Introduction to VISUAL COMPUTING

Core Concepts in Computer Vision, Graphics, and Image Processing

Aditi Majumder
M. Gopi

CPD CERTIFIED
The CPD Certification Service

CRC Press
Taylor & Francis Group
A CHAPMAN & HALL BOOK

Introduction to

VISUAL COMPUTING

Core Concepts in Computer Vision, Graphics, and Image Processing

Taylor & Francis
Taylor & Francis Group
http://taylorandfrancis.com

Introduction to VISUAL COMPUTING

Core Concepts in Computer Vision, Graphics, and Image Processing

Aditi Majumder
University of California, Irvine, USA

M. Gopi
University of California, Irvine, USA

CRC Press
Taylor & Francis Group
Boca Raton London New York

CRC Press is an imprint of the
Taylor & Francis Group, an **informa** business
A CHAPMAN & HALL BOOK

CRC Press
Taylor & Francis Group
6000 Broken Sound Parkway NW, Suite 300
Boca Raton, FL 33487-2742

Printed on acid-free paper
Version Date: 20180112

International Standard Book Number-13: 978-1-4822-4491-5 (Hardback)

Contents

Preface xi

I Fundamentals 1

1 Data 3
- 1.1 Visualization 4
- 1.2 Discretization 4
  - 1.2.1 Sampling 5
  - 1.2.2 Quantization 7
- 1.3 Representation 8
  - 1.3.1 Geometric Data 11
- 1.4 Noise 15
- 1.5 Conclusion 16
- Bibliography 17
- Summary 18
- Exercises 19

2 Techniques 21
- 2.1 Interpolation 21
  - 2.1.1 Linear Interpolation 22
  - 2.1.2 Bilinear Interpolation 23
- 2.2 Geometric Intersections 26
- 2.3 Conclusion 29
- Bibliography 29
- Summary 30
- Exercises 31

II Image Based Visual Computing 33

3 Convolution 35

3.1 Linear Systems . . . 35
3.1.1 Response of a Linear System . . . 36
3.1.2 Properties of Convolution . . . 40
3.2 Linear Filters . . . 41
3.2.1 All, Low, Band and High Pass Filters . . . 43
3.2.2 Designing New Filters . . . 51
3.2.3 2D Filter Separability . . . 54
3.2.4 Correlation and Pattern Matching . . . 56
3.3 Implementation Details . . . 60
3.4 Conclusion . . . 61
Bibliography . . . 62
Summary . . . 63
Exercises . . . 64

4 Spectral Analysis 67
4.1 Discrete Fourier Transform . . . 67
4.1.1 Why Sine and Cosine Waves? . . . 73
4.2 Polar Notation . . . 74
4.2.1 Properties . . . 76
4.2.2 Example Analysis of Signals . . . 77
4.3 Periodicity of Frequency Domain . . . 80
4.4 Aliasing . . . 81
4.5 Extension for 2D Interpretation . . . 83
4.5.1 Effect of Periodicity . . . 85
4.5.2 Notch Filter . . . 85
4.5.3 Example of Aliasing . . . 87
4.6 Duality . . . 89
4.7 Conclusion . . . 92
Bibliography . . . 93
Summary . . . 94
Exercises . . . 95

5 Feature Detection 99
5.1 Edge Detection . . . 100
5.1.1 Edgel Detectors . . . 100
5.1.2 Multi-Resolution Edge Detection . . . 111
5.1.3 Aggregating Edgels . . . 113
5.2 Feature Detection . . . 116
5.3 Other Non-Linear Filters . . . 118
5.4 Conclusion . . . 120
Bibliography . . . 120
Summary . . . 121
Exercises . . . 122

III Geometric Visual Computing 125

6 Geometric Transformations 127
6.1 Homogeneous Coordinates 127
6.2 Linear Transformations 129
6.3 Euclidean and Affine Transformations 131
6.3.1 Translation 131
6.3.2 Rotation 132
6.3.3 Scaling 134
6.3.4 Shear 135
6.3.5 Some Observations 136
6.4 Concatenation of Transformations 136
6.4.1 Scaling About the Center 137
6.4.2 Rotation About an Arbitrary Axis 138
6.5 Coordinate Systems 141
6.5.1 Change of Coordinate Systems 142
6.6 Properties of Concatenation 145
6.6.1 Global vs Local Coordinate System 146
6.7 Projective Transformation 146
6.8 Degrees of Freedom 148
6.9 Non-Linear Transformations 149
6.10 Conclusion 151
Bibliography 151
Summary 152
Exercises 153

7 The Pinhole Camera 157
7.1 The Model 157
7.1.1 Camera Calibration 161
7.1.2 3D Depth Estimation 163
7.1.3 Homography 164
7.2 Considerations in the Practical Camera 168
7.3 Conclusion 172
Bibliography 172
Summary 173
Exercises 174

8 Epipolar Geometry 177
8.1 Background 177
8.2 Correspondences in Multi-View Geometry 179
8.3 Fundamental Matrix 182
8.3.1 Properties 183
8.3.2 Estimating Fundamental Matrix 184

8.3.3 Camera Setup Akin to Two Frontal Eyes 185
8.4 Essential Matrix 187
8.5 Rectification 187
8.6 Applying Epipolar Geometry 190
8.6.1 Depth from Disparity 190
8.6.2 Depth from Optical Flow 192
8.7 Conclusion 194
Bibliography 195
Summary 196
Exercises 197

IV Radiometric Visual Computing 199

9 Light 201
9.1 Radiometry 201
9.1.1 Bidirectional Reflectance Distribution Function 203
9.1.2 Light Transport Equation 205
9.2 Photometry and Color 205
9.2.1 CIE XYZ Color Space 208
9.2.2 Perceptual Organization of CIE XYZ Space 211
9.2.3 Perceptually Uniform Color Spaces 216
9.3 Conclusion 218
Bibliography 218
Summary 220
Exercises 221

10 Color Reproduction 223
10.1 Modeling Additive Color Mixtures 224
10.1.1 Color Gamut of a Device 226
10.1.2 Tone Mapping Operator 230
10.1.3 Intensity Resolution 231
10.1.4 Example Displays 233
10.2 Color Management 235
10.2.1 Gamut Transformation 236
10.2.2 Gamut Matching 237
10.3 Modeling Subtractive Color Mixture 238
10.4 Limitations 240
10.4.1 High Dynamic Range Imaging 240
10.4.2 Multi-Spectral Imaging 243
10.5 Conclusion 245
Bibliography 246
Summary 248

Exercises 249

11 Photometric Processing 251
11.1 Histogram Processing 252
11.1.1 Handling Color Images 255
11.2 Image Composition 257
11.2.1 Image Blending 257
11.2.2 Image Cuts 263
11.3 Photometric Stereo 265
11.3.1 Handling Shadows 270
11.3.2 Computing Illumination Directions 270
11.3.3 Handling Color 271
11.4 Conclusion 272
Bibliography 272
Summary 274
Exercises 275

V Visual Content Synthesis 277

12 The Diverse Domain 279
12.1 Modeling 279
12.2 Processing 281
12.3 Rendering 282
12.4 Application 285
12.5 Conclusion 289
Bibliography 289

13 Interactive Graphics Pipeline 291
13.1 Geometric Transformation of Vertices 293
13.1.1 Model Transformation 293
13.1.2 View Transformation 294
13.1.3 Perspective Projection Transformation 297
13.1.4 Occlusion Resolution 299
13.1.5 Window Coordinate Transformation 304
13.1.6 The Final Transformation 305
13.2 Clipping and Vertex Interpolation of Attributes 305
13.3 Rasterization and Pixel Interpolation of Attributes 311
13.4 Conclusion 313
Bibliography 313
Summary 314
Exercises 315

14 Realism and Performance 317
14.1 Illumination 317
14.2 Shading 321
14.3 Shadows 322
14.4 Texture Mapping 325
14.4.1 Texture to Object Space Mapping 325
14.4.2 Object to Screen Space Mapping 329
14.4.3 Mipmapping 330
14.5 Bump Mapping 332
14.6 Environment Mapping 335
14.7 Transparency 337
14.8 Accumulation Buffer 339
14.9 Back Face Culling 340
14.10 Visibility Culling 342
14.10.1 Bounding Volumes 342
14.10.2 Spatial Subdivision 345
14.10.3 Other Uses 346
14.11 Conclusion 349
Bibliography 350
Summary 351
Exercises 352

15 Graphics Programming 357
15.1 Development of Graphics Processing Unit 357
15.2 Development of Graphics APIs and Libraries 361
15.3 The Modern GPU and CUDA 363
15.3.1 GPU Architecture 363
15.3.2 CUDA Programming Model 364
15.3.3 CUDA Memory Model 366
15.4 Conclusion 366
Bibliography 367
Summary 368

Index 369

Preface

This book is the culmination of over a decade of teaching of a newly designed umbrella course on *visual computing* that would provide students with fundamentals in the different areas of computer graphics, computer vision and image processing. Looking back, this was a very forward looking curriculum which became the launching pad for all computer graphics, computer vision and image processing students at UCI and helped future new faculty hires in this direction to count on this course to provide exposure to fundamentals that are common to all these domains. This course is a core entry-level course in the graduate curriculum providing students the opportunity to explore a larger breadth before moving on to more focused channels of computer graphics, computer vision and/or image processing. It is also being adopted as one of the core courses for our professional masters degree program which began in Fall 2017. Interestingly, the research community has also followed this trend since 2006 when we started to see researchers from one of the domains of computer graphics, computer vision and image processing having strong presence in others leading to a young and dynamic research sub-community that traverses all these domains with equal dexterity. Therefore, having a breadth of knowledge in the general area of visual computing is perceived today as a strength that helps students delve easily into inter-disciplinary domains both within CS and other domains where it is being extensively used.

The inspiration for writing this book came from many instructors and educators who inquired about our visual computing course at UCI, designed a similar course at their home institutions, and were requesting a standard single textbook to cover all the topics. The key exercises that we undertook prior to writing this book were (a) to carefully choose a *lean* set of topics that would provide adequate breadth for an introductory course in visual computing enabling the students to take one course instead of three different courses in CG, CV and IP before deciding on the direction they would like to pursue; (b) to carefully design the *depth* of material in each of these topics so that it can be dealt with nicely during the offering of a single course without being overwhelming; (c) to categorize the topics from the perspective of visual computing in such a manner that students are able to see the common threads that run through these different domains. This exercise led to the organization of the book into five different parts.

