# Computer Graphics

Edward Angel

# Computer Graphics

**Edward Angel**
University of New Mexico

Addison-Wesley Publishing Company
Reading, Massachusetts • Menlo Park, California • New York
Don Mills, Ontario • Wokingham, England • Amsterdam • Bonn
Sydney • Singapore • Tokyo • Madrid • San Juan

Cover art courtesy of LINKS Corporation.

| | |
|---|---|
| James T. DeWolf | Sponsoring Editor |
| Karen Myer | Production Supervisor |
| Patsy DuMoulin | Production Coordinator |
| Joseph K. Vetere | Technical Art Consultant |
| Rose Mary Molnar | Artist |
| Lyn Duprè | Copyeditor |
| Jean Seal | Cover Design |
| Jean Hammond | Interior Design |
| Melinda Grosser | Four-Color Insert Design |

Library of Congress Cataloging-in-Publication Data

Angel, Edward.
Computer graphics / Edward Angel.
p. cm.
Includes index.
ISBN 0-201-13548-5
1. Computer graphics. 2. C (Computer program language)
I. Title.
T385.A513 1990

006.6–dc19 89-332
CIP

ABCDEFGHIJ-HA-89

■ **To Rose Mary**

# PREFACE

The past few years have seen a revolution in the way computer graphics is practiced. Computer graphics, while still an area of much activity in computer science, is also an area of great importance to students and practitioners of engineering, science, business, and mathematics. The advances in both hardware and software have led to the modern graphics workstation as a standard tool. These workstations include not only the hardware for high-resolution, bit-mapped displays but also the necessary software tools that allow users to develop their own applications. Thus, we are seeing the electrical engineer writing her own interface to a circuit-analysis package and mechanical engineers writing menu-driven CAD packages. For the computer scientist, computer graphics has expanded to include a wealth of new and exciting problems from the generation of photorealistic displays to the design of software tools.

This book is intended for use in a first course in computer graphics for computer scientists and engineers. Although such a course is normally taught for seniors, the only prerequisites assumed are good programming skills (equivalent to CS 2 in the ACM model curriculum) and trigonometry.

There are four fundamental precepts on which the book is based. First, with the availability of present software and hardware tools, it is both possible and important that students get working on significant *applications* of computer graphics early in the course. Hence, I have adopted a top-down approach that attempts to get students working on projects using Graphical Kernel System (GKS)—or any other available system—before we spend significant time on the standard graphics algorithms, such as line drawing and fill. This approach is in contrast to most books that use a bottom up approach that begins with pixels and works up through lines and eventually gets to applications.

Second, the adoption of GKS as the standard graphics language is of great significance to the teaching of computer graphics. In spite of its faults, GKS provides a conceptual basis for the teaching of computer graphics that is shared by many systems and allows us to teach using tools that can easily be transported to other systems. Just as we do not teach our own locally-developed programming language to our beginning students, I believe we should avoid teaching home-grown graphics software in a first graphics course.

Third, the increased programming skills of both engineers and computer scientists has had a significant effect on this book. Material such as modeling with hierarchical data structures, which most texts leave to the end or completely omit, is well within the abilities of sophomores and juniors in computer science and engineering. The chapters on transformations and hierarchical modeling are core to this book and appear early.

Finally, the expansion of our knowledge in computer graphics has made it likely that there will be at least two courses in computer graphics within most computer science departments. The first will be an introduction covering the range of the field with a significant emphasis on the systems and software engineering aspects of computer graphics. Follow-on courses will emphasize algorithms, geometric modeling, ray tracing, and computational geometry. This book is designed for such a first course.

The choice of the C language and the level of detail on GKS were decisions made after great thought and with the input of a number of people. C provides a nice balance between the desire to provide some abstraction while still being close enough to the machine that implementation issues can be discussed. The fact that it is presently the language of choice by implementors of graphics systems is an added benefit. The level of C used should not present any serious problems to students who know only Pascal or FORTRAN. The level of detail on the GKS C language binding may be more controversial. My experience with other books has been that a lot of class time has been wasted clarifying or correcting sketchy material on some particular graphics language used by the text. I have tried to strike a balance by using a subset of GKS but providing the details for the functions used. Even in courses that do not have an available GKS implementation, I believe it is important to see the details even if the students write no code. For those with other software systems such as PHIGS or some of the commercial systems, the changes necessary to convert the GKS code should be minor.

