

Alan Watt


Third Edition

3D Computer Graphics



ADDISON-WESLEY





3D Computer Graphics

THIRD EDITION

ALAN WATT



ADDISON-WESLEY

An imprint of PEARSON EDUCATION

Harlow, England · London · New York · Reading, Massachusetts · San Francisco · Toronto · Don Mills, Ontario · Sydney
Tokyo · Singapore · Hong Kong · Seoul · Taipei · Cape Town · Madrid · Mexico City · Amsterdam · Munich · Paris · Milan

Pearson Education Limited
Edinburgh Gate
Harlow
Essex CM20 2JE
England

and Associated Companies around the world

Visit us on the World Wide Web at:
www.pearsoned-ema.com

First published 1989
Second edition 1993
This edition first published 2000

© 1989, 1993 Addison-Wesley Publishing Ltd, Addison-Wesley Publishing Company Inc.
© Pearson Education Limited 2000

The right of Alan Watt to be identified as author of
this work has been asserted by him in accordance with
the Copyright, Designs, and Patents Act 1988.

All rights reserved; no part of this publication may be reproduced, stored
in a retrieval system, or transmitted in any form or by any means, electronic,
mechanical, photocopying, recording, or otherwise without either the prior
written permission of the Publishers or a licence permitting restricted copying
in the United Kingdom issued by the Copyright Licensing Agency Ltd,
90 Tottenham Court Road, London W1P 0LP.

The programs in this book have been included for their instructional value.
The publisher does not offer any warranties or representation in respect of their
fitness for a particular purpose, nor does the publisher accept any liability for any
loss or damage (other than for personal injury or death) arising from their use.

Many of the designations used by manufacturers and sellers to distinguish their
products are claimed as trademarks. Pearson Education Limited has made every
attempt to supply trademark information about manufacturers and their products
mentioned in this book. A list of trademark designations and their owners
appears on page xxii.

ISBN 0 201 39855 9

British Library Cataloguing-in-Publication Data

A catalogue record for this book can be obtained from the British Library.

Library of Congress Cataloging-in-Publication Data

Available from the publisher.

Typeset by 42

Printed and bound in The United States of America



Preface

This is the third edition of a book that deals with the processes involved in converting a mathematical or geometric description of an object – a computer graphics model – into a visualization – a two-dimensional projection – that simulates the appearance of a real object. The analogy of a synthetic camera is often used and this is a good allusion provided we bear in mind certain important limitations that are not usually available in a computer graphics camera (depth of field and motion blur are two examples) and certain computer graphics facilities that do not appear in a camera (near and far clipping planes).

Algorithms in computer graphics mostly function in a three-dimensional domain and the creations in this space are then mapped into a two-dimensional display or image plane at a late stage in the overall process. Traditionally computer graphics has created pictures by starting with a very detailed geometric description, subjecting this to a series of transformations that orient a viewer and objects in three-dimensional space, then imitating reality by making the objects look solid and real – a process known as rendering. In the early 1980s there was a coming together of research – carried out in the 1970s into reflection models, hidden surface removal and the like – that resulted in the emergence of a *de facto* approach to image synthesis of solid objects. But now this is proving insufficient for the new demands of moving computer imagery and virtual reality and much research is being carried out into how to model complex objects, where the nature and shape of the object changes dynamically and into capturing the richness of the world without having to explicitly model every detail. Such efforts are resulting in diverse synthesis methods and modelling methods but at the moment there has been no emergence of new image generation techniques that rival the pseudo-standard way of modelling and rendering solid objects – a method that has been established since the mid-1970s.

