**This document is downloaded from DR-NTU (https://dr.ntu.edu.sg)
Nanyang Technological University, Singapore.**

Shapes illustration with feature lines and stylized rendering

Xie, Xuexiang

2008

Xie, X. X. (2008). Shapes illustration with feature lines and stylized rendering. Doctoral thesis, Nanyang Technological University, Singapore.

https://hdl.handle.net/10356/15659

https://doi.org/10.32657/10356/15659

Downloaded on 13 Mar 2024 18:47:44 SGT

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

Nanyang Technological University

Shapes Illustration with Feature Lines and Stylized Rendering

A thesis submitted to
the School of Computer Engineering
of Nanyang Technological University

by

Xie XueXiang

for the Degree of Doctor of Philosophy

Supervisor: Tian Feng

2008

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

Abstract

Over centuries, artists and illustrators have developed powerful techniques to convey shapes effectively in very rich forms. In recent years, non-photorealistic rendering has been improved greatly to assist artists and illustrators in their work with computer generated illustrations. However, it is more than often that hand-drawn illustrations are still the only choice in modern industry. In this dissertation, we aim to provide more effective techniques for shape illustration with feature lines and stylized rendering.

The first part of our work is to obtain new feature lines which are more perceptually consistent and flexible in conveying shapes. A number of feature lines have been proposed, but they are not really able to capture all the visually relevant features by themselves. We propose *Photic Extremum Line* (PEL) which emphasizes significant variations of illuminance over 3D surfaces based on observation in human vision and perception that a sudden change in luminance plays a critical role in faithfully representing and recovering 3D information. Compared with the existing feature lines, PEL is more flexible and offers users more freedom and control to achieve their desired illustrations. Two efficient PELs extraction algorithms are presented to incorporate PELs to various rendering systems.

Next, we simulate the traditional sketching based on PELs and contours. Sketching is one of the most widely used techniques, and in practice, architects even trace over computer generated line drawings of their initial designs to create a sketchier look. We propose a novel approach to emulate sketching strokes with good texture and break long strokes into short ones. Our approach incorporates principles of perceptual grouping with a global segmentation algorithm. By applying this approach at a succession of scales, the successive approximation effect seen in sketching can also be effectively simulated. For many applications, one of the most important goals of illustration is to achieve visually pleasing line drawings, such as cartoon. We hence present an effective and

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

flexible framework to convey shapes with stylized line drawings based on a novel stroke representation model, *disk B-spline curve*(DBSC). The flexibility of the stylized line drawing framework is further demonstrated with a wide range of styles.

Finally, stylized shading is equally important in illustration, which gives strong cues of shapes and more details. We experiment in stylized shading by modeling shading styles with low level statistical features, and present an improved feature guided texture synthesis algorithm to generate stylized shading from input examples.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

Acknowledgments

I would like to thank Dr. Tian Feng, my supervisor, for his time, effort, and constant support during this research. I am also grateful to Prof. Seah Hock Soon for his instruction and support. Especially, I would like to thank Prof. He Ying. Deep thanks also go to Dr. Wu Zhongke, Prof. Qian Kemao, Mr. Qiu Jie, Mr. Chen Quan, Mr. Lu Ji, Miss Xiao Xian, Miss Zhang Chong, Miss Li Li, Mr. Quah Chee Kwang who discussed with me about the research topic, expanded my knowledge in the field and gave me help and encouragement.

I should appreciate the financial support to this research provided by Nanyang Technological University. Finally, I would like to express my sincere gratitude to my family for their constant support, encouragement and blessings.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

Contents

Abstract i
Acknowledgments iii

1 Introduction **1**
1.1 Background 1
1.2 Overview of the Thesis 5

2 Related Work **7**
2.1 Feature Lines 7
2.1.1 Contours 8
2.1.2 Suggestive Contours 11
2.1.3 Ridges and Valleys 13
2.1.4 Other Feature Lines 15
2.1.5 Where Do People Draw Lines? 17
2.2 Stylized Line Drawing 18
2.2.1 Stylized Strokes 19
2.2.2 Stylized Line Drawing Systems 22
2.2.3 Others 24
2.3 Stylized Shading 26

3 Effective Shape Illustration with PELs **31**
3.1 Introduction 32
3.2 Photic Extremum Lines (PELs) 35
3.2.1 Definition 35
3.2.2 Lighting Dependency of PELs 36

