developing serious mathematical software to see the interplay between theory and implementation.

STEVEN PRUESS

Department of Mathematical and Computer Sciences
Colorado School of Mines
Golden, CO 80401

11[65–02, 68–02, 68U07].—JOSEF HOSCHEK & DIETER LASSER, *Fundamentals of Computer Aided Geometric Design* (translated from the German by Larry L. Schumaker), A K Peters, Wellesley, MA, 1993, xviii + 727 pp., $23\frac{1}{2}$ cm. Price $79.95.

We have here a fine addition to the textbook literature in computer-aided geometric design (CAGD). The book evolved in several stages from lecture notes (in German) prepared in 1986. The first published edition appeared in German under the Teubner imprint, and the second German edition appeared in 1992. Larry Schumaker undertook the translation and typesetting that resulted in the edition reviewed here.

The text gives a systematic account of CAGD, and is very much up-to-date. In a short review, we can only outline the contents chapter-by-chapter. Chapter 1 discusses transformations, projections, visibility methods, shading and reflection. Chapter 2 reviews elementary differential geometry, classical interpolation methods, and least squares procedures. Chapter 3 is about splines, including tension splines and exponential splines, all in one variable. Chapter 4 gives the theory of B-splines and Bézier curves. This occupies almost 100 pages. Chapter 5 continues with more technical spline topics: "FC" continuity, curvature continuity, Manning's splines, tau-splines, etc. Chapter 6 begins the study of spline surfaces, including tensor-product surfaces and splines on triangulations. Chapter 7 continues the study of surfaces, including multi-patch methods. Chapter 8 is devoted to Gordon-Coons surfaces (blending methods). Chapter 9 deals with scattered data interpolation by several procedures, including radial basis methods, Shepard's method, and methods from the finite element realm. Chapter 10 is devoted to basis transformations for the representation of curves and surfaces. In Chapter 11, methods for problems in high dimensions are discussed. The last five chapters deal with intersections of curves and surfaces, smoothing techniques, blending methods, offset curves and surfaces, and applications to milling processes.

The book has 727 pages, a bibliography of 83 pages, and a wealth of figures (approximately one per page). The authors (and translator) are to be congratulated for producing a comprehensive book on a timely topic.

E. W. C.

12[68–01, 68U05].—JOSEPH O'ROURKE, *Computational Geometry in C*, Cambridge Univ. Press, Cambridge, MA, 1994, xii + 346 pp., 26 cm. Price $59.95 hardcover, $24.95 paperback.

Computational geometry could be an ideal subject for undergraduate study. Students of mathematics would see how challenging it is to apply mathematical

ideas in a world where there is no real arithmetic, where common assumptions of "general position" or "nondegeneracy" can severely limit or completely destroy the usefulness of a program, and where "abuse of notation" only invites abusive error messages in return. Students of computer science would see applications of the calculus and linear algebra classes that (to many) represent only suffering, the importance of proof to making an algorithm work, and some advantages and disadvantages of asymptotic analysis of algorithm performance.

The textbooks available until now, however, including [1, 2, 3], have been aimed at graduate students. Taking a "high-level" approach to the subject, at best they present "pseudocode" solutions to problems, and they usually ignore the treatment of geometric degeneracies. The preface to the present text says that it is suitable for undergraduates, especially junior and senior majors in computer science or mathematics, and the "in *C*" in its title implies that it contains useful programs that implement algorithms from computational geometry.

Chapters 1–6 cover problems central to computational geometry, including computing polygon partitions, convex hulls, Voronoi diagrams, and arrangements. I would classify most of the problems in Chapters 7 and 8, on geometric search and intersection, and motion planning, as "advanced topics" in computational geometry. Section 8.6, on robot arm motion, is exceptional: it conveys to the reader both an appropriate level of mathematical and computational detail and the author's enthusiasm for the topic. Would that the more fundamental material of the first six chapters had been presented with such care.

Publishing code requires considerable bravery on the part of an author. Published code reveals to everyone all the details its author either considered or failed to consider. Even when published code is correct, readers are bound to criticize the way it is organized, the choice of data structures and algorithms, the names of variables, and the indentation convention. While some of my criticisms do indeed fall into these categories, most of them reflect larger questions about overall programming judgment. (It is disappointing to note, however, that the programs were typeset by hand, and many include typographical errors introduced by the printers. It is possible to include program source text directly from the version that was compiled and tested, which would eliminate one source of error.)