So where did it all begin? Most of the development in computer graphics as we know it today was motivated by hardware evolution and the availability of new devices. Software rapidly developed to use the image producing hardware. In this respect the most important development is the so-called raster display, a device that proliferated in the mass market shortly after the development of the PC. In this device the complete image is stored in a memory variously called a

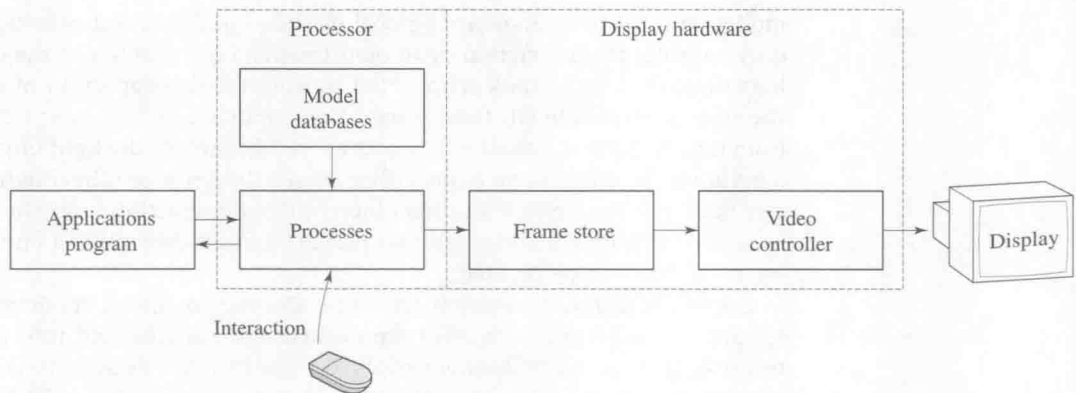


Figure P.1
The main elements of a graphics system.

frame store, a screen buffer or a refresh memory. This information – the discretized computer image – is continually converted by a video controller into a set of horizontal scan lines (a raster) which is then fed to a TV-type monitor. The image is generated by an application program which usually accesses a model or geometric description of an object or objects. The main elements in such a system are shown in Figure P.1. The display hardware to the right of the dotted line can be separate to the processor, but nowadays is usually integrated as in the case of an enhanced PC or a graphics workstation. The raster graphics device overshadows all other hardware developments in the sense that it made possible the display of shaded three-dimensional objects – the single most important theoretical development. The interaction of three-dimensional objects with a light source could be calculated and the effect projected into two-dimensional space and displayed by the device. Such shaded imagery is the foundation of modern computer graphics.

The two early landmark achievements that made shaded imagery possible are the algorithms developed by Gouraud in 1971 and Phong in 1975 enabling easy and fast calculation of the intensities of pixels when shading an object. The Phong technique is still in mainstream use and is undoubtedly responsible for most of the shaded images in computer graphics.

A brief history of shaded imagery

When we look at computer graphics from the viewpoint of its practitioners, we see that since the mid-1970s the developmental motivation has been photo-realism or the pursuit of techniques that make a graphics image of an object or scene indistinguishable from a TV image or photograph. A more recent strand of the application of these techniques is to display information in, for example, medicine, science and engineering.

The foundation of photo-realism is the calculation of light-object interaction and this splits neatly into two fields – the development of local reflection

models and the development of global models. Local or direct reflection models only consider the interaction of an object with a light source as if the object and light were floating in dark space. That is, only the first reflection of light from the object is considered. Global reflection models consider how light reflects from one object and travels onto another. In other words the light impinging on a point on the surface can come either from a light source (direct light) or indirect light that has first hit another object. Global interaction is for the most part an unsolved problem, although two partial solutions, ray tracing and radiosity, are now widely implemented.

Computer graphics research has gone the way of much modern scientific research – early major advances are created and consolidated into a practical technology. Later significant advances seem to be more difficult to achieve. We can say that most images are produced using the Phong local reflection model (first reported in 1975), fewer using ray tracing (first popularized in 1980) and fewer still using radiosity (first reported in 1984). Although there is still much research being carried out in light-scene interaction methodologies much of the current research in computer graphics is concerned more with applications, for example, with such general applications as animation, visualization and virtual reality. In the most important computer graphics publication (the annual SIGGRAPH conference proceedings) there was in 1985 a total of 22 papers concerned with the production techniques of images (rendering, modelling and hardware) compared with 13 on what could loosely be called applications. A decade later in 1995 there were 37 papers on applications and 19 on image production techniques.