1. Part 1: Fundamentals provide an exposure to all kinds of different visual data (e.g. 2D images and videos and 3D geometry) and the core mathematical techniques that are required for their processing in any of the CG, CV or IP domains (e.g. interpolation and linear regression).

2. Part 2: Image Based Visual Computing deals with several fundamental techniques to process 2D images (e.g. convolution, spectral analysis and feature detection) and corresponds to the low level retinal image processing that happens in the eye in the human visual system pathway.

3. Part 3: Geometric Visual Computing deals with the fundamental techniques used to combine the geometric information from multiple eyes creating a 3D interpretation of the object and world around us (e.g. transformations, projective and epipolar geometry). This deals with the higher level processing that happens in the brain that combines information from both the eyes helping us to navigate through the 3D world around us.

4. Part 4: Radiometric Visual Computing deals with the fundamental techniques for processing information arising from the interaction of light with the objects around us. This topic covers both lower and higher level processing in the human visual system that deals with intensity of light (e.g. interpretation of shadows, reflectance, illumination and color properties).

5. Part 5: Visual Content Synthesis presents fundamentals of creating virtual computer generated worlds that mimic all the processing presented in the prior sections.

The book is written for a 16 week long semester course and can be used for both UG and graduate teaching. The recommended timeline for teaching would be to dedicate two weeks for Part 1, three weeks each for Parts 2 and 4, and three and half weeks each for Parts 3 and 5. The exercises following each chapter can be used to provide weekly or biweekly written assignments. The ideal way to provide hands-on implementation experience would be to have one programming assignment accompany each part of the course picking a subset of topics taught in each part based on the expertise level of the students. The decision of making this book independent of any programming language or platform is to enable each instructor to choose the most convenient topics, platforms, and programming language for their assignments based on the resources at hand and the skill set of the audience. Evaluation via two midterms at the end of the 6th and 12th week and a comprehensive final is probably most conducive.

Teaching the material in this book in a 10 week quarter usually poses a challenge. There can be multiple ways to handle this. The easiest way is to increase the number of credits for this course leading to more contact hours to compensate for the reduced number of weeks. The second way is to pare down or divide the content presented in a standard semester long offering of the course. For example, Visual Computing-I can focus on low level visual computing focusing on Chapters 1-5 and 9-10 and the first two sections of Chapter 11 while Visual Computing-II can focus on higher level visual processing and representation focusing on Chapters 6-8, the last section of Chapter 11 and Chapters 12-15. Alternatively, parts of a chapter or complete chapters can be skipped to created a pared down version of the course that avoids reducing the rigor of the concepts taught in the class. Such an approach has been explored in the past in UCI by removing Chapters 8,10,15, and most of Chapter 14 beyond texture mapping. The decision of what to present, what to shorten and what to completely remove resides best with the instructors. The book has been written carefully to minimize dependencies between chapters and sections so that they can be chosen independently by instructors without worrying overtly about dependencies on other parts of the book.

We hope that the material presented in this book and its non-traditional organization inspires instructors to design a visual computing course in their institutions, use this book as a textbook for its offering, and hopefully see an increased interest amongst the students towards the study of the general domain of visual computing. We would like to get feedback from instructors who are using this book as a textbook. Please feel free to write to us about anything you faced while using this book — desired additions, details, or organization. Such feedback will be instrumental towards more refined and better suited subsequent editions of this book.

We acknowledge our colleagues at the University of California at Irvine for their support in designing non-traditional courses leading to experimentation which provided the building blocks for this book. We would like to thank the numerous students who took the Visual Computing course at UCI and the teaching assistants who helped us execute and experiment during different offerings of this course which led to the development and organization of the material presented in this book. We also acknowledge the help rendered by our students, Nitin Agrawal and Zahra Montazeri, in designing and rendering to perfection the various figures used in this book. We deeply appreciate the special efforts of Prof. Shuang Zhao of the University of California, Irvine, Prof. Amy and Bruce Gooch of the University of British Columbia, Dr. David Kirk of nVidia, Prof. Chee Yap of New York University, and Prof. Jan Verschelde of the University of Illinois, Chicago, in providing some of the images in this book on physically based modeling, non-photorealistic rendering, geometric compression and GPU architecture respectively.

Aditi Majumder
Gopi Meenakshisundaram

Author Biographies

Aditi Majumder is Professor in the Department of Computer Science at the University of California, Irvine. She received her PhD from the Department of Computer Science, University of North Carolina at Chapel Hill in 2003. She is originally from Kolkata, India and came to the US in 1996 after completing her bachelors of engineering in Computer Science and Engineering from Jadavpur University, Kolkata.

Her research resides at the junction of computer graphics, vision, visualization and human-computer interaction. Her research focuses on novel displays and cameras exploring new degrees of freedom and quality while keeping them truly a commodity, easily accessible to the common man. She has more than 50 publications in top venues like ACM Siggraph, Eurographics, IEEE Visweek, IEEE Virtual Reality (VR), IEEE Computer Vision and Pattern Recognition (CVPR) including best paper awards in IEEE Visweek, IEEE VR and IEEE PROCAMS. She is the co-author of the book *Practical Multi-Projector Display Design*. She has served as the program or general chair and program committee in several top venues including IEEE Virtual Reality (VR), ACM Virtual Reality Software and Technology (VRST), Eurographics and IEEE Workshop on Projector Systems. She has served as Associate Editor in Computer and Graphics and IEEE Computer Graphics and Applications. She has played a key role in developing the first curved screen multi-projector display being marketed by NEC/Alienware currently and was an advisor at Disney Imagineering for advances in their projection based theme park rides. She received the Faculty Research Incentive Award in 2009 and Faculty Research Midcareer Award in 2011 in the School of Information and Computer Science in UCI. She is the recipient of the NSF CAREER award in 2009 for Ubiquitous Displays Via a Distributed Framework. She was a Givens Associate and was a student fellow at Argonne National Labs from 2001-2003, a Link Foundation Fellow from 2002-2003, and is currently a Senior Member of IEEE.

Gopi Meenakshisundaram is a Professor of Computer Science in the Department of Computer Science, and Associate Dean at the Bren School of Information and Computer Sciences at the University of California, Irvine. He received his BE from Thiagarajar College of Engineering, Madurai, MS from Indian Institute of Science, Bangalore, and PhD from University of North Carolina at Chapel

Hill. His research interests include geometry and topology in computer graphics, massive geometry data management for interactive rendering, and biomedical sensors, data processing, and visualization. His work on representation of manifolds using single triangle strip, hierarchyless simplification of triangulated manifolds, use of redundant representation for big data for interactive rendering, and biomedical image processing have received critical acclaim including best paper awards in two Eurographics conferences and in ICVGIP. He is a gold medalist for academic excellence at Thiagarajar College of Engineering, a recipient of the Excellence in Teaching Award at UCI and a Link Foundation Fellow. He served as the program co-chair and papers co-chair of ACM Interactive 3D Graphics conference in 2012 and 2013 respectively, area chair for ICVGIP in 2010 and 2012, program co-chair for the International Symposium on Visual Computing 2006, an associate editor of the Journal of Graphical Models, a guest editor of IEEE Transactions on Visualization and Computer Graphics and serves in the steering committee of ACM Interactive 3D Graphics.

Part I

Fundamentals

1

Data

In the context of visual computing, data can be thought of as a function that depends on one or more independent variables. For example, audio can be thought of as one dimensional (1D) data that is dependent on the variable time. Thus, it can be represented as $A(t)$ where t denotes time. An image is data that is two dimensional (2D) data dependent on two spatial coordinates x and y and can be denoted as $I(x, y)$. A video is three dimensional (3D) data that is dependent on three variables – two spatial coordinates (x, y) and one temporal coordinate t. It can therefore be denoted by $V(x, y, t)$.

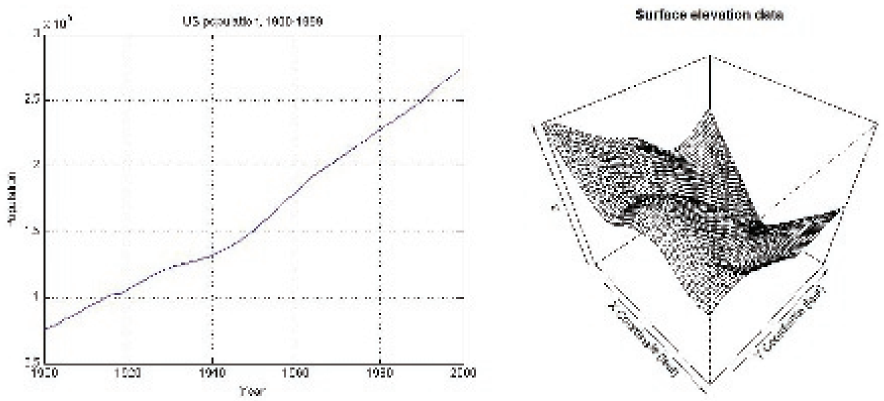


Figure 1.1. Most common visualization of 1D (left) and 2D (right) data. The 1D data shows the population of US (Y-axis) during the 20th century (specified by time in the X-axis) while the 2D data shows the surface elevation (Z-axis) of a geographical region (specified by X and Y-axes). This is often called *height field*.

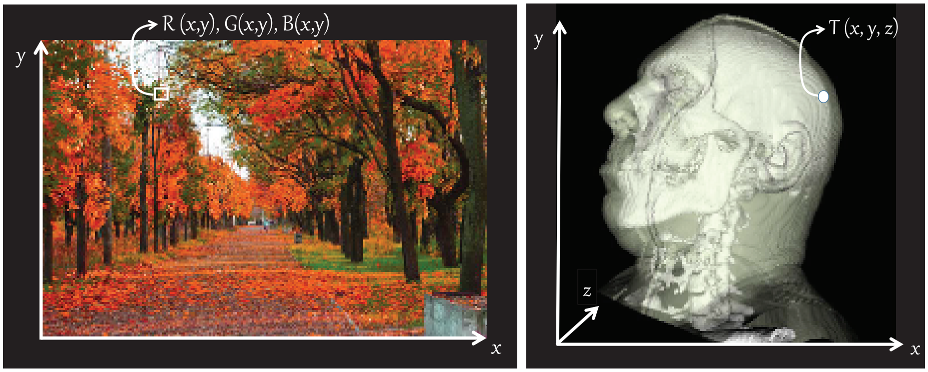


Figure 1.2. Conducive Visualizations: An image is represented as three 2D functions, $R(x, y)$, $G(x, y)$ and $B(x, y)$. But instead of three height fields, a more conducive visualization is where every pixel (x, y) is shown in RGB color (left). Similarly, volume data $T(x, y, z)$ is visualized by depicting the data at every 3D point by its transparency (right).

