MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

Dynamic Meshing Using Adaptively Sampled Distance Fields

Jackson Pope, Sarah F. Frisken, Ronald N. Perry

TR2001-13   December 2001

Abstract

Many models used in real-time graphics applications are generated automatically using techniques such as laser-range scanning. The resultant meshes typically contain one or more orders of magnitude more polygons than can be displayed by todayś graphics hardware. Numerous methods have been proposed for automatically creating level-of-detail (LOD) meshes from large input meshes. These techniques typically generate either one or more static LOD meshes, precomputed before use in the application, or a dynamic mesh, where the LOD of the mesh adapts to frame rate requirements. We present a new dynamic LOD technique ideal for applications such as games and physical simulations based upon Adaptively Sampled Distance Fields (ADFs).

SIGGRAPH 2001 Conference Abstracts and Applications

Dynamic Meshing Using Adaptively Sampled Distance Fields

Jackson Pope, Sarah F. Frisken and Ronald N. Perry, MERL

Introduction

Many models used in real-time graphics applications are generated automatically using techniques such as laser-range scanning. The resultant meshes typically contain one or more orders of magnitude more polygons than can be displayed by today's graphics hardware. Numerous methods have been proposed for automatically creating level-of-detail (LOD) meshes from large input meshes [2]. These techniques typically generate either one or more *static* LOD meshes, pre-computed before use in the application, or a *dynamic* mesh, where the LOD of the mesh adapts to frame rate requirements. We present a new dynamic LOD technique ideal for applications such as games and physical simulations based upon Adaptively Sampled Distance Fields (ADFs) [1]; ADFs also provide fast collision detection as required by these applications.

Previous Work

Existing dynamic meshing algorithms such as View Dependent Progressive Meshes (VDPM) [3] and Hierarchical Dynamic Simplification (HDS) [4] generate a hierarchy to efficiently process refinement and decimation operations. The hierarchy in VDPM is formed by creating a new parent vertex for every pair of vertices combined by an edge collapse operation. The HDS hierarchy is formed by spatially subdividing the scene into cells and grouping vertices in each cell into a single representative vertex. In both, the screen space error and normal cones (to detect back-facing and silhouette triangles) are used to determine when to refine and decimate the mesh. We present a new method that utilizes a spatial subdivision hierarchy similar to [4], enables fast collision detection, and uses the distance field to position mesh vertices to optimize mesh shape.

Generating Meshes from ADFs

ADFs are a new shape representation which adaptively sample the signed distance field of an object and store the distance values in a spatial hierarchy (we use an octree) [1]. We utilize a fast, new triangulation method that generates topologically consistent (orientable and closed) triangle meshes from the ADF structure [5]. Cells in the ADF octree which contain the object surface (where the distance field changes sign) are connected to their neighbors by triangles. The technique exploits the hierarchical nature of the octree to produce detail-directed triangles.

Algorithm

Our method creates a triangle mesh from the ADF, associating triangles with ADF cells, and then adapts the mesh in real-time to viewing parameters in such a way to optimize visual quality (by using a high level of detail in visually important regions), while meeting user defined frame rate criteria.

The algorithm is composed of two stages: a pre-processing stage and a real-time stage. The real-time stage is performed every frame or every few frames as required. The pre-processing stage initializes the data required for the real-time stage and creates an initial view-independent active cell list from which a triangle mesh is derived. Each active cell is associated with one ADF cell. Data initialization includes determining and storing normal cones in each boundary ADF cell; these cones bound the normal cones of all the cell's children. The hierarchical ADF structure enables fast view frustum and back-face culling using normal cones.

The real-time stage consists of adapting and optimizing the existing active cell list and corresponding triangle mesh for the current viewing conditions. During each adaptation, the active cells are considered to see if they contribute too many or too few triangles to the mesh according to view-dependent cell weights. If the number of triangles is appropriate, the cell is left alone. If the cell contributes too many triangles, triangles associated with the cell and its siblings are deleted from the mesh, the cell's parent is added to the active cell list, and triangles associated with the cell's parent are generated and added to the mesh. If the cell contributes too few triangles, the cell is added to an ordered list of such cells. To ensure that frame rate requirements are met, this cell list is processed in order only while there is frame time available. When processed, triangles associated with cells in the ordered list are deleted from the mesh, the cell's boundary child cells are added to the active cell list, and triangles associated with the cell's boundary child cells are generated and added to the mesh. The differential treatment of cells with too many and too few triangles avoids the mesh growing in size beyond the rendering capabilities of the graphics hardware.

