A Multiresolution Model for Non-photorealistic Rendering of Trees

Celso Campos[1], Ricardo Quirós[2], Joaquin Huerta[2], Emilio Camahort[3], Roberto Vivó[3], and Javier Lluch[3]

[1] Departamento de Lenguajes y Sistemas Informáticos, Universidad de Vigo, Spain
ccampos@ei.uvigo.es
[2] Departamento de Lenguajes y Sistemas Informáticos, Universitat Jaume I, Spain
{quiros, huerta}@lsi.uji.es
[3] Departamento de Sistemas Informáticos y Computación,
Universidad Politécnica de Valencia, Spain
{camahort, rvivo, jlluch}@dsic.upv.es

Abstract. Modeling and rendering of trees has recently received a lot of attention. Models have been developed that allow photorealistic rendering of trees at interactive frame rates. However, little attention has been devoted to expressive rendering of these models. In this work we present a multiresolution model designed specifically to speed up painterly rendering of trees. Our method proposes a novel clustering technique based on the computation of nested convex hulls. We employ variable multiresolution to obtain efficient models that contain higher resolution representations for the outside of the tree and lower resolution representations for the inner parts. This variable multiresolution method mimics the techniques used by traditional artists to paint trees.

1 Introduction

Due to the intrinsic complexity of trees, the models that represent them are made of large amounts of polygons. Therefore, we can not achieve interactive rates when rendering these models. Current research in the literature used acceleration techniques based on replacing geometry by images. Only a few methods work with geometry.

Geometry, however, is necessary in some cases like in painterly rendering of trees. In this case, the most popular method of representation is polygonal models. The mathematical simplicity of this type of representation makes it possible to render a great number of polygons with the current graphics hardware. However, due to the vast amount of polygons that compose the tree models, it is necessary to use some method that reduces the number of polygons without loss of visual appearance. The most common techniques are multiresolution models or discrete level of detail (LOD) models.

Most of the existing generic multiresolution models can not be applied to our problem, since these models work with polygonal surfaces, whereas the leaves of a tree must be handled as independent polygons. Other specific models for vegetation also present important limitations. The model proposed by Lluch et al. [3] uses precomputed images that replace the leaves, which eliminates the geometry necessary to generate the brush strokes. The work presented by Remolar et al. [11] proposes a

V.N. Alexandrov et al. (Eds.): ICCS 2006, Part II, LNCS 3992, pp. 310 – 317, 2006.

simplification algorithm (LSA: Leaf Simplification Algorithm) that collapses pairs of leaves preventing more efficient approaches.

In this work we present a geometry-based multiresolution model that allows the expressive rendering of trees using brush strokes. We introduce a clustering algorithm to organize the leaves of a tree. Then we propose a simplification algorithm suitable for rendering trees expressively. Our proposal extends the LSA algorithm and adapts it to the requirements of this type of rendering. Our models render at interactive rates on modern graphics hardware.

2 Previous Work

Our work is primarily related to two different areas of computer graphics: non-photorealistic rendering of plants and trees and multiresolution modeling.

The first methods for automatic illustration of vegetable species were introduced by Yessios [5] and Sasada [6]. They both produce tree sketches for architectural applications. Kowalski et al. [7] and Markosian et al. [8] also create abstract tree sketches using geometric primitives that approximate the tree's foliage. Deussen [9] presents a method that creates pen-and-ink illustrations of plants and trees. His method supports the representation of specific plants and trees and not just generic trees like those in [7] and [8]. More recently Di Fiore [10] proposed a method that renders cartoon shaded trees from models generated from L-systems.

In previous work [4] we presented a stroke-based method for non-photorealistic rendering of plants and trees. This method improves on Meier's algorithm [1] by supporting interactive frame rates. The method can be applied to videogames, virtual walkthroughs and other real-time computer graphics applications. It models trees using random parametric L-Systems (RL-Systems) [2]. This approach has several advantages over the surface patch algorithm of Meier. For example, it supports the simultaneous generation of both the tree's geometry and its stroke particles for future rendering. The stroke particles are distributed using the same RL-system that is used for modeling the tree's geometry. To achieve this goal we use a *shape instantiation* process. This process represents every instantiable object - a branch or leaf - using both 3D geometry and a cloud of strokes.