The first eight chapters form the basis of a one-semester senior course, primarily for computer scientists and computer engineers. The book can be used for a two-semester course by going into more depth on some of the algorithms and doing more than surveying the final two chapters. For courses with a heavy project emphasis, a two-semester (or two quarter) sequence can be obtained by using Chapters 1–5 for the first course and Chapters 6–10 for the second. Chapters 1–4 and probably 5 should be studied in order. Chapters 6–7 and 8–10 are fairly independent of each other.

The book should also be accessible to professional programmers, engineers, computer scientists, and others. The notes that preceded this book are the basis for two four-day intensive short courses, one a survey of computer graphics and the other on GKS, which have been taught to thousands of programmers, engineers, and scientists in the United States and Europe.

I would like to acknowledge a number of people who have helped me not only with this book but also in learning computer graphics. This book started when I returned from sabbatical in 1982 to find there was no one available to teach computer graphics in my department. I thank the University of New Mexico for providing me with that opportunity and the facilities to develop a laboratory in computer graphics. David Collins, Eric Garen, and Anders Amundson of Learning Tree International, Inc. (formerly Integrated Computer Systems, Inc.) gave me the chance to create two short courses. Instructors, including Jim Burk, Mark Henderson, Mike Bailey, and Kelly Booth, and the thousands of participants in these courses provided significant feedback. Many students contributed to early versions of the programs and diagrams in this book. In particular, I wish to thank Mark McLaughlin, Dan Shawver, Mathew Nordhaus, and Joe Higgins. Of the many reviewers of various versions of the manuscript, George Grinstein, University of Lowell; Michael J. Zyda, Naval Postgraduate School; Lewis Hitchner, University of California, Santa Cruz; Spencer W. Thomas, The University of Michigan; Mark Henderson, Arizona State University; and Steve Wampler, Northern Arizona University provided particularly helpful comments. My colleagues John Brayer, Dick Nordhaus, and Bernard Moret were both knowledgeable and patient in filling in gaps in my knowledge in subjects as diverse as TEX, data structures, and architecture (both computer and building design). Rab Hagy and George Schaeffer provided me with the latest versions of the GKS and PHIGS C language bindings at a crucial time. Professors Michael Duff of University College London and Mike Godfrey of Imperial College were extremely generous in providing me with friendship and facilities during my recently completed sabbatical leave.

A few comments on the illustrations in this book are appropriate. Early in this project, Jim DeWolf, of Addison-Wesley, and I agreed that a book on computer graphics should use computer graphics to generate all the figures and that all the figures must be of high quality. With the exception of the color plates, all diagrams were produced on an Apple Macintosh SE by Rose Mary Molnar. Tools range from the use of Aldus Freehand (for most figures), to Super 3D, to the direct production of PostScript files from C programs. A few images, such as the pixel image of my cat Mongo, were created using a Thunder Scanner on an ImageWriter. Hopefully the figures are both informative and illustrate the quality of interactive graphics software presently available. Writing programs to generate the images in some of the figures or to create the kinds of drawing tools needed to produce such images can lead to innumerable interesting programming projects for students.

The people at Addison-Wesley, especially Karen Myer and Mona Zeftel, could not have been more helpful. I will have no author horror stories to share with my colleagues. A final thank you is due to Bob Drake. While Bob was at Addison-Wesley, he promised to (and did) call me once a month to convince me to do this book.

At this point, most authors thank their wives for their patience during the writing of their books. When your spouse is the illustrator of the book, patience is only one of many important and necessary characteristics. Fortunately for me, Rose Mary has so many wonderful qualities we were able to survive this experience.

Edward Angel
University of New Mexico

# C O N T E N T S

■ **1 Overview** 1

Introduction 1
- 1.1 Applications of Computer Graphics 2
  - 1.1.1 Display of Information 3
  - 1.1.2 Design 3
  - 1.1.3 Simulation 4
  - 1.1.4 User Interfaces 4
- 1.2 The Development of Computer Graphics 5
  - 1.2.1 1950–1960 5
  - 1.2.2 1960–1970 6
  - 1.2.3 1970–1980 7
  - 1.2.4 1980 to the Present 8
- 1.3 A Basic Graphics System 9
  - 1.3.1 The Processor 10
  - 1.3.2 Memory 10
  - 1.3.3 Output Devices 10
  - 1.3.4 Input Devices 11
- 1.4 Graphics Software 11
  - 1.4.1 Terminal-Based Software 12
  - 1.4.2 Turnkey Software 13
- 1.5 The Rest of the Book 13
- 1.6 Suggested Readings 14
  - Exercises 15

