Lecture Notes in Computer Science

Edited by G. Goos and J. Hartmanis

337

Oliver Günther

Efficient Structures for Geometric Data Management

Springer-Verlag
Berlin Heidelberg New York London Paris Tokyo

Editorial Board

D. Barstow W. Brauer P. Brinch Hansen D. Gries D. Luckham
C. Moler A. Pnueli G. Seegmüller J. Stoer N. Wirth

Author

Oliver Günther
International Computer Science Institute
1947 Center Street, Suite 600
Berkeley, California 94704, USA

CR Subject Classification (1987): E.1, F.2.2, G.1.2, H.2.2, H.2.8, I.2.10, I.3.5, J.6

ISBN 3-540-50463-X Springer-Verlag Berlin Heidelberg New York
ISBN 0-387-50463-X Springer-Verlag New York Berlin Heidelberg

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in other ways, and storage in data banks. Duplication of this publication or parts thereof is only permitted under the provisions of the German Copyright Law of September 9, 1965, in its version of June 24, 1985, and a copyright fee must always be paid. Violations fall under the prosecution act of the German Copyright Law.

© Springer-Verlag Berlin Heidelberg 1988
Printed in Germany

Printing and binding: Druckhaus Beltz, Hemsbach/Bergstr.
2145/3140-543210

Meinen Eltern

Preface

This book is the revised and extended version of a Ph.D. dissertation submitted to the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley. Many of the ideas presented in this book have their roots in discussions with Eugene Wong, my mentor and thesis advisor. I would like to thank Gene for being most supportive throughout the ups and downs of my years in graduate school. Our cooperation could not have been better.

Thanks to Mike Stonebraker for his advice and helpful criticism. Mike helped me get started at this university, and it was he who convinced me to stay in Berkeley beyond my Master's degree. Thanks also to the members of my thesis committee, Carlo Sequin and Dorit Hochbaum, for reading the various versions of my thesis and for providing me with several suggestions for improvement.

My years at Berkeley would not have been nearly as great without the loyal support of my best friends here and abroad: Grace Fan, Markus Flik, Michael Mast, Pietro Perona, Hans, Helga & Stefan Sprung, Magan Vikramsingh, Arnold Wassner and Andreas Weigend. I would also like to thank my colleagues in the INGRES group, at the International Computer Science Institute, and elsewhere: Jeff Bilmes, Peter Deussen, Joachim Diederich, Marc Fanty, Jerry Feldman, Eric Hanson, Yannis Ioannidis, Ron Kay, Werner Kiessling, Wolfgang Klas, Kurt Mehlhorn, Hanan Samet, Timos Sellis, Wolfgang Wahlster and many others who always had time for discussions and who gave me valuable suggestions, not only concerning my scientific work.

I would like to acknowledge the German National Scholarship Foundation, which first encouraged me to go abroad and supported me throughout the first three years at Berkeley with two scholarships. The National Science Foundation and the Army Research Office provided funding under grant numbers DMC-8300463 and ARO-DAAG 29-85-K-0223. The IBM corporation supported me during the final phase of my doctoral studies through their IBM Graduate Fellowship program, and the International Computer Science Institute provided me with the opportunity of spending eight more months at Berkeley as a postdoctoral fellow.

Finally, I would like to thank my parents Helmut und Helga Günther. Their support made this book possible; they were always there when I needed them. This book is dedicated to them.

And thanks to Carolyn, for sharing the best of times.

Berkeley, July 1988 Oliver Günther

Abstract

The efficient management of geometric data, such as points, curves, or polyhedra in arbitrary dimensions, is of great importance in many complex database applications like computer-aided design and manufacturing, robotics, or computer vision. To provide optimal support for geometric operators, it is crucial to choose efficient data representation schemes. In this monograph, we first give a taxonomy of operators and representation schemes for geometric data and conduct a critical survey of common representation schemes for two- and three-dimensional objects. Then we present several new schemes for the efficient support of set operators (union, intersection, difference) and search operators (point location, range search).

Polyhedral point sets are represented efficiently as *convex polyhedral chains*, i.e. algebraic sums of convex polyhedra (*cells*). Each cell in turn is represented as an intersection of halfspaces and encoded in a vector. The notion of vertices is abandoned completely. Then the computation of set operators can be decomposed into (a) a collection of vector operations, and (b) a garbage collection where vectors that represent empty cells are eliminated. All results of the garbage collection are cached in the vectors, which speeds up future computations. No special treatment of singular intersection cases is needed. This approach to set operations is significantly different from algorithms that have been proposed in the past.

To detect intersections of hyperplanes and convex polyhedra in arbitrary dimensions, we propose a *dual representation scheme* for polyhedra. In d dimensions, the time complexities of the dual algorithms are $O(2^d \log n)$ and $O((2d)^{d-1} \log^{d-1} n)$ for the hyperplane-polyhedron and the polyhedron-polyhedron intersection detection problems, respectively. In two dimensions, these time bounds are achieved with linear space and preprocessing. In three dimensions, the hyperplane-polyhedron intersection problem is also solved with linear space and preprocessing, which is an improvement over previously known results. Quadratic space and preprocessing, however, is required for the polyhedron-polyhedron intersection problem. For general d, the dual algorithms require $O(n^{2^d})$ space and preprocessing. These results are the first of their kind for dimensions greater than three. All of these results readily extend to unbounded polyhedra.