Modelling surface reflection with local interaction

Two early advances which went hand-in-hand were the development of hidden surface removal algorithms and shaded imagery – simulating the interaction of an object with a light source. Most of the hidden surface removal research was carried out in the 1970s and nowadays, for general-purpose use, the most common algorithm is the Z-buffer – an approach that is very easy to implement and combine with shading or rendering algorithms.

In shaded imagery the major prop is the Phong reflection model. This is an elegant but completely empirical model that usually ends up with an object reflecting more light than it receives. Its parameters are based on the grossest aspects of reflection of light from a surface. Despite this, it is the most widely used model in computer graphics – responsible for the vast majority of created images. Why is this so? Probably because users find it adequate and it is easy to implement.

Theoretically based reflection models attempt to model reflection more accurately and their parameters have physical meaning – that is they can be measured for a real surface. For example, light reflects differently from an isotropic surface, such as plastic, compared to its behaviour with a non-isotropic surface

such as brushed aluminium and such an effect can be imitated by explicitly modelling the surface characteristics. Such models attempt to imitate the behaviour of light at a 'milliscale' level (where the roughness or surface geometry is still much greater than the wavelength of light). Their purpose is to imitate the material signature – why different materials in reality look different. Alternatively, parameters of a model can be measured on a real surface and used in a simulation. The work into more elaborate or theoretical local reflection models does not seem to have gained any widespread acceptance as far as its implementation in rendering systems is concerned. This may be due to the fact that users do not perceive that the extra processing costs are worth the somewhat marginal improvement in the appearance of the shaded object.

All these models, while attending to the accurate modelling of light from a surface, are local models which means that they only consider the interaction of light with the object as if the object was floating in free space. No object–object interaction is considered and one of the main problems that immediately arises is that shadows – a phenomenon due to global interaction – are not incorporated into the model and have to be calculated by a separate 'add-on' algorithm.

The development of the Phong reflection model spawned research into add-on shadow algorithms and texture mapping, both of which enhanced the appearance of the shaded object and tempered the otherwise 'floating in free space' plastic look of the basic Phong model.

Modelling global interaction

The 1980s saw the development of two significant global models – light reflection models that attempt to evaluate the interaction between objects. Global interaction gives rise to such phenomena as the determination of the intensity of light within a shadow area, the reflection of objects in each other (specular interaction) and a subtle effect known as colour bleeding where the colour from a diffuse surface is transported to another nearby surface (diffuse interaction). The light intensity within a shadow area can only be determined from global interaction. An area in shadow, by definition, cannot receive light directly from a light source but only indirectly from light reflecting from another object. When you see shiny objects in a scene you expect to see in them reflections of other objects. A very shiny surface, such as chromium plate, behaves almost as a mirror taking all its surface detail from its surroundings and distorting this geometrically according to surface curvature.

The successful global models are ray tracing and radiosity. However, in their basic implementation both models only cater for one aspect of global illumination. Ray tracing attends to perfect specular reflection – very shiny objects reflecting in each other, and radiosity models diffuse interaction which is light reflecting off matte surfaces to illuminate other surfaces. Diffuse interaction is common in man-made interiors which tend to have carpets on the floor and matte finishes on the walls. Areas in a room that cannot see the light source are

illuminated by diffuse interaction. Mutually exclusive in the phenomena they model, images created by both methods tend to have identifying 'signatures'. Ray-traced images are notable for perfect recursive reflections and super sharp refraction. Radiosity images are usually of softly-lit interiors and do not contain specular or shiny objects.

