
Undergraduate Topics in Computer Science

Undergraduate Topics in Computer Science (UTiCS) delivers high-quality instructional content for undergraduates studying in all areas of computing and information science. From core foundational and theoretical material to final-year topics and applications, UTiCS books take a fresh, concise, and modern approach and are ideal for self-study or for a one- or two-semester course. The texts are all authored by established experts in their fields, reviewed by an international advisory board, and contain numerous examples and problems. Many include fully worked solutions.

For further volumes:

<http://www.springer.com/series/7592>

Frank Klawonn

Introduction to Computer Graphics

Using Java 2D and 3D

Second Edition

 Springer

Prof. Dr. Frank Klawonn
Department of Computer Science
Ostfalia University of Applied Sciences
Wolfenbüttel, Germany

Series editor
Ian Mackie

Advisory board

Samson Abramsky, University of Oxford, Oxford, UK
Karin Breitman, Pontifical Catholic University of Rio de Janeiro, Rio de Janeiro, Brazil
Chris Hankin, Imperial College London, London, UK
Dexter Kozen, Cornell University, Ithaca, USA
Andrew Pitts, University of Cambridge, Cambridge, UK
Hanne Riis Nielson, Technical University of Denmark, Kongens Lyngby, Denmark
Steven Skiena, Stony Brook University, Stony Brook, USA
Iain Stewart, University of Durham, Durham, UK

ISSN 1863-7310 Undergraduate Topics in Computer Science
ISBN 978-1-4471-2732-1 e-ISBN 978-1-4471-2733-8
DOI 10.1007/978-1-4471-2733-8
Springer London Dordrecht Heidelberg New York

British Library Cataloguing in Publication Data
A catalogue record for this book is available from the British Library

Library of Congress Control Number: 2012930826

© Springer-Verlag London Limited 2008, 2012

Originally published in the German language by Vieweg+Teubner, 65189 Wiesbaden, Germany, as “Klawonn: Grundkurs Computergrafik mit Java. 3rd edition” © Vieweg+Teubner | Springer Fachmedien Wiesbaden GmbH 2010 Springer Fachmedien is part of Springer Science+Business Media

Apart from any fair dealing for the purposes of research or private study, or criticism or review, as permitted under the Copyright, Designs and Patents Act 1988, this publication may only be reproduced, stored or transmitted, in any form or by any means, with the prior permission in writing of the publishers, or in the case of reprographic reproduction in accordance with the terms of licenses issued by the Copyright Licensing Agency. Enquiries concerning reproduction outside those terms should be sent to the publishers.

The use of registered names, trademarks, etc., in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant laws and regulations and therefore free for general use.

The publisher makes no representation, express or implied, with regard to the accuracy of the information contained in this book and cannot accept any legal responsibility or liability for any errors or omissions that may be made.

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface to the Second Edition

The positive feedback to the first edition of the book and various encouraging requests to extend the list of topics contained in the book have led to this second edition. The new topics that are included in this second edition have been selected in a way to keep the character of the book as a textbook for undergraduate students explaining the basic concepts of computer graphics and enabling the students at the same time to implement and use these concepts in Java 2D and 3D without going into sophisticated details of programming.

Apart from minor changes in the text, the following new topics are included in this second edition of the book.

- Using functions of two variables for surface modelling is explained, especially for the purpose of representing hilly landscapes.
- The integration of text in 3D scenes.
- Particle systems are treated in more detail and an implementation of a simple particle system with a number of different control parameters in Java 3D is described.
- Billboard behaviours are explained for enforcing that objects are always oriented to the viewer that are of specific interest for text in 3D scenes and for particle systems.
- Dynamic surfaces are treated in more detail.
- The concept of level of detail (LOD) for rendering objects in different resolutions depending on the distance of the object to the viewer is explained.

As in the first edition, for all these new concepts, example programs in Java are provided and described.

I hope that this second edition will encourage even more readers to become interested in computer graphics, to understand the basic principles and to apply the concepts in their own programs.