Each cell is assigned a weight based upon its contribution to the view. Currently a cell is assigned a high weight if it is on the object's silhouette, and zero weight if the cell is back-facing or outside the view frustum. Other parameters could be considered such as the projected screen size of the cell or whether the cell contains a specular highlight. In addition, our method uses the in-place cell error of the ADF as an indicator of surface roughness/curvature in the cell, and modulates the weight by this error.

Results

The technique produces detail-directed triangle meshes of high visual quality as viewed from the camera, while minimizing the number of triangles in non-visible portions of the object. It meets frame rate criteria (currently at 30 FPS it maintains ~25K triangles), even during viewpoint changes that lead to large differences in the visible portion of the object.

Summary

A new method allowing the generation of viewpoint-dependent dynamic triangle meshes using ADFs has been presented. These meshes are of high visual quality, while maintaining a low triangle count in invisible areas.

References

[1] Frisken, S. F., Perry, R. N., Rockwood, A. P. and Jones, T. R., Adaptively Sampled Distance Fields: A General Representation of Shape for Computer Graphics, in Proceedings of SIGGRAPH 2000, pp. 249-254, 2000.
[2] Garland, M., Multiresolution Modeling: Survey and Future Opportunities, in Eurographics '99 State of the Art Reports, pp. 111-131, 1999.
[3] Hoppe, H., View-Dependent Refinement of Progressive Meshes, in Proceedings of SIGGRAPH 1997, pp. 189-198, 1997.
[4] Luebke, D. and Erikson, C., View-Dependent Simplification of Arbitrary Polygonal Environments, in Proceedings of SIGGRAPH 1997, pp. 199-208, 1997.
[5] Perry, R. N. and Frisken, S. F., Kizamu: A system for sculpting digital characters, in Proceedings of SIGGRAPH 2001.

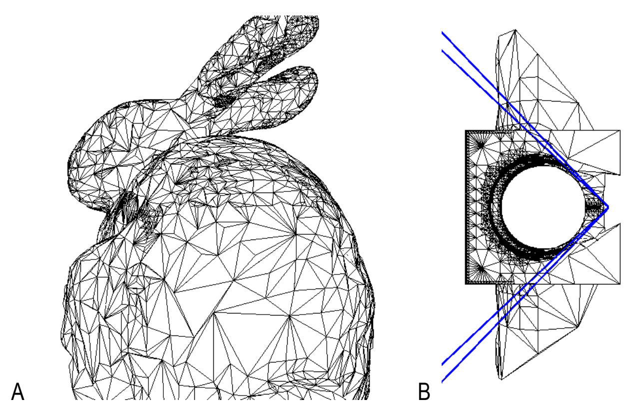

Figure 1. A) Bunny model from camera point (16984 triangles, 47 FPS), note the silhouette quality. B) CSG object showing view frustum (20364 triangles, 41 FPS), note how the areas outside the view frustum are culled.