Computer graphics is not an exact science. Much research in light-surface interaction in computer graphics proceeds by taking existing physical models and simulating then with a computer graphics algorithm. This may involve much simplification in the original mathematical model so that it can be implemented as a computer graphics algorithm. Ray tracing and radiosity are classic examples of this tendency. Simplifications, which may appear gross to a mathematician, are made by computer graphicists for practical reasons. The reason this process 'works' is that when we look at a synthesized scene we do not generally perceive the simplifications in the mathematics unless they result in visible degeneracies known as aliases. However, most people can easily distinguish a computer graphics image from a photograph. Thus computer graphics have a 'realism' of their own that is a function of the model, and the nearness of the computer graphics image to a photograph of a real scene varies widely according to the method. Photo-realism in computer graphics means the image *looks* real not that it approaches, on a pixel by pixel basis, a photograph. This subjective judgement of computer graphics images somewhat devalues the widely used adjective 'photo-realistic', but there you are. With one or two exceptions very little work has been done on comparing a human's perception of a computer graphics image with, say, a TV image of the equivalent real scene.

Acknowledgements

The author would like to thank the following:

- Lightwork Design Ltd (Sheffield, UK) and Dave Cauldron for providing the facilities to produce the front cover image (model of the Tate Gallery, St Ives, UK) and the renderer, RadioRay.
- Daniel Teece for the images on the back cover which he produced as part of his PhD thesis and which comprise three-dimensional paint strokes interactively applied to a standard polygon model.
- Lee Cooper for producing Figures 6.12, 7.8, 8.7, 8.10, 10.4, 18.1, 18.3, 18.5, 18.6, 18.7, 18.8, 18.9, 18.10, 18.11, 18.12, 18.13, 18.14, 18.16, 18.17 and 18.19 together with the majority of images on the CD-ROM. These were produced using Lightworks Application Development System kindly supplied by Lightwork Design Ltd.
- Mark Fuller for Figure 13.1.
- Steve Maddock for Figures 1.5, 4.9, 8.8, 8.26.
- Agata Opalach for Figure 2.20.
- Klaus de Geuss for Figures 13.10 and 13.11.
- Guy Brown for Figure 16.19.
- Fabio Policarpo for Figure 8.14.
- IMDM University, Hamburg, for Figure 13.3.

In addition the author would like to thank Keith Mansfield, the production staff at Addison-Wesley and Robert Chaundy of Bookstyle for his care with the manuscript.

The publishers are grateful to the following for permission to reproduce copyright material:

Figure 2.1 reproduced with the permission of Viewpoint Digital, Inc.; Figure 2.4 from *Tutorial: Computer Graphics*, 2e (Beatty and Booth, 1982), © 1982 IEEE, The Institute of Electrical and Electronics Engineers, Inc., New York; Figures 2.7 and 2.8 from *Generative Modelling for Computer Graphics and CAD* (Snyder, 1992), Academic

Press, London; Figure 2.20 reproduced with the permission of Agata Opalach; Figure 13.3 from *VOXEL-MAN, Part 1: Brain and Skull*, CD-ROM for UNIX workstations and LINUX PCs, Version 1.1 © Karl-Heinz Höhne and Springer-Verlag GmbH & Co. KG 1996, reproduced with kind permission; Figure 16.14 reproduced with the permission of Steven Seitz; Figure 17.28 from *ACM Transactions on Graphics*, 15:3, July 1996 (Hubbard, 1996), ACM Publications, New York.

Whilst every effort has been made to trace the owners of copyright material, in a few cases this has proved impossible and we take this opportunity to offer our apologies to any copyright holders whose rights we may have unwittingly infringed.

Trademark notice

Apple™ and QuickTime™ are trademarks of Apple Computer, Inc.

Luxo™ is a trademark of Jac Jacobson Industries.

Kodak™ is a trademark of Eastman Kodak Company.

RenderMan™ is a trademark of Pixar Corporation.

