

Interactive Walkthrough of Large Geometric Databases

Course Notes for SIGGRAPH '96

Course Organizer

Eric L. Brechner
Microsoft Corporation

Course Speakers

Brian Cabral
Silicon Graphics Inc.

Ned Greene
Apple Computer

Jarek Rossignac
IBM T.J. Watson Research Center

Thomas A. Funkhouser
AT&T Bell Laboratories

Abstract

This course will focus on techniques, algorithms, data-structures, and databases for displaying very large geometric databases interactively (greater than one million polygons drawn at least ten frames per second). Beginning with a discussion of basic techniques and leading to state-of-the-art algorithms, the speakers will address key issues in walkthrough, including visibility computations, automatic object simplification, and memory management through database subset pre-fetching. Speakers will show real applications of these algorithms to a variety of areas, including visual simulation, virtual reality, architecture, and digital mockup.

Attendees of this course will better understand the key issues in dealing with very large geometric databases. They will be provided basic and state-of-the-art techniques to overcome hardware and software limitations that preclude the high frames rates necessary for interactive inspection of complex geometric scenes. These techniques will be illustrated by real examples of working walkthrough applications using databases with between 1 million and 500 million polygons.

A working knowledge of interactive computer graphics, including the mechanisms of matrix transformation, perspective, and raster graphics, will form a good foundation for the course material.

Course Schedule

A. Introduction - 10 minutes

Statement of the problem. Introduction of the speakers. Agenda. - Eric Brechner

B. Graphics Techniques for Walkthrough Applications - 75 minutes

An introduction to techniques common to most walkthrough applications with examples from the IRIS Performer toolkit. The methods include multi-tasking, view frustum culling, occlusion culling, level-of-detail, frame rate control, database paging, dynamic primitive tessellation, and texture replacement. - Brian Cabral

- **Morning Break**

C. Hierarchical Visibility and Tiling - 75 minutes

Hierarchical approaches to accelerating visibility computations in extremely complex scenes. - Ned Greene

- **Lunch**

D. Geometric Simplification - 75 minutes

The generation and exploitation of multi-resolution graphic models for the interactive visualization of complex mechanical or architectural 3D scenes. - Jarek Rossignac

- **Afternoon Break**

E. Database Management - 75 minutes

Algorithms for computing and pre-fetching a small subset of a disk-resident database to store in memory during an interactive walkthrough. - Thomas Funkhouser

- **Seventh-inning Stretch**

F. Wrap-up and Future Directions - 20 minutes

Hierarchical level of detail, culling, and data staging for interactive walkthrough of infinite databases. - Eric Brechner

Contents



Abstract	iii
Course Schedule	v
About the speakers	xix
A Introduction	A-1
1 Slides	A-3
2 Introduction	A-5
2.1 Statement of the problem	A-5
2.2 Overview of the notes	A-6
B Graphics Techniques	B-1
3 Slides	B-3
4 Graphics Techniques	B-13
4.1 Introduction	B-13
4.2 Motivation of Performance Requirements	B-14
4.2.1 Visual Acuity	B-14
4.2.2 Field of View	B-15
4.2.3 Latency	B-15
4.2.4 High Frame Rate	B-15
4.2.5 Constant Frame Rate	B-17
4.3 Performance in Graphics Pipelines	B-17
4.4 Reducing Geometric Complexity	B-18
4.4.1 Texture Mapping and plenoptic rendering	B-19
4.4.2 Billboards	B-19
4.4.3 Volumetric techniques	B-20
4.4.4 Geometric Level of Detail (LOD)	B-20
4.4.5 Combining imaged based LODS with geometric LODS	B-21