Wolfenbüttel, Germany

Frank Klawonn

Preface

Early computer graphics started as a research and application field that was the domain of only a few experts, for instance in the area of computer aided design (CAD). Nowadays, any person using a personal computer benefits from the developments in computer graphics. Operating systems and application programs with graphical user interfaces (GUIs) belong to the simplest applications of computer graphics. Visualisation techniques, ranging from simple histograms to dynamic 3D animations showing changes of winds or currents over time, use computer graphics in the same manner as popular computer games. Even those who do not use a personal computer might see the results of computer graphics on TV or in cinemas where parts of scenes or even a whole movie might be produced by computer graphics techniques.

Without powerful hardware in the form of fast processors, sufficiently large memory and special graphics cards, most of these applications would not have been possible. In addition to these hardware requirements efficient algorithms as well as programming tools that are easy to use and flexible at the time are required. Nowadays, a standard personal computer is sufficient to generate impressive graphics and animations using freely available programming platforms like OpenGL or Java 3D. In addition to at least an elementary understanding of programming, the use of such platforms also requires basic knowledge about the underlying background, concepts and methods of computer graphics.

Aims of the Book

The aim of this book is to explain the necessary background and principles of computer graphics combined with direct applications in concrete and simple examples. Coupling the theory with the practical examples enables the reader to apply the technical concepts directly and to visually understand what they mean.

Java 2D and Java 3D build the basis for the practical examples. Wherever possible, the introduced concepts and theory of computer graphics are immediately followed by their counterparts in Java 2D and Java 3D. However, the intention of this book is not to provide complete introductions to Java 2D or Java 3D, which would need a multivolume edition themselves without even touching the underlying theoretical concepts of computer graphics.

In order to directly apply computer graphics concepts introduced in this book, the book focusses on the parts of Java 2D and Java 3D that are absolutely relevant for these concepts. Sometimes a simple solution is preferred over the most general one so that not all possible options and additional parameters for an implementation will be discussed. The example programs are kept as simple as possible in order to concentrate on the important concepts and not to disguise them in complex, but more impressive scenes.

There are some selected additional topics—for instance the computation of shadows within computer graphics—that are introduced in the book, although Java 3D does not provide such techniques yet.

Why Java?

There are various reasons for using Java 2D and Java 3D as application platforms. The programming language Java becomes more and more popular in applications and teaching so that extensions like Java 2D/3D seem to be the most obvious choice. Many universities use Java as the introductory programming language, not only in computer science, but also in other areas so that students with a basic knowledge in Java can immediately start to work with Java 2D/3D. Specifically, for multimedia applications Java is very often the language of first choice.

Overview

The first chapters of the book focus on aspects of two-dimensional computer graphics like how to create and draw lines, curves and geometric shapes, handling of colours and techniques for animated graphics.

Chapter 5 and all following chapters cover topics of three-dimensional computer graphics. This includes modelling of 3D objects and scenes, producing images from virtual 3D scenes, animation, interaction, illumination and shading. The last chapter introduces selected special topics, for example special effects like fog, sound effects and stereoscopic viewing.

Guidelines for the Reader

In order to be able to apply the computer graphics concepts introduced in this book, the reader will need only very elementary knowledge of the programming language Java. The example programs in this book use Java 3D but also Java 2D in the first chapters, since two-dimensional representations are essential for computer graphics and the geometrical concepts are easier to understand in two dimensions than in three. The necessary background of Java 2D and Java 3D is included as application sections in this book.

Although the coupling of theory and practice was a main guideline for writing this book, the book can also be used as an introduction to the general concepts of computer graphics without focussing on specific platforms or learning how to use Java 2D or Java 3D. Skipping all sections and subsections containing the word “Java” in their headlines, the book will remain completely self-contained in the sense of a more theoretical basic introduction to computer graphics. For some of the computer graphics concepts introduced in this book it is assumed that the reader has basic knowledge about vectors, matrices and elementary calculus.

Supplemental Resources

Including the complete source code of all mentioned example programs would have led to a thicker, but less readable book. In addition, no one would like to take the burden of typing the source code again in order to run the examples. Therefore, the book itself only contains those relevant excerpts of the source code that are referred to in the text. The complete source code of all example programs and additional programs can be downloaded from the book web site at

<http://public.ostfalia.de/~klawonn/computergraphics>

This online service also provides additional exercises concerning the theoretical background as well programming tasks including sketches of solutions, teaching material in the form of slides and some files that are needed for the example programs. The links mentioned in the appendix and further links to some interesting web sites can also be found at the online service of this book.