Multiresolution is a modeling technique that was first introduced to accelerate rendering of complex geometries [14][16]. Additionally, multiresolution techniques have been proposed for image representations [17], curve and surface modeling [12], and volumetric data sets [13].

Geometry-based simplification methods have been successfully applied to many areas of computer graphics. However, they fail to maintain the general structure of a tree. During the process, the tree's volume diminishes and loses its visual appearance. Some techniques have been proposed for multiresolution models for plants and trees, like degradation at range, space partitioning, layered depth images, volumetric textures, bidirectional textures, leaf impostors and leaf collapsing [15].

However, the proposed models only work for realistic purposes and we need specific techniques to render them expressively.

3 Multiresolution Model

RL-Systems allow modeling trees with realistic appearance preserving their structural complexity. This amount of detail can be suitable for photorealistic rendering, but it may be excessive for stroke-based rendering. We propose solving this problem using a multiresolution representation for the trees.

One of the main components of a multiresolution model is its data structure, because it affects the quality of the final rendering of the model. We start with an initial data structure that maintains the structural complexity of the tree. Then a procedure builds the tree's multiresolution model. The model is made of two different data structures: one for the branch brush strokes and the other for the leaf brush strokes.

The extended data structure stores the data implicitly, both for the branches and for the leaves. To limit the size of the data structure we do not store all possible LODs of the tree. Implicit storage also avoids sudden changes (jumps) between contiguous LODs. Although using implicit storage generally increases the cost of LOD extraction, in our case the increase is minimum.

3.1 Branch Structure

A tree modeled using an RL System undergoes an instantiation process that generates both a geometric and a brush-stoke representation for branches, leaves, flowers and fruit [4]. The brush-stroke representation is stored in a linked list to optimize memory management. The process generates a branch representation made of multiple LODs. Each LODs has a certain number of strokes starting at three strokes per branch. The minimum LOD provides a low quality result for close views of the tree. The quality can be improved by rendering a larger number of brush strokes for each branch.

To add new brush strokes, we use an adaptive scheme [4] based on the branching level. For each branch we may have several LODs. The LOD finally selected for each branch depends on its branching level. The amount of detail (number of strokes) decreases from the trunk to the outer branches. Strokes are stored in a list containing all LODs organized sequentially. At rendering time the list is traversed until the desired LOD is reached.

3.2 Leaf Structure

Each leaf is represented using a brush stroke. For large trees this may require so many brush strokes that rendering may not be interactive. We propose using a model that stores a list of nodes. Each node contains the brush strokes associated to a cluster of leaves. The model stores an implicit representation of the different LODs. This implies that we do not store all computed LODs.

To classify the original leaves in the different nodes we define criteria for clustering and simplification. These criteria are described with detail in the following section. The width of the leaf stroke data structure (see Figure 1) depends on the number of clusters, whereas the depth depends on the number of LODs selected by the user. Width and depth are computed in the clustering process (Section 4). For our test tree we use a structure of four LODs and thirty five clusters.

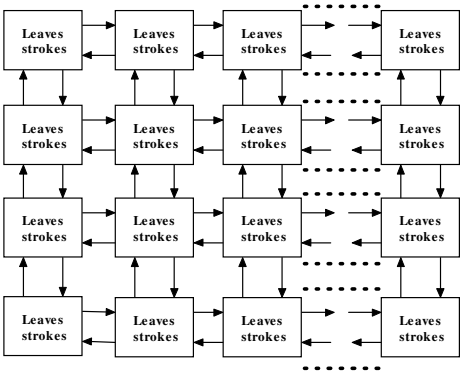

Fig. 1. Data structure of our foliage representation: the top row contains the clusters of leaves; lower rows contain lower LOD representations of the leaf clusters

4 Clustering and Simplification

The generation of the different LODs of the tree leaves requires a simplification method to decide which information is less relevant and remove it from the model. To implement a simplification method we first cluster the brush strokes of the leaves.

