Foreword

Eric Haines

On my shelf is an old book called *A Programmer's Geometry,* by Bowyer and Woodwark. It was published in 1983, reprinted twice in 1984 and 1985, but then discontinued. Over the years I have guarded my copy, keeping careful track of who borrowed it. Checking on the Web today, I found six used copies ranging in price from $50 to $100. This is a high price range for a paperback book only 140 pages in length. The reason the book is worth this much is that it describes how to program various operations related to 2D geometry. It does not just present geometric formulae; it also describes efficient ways to accomplish tasks and gives code snippets (in FORTRAN).

Now, almost two decades later, we have a worthy successor to that slim volume. The book before you distills a huge amount of literature on geometry into that which is most useful to programmers. The field of computer graphics has evolved considerably since 1983, and this volume reflects those advances. Due to the continuing improvement in computer processor performance, operations that once were only part of offline analysis packages are now commonly done in interactive programs. Polygon triangulation, collision detection and response, and surface modelling and modification are now possible at real-time rates. This book gives solid explanations and code to perform these and many other algorithms.

Beyond providing a solid reference for a wide range of geometry-related tasks, this volume also presents the underpinnings of the theory behind the algorithms. Rather than employ a pure cookbook approach, which can leave the reader with runnable code and no comprehension of how it works, the authors explain key concepts. This approach makes each algorithm a tool that, further on, can be recombined with other tools.

The dynamic nature of computer graphics makes it a particularly interesting area of study. Research and implementation of rendering methods respond to changes in the underlying hardware. For example, in the field of interactive rendering, the emerging programmable nature of the graphics accelerator has changed the relative costs of different techniques. On a broader scale, the evolution of the CPU has made memory access and caching rise in importance, compared to the older practice of minimizing the number of operations (e.g., counting multiplies and adds). However, the underlying theory and algorithms for, say, finding the convex hull of an object are considerably more long-lasting, less affected by changes. Of course, more efficient algorithms are found over time, and hardware influences which method currently is considered the fastest, but the basic principles remain the same. Years after you have shed your books on DirectX 9 or Intel's 64-bit Itanium architecture, you are likely to have some edition of this book on your shelf.

Another reason this book will have increased staying power is the Internet. I am the archivist for the "Graphics Gems" series code repository. The code for this series of books, including code by Philip Schneider, was wisely made free for reuse when the series was published in the early 1990s. Over the years readers have sent in bug fixes and improvements to the code base, so benefiting all. Similarly, Dave Eberly has carefully maintained his "Magic Software" Web site (*www.magic-software.com*), which includes working versions of many of the algorithms presented in this volume. Called "a national treasure" by a leading researcher in computer graphics, this site allows addenda and corrigenda to be made available instantly whenever they are needed. Code does not rust; it improves with age when properly supported. This is particularly true for algorithms in this book as they are not tied to particular hardware, network protocols, or other transient objects.

Over the years I and many others have used algorithms and code by the authors in products and research projects. An hour of a programmer's time often costs more than the price of a book. By this measure, you hold a volume potentially worth thousands of dollars. That it can be purchased for a fraction of this cost I consider a modern miracle. The amount of information crammed into this book is incredible. The mathematics may be slow going at times, but the alternative would be to include wordier and less precise descriptions of fewer algorithms. If you are looking for a lightweight text you can read through and check off your list, keep searching. This book sometimes requires effort and struggle to fully comprehend but then, so do most of the worthwhile things in the world.