Chapter 1 begins with some useful small functions to compute areas, point orientations, and the intersections of simple curves (line segments and circles). Mathematicians may consider these functions trivial, but computer programmers know how hard it is to make them work when areas may vanish, points may coincide, or other degeneracies may occur.

Most of the larger programs, however, are disappointing. The preface warns that the text includes implementations of only nine algorithms, but even those nine programs are mostly toys and curios.

(1) The first program triangulates a polygon by chopping off one triangle at a time, so it runs in quadratic time. This choice of a simple algorithm is reasonable since it illustrates nicely the meticulous care required to implement reliable geometric computations. Unfortunately, the program stores a polygon in an array with room for up to 1000 vertices, wasting both space and time: Since the algorithm works by repeatedly deleting vertices from the middle of the array, this choice of data structure is simply wrong. O'Rourke repeatedly defends the use of arrays in the name of making the code easier to understand.

In my experience, upper-division undergraduates can cope with linked data structures.

(2) The program to compute a Delaunay triangulation runs in quartic time, which the text admits is "rarely acceptable." It would have been far better to present an implementation of the incremental algorithm (the most popular, according to the text), which runs in quadratic time.

(3) The program to test whether a given point lies inside a given polygon could be one of the cleanest of the long programs in the book. As written, however, it destroys the vertices of the input polygon, though a footnote admits that this effect is "rarely desired."

(4) Of the program to compute the intersection of two convex polygons, "It only remains to settle the 'degenerate' cases to get code. We will not argue these details here but only claim that the algorithm is fortunately robust in that almost any decision works" (p. 246). Just *where* could it *possibly* be *more* appropriate to argue the correctness of a C program implementing a geometric algorithm than in a book entitled *Computational Geometry in C*?

Of course not every algorithm in the text needs to appear in an implementation. Nevertheless, I consider it a serious mistake not to have included any implementation of a plane-sweep algorithm. The text itself includes ample justification for the importance of the plane-sweep paradigm in connection with triangulations, Voronoi diagrams, and arrangements.

Since the book contains only nine implementations, much of the remainder is written in the style of a mathematics book, with definitions, lemmas, theorems, and proofs. Mathematicians who criticize computer scientists' attempts at writing mathematics as clumsy and exasperating will not be heartened. Almost every proof asserts that something is "clear" or "evident." This lapse is especially surprising since there are so many examples of discredited results in computational geometry that were once said to be "clear" or "proven by picture."

O'Rourke is fond of exploring questions by drawing pictures, and he can do a nice job of bringing us along on these experimental forays, both building our intuition and showing us pitfalls. Unfortunately, he often fails to establish a point beyond which the experiments are over, where he states the definitions he will use or the results he will prove.

(1) "A little experimentation can lead to a conviction that no more than two [guards] are even needed, so that $G(6) = 2$" (p. 5). Are we meant to believe that $G(6) = 2$ because of our experiments, our conviction, or the proof several pages later?

(2) The first paragraph of §3.2.2 explains what it means for an edge of a polygon to be extreme. Then the second paragraph begins "There is unfortunately an inaccuracy in our characterization of extreme edges" (p. 74), and goes on to explain a modification. Just what, finally, is "our characterization"?

(3) The proof of Euler's formula on p. 119 does not distinguish appropriately between a planar graph and a particular embedding of it.

(4) In a proof, "Because no points are on the boundary of $C(x)$ other than a and b (by hypothesis), there must be freedom to wiggle x a bit and maintain emptiness. In particular, we can move x along B_{ab}, the bisector between a and b, and maintain emptiness while keeping

the circle through a and b" (p. 177). The mention of x "wiggling" should be supplemented by a formal statement that "there exists $\varepsilon > 0$ such that $C(z)$ is empty whenever z is chosen such that $z \in B_{ab}$ and $|z - x| < \varepsilon$." When will students ever learn to express concepts like "wiggle" appropriately if textbooks don't show them?

Let me emphasize that I *do* appreciate it when books include explorations and intuitive explanations. But my experience with upper-division undergraduates suggests that they also need explicit and careful statements of results and their proofs.

A couple of minor errors in the mathematical exposition are especially apt to confuse students.

(1) The attempt to define a polygon without appeal to topology (p. 1) is incorrect: the condition that nonadjacent edges of the polygon do not intersect should be written "$e_i \cap e_j = \varnothing$ whenever $i - 1 \neq j \neq i + 1$."

(2) On p. 74, the notation "$l \neq i \neq j \neq k$" is apparently used to mean "$l \notin \{i, j, k\}$."