1.1 Visualization

The simplest visualization of a multi-dimensional data is a traditional plot of the dependent variable with respect to the independent ones, as illustrated in Figure 1.1. For example, such a visualization in 2D is called *height field*. However, as data becomes more complex, such visualization do not suffice due to the inherent inability of humans to visualize geometrical structures beyond three dimensions. Alternative perceptual modalities (e.g. color) are therefore used to encode data. For example, color image comprises of information of three color channels, usually red, green and blue, each dependent on two spatial coordinates (x, y) – $R(x, y)$, $G(x, y)$ and $B(x, y)$. However, often visualizing these three functions together is much more informative that visualizing them as three different height fields. Thus, the ideal visualization is an image where each spatial coordinate is visualized as a color which is also a 3D quantity. Similarly, a 3D volume data $T(x, y, z)$, providing scalar data at each 3D grid point, is visualized in 3D by assigning color or transparency to each grid point computed using a user defined *transfer function* $f(T(x, y, z))$ that is common to the entire data set (See Figure 1.2).

1.2 Discretization

Data exists in nature as a continuous function. For example, the sound we hear changes continuously over time; the dynamic scenes that we see around us also

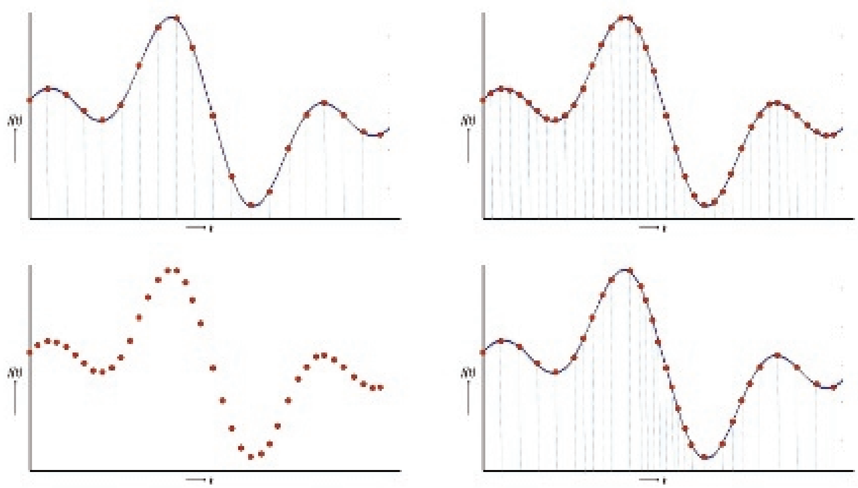


Figure 1.3. This figure illustrates the process of sampling. On top left, the function $f(t)$ (curve in blue) is sampled uniformly. The samples are shown with red dots and the values of t at which the function is sampled is shown by the vertical blue dotted lines. On top right, the same function is sampled at double the density. The corresponding discrete function is shown in the bottom left. On the bottom right, the same function is now sampled non-uniformly i.e. the interval between different values of t at which it is sampled varies.

change continuously with time and space. However, if we have to digitally represent this data, we need to change the continuous function to a discrete one, i.e. a function that is only defined at certain values of the independent variable. This process is called *discretization*. For example, when we discretize an image defined in continuous spatial coordinates (x, y), the values of the corresponding discrete function are only defined at integer locations of (x, y), i.e. pixels.

1.2.1 Sampling

A *sample* is a value (or a set of values) of a continuous function $f(t)$ at a specified value of the independent variable t. Sampling is a process by which one or more samples are extracted from a continuous signal $f(t)$ thereby reducing it to a discrete function $\hat{f}(t)$. The samples can be extracted at equal intervals of the independent variable. This is termed as *uniform* sampling. Note that the *density* of sampling can be changed by changing the interval at which the function is sampled. If the samples are extracted at unequal intervals, then it is termed as *non-uniform* sampling. These are illustrated in Figure 1.3.

The process of getting the continuous function $f(t)$ back from the discrete

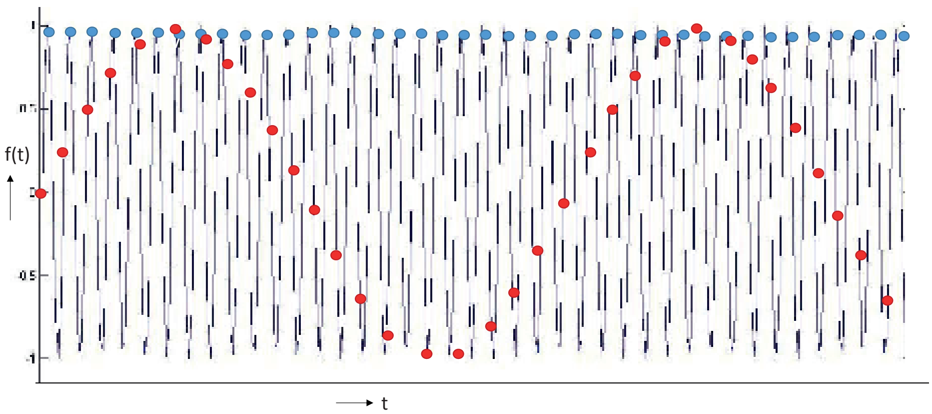


Figure 1.4. This figure illustrates the effect of sampling frequency on reconstruction. Consider the high frequency sine wave shown in blue. Consider two types of sampling shown by the blue and red samples respectively. Note that none of these sample the high frequency sine wave adequately and hence the samples represent sine waves of different frequencies.

function $\hat{f}(t)$ is called *reconstruction*. In order to get an accurate reconstruction, it is important to sample $f(t)$ *adequately* during discretization. For example, in Figure 1.4, a high frequency sine wave (in blue) is sampled in two different ways, both uniformly, shown by the red and blue samples. But in both cases the sampling frequency or rate is not adequate. Hence, a different frequency sine wave is reconstructed – for blue samples a zero frequency sine wave and for red samples a much lower frequency sine wave than the original wave. These incorrectly reconstructed functions are called aliases (for imposters) and the phenomenon is called *aliasing*.

This brings us to the question of *what is adequate sampling frequency*? As it turns out, for sine or cosine waves of frequency f, one has to sample them at a minimum of double the frequency, i.e. $2f$, to assure correct reconstruction. This rate is called the *Nyquist sampling rate*. However, note that the reconstruction is not a process of merely connecting the samples. The reconstruction process is discussed in details in later chapters.

We just discussed adequate sampling for sine and cosine waves. But, *what is adequate sampling for a general signal – not a sine or a cosine wave*? To answer this question, we have to turn to the operation complementary to reconstruction, called *decomposition*. Legendary 19th century mathematician, Fourier, showed that any periodic function $f(t)$ can be decomposed into a number of sine and cosine waves which when added together give the function back. We will revisit Fourier decomposition in greater detail at Chapter 4. For now, it is sufficient to

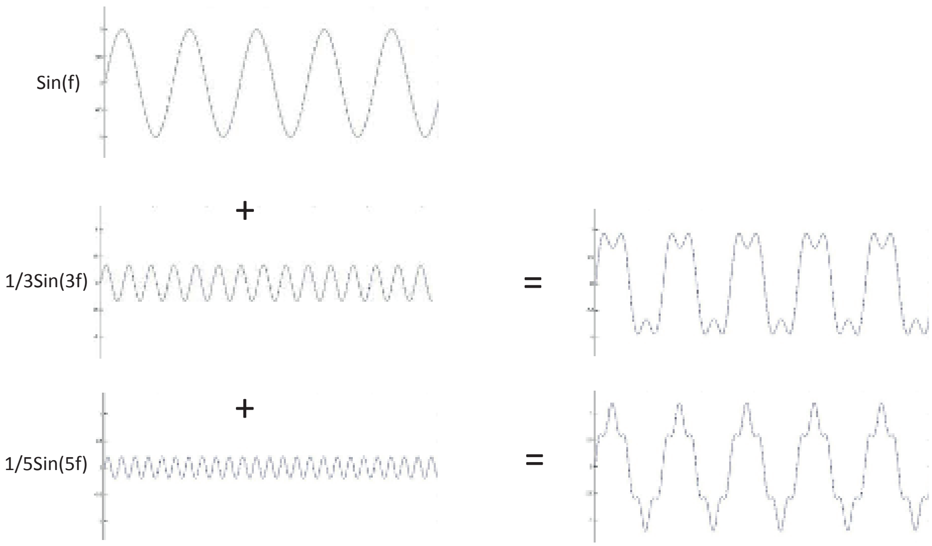


Figure 1.5. This figure illustrates how addition of different frequency sine waves results in the process of generation of general periodic signals.

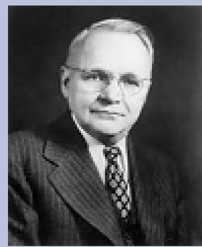
understand that there is a way by which any general signal can be decomposed into a number of sine and cosine waves. An example is shown in Figure 1.5 where different frequency sine waves are added to create new signals. Hence, the adequate sampling rate of a general signal is guided by the highest frequency sine or cosine wave present in it. If the signal is sampled at a rate that is greater than twice the highest frequency sine or cosine wave present in the signal, sampling will be adequate and the signal can be reconstructed. Therefore, the signal in Figure 1.5 has to be sampled at least at a rate of $6f$ to assure a correct reconstruction.

1.2.2 Quantization

A analog or continuous signal can have any value of infinite precision. However, whenever it is converted to digital signal, it can only have a limited set of value. So a range of analog signal values is assigned to one digital value. This process is called *quantization*. The difference between the original value of a signal and its digital value is called the *quantization error*.

The discrete values can be placed at equal intervals resulting in uniform step size in the range of continuous values. Each continuous value is usually assigned the nearest discrete value. Hence, the maximum error is half the step size. This is illustrated in Figure 1.6.