Appendix A: System Diagrams for Dynamic Meshing

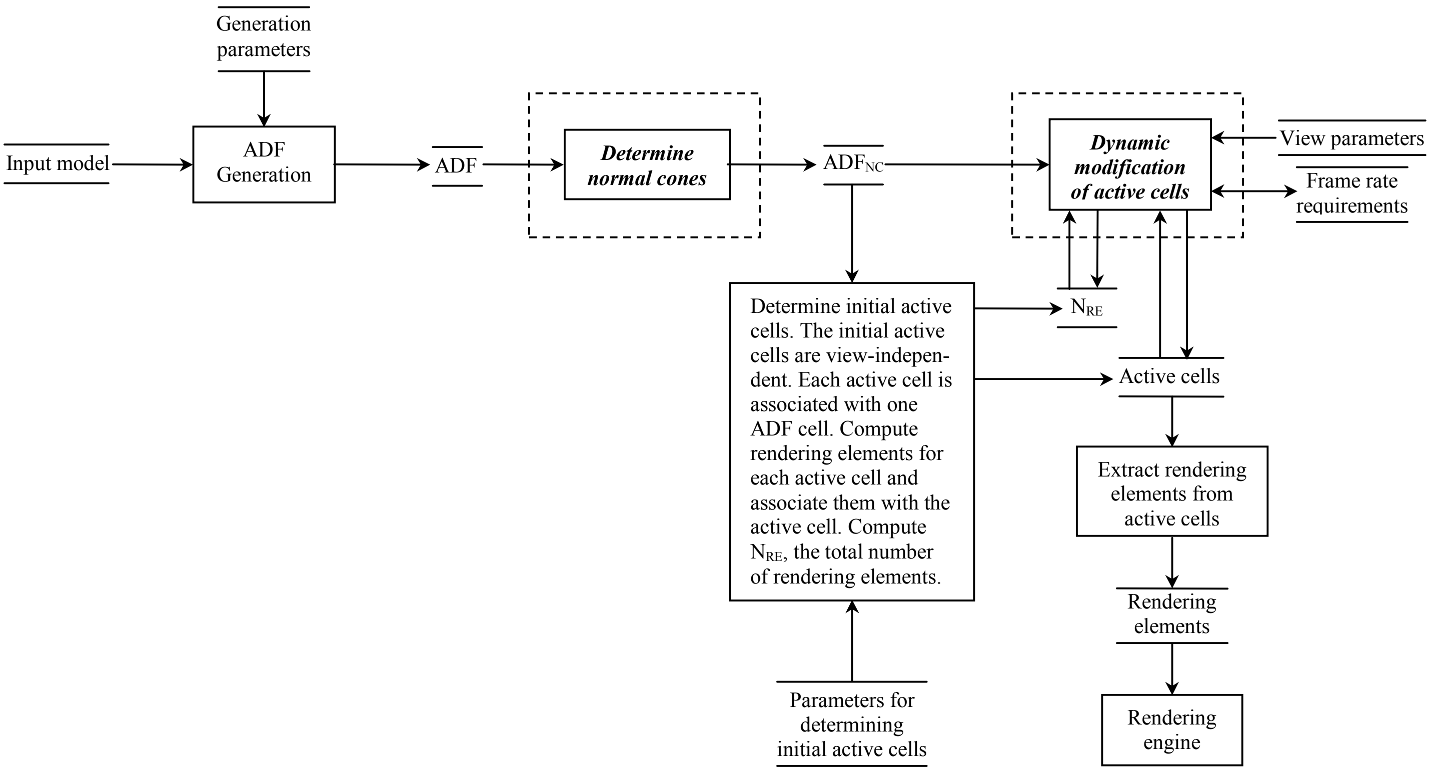

Figure A1. System diagram for dynamic meshing using ADFs

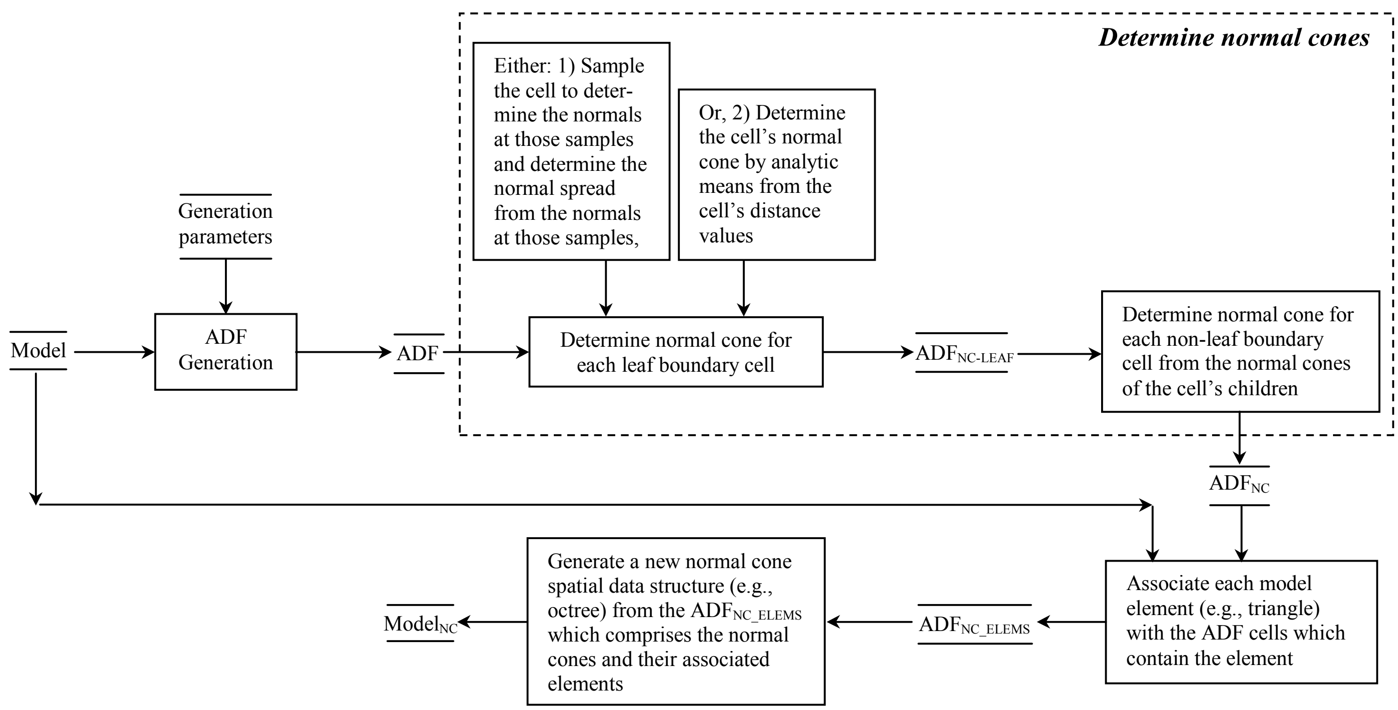

Figure A2. System diagram for building a detail-directed normal cone hierarchy using ADFs