■ **2 Fundamental Ideas** 17

Introduction 17
- 2.1 A Simple Plotting Procedure 17
- 2.2 Image Formation 19

2.3 The Synthetic Camera 23
2.3.1 Separating the Viewer from the Objects 23
2.3.2 Two-Dimensional Viewing 25
2.3.3 Device Independence 26
2.4 Device-Independent Software 26
2.5 Windows and Viewports 28
2.6 Positioning 32
2.7 Points, Lines, and Curves 36
2.7.1 Points 36
2.7.2 Vectors 37
2.7.3 Curves 39
2.7.4 Explicit Curves 39
2.7.5 Implicit Form 40
2.7.6 Parametric Form 43
2.8 Portability Considerations 45
2.8.1 Functionality versus Format 45
2.8.2 Defaults and Choices 47
2.8.3 Error Handling 48
2.9 Suggested Readings 49
Exercises 50

**3 Two-Dimensional Graphics** 53

Introduction 53
3.1 Device-Independent Graphics Standards 54
3.2 The Programmer's Model 55
3.2.1 Logical and Physical Workstations 56
3.2.2 Communicating with the Hardware 58
3.2.3 Implementation Issues 59
3.3 Graphics Functions 60
3.3.1 Output Functions 60
3.3.2 Control Functions 61
3.3.3 Attributes 61
3.3.4 Viewing and Transformation Functions 62
3.3.5 Input Functions 62
3.3.6 Segmentation Functions 63
3.3.7 Metafiles 63
3.3.8 Inquiry Functions 64
3.4 A Simple Program 64
3.4.1 The Pen-Plotter Model 64
3.4.2 Polyline and Text 66
3.5 Viewing 69
3.5.1 The Normalization Transformation 69
3.5.2 Clipping 71

3.5.3 The Workstation Transformation 73
3.6 Control 76
3.6.1 Initialization 76
3.6.2 The Error File 77
3.6.3 Opening the System 77
3.6.4 Opening and Activating Workstations 78
3.6.5 Termination 79
3.7 Polyline and Text Attributes 79
3.7.1 Geometric and Nongeometric Attributes 80
3.7.2 Polyline Attributes 80
3.7.3 Text Attributes 83
3.7.4 Bundled Attributes 85
3.8 Other Primitives 87
3.8.1 The Polymarker 87
3.8.2 The Fill Area 88
3.8.3 Cell Arrays 90
3.8.4 Generalized Drawing Primitives 91
3.9 A Self-Scaling Plotter 91
3.9.1 Setting up the Normalization Transformations 93
3.10 Metafiles 96
3.10.1 The GKS Metafile 97
3.10.2 Interpreting A GKS Metafile 98
3.10.3 The Computer-Graphics Metafile 100
3.11 Suggested Readings 100
Exercises 101

■ **4 Interactive Graphics** 105

Introduction 105
4.1 Programming with Interaction 105
4.2 A Shape-Layout Program 106
4.2.1 Choosing Windows and Viewports 107
4.2.2 The Shape Menu 108
4.3 Defining Objects 110
4.3.1 Segments 110
4.3.2 Segments and Program Flow 112
4.3.3 Buffering 113
4.4 Segment Attributes 114
4.4.1 Visibility 114
4.4.2 Priority 115
4.4.3 Other Attributes 116
4.5 Input 117
4.5.1 Logical versus Physical Input 117
4.5.2 Logical Input Classes 118
4.5.3 Measure and Trigger 119

4.5.4 Input Modes 119
4.5.5 Prompt, Echo, and Status Feedback 121
4.5.6 Programming Input 122
4.6 Physical Input Devices 123
4.6.1 The Keyboard 123
4.6.2 The Lightpen 124
4.6.3 The Joy Stick 126
4.6.4 The Trackball and the Mouse 127
4.6.5 Data Tablets 127
4.6.6 Graphical Devices 128
4.6.7 Dragging 129
4.7 The Pick 129
4.7.1 Using the Returned Status 130
4.7.2 Pick Identifiers 131
4.7.3 Setting Up the Menus 132
4.7.4 The Control Loop 134
4.7.5 Mode Selection and Initialization 135
4.7.6 General Program Flow 136
4.8 The Locator 137
4.8.1 Request Locator 137
4.8.2 Inverting the Coordinate Transformations 137
4.8.3 Entering the Data 138
4.8.4 Device Initialization 140
4.9 String Input 141
4.9.1 Using an Inquiry 142
4.9.2 Pausing During Execution 142
4.9.3 Completing the Layout Program 143
4.10 Event-Driven Input 144
4.11 The User Interface 145
4.11.1 Menus 146
4.11.2 Icons 148
4.11.3 User Feedback 150
4.11.4 User Aids 151
4.11.5 Layout 152
4.11.6 color 154
4.12 The Burden of Interaction 154
4.13 Suggested Readings 155
Exercises 155

