A GIS-based Forest Visual Simulation System

Qizhi Yu[1, 2] Chongcheng Chen[2,3] Zhigeng Pan[1] Jianwei Li[2,3]
[1]*State Key Laboratory of CAD & CG, Zhejiang University, Hangzhou 310027, China*
[2]*Spatial Information Research Center of Fujian Province, Fuzhou University, Fuzhou 350002, China*
[3]*Key Lab. of Data Mining & Information Sharing, Ministry of Education (Fuzhou University), Fuzhou, 350002, China*
E-mail: zgpan@cad.zju.edu.cn

Abstract

This paper reports on a visual simulation system that supports GIS-based realistic modeling and real-time rendering of forest scenes. Geometric models of trees are automatically generated according to inventory database and pre-designed template models. A combined image and geometry representation method for 3D tree model is given with a specific level of detail algorithm for ensuring real-time frame rates. We have tested and evaluated the system in applications of walkthrough simulation and forest fire visualization.

1. Introduction

Modeling and real-time rendering of forest scenes based on real-world data from GIS is an important and challenging problem in forestry applications [10]. 3D visualization of forest landscape has been increasingly used in planning processes as means of communication between planners, clients and the public, in particular for the discussion of visual impacts of proposed changes in the landscape [4]. In addition, real-time visualization of forest scenes is necessary in virtual reality systems towards forest industry such as VR-based fire-fighting training.

In the last two decades, modeling and rendering of plants and forest scenes has been intensively explored in the field of computer graphics. However, few of the published work focus on plants modeling from real forest data at run-time. We refer the reader to [7] and [3] for a broad review.

The goal of this project is to develop a visual simulation system that supports GIS-based realistic modeling and real-time rendering of forest scenes. We also have integrated a simulation module of forest fires into our system.

2. System overview

2.1 System components

The system consists of six major components: Input Data, Procedure Tree Geometry Engine, Template Model Designer, Forest Modeler, Forest Render, Terrain Render Engine, and GIS Software Package. Here we will only discuss the system components directly related to forest visualization.

Input Data: The input data required by forest visualization mainly include forest inventory database and DEM.

Forest inventory database are generally managed not in units of individual trees, but forest stands. A forest stand is a group of trees that have similar structures and are in the same growth stage. In this project, a forest stand table contains information on the area of a stand, together with the dominant plant species, average height, average crown width, average diameter at breast height (DBH) and density of the dominant trees in the stand. A forest stand map expresses the boundaries of each stand.

A DEM is a regular grid of terrain elevation values on which each point is interpolated from spot height measurement or contour data. It is used to render the terrain and determine the height of the base of tree objects.

Procedure Tree Geometry Engine: The procedure tree geometry engine is a key component of our system. Both the parameterized modeling algorithm and LOD algorithm later described in Section 3 are implemented in this module. It is invoked by the Template Model Designer and the Forest Modeler to generate tree geometric models for rendering. The engine also provides

Proceedings of the Third International Conference on Image and Graphics (ICIG'04)
0-7695-2244-0/04 $20.00 © 2004 **IEEE**

a function of generating an image of current geometry model at run-time that will be used when the tree model is rendered as a billboard.

Template Model Designer: The template model designer allows the users to interactively design a representative geometric model for each plant species involved in the applications. In this design tool, as parameters are changed, tree model can be previewed immediately. The parameters used to generate a template model and the resulting shape data such as tree height are saved as a template model file. At run-time, the template model file will be loaded by the forest modeler component for generating a specific individual tree.

Forest Modeler/Render: The forest modeler and the forest render are respectively used for constructing and rendering forest scenes at run-time. The details of involved algorithms will be described in next Section

2.2 Implementation

Our system links to GIS database using ESRI ArcObject that comprises an object-oriented geographic data model and an integrated library of software components. Forest inventory information, forest stand maps and DEMs were stored and managed in a relational database management system by ESRI ArcSDE, a spatial database engine. The implementation was written in C++, using OpenGL and OpenSceneGraph libraries [6]. So far we have only implemented the system in PC version.

3 Methods for forest visualization

3.1. Individual tree model

In this paper, the 3D geometric models of trees are composed of a main trunk, several levels of branches, and leaves. We use a set of parameters to depict the structure and shape characteristics, which have much in common with the method described in [11]. Enlightened by [7], we extend the work in [11] by introducing parameterized curves to depict morphogenetic gradients that describe the distributions of features along the axes. Experiences show that curve tool is more intuitive and flexible than mathematical equations used in [11].