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

3.2.3 Optimal Lighting for PELs 40
3.3 Comparisons of PELs with Previous Feature Lines 44
3.3.1 PELs versus Contours 45
3.3.2 PELs versus Suggestive Contours 46
3.3.3 PELs versus Ridges & Valleys. 49
3.3.4 Summary 51
3.4 Comparison of PELs with Apparent Ridges 53
3.5 PELs for Volume Illustration 56
3.6 Conclusion 61
3.6.1 Evaluation 61
3.6.2 Limitation and Discussion 64

4 Algorithms of PELs Extraction **67**

4.1 Introduction 67
4.2 PELs Extraction on Meshes 69
4.2.1 Preprocessing 70
4.2.2 Computing the Derivatives 71
4.2.3 Tracing PELs 73
4.2.4 Experimental Results 74
4.3 PELs Extraction in Geometry-Image Space 78
4.3.1 Geometry Images 78
4.3.2 Geometry Images Based PELs Extraction 80
4.4 Discussion and Conclusion 82

5 Conveying Shapes with NPR Sketching **87**

5.1 Introduction 88
5.1.1 Observation of Traditional Sketching 88
5.1.2 Related Work and Our Contribution 90
5.1.3 Overview of Our method 91
5.2 Stroke Visibility 91
5.3 Normalized Cuts 92
5.4 Perceptual Stroke Segmentation 96

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

5.4.1 Principles of Perceptual Grouping 96
5.4.2 Measure of Perceptual Similarity 98
5.5 Spline Fitting Stroke Segments 101
5.6 Results and Discussion 103
5.6.1 Experimental Results 103
5.6.2 Evaluation 108
5.6.3 Discussion 111

6 Stylized Line Drawing based on DBSC Strokes 112
6.1 Introduction 113
6.2 DBSC Strokes 114
6.2.1 Disk B-Spline Curve(DBSC) 115
6.2.2 Generation of DBSC Strokes 120
6.2.3 Visualization of DBSC Strokes 122
6.2.4 Results and Discussion 124
6.3 DBSC Based Pencil Style Simulation 125
6.3.1 Simulation of Pencil Materials 126
6.3.2 Pencil Rendering of 3D Models 127
6.3.3 DBSC Based Pencil Drawing 128
6.4 Procedural Stylized Line Drawing 131
6.4.1 Expressive Strokes 132
6.4.2 Artistic Media with Stroke Textures 134
6.4.3 Stroke Segmentation for Stylized Line Drawing 137
6.5 Conclusion 139

7 Stylized Shading with Feature Guided Texture Synthesis 141
7.1 Introduction 142
7.2 MRF Analysis of Texture Synthesis 144
7.3 Feature Guided Texture Synthesis 146
7.3.1 Image Analogies Framework 146
7.3.2 Feature Field Generation 148
7.3.3 Improved L_2 Distance Metric 150

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

7.4 Results and Comparisons 151
7.5 Conclusion 157

8 Conclusions and Future Work 158
8.1 Summary 158
8.2 Future Work 160
8.2.1 Style Learning from Example 160
8.2.2 Improvement and System Integration 162