**5 Transformations and Modeling** 159

Introduction 159
5.1 Affine Transformations 160
5.1.1 General Transformations 160
5.1.2 Transforming Lines to Lines 161
5.1.3 Translation 161

5.1.4 Rotation 162
5.1.5 Scaling 163
5.1.6 Shear 164
5.2 Concatenating Transformations 165
5.2.1 Rotating About a Fixed Point 165
5.2.2 Homogeneous Coordinates 167
5.2.3 Matrix Representations 169
5.2.4 Inverse Transformations 170
5.2.5 Concatenation Examples 171
5.3 Transformations in GKS 174
5.4 A Transformation Package 176
5.4.1 Evaluation Procedures 177
5.4.2 Accumulation Procedures 178
5.4.3 Applying the Transformations 179
5.5 Symbols and Instances 181
5.5.1 Symbols 182
5.5.2 Modeling with Symbols 182
5.6 Modeling With Relationships 184
5.6.1 A Simple Robot Arm 185
5.6.2 Modeling With Transformation Matrices 186
5.6.3 Animating the Model 187
5.7 Using Hierarchy and Recursion 188
5.7.1 The Robot Arm as a Tree 189
5.7.2 Representing a Tree 190
5.7.3 Traversing the Model 192
5.7.4 Discussion 193
5.8 Implementation of Abstract Data Types 194
5.8.1 Operations on a Tree 194
5.8.2 Another Implementation 195
5.9 From Segments to Structures 199
5.9.1 Segment Contents 199
5.9.2 Directed Acyclic Graphs 200
5.9.3 Structures 200
5.10 PHIGS 202
5.10.1 Viewing a Database 203
5.10.2 Programming in PHIGS 204
5.10.3 Modeling with PHIGS 205
5.11 Suggested Readings 207
Exercises 207

**6 Implementation** 211

Introduction 211
6.1 Implementation Issues 211
6.2 Following the Pipeline 213

6.3 Clipping 215
6.3.1 The Difficulty of Clipping 215
6.3.2 Text Clipping 216
6.4 Clipping Line Segments 218
6.5 The Cohen-Sutherland Algorithm 222
6.5.1 Outcodes 222
6.5.2 The Accept and Reject Checks 223
6.5.3 Computing Intersections 223
6.6 Other Clipping Methods 226
6.6.1 Reentrant Clipping 227
6.6.2 Using Bounding Boxes 227
6.7 Device Drivers 229
6.7.1 ASCII Devices 230
6.7.2 REGIS Drivers 231
6.7.3 Tektronix Drivers 233
6.8 Scan Converting Line Segments 235
6.8.1 Setting Pixels 236
6.8.2 A Simple Algorithm 237
6.9 Bresenham's Algorithm 240
6.10 Real-Time Processors 245
6.10.1 The Display Processor 246
6.10.2 Graphics Workstations 249
6.11 Suggested Readings 250
Exercises 250

**7 Raster Graphics** 253

Introduction 253
7.1 The Frame Buffer 254
7.1.1 Conceptualizing the Frame Buffer 255
7.1.2 Manipulating the Frame Buffer 255
7.2 Writing in the Frame Buffer 257
7.2.1 Swapping 258
7.2.2 Writing Modes 260
7.3 Using XOR 261
7.3.1 Swapping Revisited 262
7.3.2 Erasing, Cursors, and Rubberbanding 263
7.3.3 Simple Fill 265
7.4 BitBlt Operations 265
7.4.1 Formulating the Operations 266
7.4.2 Clipping 266
7.4.3 CharBlt 267
7.5 Polygons and Raster 268
7.5.1 Representation 269

7.5.2 Clipping 272
7.6 Fill 274
7.6.1 GKS Fill Areas 275
7.6.2 Where Are the Intersections? 276
7.6.3 Edge-Flag Methods 277
7.6.4 Priority Methods 277
7.6.5 Recursive Methods 279
7.6.6 Sorting Methods 281
7.7 Color 282
7.7.1 Brightness and Intensity 283
7.7.2 Three-Color Theory 284
7.7.3 The Color Solid 285
7.7.4 Producing Color 286
7.7.5 Color Matching and Color Systems 288
7.7.6 Perceptual Color 289
7.8 Using Multiple-Bit Pixels 291
7.8.1 Lookup Tables 291
7.8.2 Antialiasing 293
7.9 Suggested Readings 295
Exercises 295