The representation of a 3D tree model can be separated into two different parts: the stems, including main trunk and branches, and the leaves. In the system, each of these parts has been treated in a different manner. The surface of a stem could be considered as a generalized cylinder with a circular cross section of varying radius [2]. To generate triangle strips of the stem surface for final rendering, a finite number of cross sections are evaluated along the stem axis and connected together. Each cross section consists of a finite number of points. It is meaningless and impractical to model and render individual leaves in an interactive landscape visualization system. For greatly reducing the geometry detail, we use a textured quadrilateral to represent a cluster of leaves. This representation enable us greatly reduce polygons. In addition, screen-aligned billboard technique is used to orient the textured quadrilateral.

3.2 Using Template Model

This section describes how to adjust parameters in template model file for generating geometry models that with given size information of trees including DBH, tree height and crown width. In discussion below, a variable with subscript *stand* represents corresponding value in forest stand inventory data; a variable with subscript *template* represents corresponding parameter in template.

To generate a model with a given DBH, the following $scale_{radius}$ is used to scale the radius parameters of trunk and branches:

$$scale_{radius} = DBH_{stand} / DBH_{template}$$

Then we adjust the length of trunk and branches for generating a tree model with a given height and a given crown width. Different methods are used according to branching patterns.

For monopodial branching tree (where apical meristems are the primary creators of new stems and leaves), its height is linear with the length of trunk and independent with length of branches. Therefore, we scale the length of branches in parameter template with:

$$scale_{branch\_length} = crown\_width_{stand} / crown\_width_{template} \quad (1)$$

and scale the length of trunk with:

$$scale_{trunk\_length} = height_{stand} / height_{template}$$

For sympodial branching tree (where each apical meristem dies, and an axillary meristem creates the next stem and leaves), tree height is dependent with both trunk length and branches length. Firstly, we still use Eq. (1) to scale branches length for getting desired crown width. Then, we compute $height_{\text{new}}$, the height of the tree model generated with the adjusted parameters in template. At last, we replace the length of trunk in template by:

$$trunk\_length = trunk\_length_{template} + (height_{new} - height_{stand})$$

3.3. Modeling and rendering of forest scenes

The algorithm for transforming each forest stand into a list of tree models follows these steps:

1. Load forest inventory data of a forest stand including dominant plant species, average height, average DBH, and distribution density etc.

2. Load a pre-designed template model according to dominant plant species.
3. Adjust parameters in the loaded template using the method introduced in Section 3.2.
4. Invoke the procedural tree geometry engine to generate a tree geometric model by inputting the parameters computed in *step 3* and a random seed.
5. Repeat *step 4 N* times with different random seed to get *N* models with slight differences. The value of *N* determines the variation of trees in a stand.
6. Compute number of trees using the area of and trees density of current stand.
7. Distribute trees in current stand in random and then assign spatial coordinates to individual trees. The height of the base of tree objects is computed from DEM.
8. Assign each tree a geometric model random selected from the results of *step 5*.

We organize the forest scenes using a scene graph. Trees in the same forest stand are grouped as one group node. The scene graph is a bounding volume hierarchy that supports hierarchical culling techniques. At run-time, the scene graph is traversed recursively and render functions are called to process each node. We render tree models using different levels of detail (LODs) [12].

Branches and leaves are treated in different ways. For branches, the automatically computed LODs start with branches generated by the modeling engine and produce a user controlled number of discrete LODs where each new LOD is a reduced version of its predecessor. Reduction occurs by not only eliminating the least important branches but also reducing the cross-sectional resolution of those that remain. The criterion for evaluating the importance of branches is: the branches in a lower recursive level are more important than those in higher recursive levels; in the same recursive level, the longer branch is more important than the shorter one. The leaves number is reduced when a lower LOD leaf set is created from an existing set. Meanwhile, in coarser LOD, leaves sizes are increased to represent original tree without loss leafiness. In particular, for trees in far away, we can simply draw them as rotating billboards.

4 Applications and results

4.1 Visualization of real forest landscape

Visualization of real forest landscape is a direct application of our system. We ran our tests on a PC with a 2.4G Pentium 4 processor, 512M RAM and an NVIDIA GeForece4 display adaptor. The system allows user to select an area of interest on 2-D map and then jump into the site and walkthrough in the corresponding 3D scenes. Figure 1 shows a screenshot of moving through in a virtual forest. The frame rate is typically 25-35 Hz. Figure 2 is a screenshot of showing an example of querying forest stand information in virtual forest scenes.

Figure 1. Moving through the forest, rendered with typically 25-35Hz.