4.5	Optimizing Runtime Rendering	B-21
4.5.1	Visibility and occlusion Culling	B-21
4.5.2	Level of Detail Switching	B-22
4.5.3	Maintaining Constant Frame Rate	B-23
4.5.4	Parallel Processing	B-24
4.5.5	Mode Sorting	B-25
4.5.6	Latency Control	B-25
4.5.7	Scene graph management	B-26
4.6	Optimizing the Graphical Database	B-27
4.6.1	Triangle Meshes	B-27
4.6.2	Packed vertex arrays and Display lists	B-27
4.6.3	Spatial Organization	B-27
4.6.4	Multiple concurrent organizations	B-28
4.6.5	Graphics Attributes	B-28
4.6.6	Eliminating Transformations	B-28
4.6.7	Precomputed Animation	B-28
4.7	Database Examples	B-28
4.7.1	Architectural Exteriors	B-28
4.7.2	Architectural Interiors	B-29
4.8	Handling Very Large Databases	B-29
4.8.1	Paging Geometry and Texture from Disk	B-31
4.8.2	Paging Texture and Geometry	B-31
4.9	Beyond the Rendering	B-31
4.9.1	Interface with M/CAD Systems	B-32
4.9.2	Collision Detection	B-32
4.9.3	Networking	B-32
4.10	Conclusions	B-33
 C Hierarchical Visibility and Tiling		C-1
5	Introduction	C-3
6	Hierarchical Z-Buffer Visibility	C-5
6.1	Overview	C-5
6.2	Introduction	C-5
6.3	The Hierarchical Z-Buffer Visibility Algorithm	C-7
6.3.1	The Object-Space Octree	C-7
6.3.2	The Image-Space Z-Pyramid	C-11
6.3.3	The Temporal Coherence Scheme	C-15
6.4	Building and Maintaining The Octree	C-16
6.4.1	Building an Octree from Unorganized Primitives	C-16
6.5	Implementation and Results	C-18
6.5.1	Parallel Performance	C-23
6.5.2	Use of Graphics Hardware	C-24
6.6	Conclusion	C-25

7 Hierarchical Polygon Tiling	C-27
7.1 Overview	C-27
7.2 Introduction	C-28
7.3 Previous Work	C-30
7.3.1 Warnock Subdivision	C-30
7.3.2 Coverage Masks	C-30
7.4 Triage Coverage Masks	C-31
7.4.1 Triage Edge Masks	C-32
7.4.2 Compositing Triage Masks	C-32
7.4.3 Building Lookup Tables	C-35
7.5 The Hierarchical Tiling Algorithm	C-37
7.5.1 Data Structures	C-37
7.5.2 The Basic Hierarchical Tiling Algorithm	C-39
7.6 Hierarchical Object-Space Culling	C-42
7.7 Implementation and Results	C-44
7.8 Conclusion	C-46
D Geometric Simplification	D-1
8 Slides	D-3
9 Geometric Simplification	D-13
9.1 Introduction	D-14
9.2 Rendering cost	D-15
9.3 Review of graphics acceleration techniques	D-16
9.3.1 Meshing or storing transformed vertices	D-16
9.3.2 Frustum culling	D-16
9.3.3 Smart caching and pre-fetching	D-17
9.3.4 Visibility	D-17
9.3.5 Polygon count reducing techniques	D-17
9.3.6 Vertex clustering techniques	D-18
9.4 Geometric clustering of vertices	D-18
9.4.1 Overview	D-19
9.4.2 Grading	D-19
9.4.3 Triangulation	D-19
9.4.4 Clustering	D-19
9.4.5 Synthesis	D-20
9.4.6 Elimination	D-20
9.4.7 Adjustment of normals	D-20
9.4.8 An efficient implementation	D-20
9.4.9 Exploitation	D-21
9.4.10 Advantages	D-21
9.4.11 Experiments	D-22
9.5 Superfaces	D-22
9.5.1 Overview	D-22

9.5.2	Advantages	D-23
9.6	Edge pinching algorithms	D-23
9.6.1	Ronfard and Rossignac	D-24
9.6.2	Guezic	D-24
9.7	Conclusion	D-25
10	Figures	D-31
10.1	Multi-resolution 3D approximations	D-31
10.2	Superfaces	D-34
10.3	Full-range approximation of polyhedra	D-36
10.4	Surface Simplification inside a tolerance volume	D-37
E	Database Management	E-1
11	Slides	E-3
12	Database Management	E-15
12.1	Introduction	E-16
12.1.1	Previous Work	E-17
12.1.2	Goals	E-19
12.1.3	Problem Statement	E-23
12.1.4	System Organization	E-24
12.2	Modeling	E-24
12.2.1	Model Loading	E-24
12.2.2	Model Representation	E-26
12.2.3	Object Abstraction	E-27
12.2.4	Results	E-28
12.3	Precomputation	E-29
12.3.1	Spatial Subdivision	E-29
12.3.2	Visibility Precomputation	E-30
12.3.3	Results	E-34
12.4	Interactive Walkthrough	E-34
12.4.1	User Interface	E-36
12.4.2	Display Management	E-37
12.4.3	Memory Management	E-43
12.5	Results	E-48
12.5.1	Display Management	E-48
12.5.2	Memory Management	E-52
12.6	Conclusion	E-54
12.7	Acknowledgements	E-55
F	Wrap-up and Future Directions	F-1
13	Slides	F-3