References 163

Publications 177

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

Chapter 1

Introduction

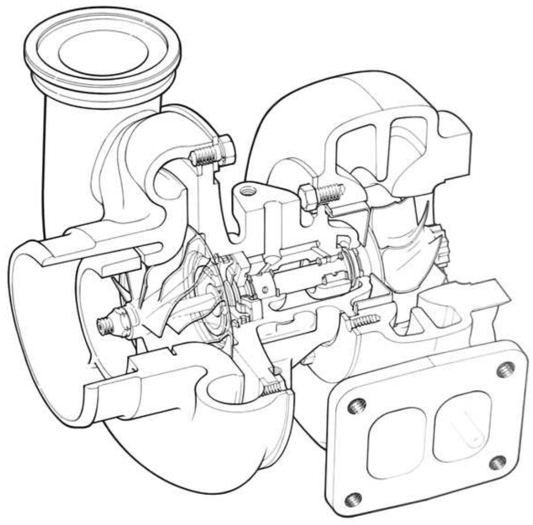
1.1 Background

Computer graphics has become a popular medium for science, art, and communication. It has long been defined as a quest to achieve photorealism. Photo can convey shapes and scenes with great detail. But too many details can also be distracting. Experience shows that, so frequently, an artistic drawing or painting is both more effective in communication and more visually pleasing than a photograph. This is mainly due to the abstract nature of various artistic styles. It's also demonstrated that photo is not always the best way of presenting visual information. Some kinds of visual abstraction are required in many cases. This is well studied in traditional illustration which focuses on conveying complex procedures or shapes' structures under the rule, "as detailed as necessary while as simple as possible". In contrast to photorealistic rendering, the goal of non-photorealistic rendering(NPR) is to effectively communicate information in digital media to the viewer with interpretive and expressive rendering. It is quite in line with the goal of traditional illustration. Taking both the beauty of traditional media and the power of computer graphics, NPR has advanced significantly in recent years in assisting artists and illustrators.

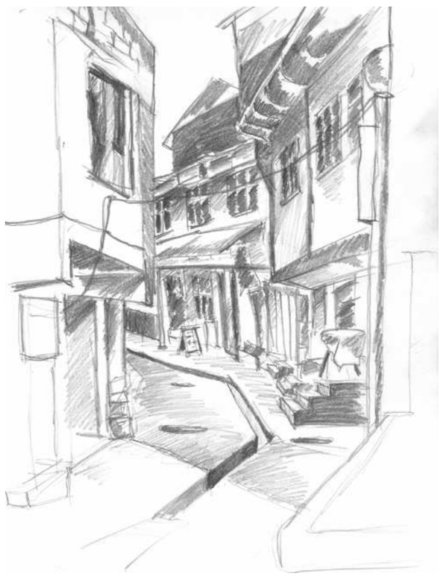
Traditional illustration has been developed for centuries by artists and illustrators to effectively convey shapes in very rich forms. In comparison with photograph, illustration

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

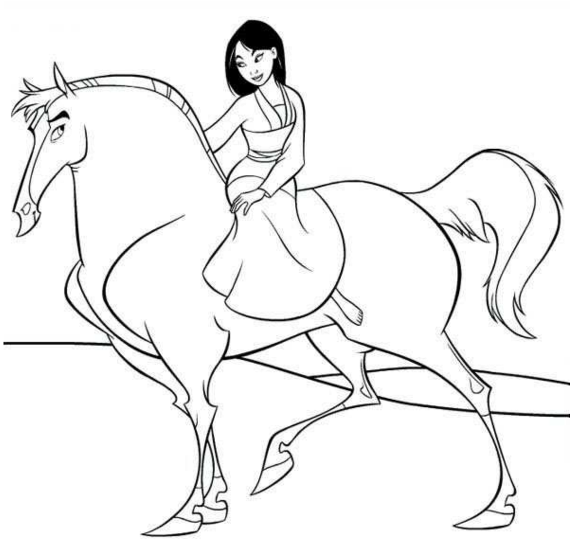
has many advantages such as focusing attention on specific features, omitting extraneous details, and exposing hidden parts of shapes. It may also express complex scenes with distinct stylistic choices and simplify shapes to prevent visual overload. These can be demonstrated in practical cases, such as the maintenance and repair manuals of mechani-



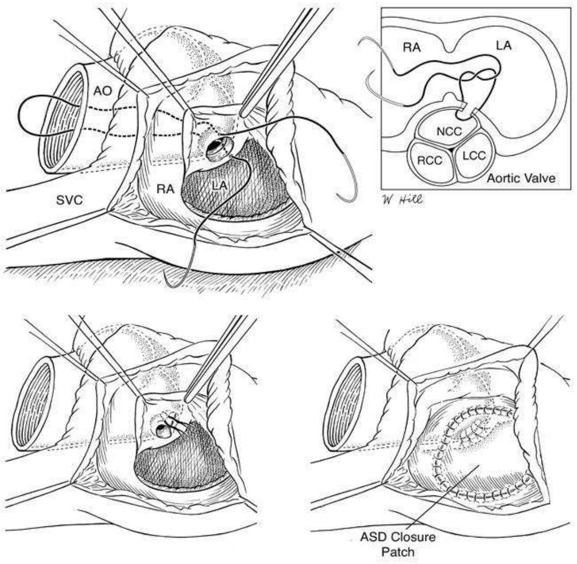
(a) Technical illustration



(b) Sketching for archaeological illustration



(c) Illustration for cartoon animation
(by courtesy of Walt Disney)



(d) Medical illustration
(by courtesy of Wendolyn Hill, MFA, CMI)

Figure 1.1: Examples of traditional illustration.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

cal products, where photographs have no point to machinists who have the real objects in hand. What they actually need is the clear illustration of the structure of the products. The advantages are also apparent in medical textbooks, where illustrations are used in place of photographs to expose subtle and hidden structures for educational purposes. More examples on illustration are shown in Figure 1.1.