Acknowledgements

Over the years, the questions, remarks and proposals of my students had a great influence on how this book was written. I cannot list all of them by name, but I would like to mention at least Daniel Beier, Thomas Weber, Jana Volkmer and especially Dave Bahr for reading the manuscript and their extremely helpful comments. I also would like to thank Katharina Tschumitschew and Gerry Gehrmann for designing the online service of the book and for some 3D models that I could use in my programs. The book was first published in German and without the encouragement and support of Catherine Brett from Springer Verlag in London this English version would have been impossible. Thanks also to Frank Ganz from Springer, who seems to know everything about \LaTeX . My very personal thanks go to my parents and my wife Keiko for their love and for always accepting my sometimes extremely heavy overload of work.

Wolfenbüttel, Germany

Frank Klawonn

Contents

1	Introduction	1
1.1	Application Fields	1
1.2	From a Real Scene to an Image	3
1.3	Organisation of the Book	4
	References	5
2	Basic Principles of Two-Dimensional Graphics	7
2.1	Raster Versus Vector Graphics	7
2.2	The First Java 2D Program	9
2.3	Basic Geometric Objects	13
2.4	Basic Geometric Objects in Java 2D	15
2.5	Geometric Transformations	20
2.6	Homogeneous Coordinates	24
2.7	Applications of Transformations	27
2.8	Geometric Transformations in Java 2D	29
2.9	Animation and Movements Based on Transformations	32
2.10	Movements via Transformations in Java 2D	34
2.11	Interpolators for Continuous Changes	35
2.12	Implementation of Interpolators in Java 2D	38
2.13	Single or Double Precision	39
2.14	Exercises	41
	References	41
3	Drawing Lines and Curves	43
3.1	Lines and Pixel Graphics	43
3.2	The Midpoint Algorithm for Lines	45
3.3	Structural Algorithms	53
3.4	Pixel Densities and Line Styles	55
3.4.1	Different Line Styles with Java 2D	58
3.5	Line Clipping	59
3.6	The Midpoint Algorithm for Circles	65
3.7	Drawing Arbitrary Curves	69
3.8	Antialiasing	70
3.8.1	Antialiasing with Java 2D	72

3.9	Drawing Thick Lines	72
3.9.1	Drawing Thick Lines with Java 2D	73
3.10	Exercises	74
	References	75
4	Areas, Text and Colours	77
4.1	Filling Areas	77
4.2	Buffered Images in Java 2D	80
4.2.1	Double Buffering in Java 2D	81
4.2.2	Loading and Saving of Images with Java 2D	83
4.2.3	Textures in Java 2D	83
4.3	Displaying Text	84
4.4	Text in Java 2D	85
4.5	Grey Images and Intensities	87
4.6	Colour Models	89
4.6.1	Colours in Java 2D	93
4.7	Colour Interpolation	93
4.8	Colour Interpolation with Java 2D	96
4.9	Exercises	98
	References	98
5	Basic Principles of Three-Dimensional Graphics	99
5.1	From a 3D World to a Model	99
5.2	Geometric Transformations	100
5.2.1	Java 3D	104
5.2.2	Geometric Transformations in Java 3D	104
5.3	The Scenegraph	105
5.4	Elementary Geometric Objects in Java 3D	107
5.5	The Scenegraph in Java 3D	108
5.6	Animation and Moving Objects	113
5.7	Animation in Java 3D	116
5.8	Projections	121
5.8.1	Projections in Java 3D	127
5.9	Exercises	128
	References	128
6	Modelling Three-Dimensional Objects	129
6.1	Three-Dimensional Objects and Their Surfaces	129
6.2	Topological Notions	131
6.3	Modelling Techniques	133
6.4	Surface Modelling with Polygons in Java 3D	137
6.5	Importing Geometric Objects into Java 3D	139
6.6	Surfaces as Functions of Two Variables	141
6.6.1	Representation of Landscapes	143
6.6.2	Representation of Functions in Java 3D	145
6.7	Text in 3D	146
6.7.1	Text in Java 3D	146