- 14 Wrap-up and Future Directions** **F-7**
- 14.1 Solved and unsolved problems F-7
- 14.2 The infinite database problem F-7
 - 14.2.1 Four key concepts F-7
 - 14.2.2 Choosing the right approximation F-8
 - 14.2.3 A possible approximation choice F-9

List of Figures

4.1	Sources of latency in a hypothetical virtual environment system. . . .	B-16
4.2	Multiple image artifact when fixating on moving object.	B-17
4.3	View into a simple town using texture mapping and billboards.	B-19
4.4	Transition from a higher level-of-detail model to a lower one using fade (top) and morphing (bottom).	B-22
4.5	Graphics Load Management.	B-23
4.6	Pipelined processing model for multiprocessor graphics workstations.	B-25
4.7	Trading off throughput for latency in pipeline multiprocessor.	B-26
4.8	Visualization of a building used for city zoning discussions.	B-29
4.9	Building interior using radiosity solution, no texture mapping.	B-30
4.10	Architectural scene using radiosity solution and texture mapping.	B-30
6.1	If a cube is hidden, then all geometry it contains is also hidden.	C-8
6.2	Ordering of octants by visibility priority.	C-10
6.3	A primitive inside a visible cube is “nearly visible.”	C-11
6.4	A scene and its corresponding z-pyramid.	C-13
6.5	Does a z-pyramid hide a primitive?	C-14
6.6	An office environment rendered with hierarchical visibility.	C-20
6.7	Top view of model space showing viewing frustum and octree subdivision.	C-21
6.8	Terrain models rendered with hierarchical visibility.	C-22
6.9	Parallel performance graph.	C-23
7.1	Triage coverage masks classify subcells as inside, outside, or intersecting an edge.	C-31
7.2	Constructing a triage coverage mask for a convex polygon from the triage masks of its edges.	C-33
7.3	Compositing triage coverage masks.	C-34
7.4	With triage coverage masks, classification of subcells as <i>covered</i> or <i>vacant</i> is definitive.	C-36
7.5	Schematic diagram of a mask pyramid.	C-38
7.6	PSEUDOCODE LISTING	C-41
7.7	Antialiased frame rendered with the hierarchical polygon tiling algorithm.	C-45
10.1	Simplified Chess Set	D-32
10.2	A perspective view of the same chess pieces.	D-32

10.3	Simplified chair	D-33
10.4	Original skull model (349,792 triangles)	D-34
10.5	Simplified skull (36.6% of original)	D-35
10.6	Simplified skull (2.55% of original)	D-35
10.7	Simplified skull model	D-36
10.8	Simplified arterial network	D-37
10.9	Simplified human femur	D-37
12.1	Exterior view of Soda Hall.	E-20
12.2	Exterior view of Soda Hall “cut open” by a horizontal plane at the sixth floor.	E-20
12.3	Typical office with furniture.	E-20
12.4	Board room with furniture.	E-20
12.5	Radiosity rendering of hallway.	E-22
12.6	Radiosity rendering of office.	E-22
12.7	Radiosity computation and results are independent of observer viewpoint.	E-22
12.8	Polygonal mesh generated for the office during radiosity computation.	E-22
12.9	System overview.	E-24
12.10	Loading operations of the modeling phase.	E-25
12.11	Three LODs for a chair.	E-26
12.12	Object instances can share geometries stored in an object definition.	E-27
12.13	An object instance can store its own geometries.	E-27
12.14	Different door handle representations constructed using a parameterized generator.	E-27
12.15	Three LODs for a canary.	E-28
12.16	Functional steps of the precomputation phase.	E-30
12.17	Spatial subdivision of the sixth floor of Soda Hall. The image on the left shows the actual three dimensional subdivision, while the image on the right shows a two dimensional schematic representation.	E-31
12.18	A sight-line stabbing a portal sequence.	E-31
12.19	Bowtie-shaped region containing sightlines stabbing a portal sequence (stipple gray). Opaque boundaries are shown in solid black. Extremal sightlines are shown as dashed lines.	E-32
12.20	Cell visibility (stipple gray) for source cell (dark gray).	E-32
12.21	Two dimensional schematic diagram of a) cell-to-cell visibility (stipple gray), and b) cell-to-object visibility (shown as solid black squares) for a particular source cell (dark gray).	E-33
12.22	Cell visibility in three dimensions. Visible region (beams) and cell-to-cell visibility (outlined) are shown for one source cell.	E-33
12.23	Organization of data in the display database.	E-35
12.24	Functional operations of the walkthrough phase.	E-35
12.25	View frustum variables.	E-36
12.26	Mapping view frustum to perspective projection.	E-36
12.27	Visualization program supports two views.	E-37
12.28	Panels used for controlling parameters for a) observer navigation, b) visibility determination, and c) memory management.	E-38