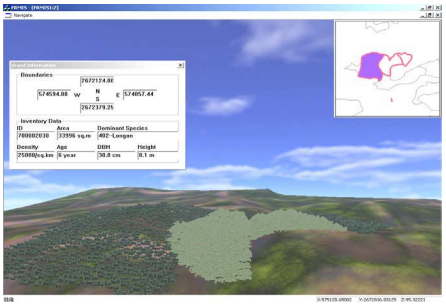

Figure 2: Screenshot of querying in virtual forest scenes. (The eagle eye window highlights boundaries of three interested stands in the map. The left dialogue shows forest inventory data of the stand currently queried, cyan area in eagle eye window.)

4.2 Simulation of forest fires

The major task in adding the feature of forest fire simulation into our system is to implement a fire growth model and a fire behavior model for predicting the spread and intensity of forest fires. We choose to implement Huygen's principle of wave propagation [1] as the fire growth model. The growth of a fire front is simulated as a 2D elliptical wave [8] using spatial data from GIS. The fire front is then projected over a finite time step using fire behavior at discrete points along the fire's edge. We compute fire behavior for those points using the Rothermel model [9] together with local raster information on fuels, weather, and topography. This provides a 1D rate and direction of fire spread for each point that produces a 2D fire growth when aggregated for all points around the fire perimeter.

A GIS-based forest fire simulation sub-system thus can be developed that provides the user with information like forest landscape features, and predicted position of a fire front. Figure 3 shows an example of forest fire simulation.

5. Conclusion and future work

We have presented a forest visual simulation system combining technologies of 3-D visualization and GIS. In this system, geometric models of individual trees are generated on the fly and faithful to underlying forest data stored in GIS. Thus it supports the accurate depiction of forest scenes. With the hybrid representation method for 3D tree models and specific LOD algorithm, the system can provide real-time frame rates to ensure user-steered interactive displays and query. In addition, we implemented a forest fire simulation module that is useful when constructing virtual environment for forest fire-fighting training and forest fire management.

Our work is but an early step in the development of techniques for automatically constructing and real-time rendering real forest scenes. We plan to introduce simulation of plant distribution [5] in our system for converting stand level inventory data into individual plant level information. Furthermore, rendering other important features besides trees such as rivers, lakes and roads will make virtual forest scenes more realistic. Finally, we are interested in developing more applications for forest management based on our visual simulation system.

Acknowledgements

This project is sponsored by 973 Project (No. 2002CB312100) and Preliminary Project of 973 Program (No.2002CCC01900) in China.

References

[1] D.G. Anderson, E.A. Catchpole, N.J. DeMestre, and T. Parkes, "Modeling the Spread of Grass Fires", *J. Austral. Math. Soc. (Ser. B.),* 451-466, 1982.

[2] J. Bloomenthal, "Modeling the Mighty Maple", *Proceedings of SIGGRAPH '85*, pp. 305-311, 1985.

[3] P. Decaudin and F. Neyret, "Rendering Forest Scenes in Real-Time", In *Proceedings of Rendering Techniques '04* (Eurographics Symposium on Rendering), 93-102, 2004.

[4] E. Karjalainen and L. Tyrvainen, "Visualization in Forest Landscape Preference Research: a Finnish perspective", *Landscape and Urban Planning*, 59, 13-28, 2002.

[5] B. Lane and P. Prusinkiewicz, "Generating Spatial Distribution for Multilevel Models of Plant Communities", In *Proceedings of Graphics Interface'02*, volume 1, 69-80, 2002.

[6] OpenSceneGraph, http://www.openscenegraph.org

[7] P. Prusinkiewicz, L. Mundermann, R. Karwowski, and B. Lane. "The Use of Positional Information in the Modeling of Plants", In *Proceedings of SIGGRAPH 2001*, ACM Press / ACM SIGGRAPH, E. Fiume, Ed., Computer Graphics Proceedings, Annual Conference Series, ACM, 15–22, 2001

[8] G.D. Richards, "An Elliptical Growth Model of Forest fire Fronts and Its Numerical Solution", *Int. J. Numer. Meth. Eng*., 30:1163-1179, 1990.

[9] R.C. Rothermel, "A Mathematical Model for Predicting Fire Spread in Wildland Fuels", *USDA For. Serv. Res. Pap*. INT-1 15, 1972.

[10] J. Uusitalo and B. Orland, "Virtual Forest Management: Possiblities and Challenges", *International Journal of Forest Engineering,* Vol. 12 No. 2, July 2001.

[11] J. Weber and J. Penn, "Creation and Rendering of Realistic Tree", In *SIGGRAPH 95 Conference Proceedings,* pages 119-128, 1995.

[12] Q. Yu, C. Chen, Z. Pan and X. Chi, "Interactive 3-D Visualization of Forests", *Proceedings of 13th International Conference on Artificial Reality and Telexistence*, pp. 263-269, 2003.

a) After 20 mins **b) After 1 hour** **c) After 2 hours**

Figure 3. Simulation of forest fires.