6.8	Parametric Curves and Freeform Surfaces	148
6.8.1	Parametric Curves	148
6.8.2	Efficient Computation of Polynomials	153
6.8.3	Freeform Surfaces	155
6.9	Normal Vectors for Surfaces	157
6.9.1	Normal Vectors in Java 3D	159
6.10	Exercises	160
	References	160
7	Visible Surface Determination	161
7.1	The Clipping Volume	161
7.1.1	Clipping in Java 3D	163
7.2	Principles of Algorithms for Visible Surface Determination	164
7.2.1	Image-Precision and Object-Precision Algorithms	164
7.2.2	Back-Face Culling	165
7.2.3	Spatial Partitioning	166
7.3	Image-Precision Techniques	167
7.3.1	The z -Buffer Algorithm	167
7.3.2	Scan Line Technique for Edges	170
7.3.3	Ray Casting	171
7.4	Priority Algorithms	174
7.5	Exercises	176
8	Illumination and Shading	177
8.1	Light Sources	178
8.2	Light Sources in Java 3D	181
8.3	Reflection	183
8.4	Shading in Java 3D	190
8.5	Shading	191
8.5.1	Constant and Gouraud Shading in Java 3D	195
8.6	Shadows	195
8.7	Transparency	197
8.7.1	Transparency in Java 3D	199
8.8	Textures	199
8.9	Textures in Java 3D	201
8.10	The Radiosity Model	203
8.11	Ray Tracing	207
8.12	Exercises	208
	References	209
9	Special Effects and Virtual Reality	211
9.1	Fog	211
9.2	Fog in Java 3D	213
9.3	Particle Systems	214
9.4	A Simple Implementation of a Particle System in Java 3D	216
9.5	Dynamic Surfaces	218

9.6	Dynamic Surfaces in Java 3D	221
9.7	Interaction	223
9.8	Interaction in Java 3D	223
9.9	Collision Detection	226
9.10	Collision Detection in Java 3D	227
9.11	Level of Detail (LOD) in Java 3D	232
9.12	Sound Effects	234
9.13	Sound Effects in Java 3D	235
9.14	Stereoscopic Viewing	236
9.15	Exercises	239
	References	239
Appendix A	Useful Links	241
Appendix B	Example Programs	243
Appendix C	References to Java 2D Classes and Methods	247
Appendix D	References to Java 3D Classes and Methods	249
Index	251

List of Figures

Fig. 1.1	From a scene to an image	3
Fig. 2.1	Original image, vector and pixel graphics	8
Fig. 2.2	The tip of an arrow drawn as raster graphics in two different resolutions	9
Fig. 2.3	An alternative representation for pixels	9
Fig. 2.4	The Java 2D API extends AWT	10
Fig. 2.5	The result of the first Java 2D program	11
Fig. 2.6	A self-overlapping, a nonconvex and a convex polygon	14
Fig. 2.7	Definition of quadratic and cubic curves	14
Fig. 2.8	Fitting a cubic curve to a line without sharp bends	14
Fig. 2.9	Union, intersection, difference and symmetric difference of a circle and a rectangle	15
Fig. 2.10	An example for a <code>GeneralPath</code>	17
Fig. 2.11	An example for a rectangle and an ellipse	18
Fig. 2.12	An arc of an ellipse, a segment and an arc with its corresponding chord	19
Fig. 2.13	Scaling applied to a rectangle	22
Fig. 2.14	A rotation applied to a rectangle	22
Fig. 2.15	A shear transformation applied to a rectangle	23
Fig. 2.16	Translation of a rectangle	24
Fig. 2.17	Homogeneous coordinates	25
Fig. 2.18	Differing results on changing the order for the application of a translation and a rotation	26
Fig. 2.19	From world to window coordinates	28
Fig. 2.20	A moving clock with a rotating hand	33
Fig. 2.21	Changing one ellipse to another by convex combinations of transformations	36
Fig. 2.22	Two letters each defined by five points and two quadratic curves	37
Fig. 2.23	Stepwise transformation of two letters into each other	38
Fig. 3.1	Pseudocode for a naïve line drawing algorithm	44
Fig. 3.2	Lines resulting from the naïve line drawing algorithm	44
Fig. 3.3	The two candidates for the next pixel for the line drawing algorithm	46
Fig. 3.4	The new midpoint depending on whether the previously drawn pixel was <i>E</i> or <i>NE</i>	49