Put a Face to the Name



Harry Theodore Nyquist is considered to be one of the founders of communication theory. He was born to Swedish parents in February 1886 and immigrated to the United States at the age of 18. He received his B.S. and M.S. in electrical engineering from the University of North Dakota in 1914 and 1915 respectively. He received his PhD in physics in 1917 from Yale University. He worked in the Department of Development and Research at AT&T from 1917 to 1934, and continued there when it became Bell Telephone Laboratories until his retirement in 1954. He died in April 1976.

However, human perception is usually not linear. For example, human perception of brightness of light is non-linear, i.e. if the brightness is increased by a factor of 2, its perception increases by less than a factor of 2. In fact, any modality of human perception (e.g. vision, audio, nervous) is known to be non-linear. It has been shown that most human perception modalities follow Steven's power law which says that for input I, the perception P is related by the equation $P \propto I^{\gamma}$. If $\gamma < 1$, as is the case of human response to brightness of light, the response is said to be sub-linear. If $\gamma > 1$, as is the case for human response to electric shock, the response is said to be super-linear.

Due to such non-linear response of the human system, in many cases, a non-uniform step size is desired when converting a continuous signal to digital. For example, in displays, the relationship of the input voltage to the produced brightness needs to be super-linear to compensate for the sub-linear response of the human eye. This function in displays (e.g. projectors, monitors) is commonly termed as the gamma function. When such non-uniform step size is used during the conversion of the continuous signal to digital, the maximum quantization error is half the maximum step size, as illustrated in Figure 1.6.

1.3 Representation

In this section we will discuss data representation in the context of visual computing – namely audio, images, videos and meshes. An analytical representation of data is in the form of a function of one or more independent variables. Audio data $A(t)$, where t denotes time, can be represented as $A(t) = sin(t) + \frac{1}{2}sin(2t)$. However, for digital representation of an arbitrary audio signal, we usually use a 1D array to represent the audio data. From now on, we will distinguish the digital representation from the analog by using $A[t]$ instead of $A(t)$. Note that

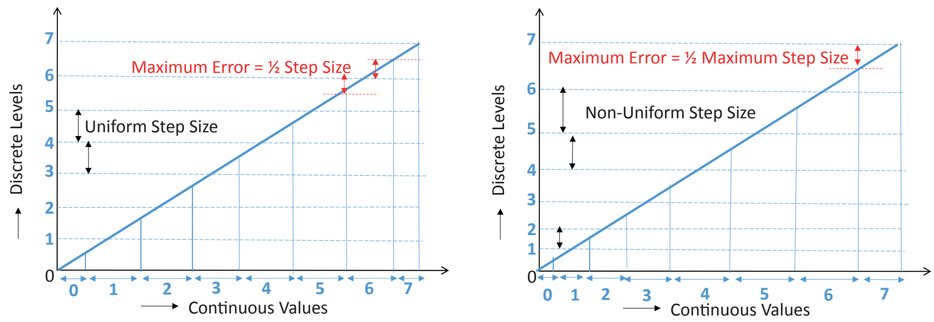


Figure 1.6. This figure illustrates the effect of step size on quantization error. The blue dotted lines show the eight discrete values. Note that these can be distributed at equal intervals resulting in uniform step size throughout the range of continuous values. The intervals can also change to create non-uniform step size. The range of continuous signal values that are assigned a particular discrete value is shown on the independent axis leading to maximum quantization error of half the maximum step size. Hence, for uniform step size, the maximum error is half the uniform step size.

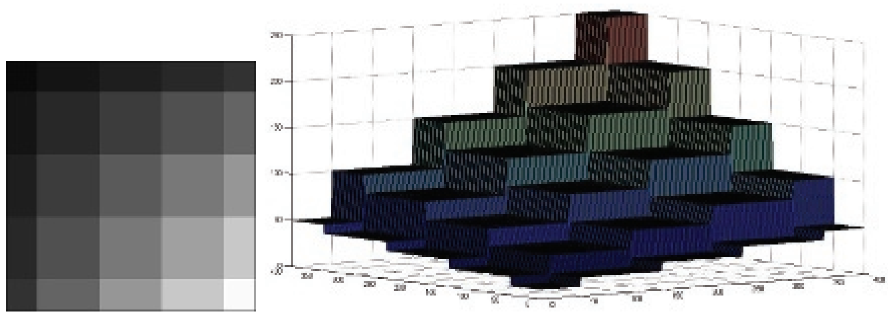


Figure 1.7. This figure illustrates the gray scale image (left) being represented as a height field (right).

representation using an 1D array follows an underlying assumption that the data is *structured*, which in this case means uniformly sampled.

Similarly, a 2D digital grayscale image I is denoted by the 2D array $I[x, y]$ where x, y stands for spatial coordinates. This also assumes structured data. This can be visualized as an image with a grayscale color assigned to every (x, y) coordinates. It can also be visualized as a *height field* in which the height (Z-value) is the grayscale value at every (x, y) coordinate forming a surface (Figure 1.7).

Color images also have multiple channels, typically red, green and blue.

Hence, they are represented by a three dimensional array $I[c, x, y]$ where c denotes the channel, $c \in \{R, G, B\}$. Video involves the additional dimension of time and hence is represented by a four dimensional array $V[t, c, x, y]$. Note that all of these data are structured, which assumes a uniform sampling in each dimension. All these aforementioned representations are called the time or spatial domain representation.

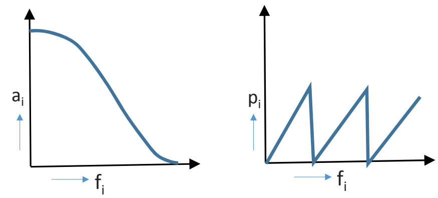


Figure 1.8. Informal representation of the frequency domain response of a 1D signal

An alternate representation, called the *frequency domain representation*, considers the signal as a composition (e.g. linear combination) of a number of more fundamental signals (e.g. sine or cosine waves). Then the signal can be represented by the coefficients of these fundamental signals in the composition that would result in the original signal.. For example, the Fourier transform provides us with a way to find the weights of the sine and cosine waves that form the signal. Since the frequencies of these fundamental signals are predefined based on their sampling rate, the signal can then be represented by a set of coefficients for these waves. In this chapter we will briefly discuss the Fourier transformation, and will revisit this topic in greater detail in Chapter 4.

Let us consider a 1D signal $c(t)$ (e.g. audio). This can be represented as

$$c(t) = \sum_{i=1}^{\infty} a_i Cos(f_i + p_i)$$

where a_i and p_i denote respectively the amplitude and the phase of the constituting cosine waves. Therefore, the frequency domain representation of $c(t)$ is two plots – amplitude plot that shows a_i plotted with respect to f_i and phase plot that shows p_i plotted with respect to f_i. Together they show the amplitude and phase of each wave of frequency f_i. A typical 1D frequency response plot is shown in Figure 1.8. Since higher frequency waves only create the sharp features, they are usually present in very small amplitudes. Hence, most amplitude plots, especially for natural signals, taper away at higher frequencies as shown in 1.8.

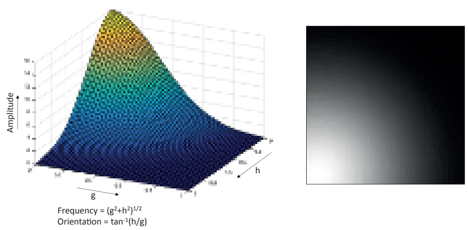


Figure 1.9. Left: Informal representation of the amplitude part of the frequency domain response of a 2D signal. Right: A grayscale representation of the same plot on the left.

Let us now try to extend this concept intuitively to 2D signals (e.g. grayscale image). Note that when considering these waves in 2D, they can now not only differ in frequency f but also in orientation o. A horizontal cosine wave is entirely different than a vertical one even if they have the same frequency. Therefore, the frequency response of 2D signals results in 2D plots where the amplitude/phase are functions of both frequency and orientation. However, understanding a 2D plot whose one axis is frequency and other orientation is very hard for us to comprehend. An easier way to plot these is to use polar coordinates g and h such that frequency f at coordinate (g, h) is given by the length $\sqrt{g^2 + h^2}$ and the orientation is given by the angle $\tan^{-1} \frac{h}{g}$. This means that a circle in (g, h) would provide cosine waves of the same frequency and different orientation and a ray from the origin will provide cosine waves of the same orientation and different frequencies. Figure 1.9 shows an example 2D amplitude plot. Note that here also the higher frequencies have much less amplitude than the lower frequencies given by the radially decreasing values of the plot. Alternatively, the same plot can be visualized as a grayscale image where the amplitude is normalized and plotted as a gray value between black and white (Figure 1.9).

1.3.1 Geometric Data

A geometric entity (e.g. lines, planes or surfaces) can be represented analytically. Alternatively, a discrete representation can also be used. Continuous representations can be implicit, explicit or parametric.

An explicit representation is one where one dependent variable is expressed as a function of all the independent variables and constants. The explicit equation of a 2D line is

$$y = mx + c$$

where m and c are the slope and y-intercept of the line. Similarly, the explicit representation of a 2D quadratic curve can be

$$y = ax^2 + bx + c$$

where a, b and c are the coefficients of the quadratic function representing the curve. Another popular explicit function occuring in physics and signal processing is

$$y = Asin(\omega t + \phi).$$

This represents a sine wave of amplitude A, frequency ω and phase ϕ. Note that an explicit representation allows easy evaluation of the function at different values of the independent variables.