**8 Three-Dimensional Graphics** 299

Introduction 299
8.1 Three-Dimensional Representations 300
8.1.1 Three-Dimensional Curves and Surfaces 301
8.1.2 Planes 302
8.2 Three-Dimensional Primitives 304
8.2.1 The Polyline 304
8.2.2 Extending the GKS Primitives 305
8.3 Transformations 306
8.3.1 Homogeneous Coordinates 306
8.3.2 Translation 307
8.3.3 Scaling and Shear 307
8.3.4 Rotation 309
8.3.5 A Three-Dimensional Transformation Package 311
8.4 An Example 312
8.4.1 Instancing a Cube 312
8.4.2 Direction Cosines 315
8.5 Projections and Normalization 318
8.5.1 The Normalization Transformation 319
8.5.2 Specifying a Projection Plane 321
8.5.3 Viewing Coordinates 321
8.5.4 Projection 323

8.5.5 Clipping 324
8.6 Classical and Computer Graphics 326
8.6.1 Classical Viewing 326
8.6.2 Orthographic Projections 327
8.6.3 Axonometric Projections 327
8.6.4 Oblique Projections 330
8.6.5 Perspective Viewing 330
8.7 Implementation 332
8.7.1 Orthogonal Viewing 333
8.7.2 Computing the View-Orientation Transformation 335
8.7.3 An Example 339
8.7.4 Projection 340
8.7.5 Oblique Viewing 341
8.7.6 Implementing Perspective Viewing 343
8.8 Suggested Readings 346
Exercises 346

**9 Working with Polygons** 351

Introduction 351
9.1 Polygons and Realism 351
9.2 Representing Polygons in Three Dimensions 353
9.2.1 Polygons and Normals 353
9.2.2 Computing the Normal 354
9.3 Polygon Meshes 356
9.3.1 Edges, Surfaces, and Volumes 357
9.3.2 Quadrilateral and Triangular Meshes 358
9.3.3 Approximating Spheres 360
9.4 Hidden-Surface Removal 361
9.4.1 Hidden-Surface Removal and Sorting 362
9.4.2 Object-Space versus Image-Space Approaches 363
9.5 Object-Space Algorithms 364
9.5.1 Removing Back-Facing Polygons 365
9.5.2 Depth Sort 366
9.5.3 The General Case 367
9.6 Image-Space Algorithms 369
9.6.1 The $z$-buffer Algorithm 369
9.6.2 The Scan-Line Algorithm 371
9.7 Rendering 372
9.7.1 Ray Tracing 373
9.7.2 Ray Casting 375
9.7.3 Aliasing and Rendering 376
9.7.4 Shadow Rays 376

9.7.5 Rendering Without Ray Tracing 377
9.8 Shading Models 378
9.8.1 Diffuse Reflections 380
9.8.2 Ambient Light 380
9.8.3 Specular Reflections 381
9.9 Polygonal Shading 382
9.9.1 Gouraud and Phong Shading 382
9.10 Suggested Readings 384
Exercises 384

**10 Curves and Surfaces** 387

Introduction 387
10.1 Explicit, Implicit, and Parametric Curves 388
10.1.1 Explicit Form 388
10.1.2 Implicit Form 389
10.1.3 Parametric Form 390
10.1.4 Example 390
10.2 Polynomial Curves 392
10.2.1 Cubic Polynomials 393
10.3 Interpolating Form 395
10.3.1 The Cubic Interpolating Polynomial 396
10.3.2 Joining Curve Segments 397
10.3.3 Blending Polynomials 399
10.3.4 Approximating a Circle 400
10.4 Smoothing Polynomials 404
10.4.1 Hermite Polynomials 405
10.4.2 Bezier Polynomials 406
10.4.3 Splines 408
10.4.4 Example 411
10.5 Scan Converting Polynomials 412
10.5.1 Forward Differences 413
10.5.2 Example 415
10.5.3 Scan Conversion by Subdivision 415
10.6 Parametric Surfaces 417
10.6.1 The Plane and the Sphere 418
10.6.2 Bicubic Polynomials 419
10.6.3 Interpolation 421
10.6.4 Bezier Patches 423
10.7 Realism 425
10.7.1 Hidden Surface Removal 425
10.7.2 Rendering 427
10.7.3 Scan Conversion 429