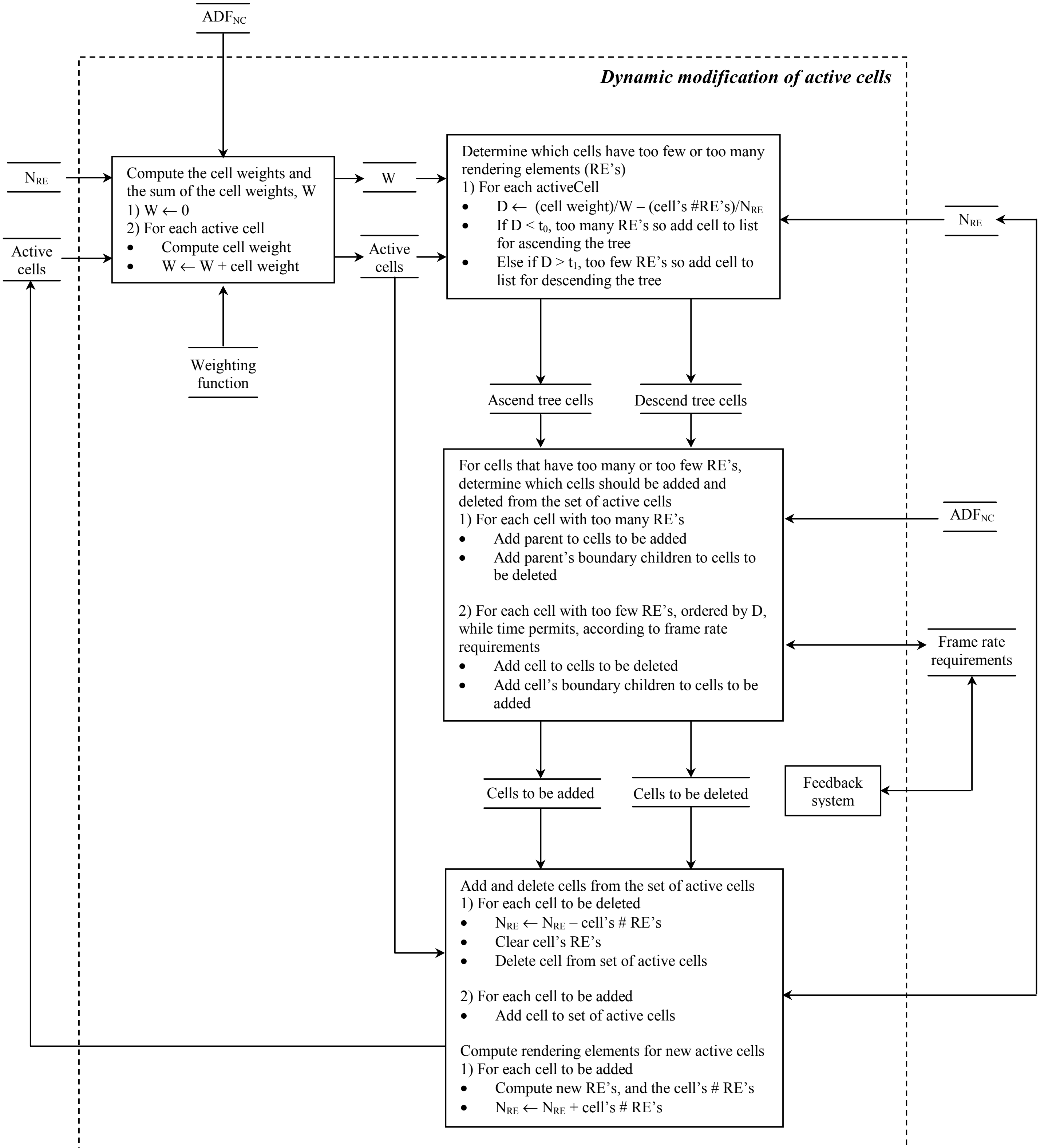

Figure A3. System diagram for dynamic modification of active cells during dynamic meshing