However, more complex functions are sometimes not easy to represent using explicit form. Implicit representations consider a point $\mathbf{p}$ to be of interest if it satisfies an equation $F(\mathbf{p}) = c$, where c is a constant. The implicit equation of a 2D line is

$$ax + by + c = 0,$$

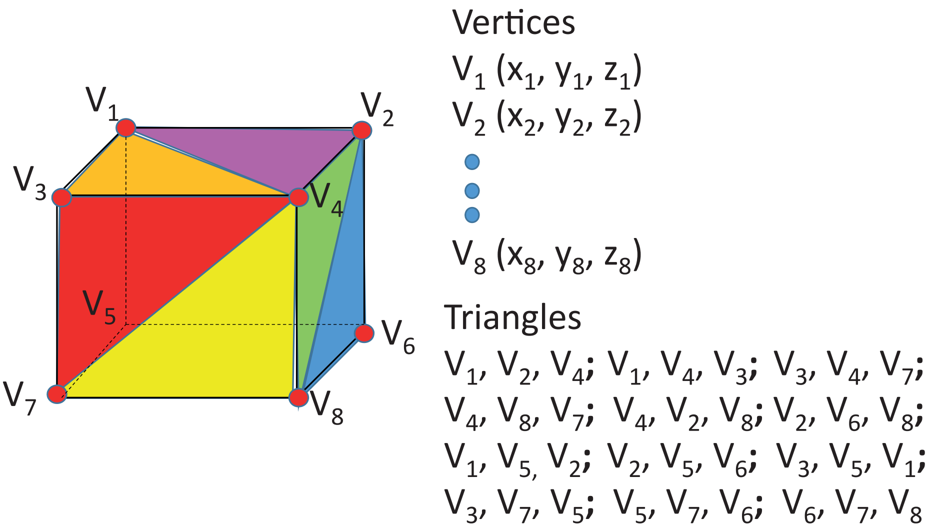


Figure 1.10. This figure shows the representation of a 3D mesh of a cube. It comprises of a list of vertices followed by a list of triangles. Each triangle is described by the indices of the vertices it comprises.

while that of a 3D plane is

$$ax + by + cz + d = 0.$$

Similarly, the implicit equation of a 2D circle is

$$(x - a)^2 + (y - b)^2 = r^2$$

where (a, b) is the center and r is the radius of the circle. The implicit equation of a 3D sphere is

$$(x - a)^2 + (y - b)^2 + (z - c)^2 = r^2$$

where (a, b, c) is the center and r is the radius of the sphere. In explicit function, sometimes dependent and independent variables have to be swapped to represent special cases. For example, it is not possible to represent a vertical line using explicit equation $y = mx + c$ since $m = \infty$. So we need to change x to be a dependent variable to represent this horizontal line $x = m'y + c'$ where $m' = 0$. On the other hand, there are no special cases in implicit function representation. The advantage of an implicit representation is an easy inside or outside test. If $F(\mathbf{p}) < 0$, the point is 'above' or 'outside' the surface and if $F(\mathbf{p}) < 0$, the point is 'below' or 'inside' the surface.

Finally, the parametric equation allows the representation of the function using one or more parameters. For example, a point $\mathbf{p} = L(t)$ on a line segment

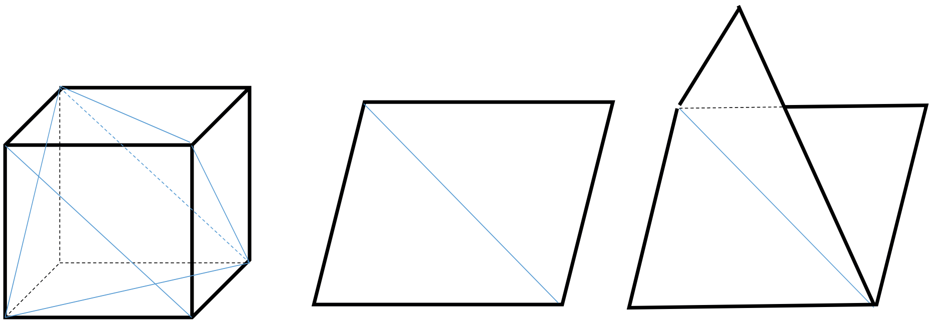


Figure 1.11. This figure illustrates manifold (closed objects), manifolds with boundaries (objects with holes) and non-manifolds (objects with folds and creases.

between two points and P and Q can be represented in the parametric form as

$$L(t) = P + t(Q - P),$$

where the parameter is t and $0 \leq t \leq 1$. Similarly, the parametric equation of a point inside the triangle formed by P, Q and R is given by the two parameter equation given by

$$\mathbf{p} = P + u(Q - P) + v(R - P).$$

where the parameters are u and v such that $0 \leq u, v \leq 1$ and $u + v \leq 1$. The parametric equation allows easy sampling of the parametric space and evaluating any function at these different sampled values.

In a discrete representation, a geometric entity is represented as a collection of other geometric entities as opposed to an analytical equation. For example, a 2D square can be defined by a set of lines embedded in the 2D space; a 3D cube can be defined by a set of quadrilaterals or traingles embedded in the 3D space. Such a representation is called a *mesh*. For example, when using triangles to define a 3D object, we call it a *triangular mesh*. The entities that make up the mesh (e.g. lines, triangles or quadrilaterals) are called the *primitives*.

Though there are many different ways to represent 3D geometry, the most common is a triangular mesh. So, we discuss some key elements of triangular mesh representation here. More details of other geometric representations and their use are presented in later chapters. A triangular mesh is defined by a set of vertices and a set of triangles formed by connecting those vertices. The representation therefore consists of two parts: (a) a list of vertices represented by their 3D coordinates; and (b) a list of triangles each defined by indices of the three vertices of its corners. Figure 1.10 shows an example of the mesh

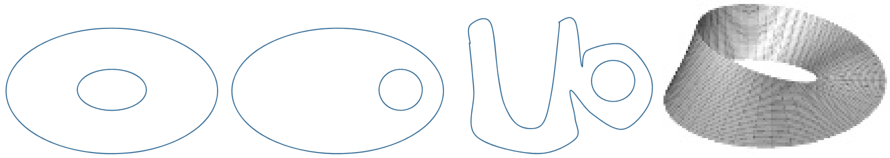


Figure 1.12. Left: This shows how a genus 1 donut is transformed to a genus 1 cup by just changing the geometry. Right: This shows the diagram of a mobius strip.

representation of a simple 3D object, a cube. The coordinates of the vertices define the *geometry* of the mesh. In other words, changing these coordinates changes the geometry of the object. For example, if we want change the cube into a rectangular parallelepiped or a bigger cube, the 3D coordinates of the vertices will be changed. However, note that this will not change the triangle list since the connectivity of the vertices forming the triangles does not change. Hence, the latter is termed as the *topological* property of the mesh. Topology refers to connectivity that remains invariant to changes in geometric properties of the data.

Next, we will define certain geometric and topological properties of meshes, but not in a rigorous fashion. We will give you intuitions and informal definitions. Closed meshes (informally defined to have no holes) have several nice properties in the context of computer graphics operations like morphing, mesh simplification and editing. Such meshes are manifolds where every edge has exactly two incident triangles. A mesh where every edge has one or two incident triangles is called manifold with boundaries. For example, a piece of paper denoted by two triangles where the four edges forming the sides of the paper have only one incident triangle, is a manifold with boundaries. Note that manifold with boundaries are less restrictive than manifolds and hence a superset of manifolds. Meshes where edges can have more than two incident triangles are called non-manifolds. Note that non-manifolds are a superset of manifolds with boundaries. Figure 1.11 illustrate this.

Meshes can be defined with geometric properties or attributes. In Figure 1.10 each vertex has 3D spatial coordinates. In addition to this basic information, each vertex can have RGB color, normal vectors, or 2D coordinates of an image to be pasted on the mesh (formally known as texture coordinates), or any other vertex-based attribute that is useful for the given application. Topological properties are properties that do not change with change in geometric properties. For mesh processing, a few topological properties are very important. First is *Euler characteristics* e defined as $V - E + F$ where V is the number of vertices, E is the number of edges, and F is the number of faces (not necessarily triangular) of the mesh. Note that if you change the cube to a parallelepiped by changing the position of the vertices which is a geometric property, e does not change.

Essentially e may change only when the object undergoes some change in the mesh connectivity. *Genus* of a mesh is defined as the number of handles. For example, a sphere has a genus zero, a donut has a genus 1 and a double donut has a genus 2. One will need to change the topology of the mesh to change from one genus to another while only geometric changes are sufficient to change one object to another with same genus (Figure 1.12). Finally, a mesh is not orientable if you start walking on the top of the mesh and end up in its backside. An example of a non-orientable mesh is the mobius strip (Figure 1.12).

Fun Facts

A non-orientable surface that has been intriguing to topologists is the Klein bottle. Unlike a mobius strip, it does not have any boundary. It is what you get when you put two mobius strips together. The Klein bottle was first described in 1882 by the German mathematician Felix Klein. It cannot be embedded in 3D space, only in 4D space. It is hard to say how much water Klein bottles would hold, they contain themselves when embedded in 4D space! This has not stopped people from trying to embed them in 3D however, and there are some beautifully-made representations on display at the London Science Museum!

We have so far only considered triangular primitives for meshes. Though other primitives can be used (e.g. six quadrilaterals instead of 12 triangles for mesh representation of a cube), triangles are preferred for various reasons. First, triangles are always planar since three non-collinear points define a plane. Hence, modeling packages do not need to assure that a surface fits the vertices when they output the mesh representation. Second, as we will see in the next chapter, in computer graphics it is important to find out the attributes or properties of points lying inside a primitive from the properties at its vertices via techniques called interpolation where triangular primitives hold a great advantage.

1.4 Noise

Any discussion on data cannot be complete without discussing noise. Noise can be caused due to several factors like mechanical imprecision, sensor imprecision (e.g. occasional always-dead or always-live pixels) and so on. The origins of noise in different systems are different. It is best described as addition of random values as random locations in the data. In this chapter we will discuss some common types of noise.

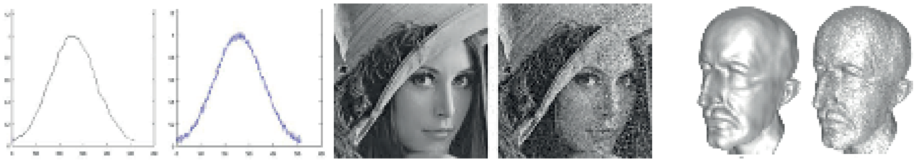


Figure 1.13. This figure illustrates random noise in 1D audio data (left), 2D image data (middle), and 3D surface data (right). In each example, the clean data is shown on the left and the corresponding noisy data is shown on the right.

The most common and general kind of noise is what we call random noise i.e. addition of small random values at any location of the data. Figure 13 shows some examples. A common technique to reduce such noise in data is what we call low pass filtering and it will be dealt with in detail in Chapter 3.

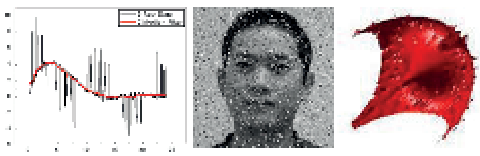


Figure 1.14. This shows the outliers or salt and pepper noise in 1D (left), 2D (middle) and 3D (right) data. On the left, we show the effect of a median filter in removing the outliers in red.