VAX™ is a trademark of Digital Equipment Corporation.

3D Dataset™ is a trademark of Viewpoint Digital, Inc.

Contents

Colour plates appear between pages 506 and 507

<i>Preface</i>	xvi
<i>Acknowledgements</i>	xxi
1 Mathematical fundamentals of computer graphics	1
1.1 Manipulating three-dimensional structures	1
1.1.1 Three-dimensional geometry in computer graphics – affine transformations	2
1.1.2 Transformations for changing coordinate systems	8
1.2 Structure-deforming transformations	9
1.3 Vectors and computer graphics	11
1.3.1 Addition of vectors	12
1.3.2 Length of vectors	12
1.3.3 Normal vectors and cross products	12
1.3.4 Normal vectors and dot products	14
1.3.5 Vectors associated with the normal vector reflection	15
1.4 Rays and computer graphics	17
1.4.1 Ray geometry – intersections	17
1.4.2 Intersections – ray–sphere	18
1.4.3 Intersections – ray–convex polygon	19
1.4.4 Intersections – ray–box	21
1.4.5 Intersections – ray–quadric	23
1.4.6 Ray tracing geometry – reflection and refraction	23
1.5 Interpolating properties in the image plane	25
2 Representation and modelling of three-dimensional objects (1)	27
Introduction	27
2.1 Polygonal representation of three-dimensional objects	33
2.1.1 Creating polygonal objects	37

2.1.2	Manual modelling of polygonal objects	38
2.1.3	Automatic generation of polygonal objects	38
2.1.4	Mathematical generation of polygonal objects	39
2.1.5	Procedural polygon mesh objects – fractal objects	44
2.2	Constructive solid geometry (CSG) representation of objects	46
2.3	Space subdivision techniques for object representation	51
2.3.1	Octrees and polygons	53
2.3.2	BSP trees	55
2.3.3	Creating voxel objects	56
2.4	Representing objects with implicit functions	56
2.5	Scene management and object representation	58
2.5.1	Polygon mesh optimization	59
2.6	Summary	64
3	Representation and modelling of three-dimensional objects (2)	66
	Introduction	66
3.1	Bézier curves	69
3.1.1	Joining Bézier curve segments	75
3.1.2	Summary of Bézier curve properties	77
3.2	B-spline representation	78
3.2.1	B-spline curves	78
3.2.2	Uniform B-splines	80
3.2.3	Non-uniform B-splines	84
3.2.4	Summary of B-spline curve properties	90
3.3	Rational curves	90
3.3.1	Rational Bézier curves	91
3.3.2	NURBS	93
3.4	From curves to surfaces	94
3.4.1	Continuity and Bézier patches	98
3.4.2	A Bézier patch object – the Utah teapot	100
3.5	B-spline surface patches	101
3.6	Modelling or creating patch surfaces	106
3.6.1	Cross-sectional or linear axis design example	107
3.6.2	Control polyhedron design – basic technique	110
3.6.3	Creating patch objects by surface fitting	115
3.7	From patches to objects	121
4	Representation and rendering	123
	Introduction	123
4.1	Rendering polygon meshes – a brief overview	124