There are two possible approaches. We can cluster the leaves in pairs and then collapse each pair into a single object [11]. Or we can use information about the topology of the tree to cluster the leaves into groups that, for example, share the same branch [10]. This second approach preserves the natural layout of the leaves but it produces a suboptimal simplification.

Our simplification method could benefit from this clustering method, since it gives better visual consistency to certain trees like coniferous. However, we propose a new clustering method based on nested convex hulls.

4.1 Clustering Method

To cluster the brush strokes of the leaves we compute a set of nested convex hulls. Then we fill the clusters with the strokes between pairs of neighboring convex hulls. This method is based on techniques used by traditional artists to paint trees. The painter uses initially large rough strokes to paint the tree, and then adds detail using fine strokes over the original ones.

To compute the set of convex hulls, we take the cloud of points formed by the centers of the polygons of all leaves. We obtain their convex hull and remove the points belonging to the convex hull from the initial cloud. We repeat this process until there are three or less points left. This produces a large set of convex hulls. So we only keep a user-selected number of them. They are chosen equally spaced between the largest convex hull and the smallest convex hull that contains at least 10% of the total number of leaf strokes. Given this set of leaf clusters we apply our simplification algorithm to each of them. This approach supports simplifying inner clusters more than outer clusters.

4.2 Simplification Method

For simplification we use the LSA algorithm [11] with two improvements: we allow leaf collapses of more than two leaves, and we obviate the requirement to compute the orientation of the tree with respect to the viewer during rendering. This speeds up stroke extraction, thus reducing rendering time.

We collapse leaves using the distance and coplanarity criteria of the LSA. As simplification proceeds and coarser LODs are generated, we increase the distance and coplanarity angle thresholds. After running, our algorithm produces a set of LODs for each cluster of strokes. Each LOD contains groups of brush strokes simplified to single strokes.

5 Rendering

Multiresolution modeling requires a data representation, a simplification algorithm and an LOD extraction algorithm for rendering. In this Section we describe how to extract LODs for both branch and leaf brush strokes.

5.1 Extraction of Brush Strokes Associated to Branches

The brush strokes associated to the branches are stored in a linked list (see Section 3.1). To extract an LOD we traverse the list rendering all brush strokes we find until a maximum number is reached. This number depends on the branching level and the distance from the viewer to the tree. Our method includes a pruning step that removes branches whose projected length is so short that they can not be perceived.

5.2 Extraction of Brush Strokes Associated to Leaves

After applying the simplification algorithm, we know the number of strokes stored at each node of the resulting data structure. For each row of the structure we store the total number of strokes. We use these numbers to decide how to traverse the structure to extract the desired LOD.

The process begins computing the distance from the observer to the model. Based on this distance we determine the number of brush strokes to display. With that number we select the two rows that contain the closest numbers of strokes. We traverse the two rows and select the combination of nodes that better fits the desired number of strokes. Figure 2 shows two example rows with all possible traversals in different

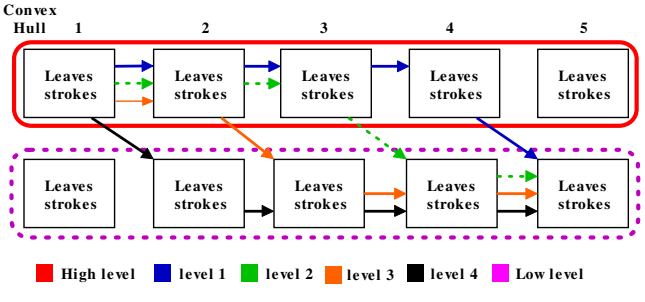

Fig. 2. LOD extraction from two rows of the leaf stroke data structure

colors. Note that the first nodes in the selection always belong to the top row and the last nodes to the bottom row.