Another common type of noise originates from having outliers in the data i.e. samples which clearly cannot belong to the data. For example, in a camera some sensor pixels may be dead making thereby blocking or allowing all the light providing pixels that are always either black or white respectively. The locations of such pixels may be random. In the specific case of 2D images, this noise is called salt and pepper noise (see Figure 1.14). Such outliers are handled adequately by median filters or other order statistics filters. We will see some of these in Chapter 5.

Finally, some noise may look random in the spatial domain but can be isolated to a few frequencies in the spectral domain. An example of such noise is shown in Figure 1.15. Such noise can be removed by applying a filter in the frequency domain called the notch filter and we are going to talk about that in detail in Chapter 4.

1.5 Conclusion

In this chapter we discussed the fundamentals of representing and visualizing different kinds of visual data like images, 3D surfaces and point clouds. We also learned about two alternate representations of data in the spatial/time domain and frequency domain. We talked about practical issues involving noise in data

Figure 1.15. This figure shows the frequency domain noise that can be removed or reduced by notch filters.

and how it needs to be handled on a case by case basis. Here are some references for familiarizing yourself for some advanced concepts. [Ware 04] explores in details all about information visualization. [Goldstein 10] provides an excellent first reading for topics related to sensation and human perception. The chapter on Data Structures for 3D graphics in [Ferguson 01] provides a detailed description of representation of 3D models. The chapter on noise on [Gonzalez and Woods 06] provides a very detailed treatise on noise that is worth reading.

Bibliography

[Ferguson 01] R. Stuart Ferguson. *Practical Algorithms for 3D Computer Graphics.* A. K. Peters, 2001.

[Goldstein 10] Bruce E. Goldstein. *Sensation and Perception.* Thomas Wadsworth, 2010.

[Gonzalez and Woods 06] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing (3rd Edition).* Prentice-Hall, Inc., 2006.

[Ware 04] Colin Ware. *Information Visualization: Perception for Design.* Morgan Kaufmann Publishers Inc., 2004.

Summary: Do you know these concepts?

- Height Field
- Discretization
- Sampling and Nyquist Sampling Theorem
- Decomposition and Reconstruction
- Aliasing
- Quantization
- Time and frequency domain representation
- Mesh - Geometry and Topology
- Manifold, manifold with boundaries and non-manifolds
- Euler characteristics, genus, orientability
- Random Noise
- Salt and Pepper Noise

Exercises

1. Consider an 8-bit grayscale image $I(x, y)$ whose size is 256×256. Each column of the image has the same gray value which starts from 0 for the left most column and increases by 1 as we sweep from left to right. What kind of shape does the height field of this image form? Find its equation?

2. Consider a height field $H(x, y)$ of size 256×256 given by the function $H(x, y) = (x \mod 16) * 16$. What kind of shape would this height field have? How many gray levels would this image have? Create a table to show the percentage of pixels that belong to each of these gray levels.

3. Consider a gray scale spatial function $A(x, y)$ which does not vary in the y-direction but form a sine wave as we go from left to right in x-direction making 50 cycles. What will be the minimum horizontal resolution of an digital image that can sample this function adequately? Consider another function $B(x, y)$ formed by rotating A about the axis perpendicular to the plane formed by x and y. Now consider the function formed by adding A and B. What is the minimum horizontal and vertical resolution of the image required to sample $A + B$ adequately?

4. Consider an object moving at 60 units per second. How many frames per second video is required to adequately capture this motion? What kind of artifact would you expect if the frame rate is lesser than this desired rate? What is this artifact more commonly known as?

5. The image of your TV looks washed out. The technician says that the intensity response curve of the TV is linear and hence the problem. To correct the problem, he has to make it non-linear. Why? What kind of non-linear response do you think he will put in?

6. If the number of bits used for representing the color of each pixel is increased quantization error is reduced. Justify this statement.

7. Can quantization be explained as an artifact of insufficient sampling? Justify your answer.

8. Your TV has three channels – R, G and B. However one of these channels is broken and now you can only see blacks and purples. Which channel is broken?

9. A 1D function contains all the harmonics of the sine wave that makes 1 cycle with a spatial span of 1 unit. Choose the correct answer.

   (a) The amplitude plot of the frequency domain response of this function is a (i) a sine wave; (ii) a horizontal line; (iii) a comb function.

(b) The phase plot of the frequency domain response of this function is a (i) a sine wave; (ii) a horizontal line; (iii) a comb function.

10. What is the euler characteristics of a cube represented by six planar quadrilaterals. Euler characteristics of an object are related to its genus by the formula $e = 2 - 2g$. Can you derive the genus of a sphere from the Euler characteristics of a cube? If so, how?

11. Topologically, a cube is an approximation of a sphere using quadrilateral faces. In such a cube, all vertices have degree three. It is claimed that one can construct an approximation of a sphere using quadrilaterals where each vertex has degree 4. Prove or disprove this claim.

12. Objects like spheres are usually approximated in computer graphics by simpler objects made of flat polygons. Start with a regular tetrahedron constructed from four triangles. Derive one or more methods to obtain a close approximation of a sphere based on subdividing each face of the tetrahedron recursively using the same geometric operation. Does these constructions change the topological properties of the sphere? Can you think of some criteria to evaluate the quality of these constructions?

13. Match the noisy images in the top row with the filters that will remove the noise in the bottom row.

Notch Filter   Low Pass Filter   Median Filter

14. Consider the mesh representing a pyramid with a quadrilateral as base and four triangles attached to each of its sides to form the structure of the pyramid. Find its Euler characteristics and genus.

Fundamentals

R. Stuart Ferguson . Practical Algorithms for 3D Computer Graphics. A. K. Peters , 2001.
Bruce E . Goldstein. Thomas Wadsworth : Sensation and Perception; 2010.
Rafael C. Gonzalez and Richard E. Woods .Digital Image Processing (3rd Edition). Prentice-Hall,, 2006.
Colin Ware. Information Visualization: Perception for Design. Morgan Kaufmann Publishers Inc., 2004.
Eric Lengyel . Mathematics for 3D Game Programming and Computer Graphics. Charles River Media Inc., 2002.
Frank Nielsen and Rajendra Uhatia. Matrix Information Geometry. Springer Verlag Uerlin Heidelberg, 2013.
Edward Uarry Saff and Arthur David Snider. Fundamentals of Matrix Analysis with Applications. John Wiley and Sons; 2015.

Data

R. Stuart Ferguson . Practical Algorithms for 3D Computer Graphics. A. K. Peters , 2001.
Bruce E . Goldstein. Thomas Wadsworth : Sensation and Perception; 2010.
Rafael C. Gonzalez and Richard E. Woods .Digital Image Processing (3rd Edition). Prentice-Hall,, 2006.
Colin Ware. Information Visualization: Perception for Design. Morgan Kaufmann Publishers Inc., 2004.

Techniques

Eric Lengyel . Mathematics for 3D Game Programming and Computer Graphics. Charles River Media Inc., 2002.
Frank Nielsen and Rajendra Uhatia. Matrix Information Geometry. Springer Verlag Uerlin Heidelberg, 2013.
Edward Uarry Saff and Arthur David Snider. Fundamentals of Matrix Analysis with Applications. John Wiley and Sons; 2015.

Image Based Visual Computing

Rafael C. Gonzalez and Richard E. Woods. Digital Image Processing (3rd Edition). Prentice-Hall, Inc., 2006.
William, K. Pratt. Digital Image Processing. John Wiley and Sons, 2007.
Folland, Gerald B. . Fourier Analysis and Its Applications (Pure and Applied Undergraduate Texts). Pacific Grove: California and Wadsworth & Brooks/Cole Advanced Books and Software, 2009.
David, W. . Kammler. A First Course in Fourier Analysis: Cambridge University Press, 2008.
Morrison, Norman (1994). Introduction to Fourier Analysis. John Wiley and Sons.
John, G. . Proakis and Dimitris K Manolakis. Digital Signal Processing: Prentice Hall, 2006.
Steven, W. . Smith. The Scientist ty Engineers Guide to Digital Signal Processing: California Technical Publishing, 1997.
Murray Spiegel. Schaums Outline of Fourier Analysis with Applications to Boundary Value Problems. Mcgraw Hill, 1974.
Elias, M. Stein and Rami Shakarchi. Fourier Analysis: An Introduction. Princeton University Press, 2003.
Tolstov, Georgio P. . Fourier Series (Dover Books on Mathematics). Prentice Hall, 1976.
David A. Forsyth and Jean Ponce . Computer Vision: A Modern Approach. Pearson, 2011.
David G. Lowe . Distinctive Image Features from Scale-Invariant Keypoints International Journal Computer Vision 60:2 (2004), 91-110.
Simon J. D. Prince . Computer Vision: Models, Learning, and Inference. Cambridge University Press, 2012.

Convolution

Rafael C. Gonzalez and Richard E. Woods. Digital Image Processing (3rd Edition). Prentice-Hall, Inc., 2006.
William, K. Pratt. Digital Image Processing. John Wiley and Sons, 2007.

Spectral Analysis

Folland, Gerald B. . Fourier Analysis and Its Applications (Pure and Applied Undergraduate Texts). Pacific Grove: California and Wadsworth & Brooks/Cole Advanced Books and Software, 2009.
David, W. . Kammler. A First Course in Fourier Analysis: Cambridge University Press, 2008.
Morrison, Norman (1994). Introduction to Fourier Analysis. John Wiley and Sons.
John, G. . Proakis and Dimitris K Manolakis. Digital Signal Processing: Prentice Hall, 2006.
Steven, W. . Smith. The Scientist ty Engineers Guide to Digital Signal Processing: California Technical Publishing, 1997.
Murray Spiegel. Schaums Outline of Fourier Analysis with Applications to Boundary Value Problems. Mcgraw Hill, 1974.
Elias, M. Stein and Rami Shakarchi. Fourier Analysis: An Introduction. Princeton University Press, 2003.
Tolstov, Georgio P. . Fourier Series (Dover Books on Mathematics). Prentice Hall, 1976.

Feature Detection

David A. Forsyth and Jean Ponce . Computer Vision: A Modern Approach. Pearson, 2011.
David G. Lowe . Distinctive Image Features from Scale-Invariant Keypoints International Journal Computer Vision 60:2 (2004), 91-110.
Simon J. D. Prince . Computer Vision: Models, Learning, and Inference. Cambridge University Press, 2012.