4.2	Rendering parametric surfaces	125
4.2.1	Rendering directly from the patch descriptions	125
4.2.2	Patch to polygon conversion	128
4.2.3	Object space subdivision	128
4.2.4	Image space subdivision	135
4.3	Rendering a CSG description	138
4.4	Rendering a voxel description	140
4.5	Rendering implicit functions	141
5	The graphics pipeline (1): geometric operations	142
	Introduction	142
5.1	Coordinate spaces in the graphics pipeline	143
5.1.1	Local or modelling coordinate systems	143
5.1.2	World coordinate systems	143
5.1.3	Camera or eye or view coordinate system	143
5.2	Operations carried out in view space	147
5.2.1	Culling or back-face elimination	147
5.2.2	The view volume	147
5.2.3	Three-dimensional screen space	149
5.2.4	View volume and depth	152
5.3	Advanced viewing systems (PHIGS and GKS)	156
5.3.1	Overview of the PHIGS viewing system	157
5.3.2	The view orientation parameters	159
5.3.3	The view mapping parameters	159
5.3.4	The view plane in more detail	162
5.3.5	Implementing a PHIGS-type viewing system	164
6	The graphics pipeline (2): rendering or algorithmic processes	167
	Introduction	167
6.1	Clipping polygons against the view volume	168
6.2	Shading pixels	171
6.2.1	Local reflection models	173
6.2.2	Local reflection models – practical points	177
6.2.3	Local reflection models – light source considerations	179
6.3	Interpolative shading techniques	179
6.3.1	Interpolative shading techniques – Gouraud shading	180
6.3.2	Interpolative shading techniques – Phong shading	181
6.3.3	Renderer shading options	182
6.3.4	Comparison of Gouraud and Phong shading	183
6.4	Rasterization	183
6.4.1	Rasterizing edges	183
6.4.2	Rasterizing polygons	185

6.5	Order of rendering	187
6.6	Hidden surface removal	189
6.6.1	The Z-buffer algorithm	189
6.6.2	Z-buffer and CSG representation	190
6.6.3	Z-buffer and compositing	191
6.6.4	Z-buffer and rendering	192
6.6.5	Scan line Z-buffer	193
6.6.6	Spanning hidden surface removal	193
6.6.7	A spanning scan line algorithm	194
6.6.8	Z-buffer and complex scenes	196
6.6.9	Z-buffer summary	198
6.6.10	BSP trees and hidden surface removal	199
6.7	Multi-pass rendering and accumulation buffers	202
7	Simulating light-object interaction: local reflection models	205
	Introduction	205
7.1	Reflection from a perfect surface	206
7.2	Reflection from an imperfect surface	207
7.3	The bi-directional reflectance distribution function	208
7.4	Diffuse and specular components	211
7.5	Perfect diffuse – empirically spread specular reflection	212
7.6	Physically based specular reflection	213
7.6.1	Modelling the micro-geometry of the surface	214
7.6.2	Shadowing and masking effects	214
7.6.3	Viewing geometry	216
7.6.4	The Fresnel term	216
7.7	Pre-computing BRDFs	219
7.8	Physically based diffuse component	221
8	Mapping techniques	223
	Introduction	223
8.1	Two-dimensional texture maps to polygon mesh objects	228
8.1.1	Inverse mapping by bilinear interpolation	229
8.1.2	Inverse mapping by using an intermediate surface	230
8.2	Two-dimensional texture domain to bi-cubic parametric patch objects	234
8.3	Billboards	235
8.4	Bump mapping	236
8.4.1	A multi-pass technique for bump mapping	238
8.4.2	A pre-calculation technique for bump mapping	239

8.5	Light maps	240
8.6	Environment or reflection mapping	243
8.6.1	Cubic mapping	245
8.6.2	Sphere mapping	247
8.6.3	Environment mapping: comparative points	248
8.6.4	Surface properties and environment mapping	249
8.7	Three-dimensional texture domain techniques	251
8.7.1	Three-dimensional noise	251
8.7.2	Simulating turbulence	252
8.7.3	Three-dimensional texture and animation	254
8.7.4	Three-dimensional light maps	256
8.8	Anti-aliasing and texture mapping	256
8.9	Interactive techniques in texture mapping	260
9	Geometric shadows	263
	Introduction	263
9.1	Properties of shadows used in computer graphics	265
9.2	Simple shadows on a ground plane	265
9.3	Shadow algorithms	267
9.3.1	Shadow algorithms: projecting polygons/scan line	267
9.3.2	Shadow algorithms: shadow volumes	268
9.3.3	Shadow algorithms: derivation of shadow polygons from light source transformations	271
9.3.4	Shadow algorithms: shadow Z-buffer	271
10	Global illumination	275
	Introduction	275
10.1	Global illumination models	276
10.1.1	The rendering equation	277
10.1.2	Radiance, irradiance and the radiance equation	278
10.1.3	Path notation	281
10.2	The evolution of global illumination algorithms	283
10.3	Established algorithms – ray tracing and radiosity	284
10.3.1	Whitted ray tracing	284
10.3.2	Radiosity	286
10.4	Monte Carlo techniques in global illumination	288
10.5	Path tracing	292
10.6	Distributed ray tracing	294
10.7	Two-pass ray tracing	297
10.8	View dependence/independence and multi-pass methods	300