O'Rourke himself explains one of the overarching problems of the book on p. ix of his preface: "It has become the fashion for textbooks to include far more material than can be covered in one semester. This is not such a text. It is a written record of what I cover with undergraduates in one 40 class-hour semester." The book does indeed have the feel of lecture notes. Terms are used before they are defined, and conditions relied upon before they are stated. While generalizations and failures to generalize (especially to higher dimensions) are mentioned, most details are omitted. And the many footnotes and digressions suggest places where students asked questions about definitions, notation, floating-point arithmetic, algorithmic complexity, topology, etc.

But lecture notes do not necessarily make good textbooks. An obvious reason is that every class is different. O'Rourke's students, for example, apparently understand without explanation topological terms like "open," "closed," and "bounded," but require explanations of cardinality, the floor function, and the relationship between logical conjunction and set intersection. I would expect most of my students to understand the latter but to need help with the former.

Lecture notes also impinge on the instructor's prerogatives, unlike more comprehensive textbooks that include more material than can be covered in one semester. A class that uses a textbook can explore various ways to define something or to pose a question, then turn to the text for a definitive rendering. When the text includes appropriately precise formulations, the instructor can explain in class that one of them means "let x wiggle." When the text includes complete verification of code correctness, the instructor can decide how many degenerate cases to check in class. In brief, lecture notes make it harder for the instructor to choose which issues to cover carefully, which to skim, and which to omit.

The preface suggests that the book is appropriate for undergraduates, graduate students, programmers, and those not interested in programming. Attempting to satisfy such a mixed audience, and simultaneously to cover a broad variety of topics, O'Rourke has opted for such superficial coverage that I wonder how much his audience can possibly understand about them. This is particularly noticable in Chapter 6, where duality and higher-order Voronoi diagrams re-

ceive only four pages each, and in Chapter 8, where "conceptual algorithms" abound. This would be a much better book if it devoted more attention to fewer topics, especially if those topics coincided better with O'Rourke's interests. The marked contrast between the author's obvious enthusiasm for clever mathematical ideas and his apologetic tone in explanations of code suggests that he might have done better to omit most of the programs altogether, while taking more care over the mathematical details.

CHRISTOPHER J. VAN WYK

Department of Mathematics and Computer Science
Drew University
Madison, NJ 07940

1. H. Edelsbrunner, *Algorithms in combinatorial geometry*, EATCS Monographs on Theoretical Computer Science, Springer, Berlin, 1987.
2. K. Mehlhorn, *Data structures and algorithms. Vol.* 3: *Multidimensional searching and computational geometry*, EATCS Monographs on Theoretical Computer Science, Springer, Berlin, 1984.
3. F. P. Preparata, *Computational geometry : an introduction*, Texts and Monographs in Computer Science, Springer, New York, 1985.

13[43-06, 43A32, 43A70, 42C15].—LARRY L. SCHUMAKER & GLENN WEBB (Editors), *Recent Advances in Wavelet Analysis*, Wavelet Analysis and Its Applications, Vol. 3, Academic Press, Boston, MA, 1994, xii + 364 pp., $23\frac{1}{2}$ cm. Price $59.95.

This is an important collection of papers by a truly distinguished group of contributors, and it should be in the hands of anyone who may be trying to keep abreast of the frantic activity in wavelet theory. All ten papers provide discussions in depth of the topics they address; they range in length from 22 to 62 pages.

In the first paper, Andersson, Hall, Jawerth, and Peters explore orthogonal and biorthogonal wavelets on prescribed subsets of the real line. A new concept of "wavelet probing" is introduced, and there are applications of wavelets to boundary value problems for ordinary differential equations. In the second paper, Auscher and Tchamitchian construct wavelet bases for elliptic differential operators in one dimension. These lead to estimates of the corresponding Green's kernel. Along the way, they create an extension of the multiresolution theory. In the third paper, Battle identifies the wavelets associated with Wilson's recursion formula in statistical mechanics. He then uses them to refine the formula. In the fourth paper, Benassi and Jaffard study a class of Gaussian random fields by means of wavelet decomposition. This work has applications in the theory of Brownian motion. In the fifth paper, Chui and Shi introduce multivariate wavelets in which the scaling (dilation) is different in each dimension, or, more generally, is defined by any nonsingular linear map. This innovation opens the possibility of constructing Riesz bases for multivariable functions using only a single function, together with the generalized notion of dilation and the usual translation by multi-integers. In the sixth paper, Dahmen, Prössdorf, and Schneider develop numerical schemes of the Petrov-Galerkin genre with trial spaces generated by a single scaling function. These schemes are applicable to a large class of pseudodifferential equations, and the authors discuss