Fig. 3.5	Drawing a line with the Bresenham algorithm	52
Fig. 3.6	A repeated pixel pattern for drawing a line on pixel raster	54
Fig. 3.7	Different pixel densities depending on the slope of a line	56
Fig. 3.8	Different line styles	57
Fig. 3.9	Different dash lengths for the same bitmask	58
Fig. 3.10	Examples for different line styles	59
Fig. 3.11	Different cases for line clipping	60
Fig. 3.12	Bit code for Cohen–Sutherland clipping	61
Fig. 3.13	Cohen–Sutherland line clipping	63
Fig. 3.14	Cyrus–Beck line clipping	63
Fig. 3.15	Potential intersection points with the clipping rectangle	64
Fig. 3.16	Finding the pixel where a line enters the clipping rectangle	65
Fig. 3.17	Exploiting symmetry for drawing circles	66
Fig. 3.18	Midpoint algorithm for circles	66
Fig. 3.19	Drawing arbitrary curves	69
Fig. 3.20	Unweighted area sampling	71
Fig. 3.21	Estimation of the area by sampling with refined pixels	71
Fig. 3.22	Weighted area sampling	72
Fig. 3.23	Pixel replication and the moving pen technique	73
Fig. 3.24	Different line endings and joins	73
Fig. 4.1	Odd parity rule	78
Fig. 4.2	Scan line technique for filling polygons	78
Fig. 4.3	A scan line intersecting two vertices of a polygon	79
Fig. 4.4	Filling a polygon can lead to aliasing effects	79
Fig. 4.5	Filling an area with a texture	80
Fig. 4.6	Italic and boldface printing for letters given in raster graphics	85
Fig. 4.7	Grey-level representation based on halftoning for a 2×2 (<i>top line</i>) and on 3×3 pixel matrices (<i>3 bottom lines</i>)	88
Fig. 4.8	Distribution of the energies over the wavelengths for high (<i>left</i>) and low (<i>right</i>) saturation	90
Fig. 4.9	RGB and CMY model	91
Fig. 4.10	HSV model	92
Fig. 4.11	HLS model	93
Fig. 4.12	Compatible triangulations of two images	95
Fig. 4.13	Computation of the interpolated colour of a pixel	96
Fig. 5.1	A right-handed coordinate system	101
Fig. 5.2	A chair constructed with elementary geometric objects	106
Fig. 5.3	A scene composed of various elementary objects	107
Fig. 5.4	The scenegraph for Fig. 5.3	107
Fig. 5.5	The overall scenegraph for Java 3D	109
Fig. 5.6	Excerpt of the scenegraph with dynamic transformations	115
Fig. 5.7	Progression of the Alpha-values	117
Fig. 5.8	Perspective and parallel projection	122
Fig. 5.9	Mapping an arbitrary plane to a plane parallel to the x/y -plane	122
Fig. 5.10	Derivation of the matrix for the perspective projection	123