10.9	Caching illumination	301
10.10	Light volumes	303
10.11	Particle tracing and density estimation	304
11	The radiosity method	306
	Introduction	306
11.1	Radiosity theory	308
11.2	Form factor determination	310
11.3	The Gauss–Seidel method	314
11.4	Seeing a partial solution – progressive refinement	315
11.5	Problems with the radiosity method	318
11.6	Artefacts in radiosity images	319
	11.6.1 Hemicube artefacts	319
	11.6.2 Reconstruction artefacts	321
	11.6.3 Meshing artefacts	323
11.7	Meshing strategies	325
	11.7.1 Adaptive or <i>a posteriori</i> meshing	325
	11.7.2 <i>A priori</i> meshing	332
12	Ray tracing strategies	342
	Introduction – Whitted ray tracing	342
12.1	The basic algorithm	343
	12.1.1 Tracing rays – initial considerations	343
	12.1.2 Lighting model components	344
	12.1.3 Shadows	345
	12.1.4 Hidden surface removal	346
12.2	Using recursion to implement ray tracing	347
12.3	The adventures of seven rays – a ray tracing study	350
12.4	Ray tracing polygon objects – interpolation of a normal at an intersection point in a polygon	352
12.5	Efficiency measures in ray tracing	354
	12.5.1 Adaptive depth control	354
	12.5.2 First hit speed up	355
	12.5.3 Bounding objects with simple shapes	355
	12.5.4 Secondary data structures	357
	12.5.5 Ray space subdivision	363
12.6	The use of ray coherence	364
12.7	A historical digression – the optics of the rainbow	367

13 Volume rendering	370
Introduction	370
13.1 Volume rendering and the visualization of volume data	373
13.2 'Semi-transparent gel' option	377
13.2.1 Voxel classification	378
13.2.2 Transforming into the viewing direction	379
13.2.3 Compositing pixels along a ray	379
13.3 Semi-transparent gel plus surfaces	380
13.3.1 Explicit extraction of isosurfaces	382
13.4 Structural considerations in volume rendering algorithms	384
13.4.1 Ray casting (untransformed data)	385
13.4.2 Ray casting (transformed data)	387
13.4.3 Voxel projection method	388
13.5 Perspective projection in volume rendering	390
13.6 Three-dimensional texture and volume rendering	391
14 Anti-aliasing theory and practice	392
Introduction	392
14.1 Aliases and sampling	393
14.2 Jagged edges	397
14.3 Sampling in computer graphics compared with sampling reality	398
14.4 Sampling and reconstruction	400
14.5 A simple comparison	401
14.6 Pre-filtering methods	402
14.7 Supersampling or post-filtering	404
14.8 Non-uniform sampling – some theoretical concepts	406
14.9 The Fourier transform of images	411
15 Colour and computer graphics	418
Introduction	418
15.1 Colour sets in computer imagery	419
15.2 Colour and three-dimensional space	420
15.2.1 RGB space	423
15.2.2 The HSV single hexcone model	424
15.2.3 YIQ space	427
15.3 Colour, information and perceptual spaces	427
15.3.1 CIE XYZ space	429
15.3.2 CIE xyY space	433
15.4 Rendering and colour spaces	435