6 Results

We have applied our algorithms to a generic ternary tree with 9841 branches and 8748 leaves. We have rendered the tree using OpenGL in a PC with Windows 2000, a 2.8 GHz AMD Athlon processor and an nVidia GeForce FX 5200 with 128 Mb. Figure 3 shows four renderings of the branches of the tree at different LODs.

We have clustered the leaves into 35 convex hulls each of them simplified four imes. Figure 4 shows four LODs of the leaves of our test tree. Simplified leaf trokes have been drawn in red to show how simplification proceeds from the nside to the outside of the tree. Figure 5 shows a front view of the tree endered at four different

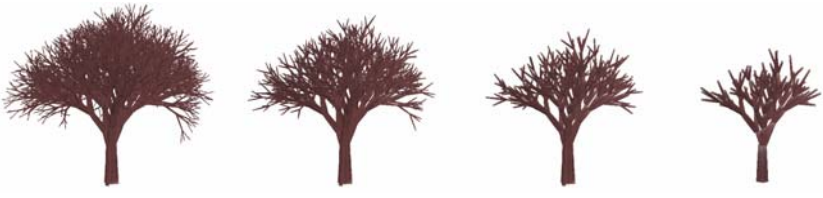

Fig. 3. The branches of the test tree and three finer LODs generated with our method. The trees contain 33663, 3754, 1159 and 363 branch strokes with pruning.

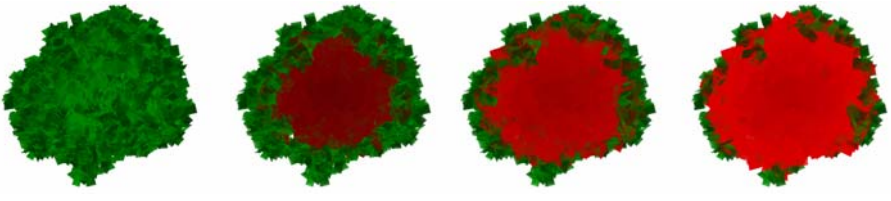

Fig. 4. The test tree and three finer LODs. They have 8748, 4156, 2463 and 761 leaf strokes.

Fig. 5. The test tree and three finer LODs. They have 42411, 7910, 3622 and 1124 strokes.

Table 1. Rendering times of the tree models shown in Figures 3 and 5 (in msec)

Only branches (Fig. 3)	Test Tree	LOD 1	LOD 2	LOD 3
Number of strokes	33663	3754	1159	363
Rendering time	66.7	27.8	13.5	13.5
Full tree (Fig. 5)	Test Tree	LOD 1	LOD 2	LOD 3
Number of strokes	42411	7910	3622	1124
Rendering time	250	125	91	66.7

LODs. The top row shows the LODs at the same size, he bottom row shows real rendering sizes. This view shows better the results f our simplification algorithm.

Table 1 shows the rendering times of the models depicted in Figures 3 and 5.

7 Conclusions and Future Work

We have presented a multiresolution model specially designed for painterly rendering of trees. For each tree we generate and render brush strokes. Our solution separately handles the branches and the leaves of a tree. The storage of the LODs is implicit, avoiding an excessive increase in the spatial cost of the data structure. Our method allows efficient extraction and rendering of LODs. Rendering runs at interactive rates and sudden changes between contiguous LODs are not noticeable.

We can extend our method to include viewer-distance dependent adaptive simplification. We are also considering applying our model to photorealistic rendering and including the structural information of the trees in the clustering process. This would produce better approximations for trees like coniferous.

We want to extend our model to use occlusion techniques for modeling large tree populations. We also want to create a tree database, and we want to extend our expressive rendering algorithms to support other artistic techniques like sketching, engraving and half-toning.

