NVidia GeForce 6800 ultra graphics card.

Although implementation on the GPU may be used to further improve the frame rates, our overall goal was to distribute our program over the web for city planning purpose. Thus, our system should run on any personal computer with any hardware configuration. Thus, we decided not to require specific graphics hardware.

6 Conclusions and Future Work

We have presented a real-time tree-growth animation for urban street trees. The real-time computations include all tree-growth modeling steps, geometry generation, and rendering. The growth of the trees is controlled by given, global functions of time for tree height, diameter at breast height, and crown height and width. In our model, we have mapped these global parameters to local production rules. The trees are generated by simultaneously and iteratively applying lo-

 ISBN 978-80-86943-98-5

# branch segments	computation times (in *ms*)			rendering time (in *ms*)
	growth	light	mesh	
5,000	7	7	2	16
10,000	13	8	12	16
20,000	27	12	30	33
30,000	40	13	60	33

Table 1: The computation time in milliseconds for each frame, dependent on the number of branch segments. A tree with 30,000 branch segments is quite a large and old tree. It consists of nearly half a million polygons.

cal production rules to all branches. This approach produces trees with a natural branching structure that fits the global properties. In our model, we have also included the influence of light on the growth of the tree. We used a data structure giving us global control of the tree and allowing us to compute the available light at every position inside the crown at any time. As a result, crown density conforms to characteristics of measured trees, as does crown size.

For rendering purposes, we used a polygon-based approach with photographs and scans for textures. The appearance of the tree is highly realistic when considering real-time constraints. In our animation we reached interactive rates due to the sophisticated yet simple modeling and rendering methods. For shadow computations including self shadowing we made use of the light distributions computed during modeling.

We applied our model to seven different single-stemmed tree species. For future work we plan to apply our method to additional tree species, including multi-stemmed ones. We will also model conifers when the necessary global functions are available. Another objective for future work is to include the effects of buildings on light distribution and tree-growth.

Acknowledgments

This research was supported in part by funds provided by the U.S.D.A. Forest Service, Pacific Southwest Research Station, and the Elvenia Slosson Research Endowment for Ornamental Horticulture at the University of California, Davis.

References

[AK85] M. Aono and T.L. Kunii. Botanical tree image generation. *IEEE Computer Graphics & Applications*, 4(5):10–34, 1985.

[Ben96] B. Benes. An efficient estimation of light in simulation of plant development. In *Computer Animation and Simulation '96*, pages 153–165. Springer-Verlag, 1996.

[Blo85] J. Bloomenthal. Modeling the mighty maple. In *SIGGRAPH '85: Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, pages 305–311, New York, NY, USA, 1985. ACM Press.

[Bre65] J. E. Bresenham. Algorithm for computer control of a digital plotter. *IBM System Journal*, 4(1):25–30, July 1965.

[CSHD03] M. Cohen, J. Shade, S. Hiller, and O. Deussen. Wang tiles for texture and image generation. In A.P. Rockwood, editor, *SIGGRAPH '03 Proceedings*. ACM SIGGRAPH, 2003.

[DCSD02] O. Deussen, C. Colditz, M. Stamminger, and G. Drettakis. Interactive visualization of complex plant ecosystems. In M. Gross, K. I. Joy, and R. J. Moorhead, editors, *Proceedings of IEEE Visualization '02 conference*. IEEE Computer Society Press, 2002.

[GMW04] C. Galbraith, L. Mündermann, and B. Wyvill. Implicit visualization and inverse modeling of growing trees. *Computer Graphics Forum (Proceedings of Eurographics 2004)*, 23(3), 2004.

[HB03] W. Van Haevre and P. Bekaert. A simple but effective algorithm to model the competition of vitual plants for light and space. *Journal of WSCG*, 11(1), 2003.

[HdFBR04] W. Van Haevre, F. di Fiore, P. Bekaert, and F. Van Reeth. A ray density estimation approach to take into account environmental illumination in plant growth simulation. In *SCCG '04: Proceedings of the 20th spring conference on Computer graphics*, pages 121–131. ACM Press, 2004.

[KK86] R. E. Kendrick and G. H. M. Kronenberg. *Photomorphogenesis in Plants*. Kluwer Academic Publishers, 1986.

[LD99] B. Lintermann and O. Deussen. Interactive modeling of plants. *IEEE Computer Graphics & Applications*, 19(1), 1999.

[Lin68] A. Lindenmayer. Mathematical models for cellular interaction in development. *Journal of theoretical biology*, 18:280–315, 1968.

[LKMH05] L. Linsen, B. J. Karis, E. G. McPherson, and B. Hamann. Tree growth visualization. *Journal of WSCG*, 13((1)–(3)):81–88, 2005.

[Mat91] C. Mattheck. *Trees: The mechanical Design*. Springer-Verlag, Berlin Heidelberg, 1991.

[Nul68] W. Nultsch. *Allgemeine Botanik*. Georg Thieme Verlag, Stuttgart, 1968.

[PHHM97] P. Prusinkiewicz, M. Hammel, J. Hanan, and R. Mech. Visual models of plant development. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages, Vol. III: Beyond Words*, pages 535–597. Springer-Verlag, Berlin, 1997.

[PHM93] P. Prusinkiewicz, M. Hammel, and E. Mjolsness. Animation of plant development. In J.T. Kajiya, editor, *Computer Graphics (SIGGRAPH '93 Proceedings)*, volume 27, pages 351–360. ACM SIGGRAPH, 1993.

[PHMH95] P. Prusinkiewicz, M. Hammel, R. Mech, and J. Hanan. The artificial life of plants: Artificial life for graphics, animation, and virtual reality. In *SIGGRAPH '95 Course Notes*, pages 1–38. ACM SIGGRAPH, 1995.

[PL90] P. Prusinkiewicz and A. Lindenmayer. *The algorithmic beauty of plants*. Springer-Verlag, New York, 1990.

[PMKL01] P. Prusinkiewicz, L. Mündermann, R. Karwowski, and B. Lane. The use of positional information in the modeling of plants. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 289–300, New York, NY, USA, 2001. ACM Press.

[SS02] E. Strasburger and P. Sitte. *Lehrbuch der Botanik für Hochschulen*. Spektrum, Akad. Verlag, Heidelberg, 2002.

[SSBD03] C. Soler, F.X. Sillion, F. Blaise, and P. Dereffye. An efficient instantiation algorithm for simulationg radiant energy transfer in plnat models. *ACM Trans. Graph.*, 22(2):204–233, 2003.