To support search operations, we introduce the *cell tree*, an index structure for geometric databases that is related to R-trees and BSP-trees. The data objects in the database are represented as convex polyhedral chains. The cell tree is a balanced tree structure whose leaves contain the cells and whose interior nodes correspond to a hierarchy of nested convex polyhedra. This index structure allows quick access to the cells (and thereby to the data objects), depending on their location in space. Furthermore, the cell tree is designed for paged secondary memory: each node corresponds to a disk page. This minimizes the number of page faults occuring during a search operation. Point locations and range searches can therefore be carried out very efficiently using the cell tree. The cell tree is a dynamic structure; insertions and deletions of cells cause only incremental changes. These update operations can be interleaved with searches and no periodic reorganization is required.

For the representation of arbitrary curved shapes, we introduce a hierarchical data structure termed *arc tree*. The arc tree is a balanced binary tree that represents a curve of length l such that any subtree whose root is on the k-th tree level is representing a subcurve of length $l/2^k$. Each tree level is associated with an approximation of the curve; lower levels correspond to approximations of higher resolution. Based on this hierarchy of detail, queries such as point inclusion or intersection detection and computation can be solved in a hierarchical manner. We present the results of a practical performance analysis for various kinds of set and search operators. Several related schemes are also discussed. Finally, we discuss various options to embed arc trees as complex objects in an extensible database management system like POSTGRES.

Table of Contents

Chapter 1. Introduction 1

Chapter 2. Operators and Representation Schemes for Geometric Data 5

2.1. Introduction 5

2.2. Properties of Operators 6

2.2.1. Operand and Result Spaces 6

2.2.2. Order 7

2.2.3. Invariants 7

2.2.4. Commutativity and Associativity 7

2.2.5. Examples: Numerical and Geometric Operators 7

2.3. Properties of Representation Schemes 9

2.3.1. Domain and Range 9

2.3.2. Unambiguous and Unique Representations 9

2.3.3. Irredundant and Concise Representations 10

2.3.4. Invariants 10

2.3.5. Distance Functions 11

2.3.6. Continuity 12

2.4. Elementary Representation Schemes 12

2.4.1. Boundary Representation Schemes 13

2.4.1.1. Vertex Lists for General Polygons 13

2.4.1.2. Fourier Descriptors for Planar Curves 16

2.4.1.3. B-Rep and Wireframe for 3-D Objects 20

2.4.2. Sweep Representation Schemes 22

2.4.3. Skeleton Representation Schemes 22

2.5. Hierarchical Representation Schemes 24

2.5.1. Occupancy Representation Schemes 24

2.5.2. Constructive Solid Geometry (CSG) 26
2.5.3. Halfspaces for Convex Polyhedra 28
2.6. Summary - Evaluation of Representation Schemes 29
Chapter 3. Polyhedral Chains 31
3.1. Introduction 31
3.2. Definition 32
3.3. Properties 36
3.4. Convex Polyhedral Chains as Representation Scheme 37
3.5. The **h**-Vector 39
3.6. Set Operators 41
3.7. Summary 48
Chapter 4. A Dual Approach to Detect Polyhedral Intersections in Arbitrary Dimensions 49
4.1. Introduction 49
4.2. The Dual Representation Scheme 51
4.3. Hyperplane-Polyhedron Intersection Detection 54
4.4. Polyhedron-Polyhedron Intersection Detection 57
4.5. Extensions 62
4.5.1. Unbounded Polyhedra 62
4.5.2. Vertical Hyperplanes 63
4.6. Summary 64
Chapter 5. The Cell Tree: An Index for Geometric Databases 65
5.1. Introduction 65
5.2. Geometric Index Structures 65
5.3. The Geometric Database 72
5.4. The Cell Tree 73
5.4.1. Description 73
5.4.2. Searching 77
5.5. Updating the Cell Tree 78

5.5.1. Insertion 78
5.5.2. Deletion 78
5.5.3. Node Splitting 79
5.5.4. Tree Condensation 82
5.6. Summary 84
Chapter 6. The Arc Tree: An Approximation Scheme To Represent Arbitrary Curved Shapes 85
6.1. Introduction 85
6.2. Definition 86
6.3. Generalization 92
6.4. Hierarchical Point Inclusion Test 98
6.5. Hierarchical Set Operations 102
6.5.1. Curve-Curve Intersection Detection 102
6.5.2. Curve-Curve Intersection Computation 106
6.5.3. Curve-Area Intersection Detection 109
6.5.4. Curve-Area Intersection Computation 109
6.5.5. Area-Area Intersection Detection 113
6.5.6. Area-Area Set Operations 113
6.6. Implementation in a Database System 116
6.6.1. The Pure Relational Approach 116
6.6.2. User Defined Data Types and Operators 117
6.6.3. Procedure as a Data Type 118
6.6.4. Abstract Data Types 119
6.7. Summary 121
Chapter 7. Conclusions 123
References 127