Acknowledgements

This work was partially supported by grant 05VI-1C02 of the University of Vigo, rant TIN2005-08863-C03 of the Spanish Ministry of Education and Science and by STREP project IST-004363 of the 6th Framework Program of the European Union.

References

1. B. J. Meier, "Painterly rendering for animation", Proceedings of SIGGRAPH 96, pp 477-484, Agosto 1996. New Orleans, Louisiana.
2. J. Lluch, M. J. Vicent, R. Vivó, and R. Quirós, "GREEN: A new tool for modelling natural elements", Proceedings of WSCG'2000 International Conference on Computer Graphics and Rendering, Plzen, 2000.
3. J. Lluch, E. Camahort, R. Vivó, "An Image-Based Multiresolution Model for Interactive Foliage Rendering", The 12-th International Conference in Central Europe on Computer Graphics, Rendering and Computer Vision 2004, Czech Republic.

4. C. Campos, R. Quirós, J. Huerta, M. Chover, J. Lluch, and R. Vivó, "Non Photorealistic Rendering of Plants and Trees", International Conference on Augmented, Virtual Environments and Three-Dimensional Imaging, ICAV3D, Greece, 2001.
5. C. I. Yessios, "Computer drafting of stones, wood, plant and ground materials." Proceedings of SIGGRAPH'79 , pp 190-198, 1979.
6. T. T. Sasada, "Drawing Natural Scenery by Computer Graphics", Computer-Aided Design, vol. 19, pp 212-218, 1987.
7. M. A. Kowalski, L. Markosian, J. D. Northrup, L. D. Bourdev, R. Barzel, L. S. Holden, and J. F. Hughes, "Art-Based Rendering of Fur, Grass and Trees", Proceedings of SIGGRAPH 99, pp 433-438, Agosto 1999. Los Angeles, California.
8. L. Markosian, B. J. Meier, M. A. Kowalski, L. S. Holden, J. D. Northrup, and J. F. Hughes, "Art-based Rendering with Continous Levels of Detail", Proceedings of NPAR 2000, Annecy, France, 2000.
9. O. Deussen and T. Strothotte, "Computer-Generated Pen-and-Ink Illustration of Trees", Proceedings of SIGGRAPH 2000, pp 13-18, Julio 2000.
10. F. Di Fiore, W. Van Haevre, and F. Van Reeth, "Rendering Artistic and Believable Trees for Cartoon Animation", Proceedings of CGI2003, 2003.
11. I. Remolar, C. Rebollo, M. Chover, J. Ribelles, Real-Time Tree Rendering, Lecture Notes in Computer Science, Proc. of Computational Science ICCS 2004, Springer, ISBN/ISSN 3-540-22129-8, krakow (Poland), vol. 3039, pp. 173-180, June, 2004.
12. Matthias Eck, Tony DeRose, Tom Duchamp, Hugues Hoppe, Michael Lounsbery, and Werner Stuetzle. Multiresolution analysis of arbitrary meshes. Proceedings of SIGGRAPH 95, Computer Graphics Proceedings, Annual Conference Series, August 1995, pages 173-182.
13. T. He, L. Hong, A. Kaufman, A. Varshney and S. Wang. Voxel-based object simplification. Proceedings of Rendering 95, IEEE Computer Society Press, 1995.
14. Paul Heckbert and Michael Garland. Multiresolution Modeling For Fast Rendering. In Proceedings of Graphics Interface '94, pages 43-50.
15. O.Deussen, P.Hanrahan, B.Lintermann, R.Měch, M.Pharr, P.Prusinkiewicz. "Realistic modeling and rendering of plant ecosystems." In Proceedings of SIGGRAPH 98, 275-286.
16. E. Puppo and R. Scopigno. Simplification, LOD and Multiresolution Principles and Applications. In Proceedings of EUROGRAPHICS 97. CG Forum, vol. 16, no. 3, 1997.
17. Azriel Rosenfeld. Multiresolution Image Processing and Analysis. Springer-Verlag, Berlin, 1984.