12.29	Visible region is a wedge that typically narrows as it traverses through more portals.	E-39
12.30	Visible region for view frustum in two dimensions.	E-40
12.31	Objects in the eye-to-object visibility are shown as by solid squares.	E-40
12.32	All objects incident upon cells in the eye-to-cell visibility.	E-41
12.33	Objects in the eye-to-object visibility.	E-41
12.34	The observer range cells (cross-hatch) contain all observer view positions possible during the upcoming $N = 4$ frames. Each cell is labeled by the number of frames before the observer can be resident in it.	E-44
12.35	The lookahead cells (stipple gray) contain all objects that can be visible to the observer during the upcoming N frames. Each cell is labeled by the number of frames before it can become visible to the observer.	E-45
12.36	Lookahead objects are stored in memory only up to the LOD at which they can possibly be rendered during the next N frames. Each cell is labeled and shaded by the maximum level of detail any object incident upon it is stored in memory – darker shades of gray represent higher levels of detail.	E-46
12.37	Cache management algorithm results. Each cell is labeled by the number of frames since objects incident upon it were included in the lookahead set. Shading for each cell indicates whether or not it contains objects in the memory resident cache (stipple gray), read set (left-hatch), and resident set (right-hatch).	E-47
12.38	Test observer path through the top four floors of Soda Hall.	E-49
12.39	Frame time for each observer viewpoint along the entire test observer path during the test using both visibility determination and detail elision.	E-50
12.40	Images of library generated using (left) no detail elision (19,821 polygons), and (right) using the <i>Optimization</i> detail elision algorithm with a target frame time of 0.05 seconds (3,568 polygons).	E-51
12.41	Rendering times during <i>Detail Elision</i> test for each observer viewpoint along the test observer path.	E-52
12.42	Compute time and rendering time during <i>Detail Elision</i> test for each observer viewpoint along the portion of test observer path where computation was most expensive.	E-52
12.43	Frame time for each observer viewpoint along the entire test observer path during the test with memory management.	E-53
14.1	Approximating a cluster of polygons.	F-10

List of Tables

7.1	Rules for compositing subcells of triage masks.	C-33
12.1	Multi-resolution modeling statistics for Soda Hall.	E-29
12.2	Mean and maximum statistics for cells in the spatial subdivision of Soda Hall.	E-34
12.3	Mean and maximum compute time, rendering time, and frame time statistics collected during display management tests.	E-50
12.4	Mean and maximum numbers of cells, objects, and polygons rendered during display management tests.	E-50
12.5	Mean and maximum compute time, rendering time, swap time, and frame time statistics collected during tests with and without memory management.	E-53
12.6	Mean and maximum set statistics collected during memory management tests.	E-53

About the speakers

Eric L. Brechner

Software Design Engineer
Microsoft Corporation
One Microsoft Way, Redmond, WA, 98052-6399
ericbrec@microsoft.com

Eric Brechner is a member of the Media Foundation group at Microsoft. He was formally a Senior Principal Scientist at The Boeing Company where he worked in the areas of large scale visualization, computational geometry, network communications, data-flow languages, and software integration. He was the principal architect of FlyThru™, the walkthrough program for the 20GB, 500+ million polygon model of the Boeing 777 aircraft, authoring three invention disclosures relating to that work. Before coming to Boeing, Eric had worked in computer graphics and CAD for Silicon Graphics Inc., GRAFTEK, the Rensselaer Design Research Center (formally the Center for Interactive Computer Graphics), and the Jet Propulsion Laboratory. He holds a BS and MS in mathematics, and a PhD in applied mathematics from Rensselaer Polytechnic Institute.