Fig. 5.11	Vanishing point for perspective projection	126
Fig. 5.12	One-, two- and three-point perspective projections	127
Fig. 6.1	Isolated and dangling edges and faces	130
Fig. 6.2	Triangulation of a polygon	130
Fig. 6.3	Orientation of polygons	131
Fig. 6.4	A tetrahedron	131
Fig. 6.5	A set $M \subset \mathbb{R}^2$ of points, its interior, boundary, closure and regularisation	132
Fig. 6.6	Modelling a three-dimensional object with voxels	134
Fig. 6.7	Recursive partition of an area into squares	134
Fig. 6.8	The quadtree for Fig. 6.7	135
Fig. 6.9	An object that was constructed using elementary geometric objects and set-theoretic operations shown on the <i>right</i>	135
Fig. 6.10	Two objects and their sweep representations	136
Fig. 6.11	Tessellation of the helicopter scene in Fig. 5.3	136
Fig. 6.12	Representation of a sphere with different tessellations	137
Fig. 6.13	The surface defined by the function $z = x \sin(7x) \cos(4y)$	142
Fig. 6.14	Approximation of a surface defined by a function using triangles	143
Fig. 6.15	LOD resolution of a landscape with geometry clipmaps	144
Fig. 6.16	Labels for the axes: A Billboard behaviour is used only for the letter <i>Y</i>	147
Fig. 6.17	Two curves obtained from a surface that is scanned along the coordinate axes	149
Fig. 6.18	An interpolation polynomial of degree 5 defined by the control points $(0, 0)$, $(1, 0)$, $(2, 0)$, $(3, 0)$, $(4, 1)$, $(5, 0)$	150
Fig. 6.19	B-spline with knots P_1 , P_4 , P_7 and inner Bézier points P_2 , P_3 , P_5 , P_6	151
Fig. 6.20	Condition for the inner Bézier points for a twice differentiable, cubic B-spline	152
Fig. 6.21	A parametric freeform surface	156
Fig. 6.22	A net of Bézier points for the definition of a Bézier surface	156
Fig. 6.23	A triangular grid for the definition of a Bézier surface	157
Fig. 6.24	Normal vectors to the original surface in the vertices of an approximating triangle	159
Fig. 6.25	Interpolated and noninterpolated normal vectors	160
Fig. 7.1	The angle α determines the range on the projection plane that corresponds to the width of the display window	162
Fig. 7.2	The clipping volume for parallel projection (<i>top</i>) and perspective projection (<i>bottom</i>)	163
Fig. 7.3	A front face whose normal vector forms an acute angle with the direction of projection and a back face whose normal vector forms an obtuse angle with the direction of projection	166
Fig. 7.4	Partitioning of the clipping volume for image-precision (<i>left</i>) and object-precision algorithms (<i>right</i>)	167
Fig. 7.5	Principle of the z -buffer algorithm	169

Fig. 7.6	Determining the active edges for the scan lines v_1, v_2, v_3, v_4 . . .	171
Fig. 7.7	Ray casting	172
Fig. 7.8	Projection of a polygon to decide whether a point lies within the polygon	173
Fig. 7.9	Supersampling	174
Fig. 7.10	No overlap in the x -coordinate (<i>left</i>) or the y -coordinate (<i>right</i>) . .	175
Fig. 7.11	Does one polygon lie completely in front or behind the plane induced by the other?	175
Fig. 7.12	Determining whether a polygon lies completely in front of the plane induced by the other polygon	176
Fig. 7.13	A case where no correct order exists in which the polygons should be projected	176
Fig. 8.1	Objects with and without illumination and shading effects	178
Fig. 8.2	Cone of light from a spotlight	180
Fig. 8.3	The Warn model for a spotlight	180
Fig. 8.4	The functions $(\cos \gamma)^{64}$, $(\cos \gamma)^8$, $(\cos \gamma)^2$, $\cos \gamma$	181
Fig. 8.5	Light intensity depending on the angle of the light	185
Fig. 8.6	Diffuse reflection	185
Fig. 8.7	Diffuse and specular reflection	186
Fig. 8.8	Computation of ideal specular reflection	187
Fig. 8.9	The halfway vector \mathbf{h} in Phong's model	189
Fig. 8.10	A sphere in different tessellations rendered with flat shading	192
Fig. 8.11	The colour intensity as a function over a triangle for Gouraud shading	193
Fig. 8.12	Scan line technique for the computation of Gouraud shading	194
Fig. 8.13	Interpolated normal vectors for Phong shading	194
Fig. 8.14	Shadow on an object	196
Fig. 8.15	50% (<i>left</i>) and 25% (<i>right</i>) screen-door transparency	198
Fig. 8.16	Using a texture	200
Fig. 8.17	Modelling a mirror by a reflection mapping	200
Fig. 8.18	Bump mapping	201
Fig. 8.19	Illumination among objects	204
Fig. 8.20	Determination of the form factors	205
Fig. 8.21	Determination of the form factors according to Nusselt	206
Fig. 8.22	Recursive ray tracing	208
Fig. 9.1	Linear and exponential fog	213
Fig. 9.2	The sparks of a sparkler as a particle system	215
Fig. 9.3	Intermediate steps for the interpolation between surfaces that are defined by points and triangles with a one-to-one correspondence .	219
Fig. 9.4	Skeleton and skinning	220
Fig. 9.5	Bounding volume in the form of a cube and a sphere	227
Fig. 9.6	Parallax and accommodation for natural and artificial stereoscopic viewing	238
Fig. 9.7	Parallax for stereoscopic viewing	238