Geometric Visual Computing

Faugeras, Olivier . Three-dimensional Computer Vision: A Geometric Viewpoint. MIT Press, 1993.
John F. Hughes, Andries van Dam, Morgan McGuire, David F. Sklar, James D. Foley, Steven K. Feiner, and Kurt Akeley. Computer Graphics: Principles and Practice (3rd ed Addison-Wesley Professional, 2013.
Peter Shirley and Steve Marschner. Fundamentals of Computer Graphics, Third edition. A. K. Peters, Ltd., 2009.
Faugeras, Olivier , 1993. Three-dimensional Computer Vision: A Geometric Viewpoint. MIT Press, 1993.
Lukac, Rastislav , 2010. Computational Photography: Methods and Applications. 1st ed. CRC Press Inc,. 2010.
Szeliski, Richard , 2010. Computer Vision: Algorithms and Applications. New York Inc: Springer-Verlag.
Hartley Richard , Zisserman Andrew . Multiple View Geometry in Computer Vision. Cambridge University Press; 2003.
Leonard Mcmillan and Gary Bishop . Plenoptic Modeling: An Image-based Rendering System. In Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques (SIG-GRAPH), pp. 39-46, 1995.
Heung Yeung Shum. Shing Chow Chan, and Sing Bing Kang. Image Based Rendering: Springer; 2007.
Szeliski Richard . Computer Vision: Algorithms and Applications. New York Inc: Springer-Verlag; 2010.

Geometric Transformations

Faugeras, Olivier . Three-dimensional Computer Vision: A Geometric Viewpoint. MIT Press, 1993.
John F. Hughes, Andries van Dam, Morgan McGuire, David F. Sklar, James D. Foley, Steven K. Feiner, and Kurt Akeley. Computer Graphics: Principles and Practice (3rd ed Addison-Wesley Professional, 2013.
Peter Shirley and Steve Marschner. Fundamentals of Computer Graphics, Third edition. A. K. Peters, Ltd., 2009.

The Pinhole Camera

Faugeras, Olivier , 1993. Three-dimensional Computer Vision: A Geometric Viewpoint. MIT Press, 1993.
Lukac, Rastislav , 2010. Computational Photography: Methods and Applications. 1st ed. CRC Press Inc,. 2010.
Szeliski, Richard , 2010. Computer Vision: Algorithms and Applications. New York Inc: Springer-Verlag.

Epipolar geometry

Hartley Richard , Zisserman Andrew . Multiple View Geometry in Computer Vision. Cambridge University Press; 2003.
Leonard Mcmillan and Gary Bishop . Plenoptic Modeling: An Image-based Rendering System. In Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques (SIG-GRAPH), pp. 39-46, 1995.
Heung Yeung Shum. Shing Chow Chan, and Sing Bing Kang. Image Based Rendering: Springer; 2007.
Szeliski Richard . Computer Vision: Algorithms and Applications. New York Inc: Springer-Verlag; 2010.

Radiometric Visual Computing

James Arvo and David Kirk . Particle transport and image synthesis SIGGRAPH Computer Graphics, pp. 6366.
Immel David S , Cohen Michael F , Greenberg Donald P . A Radiosity Method for Non-diffuse Environments SIGGRAPH. Computer Graphics. 1986;20(4):133142.
Kajiya James T . The Rendering Equation SIGGRAPH . Computer Graphics. 1986;20(4):143150.
Erik Reinhard , Erum Arif Khan , Ahmet Oguz Akyz , and Garrett M. Johnson . Color Imaging: Fundamentals and Applications. A. K. Peters, Ltd., 2008.
Maureen C . Stone. A K Peters: A Field Guide to Digital Color; 2003.
Roy S. Berns, FredW. Billmeyer, and Max Saltzman. Billmeyer and Saltzman's Principles of Color Technology. Wiley Interscience, 2000.
Paul E. Debevec and Jitendra Malik. Recovering High Dynamic Range Radiance Maps from Photographs." In Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '97, pp. 369-378, 1997.
O. Gallo, N. Gelfand, W. Chen, M. Tico, and K. Pulli. Artifact-free High Dynamic Range Imaging." IEEE International Conference on Computational Photography (ICCP).

M. Gupta, D. Iso, and S.K. Nayar. Fibonacci Exposure Bracketing for High Dynamic Range Imaging." pp. 1-8.
R. W. G. Hunt. The Reproduction of Color. Fountain Press, 1995.
G. W. Larson , H. Rushmeier , and C. Piatko . A Visibility Matching Tone Reproduction Operator for High Dynamic Range Scenes." IEEE Transactions on Visualization and Computer Graphics 3:4.
Yuqi Li , Aditi Majumder , Dongming Lu , and Meenakshisundaram Gopi . Content-Independent Multi-Spectral Display Using Superimposed Projections." Computer Graphics Forum.
Margaret Livingstone and H. David Hubel . Vision and Art : The Biology of Seeing. Harry N Abrams, 2002.
Stephen, E. Palmer. Vision Science: MIT Press, 1999.
Erik Reinhard , Greg Ward , Sumanta Pattanaik , and Paul Debevec . High Dynamic Range Imaging: Acquisition, Display, and Image-Based Lighting (The Morgan Kaufmann Series in Computer Graphics). Morgan Kaufmann Publishers Inc., 2005.
Helge Seetzen , W. Heidrich , W. Stuezlinger , G. Ward , L. Whitehead , M. Trentacoste , A. Ghosh , and A. Vorozcovs . High Dynamic Range Display Systems." ACM Transactions on Graphics (special issue SIGGRAPH).
R. Shogenji , Y. Kitamura , K. Yamada , S. Miyatake , and J. Tanida . Multispectral imaging using compact compound optics." Opt. Exp., p. 16431655.
Peterescu S. Nanu F. Capata A. Corcoran P. Susanu, G. RGBW Sensor Array." US Patent 2009/0,167,893.
J. Tumblin and G. Turk . Low Curvature Image Simplifiers (LCIS): A Boundary Hierarchy for Detail-Preserving Contrast Reduction." pp. 83-90.
F. Yasuma , T. Mitsunaga , D. Iso , and S.K. Nayar . Generalized Assorted Pixel Camera: Post-Capture Control of Resolution, Dynamic Range and Spectrum." IEEE Transactions on Image Processing 99.
Basri Ronen , Jacobs David , Kemelmacher Ira . (Photometric Stereo with General. Unknown Lighting International Journal of Computer Vision. 2007;72:239257.
M. Brown and D. G. Lowe . Recognising Panoramas In Proceedings of the Ninth IEEE International Conference on Computer VisionVolume 2 , 2003.
Peter J. Burt and Edward H. Adelson . A Multiresolution Spline with Application to Image Mosaics ACM Trans. Graph . 2:4.
Alexei A. Efros and William T. Freeman . Image Quilting for Texture Synthesis and Transfer In Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 01, 2001.
Bruce E . Goldstein. Thomas Wadsworth: Sensation and Perception; 2010.
Vivek Kwatra , Arno Schdl , Irfan Essa , Greg Turk , and Aaron Bobick . Graphcut Textures: Image and Video Synthesis Using Graph Cuts ACM Transactions on Graphics , SIGGRAPH 2003 22:3 (2003), 277286.
Aditi Majumder and Sandy Irani. Perception-based Contrast Enhancement of Images ACM Trans. Appl. Percept. 4:3.
Prez Patrick , Gangnet Michel , Blake Andrew . Poisson Image Editing ACM Trans . Graph. 2003;22(3):313318.
Lun Wu , Arvind Ganesh , Boxin Shi , Yasuyuki Matsushita , Yongtian Wang , and Yi Ma . Robust Photometric Stereo via Low-rank Matrix Completion and Recovery pp. 703717.

Light

James Arvo and David Kirk . Particle transport and image synthesis SIGGRAPH Computer Graphics, pp. 6366.
Immel David S , Cohen Michael F , Greenberg Donald P . A Radiosity Method for Non-diffuse Environments SIGGRAPH. Computer Graphics. 1986;20(4):133142.
Kajiya James T . The Rendering Equation SIGGRAPH . Computer Graphics. 1986;20(4):143150.
Erik Reinhard , Erum Arif Khan , Ahmet Oguz Akyz , and Garrett M. Johnson . Color Imaging: Fundamentals and Applications. A. K. Peters, Ltd., 2008.
Maureen C . Stone. A K Peters: A Field Guide to Digital Color; 2003.

Color Reproduction

Roy S. Berns, FredW. Billmeyer, and Max Saltzman. Billmeyer and Saltzman's Principles of Color Technology. Wiley Interscience, 2000.
Paul E. Debevec and Jitendra Malik. Recovering High Dynamic Range Radiance Maps from Photographs." In Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '97, pp. 369-378, 1997.
O. Gallo, N. Gelfand, W. Chen, M. Tico, and K. Pulli. Artifact-free High Dynamic Range Imaging." IEEE International Conference on Computational Photography (ICCP).
M. Gupta, D. Iso, and S.K. Nayar. Fibonacci Exposure Bracketing for High Dynamic Range Imaging." pp. 1-8.
R. W. G. Hunt. The Reproduction of Color. Fountain Press, 1995.
G. W. Larson , H. Rushmeier , and C. Piatko . A Visibility Matching Tone Reproduction Operator for High Dynamic Range Scenes." IEEE Transactions on Visualization and Computer Graphics 3:4.
Yuqi Li , Aditi Majumder , Dongming Lu , and Meenakshisundaram Gopi . Content-Independent Multi-Spectral Display Using Superimposed Projections." Computer Graphics Forum.
Margaret Livingstone and H. David Hubel . Vision and Art : The Biology of Seeing. Harry N Abrams, 2002.
Stephen, E. Palmer. Vision Science: MIT Press, 1999.
Erik Reinhard , Greg Ward , Sumanta Pattanaik , and Paul Debevec . High Dynamic Range Imaging: Acquisition, Display, and Image-Based Lighting (The Morgan Kaufmann Series in Computer Graphics). Morgan Kaufmann Publishers Inc., 2005.
Helge Seetzen , W. Heidrich , W. Stuezlinger , G. Ward , L. Whitehead , M. Trentacoste , A. Ghosh , and A. Vorozcovs . High Dynamic Range Display Systems." ACM Transactions on Graphics (special issue SIGGRAPH).
R. Shogenji , Y. Kitamura , K. Yamada , S. Miyatake , and J. Tanida . Multispectral imaging using compact compound optics." Opt. Exp., p. 16431655.
Peterescu S. Nanu F. Capata A. Corcoran P. Susanu, G. RGBW Sensor Array." US Patent 2009/0,167,893.
J. Tumblin and G. Turk . Low Curvature Image Simplifiers (LCIS): A Boundary Hierarchy for Detail-Preserving Contrast Reduction." pp. 83-90.
F. Yasuma , T. Mitsunaga , D. Iso , and S.K. Nayar . Generalized Assorted Pixel Camera: Post-Capture Control of Resolution, Dynamic Range and Spectrum." IEEE Transactions on Image Processing 99.