Brian Cabral

Member Technical Staff
Silicon Graphics
2011 N. Shoreline Blvd
cabral@sgi.com

Brian Cabral is a member of the Core Rendering technical staff within the Advanced Graphics Division at Silicon Graphics Computer Systems. His current technical focus is on Perfomer™ based large geometric database walkthrough software and tools. Previously, Brian, developed algorithms and techniques in the areas of Medical visualization and image processing, scientific visualization, and physical based shading and lighting. Prior to working at SGI, Brian, was a lead engineer and researcher at Lawrence Livermore National Laboratory where he worked a variety of visualization algorithms, tools and systems. He received a BS in computer science from California State University Stanislaus and a MS in computer science from University of California, Davis. His interests include differential geometry, image processing, shading techniques, computational geometry and signal processing. When he's not toiling over a large equation or

chunk of code Brian enjoys the good life of California.

Ned Greene

Apple Computer
MS301-3J, 1 Infinite Loop, Cupertino, CA 95014
greene@apple.com

Ned Greene is a member of the Advanced Technology Group at Apple Computer in Cupertino, California, where he conducts research in computer graphics. He holds a PhD in computer science from the University of California at Santa Cruz. From 1980 to 1989 Ned worked at the Computer Graphics Lab at The New York Institute of Technology where he developed software for computer animation and contributed to pioneering animation projects such as *The Works* and *The Magic Egg*. Over the years he has been a frequent contributor to the SIGGRAPH technical program and Electronic Theatre.

Jarek Rossignac

Senior Manager
IBM T.J. Watson Research
PO Box 704, Yorktown Heights, NY 10598
*jarek@watson.ibm.com*¹

Jarek Rossignac is Senior Manager of the 3D Graphics and Interactions Department at IBM Research, covering a wide range of activities in CAD, graphics, and virtual reality. He joined IBM Research in 85 with a PhD in EE on Solid Modeling from the University of Rochester. In 89, he created the Interactive Geometric Modeling group and managed research projects in topological representations and algorithms for CAD and in the interactive design and inspection of complex animated assemblies. He published over 40 papers winning three Best Paper external awards (Computer and Graphics'90, Eurographics'91, and IEEE Computer Graphics and Applications'93). He authored 12 invention disclosures in modeling and graphics and received numerous IBM internal awards (including the Best Paper award from Computer Science and a Research Division Award for the 3D Interaction Accelerator software system, which supports graphics and VR interactions with 3D CAD models of industrial complexity, i.e.: comprising millions of faces). Rossignac lectures at SIGGRAPH and at Eurographics. He is Associate Editor of four professional journals (ACM Transactions on Graphics, Computer-Aided Design, The Visual Computer, and Computer Graphics Forum) and Guest Editor for 7 special issues (three in Computer-Aided Design, one in the International Journal of Computational Geometry and Applications, two in the IEEE Computer Graphics and Applications, and one in the ACM transactions on Graphics). He refereed several hundred manuscripts and numerous grant proposals, serves in various international conference committees (including Eurographics 91-93-94 and SIGGRAPH 93-94). Rossignac is also co-chairman of the ACM/SIGGRAPH Sym-

¹Jarek Rossignac is now Director of the Graphics Visualization and Usability Center, Georgia Institute of Technology, College of Computing, Atlanta, GA 30332-0280, *rossignac@cc.gatech.edu*

posium on Solid Modeling for 91-95, of the Eurographics'94 Workshop on Graphics Hardware. He is co-chairman of the International Program Committees for Eurographics'96 and for CAD/Graphics'95. Finally, He serves as Program Director for SIAM's Activity Group on Geometric Design.

Thomas A. Funkhouser

MTS

AT&T Bell Laboratories

600 Mountain Avenue, 2A-202, Murray Hill, NJ 07974

funk@research.att.com

Thomas Funkhouser is a member of the technical staff at AT&T Bell Laboratories. His research interests include multi-user systems, global illumination, and algorithms for managing large amounts of three-dimensional data in interactive computer graphics and visualization systems. He is a principal developer of the UC Berkeley Architectural Walkthrough System which is able to maintain thirty frames per second during interactive visualization of a building model containing 1.5 million polygons. He received a B.S. in biological sciences from Stanford University in 1983, a M.S. in computer science from UCLA in 1989, and a PhD in computer science from UC Berkeley in 1993.