Photometric Processing

Basri Ronen , Jacobs David , Kemelmacher Ira . (Photometric Stereo with General. Unknown Lighting International Journal of Computer Vision. 2007;72:239257.
M. Brown and D. G. Lowe . Recognising Panoramas In Proceedings of the Ninth IEEE International Conference on Computer VisionVolume 2 , 2003.
Peter J. Burt and Edward H. Adelson . A Multiresolution Spline with Application to Image Mosaics ACM Trans. Graph . 2:4.
Alexei A. Efros and William T. Freeman . Image Quilting for Texture Synthesis and Transfer In Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 01, 2001.
Bruce E . Goldstein. Thomas Wadsworth: Sensation and Perception; 2010.
Vivek Kwatra , Arno Schdl , Irfan Essa , Greg Turk , and Aaron Bobick . Graphcut Textures: Image and Video Synthesis Using Graph Cuts ACM Transactions on Graphics , SIGGRAPH 2003 22:3 (2003), 277286.
Aditi Majumder and Sandy Irani. Perception-based Contrast Enhancement of Images ACM Trans. Appl. Percept. 4:3.
Prez Patrick , Gangnet Michel , Blake Andrew . Poisson Image Editing ACM Trans . Graph. 2003;22(3):313318.
Lun Wu , Arvind Ganesh , Boxin Shi , Yasuyuki Matsushita , Yongtian Wang , and Yi Ma . Robust Photometric Stereo via Low-rank Matrix Completion and Recovery pp. 703717.

Visual Content Synthesis

Angel Edward . Interactive Computer Graphics: A Top-Down Approach Using OpenGL. Addison Wesley; 2008.
John Cheng Max Grossman , and Ty McKercher . Professional CUDA C Programming. Wrox, 2014.
290 Cook Shane . CUDA Programming: A Developers Guide to Parallel Computing with GPUs. Morgan Kaufmann; 2012.
James D. Foley , Andries van Dam , Steven K. Feiner , and John F. Hughes . Computer Graphics: Principles and Practice (2 Nd Ed.) Addison-Wesley Longman Publishing Co., Inc., 1990.
Donald D . Hearn and M. Pearson: Pauline Baker. Computer Graphics with OpenGL; 2010.
David B. Kirk and Wen mei W. Hwu . Programming Massively Parallel Processors: A Hands-on Approach. Morgan Kaufmann, 2012.
Shirley Peter , Marschner Steve . Fundamentals of Computer Graphics. Peters Ltd: A. K; 2009.
Foley James D , van Dam Andries , Feiner Steven K , Hughes John F . Computer Graphics: Principles and Practice. 2nd ed. Boston, MA: Addison-Wesley Longman Publishing Co., Inc; 1990.
Shirley Peter , Marschner Steve . Fundamentals of Computer Graphics. Peters A. K. ; Ltd: 2009.
Watt Alan . 3D Computer Graphics. Addison Wesley; 1999.
Stefan Gottschalk , Ming Lin , and Dinesh Manocha . OBB-Tree: A Hierarchical Structure for Rapid Interference Detection. Computer Graphics (SIGGRAPH 1996 Proceedings), pp. 171-180.
P. Jimenez , F. Thomas , & C. Torras . 3D Collision Detection: a Survey. Computers and Graphics, 25:2 (2001), 269285.
Shankar Krishnan , Amol Pattekar , Ming Lin , and Dinesh Manocha . Spherical Shell: A Higher Order Bounding Volume for Fast Proximity Queries. Robotics: The Algorithmic Perspective: WAFR, pp. 177-190.
Edward Angel and Dave Shreiner .Interactive Computer Graphics: A TopDown Approach with WebGL, 7th edition. Pearson, 2014.
Edward Angel . Interactive Computer Graphics, Third edition. AddisonWesley Longman Publishing Co. Inc., 2002.
Azad, Hamid , 2016. Advances in GPU Research and Practice. 1st ed. Morgan Kaufman, 2016
John Cheng and Max Grossman . Professional CUDA C Programming. Wrox, 2014.
Donald D. Hearn and M. Pauline Baker . Computer Graphics with OpenGL, Third edition. Prentice Hall Professional Technical Reference, 2003.
Hill, F.S. and Stephen,M.Kelly . Computer Graphics using OpenGL: Prentice Hall, 2006.
John Kessenich , Graham Sellers , and Dave Shreiner . OpenGL Programming Guide: The Official Guide to Learning OpenGL, Version 4.5, 9th edition. AddisonWesley Longman Publishing Co., Inc., 2016.
Aaron E. Lefohn , Shubhabrata Sengupta , Joe Kniss , Robert Strzodka , and John D. Owens . Glift: Generic, efficient, randomaccess GPU data structures ACM Trans. Graph. 25:1 (2006), 6099. Available online (http://doi.acm.org/10.1145/1122501.1122505).
David P. Luebke . Graphics hardware & GPU computing: past, present, and future. In Proceedings of the Graphics Interface 2009 Conference, May 2527, 2009, Kelowna, British Columbia, Canada, p. 6, 2009. Available online (http://doi.acm.org/10.1145/1555880.1555888).
Patrick McCormick , Jeff Inman , James Ahrens , Jamaludin Mohd-Yusof , Greg Roth , and Sharen Cummins . Scout: A Dataparallel Programming Language for Graphics Processors. Parallel Comput. 33:1011 (2007), 648662. Available online (https://doi.org/10.1016/j.parco.2007.09.001).
Sanders, Jason , 2010. CUDA by Example: An Introduction to General-Purpose GPU Programming. 1st ed. Addison Wesley.

The Diverse Domain

Angel Edward . Interactive Computer Graphics: A Top-Down Approach Using OpenGL. Addison Wesley; 2008.
John Cheng Max Grossman , and Ty McKercher . Professional CUDA C Programming. Wrox, 2014.
290 Cook Shane . CUDA Programming: A Developers Guide to Parallel Computing with GPUs. Morgan Kaufmann; 2012.
James D. Foley , Andries van Dam , Steven K. Feiner , and John F. Hughes . Computer Graphics: Principles and Practice (2 Nd Ed.) Addison-Wesley Longman Publishing Co., Inc., 1990.
Donald D . Hearn and M. Pearson: Pauline Baker. Computer Graphics with OpenGL; 2010.
David B. Kirk and Wen mei W. Hwu . Programming Massively Parallel Processors: A Hands-on Approach. Morgan Kaufmann, 2012.
Shirley Peter , Marschner Steve . Fundamentals of Computer Graphics. Peters Ltd: A. K; 2009.

Interactive Graphics Pipeline

Foley James D , van Dam Andries , Feiner Steven K , Hughes John F . Computer Graphics: Principles and Practice. 2nd ed. Boston, MA: Addison-Wesley Longman Publishing Co., Inc; 1990.
Shirley Peter , Marschner Steve . Fundamentals of Computer Graphics. Peters A. K. ; Ltd: 2009.
Watt Alan . 3D Computer Graphics. Addison Wesley; 1999.

Realism and Performance

Stefan Gottschalk , Ming Lin , and Dinesh Manocha . OBB-Tree: A Hierarchical Structure for Rapid Interference Detection. Computer Graphics (SIGGRAPH 1996 Proceedings), pp. 171-180.
P. Jimenez , F. Thomas , & C. Torras . 3D Collision Detection: a Survey. Computers and Graphics, 25:2 (2001), 269285.
Shankar Krishnan , Amol Pattekar , Ming Lin , and Dinesh Manocha . Spherical Shell: A Higher Order Bounding Volume for Fast Proximity Queries. Robotics: The Algorithmic Perspective: WAFR, pp. 177-190.

Graphics Programming

Edward Angel and Dave Shreiner .Interactive Computer Graphics: A TopDown Approach with WebGL, 7th edition. Pearson, 2014.
Edward Angel . Interactive Computer Graphics, Third edition. AddisonWesley Longman Publishing Co. Inc., 2002.
Azad, Hamid , 2016. Advances in GPU Research and Practice. 1st ed. Morgan Kaufman, 2016
John Cheng and Max Grossman . Professional CUDA C Programming. Wrox, 2014.
Donald D. Hearn and M. Pauline Baker . Computer Graphics with OpenGL, Third edition. Prentice Hall Professional Technical Reference, 2003.
Hill, F.S. and Stephen,M.Kelly . Computer Graphics using OpenGL: Prentice Hall, 2006.
John Kessenich , Graham Sellers , and Dave Shreiner . OpenGL Programming Guide: The Official Guide to Learning OpenGL, Version 4.5, 9th edition. AddisonWesley Longman Publishing Co., Inc., 2016.
Aaron E. Lefohn , Shubhabrata Sengupta , Joe Kniss , Robert Strzodka , and John D. Owens . Glift: Generic, efficient, randomaccess GPU data structures ACM Trans. Graph. 25:1 (2006), 6099. Available online (http://doi.acm.org/10.1145/1122501.1122505).
David P. Luebke . Graphics hardware & GPU computing: past, present, and future. In Proceedings of the Graphics Interface 2009 Conference, May 2527, 2009, Kelowna, British Columbia, Canada, p. 6, 2009. Available online (http://doi.acm.org/10.1145/1555880.1555888).
Patrick McCormick , Jeff Inman , James Ahrens , Jamaludin Mohd-Yusof , Greg Roth , and Sharen Cummins . Scout: A Dataparallel Programming Language for Graphics Processors. Parallel Comput. 33:1011 (2007), 648662. Available online (https://doi.org/10.1016/j.parco.2007.09.001).
Sanders, Jason , 2010. CUDA by Example: An Introduction to General-Purpose GPU Programming. 1st ed. Addison Wesley.