NPR has improved greatly in assisting artists and illustrators with computer generated imagery. In so many cases, however, hand drawn illustrations remain the only choice. For example, most of the medical illustrations are still hand drawn in spite being laborious. At Boeing, though the CAD database of airplane is well developed, all high-quality manuals are still illustrated by hand [67]. Based on these observations, we seek to provide more effective techniques for shapes illustration with feature lines and stylized rendering.

Depicting shapes using feature lines is one of the most important techniques in traditional illustration, as we can see in Figure 1.1. A number of feature lines have been proposed and applied to non-photorealistic line drawing. However, none of them seems able by themselves to capture all visually relevant features. Thus, the first motivation of our work is to obtain new feature lines which are more perceptually consistent and flexible in conveying shapes. We find a new type of feature lines based on principles of human vision and perception, offering users more flexibility to achieve their desired illustrations.

Sketching is one of the most widely used traditional illustration styles to convey shapes with successive approximation. It describes a drawing which is not a final, perfect result, as shown in Figure 1.1 (b). In practice, architects even trace over computer generated line drawings of their initial designs to create a sketchier look [113]. Based on these observations, we have developed a novel algorithm to emulate sketching. A lot of research has been carried out to tackle issues of stroke based rendering, but insufficient attention

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

has been paid to the problem of breaking strokes in artists' way. Our algorithm generates sketching strokes qualitatively resemble those produced by real artists with perceptual stroke segmentation methods.

Though feature lines alone can convey shapes effectively, in many applications one of the most important goals of illustration is to achieve visually pleasing line drawings. As shown in Figure 1.1 (c), the application of strokes based on feature lines makes the line drawing more vivid and attractive. Thus, we propose and develop an effective and flexible framework for conveying shapes with stylized line drawings based on a novel stroke representation model. In comparison with stylized line drawing, stylized shading is equally important in traditional illustration. It gives strong cues of shapes and more details, as can be seen in Figure 1.1 (b) and (d). We experiment stylized shading with texture synthesis and get intuitive cues for further research towards this direction.

Inspired by observations of characteristic of traditional illustration, and the application of existing NPR techniques in conveying shapes, several novel algorithms have been proposed in our work for more effective shapes illustration with feature lines and stylized rendering. In summary, the contributions of this dissertation are:

- We present a new type of feature lines, *Photic Extremum Line* (PEL), which is more perceptually consistent and flexible in conveying shapes effectively. PEL is defined based on observations of human vision and perception. In comparison with existing feature lines, PEL offers users more freedom to achieve user controllable illustrations.

- We propose novel algorithms for efficient PELs extraction. With these methods, PEL may become a convenient tool to be incorporated into various rendering systems.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

- We simulate traditional sketching based on PELs and contours. Sketching strokes are emulated with a novel method which incorporates principles of perceptual grouping with a global segmentation approach. The sketching strokes generated with our approach qualitatively resemble those produced by artists.
- We present an effective and flexible framework for conveying shapes with stylized line drawings based on a novel stroke representation model, *disk B-spline curve*(DBSC). An improved algorithm on stylized shading based on texture synthesis is also presented to give us intuitive cues for further research.

1.2 Overview of the Thesis

The rest of this thesis is organized as follows:

- Chapter 2 reviews related works with emphasis on feature lines, stylized line drawing and stylized shading.
- Chapter 3 describes the definition of *Photic Extremum Line* (PEL), and compares PEL with existing feature lines for both 3D shapes and volume illustrations.
- Chapter 4 presents two novel algorithms for efficient PELs extraction. One is to extract PELs in object space for triangle meshes. The other is to extract PELs in geometry-image space.
- Chapter 5 introduces our sketching emulation algorithm with more optimal sketching strokes based on stroke segmentation.
- Chapter 6 presents an effective and flexible framework for stylized line drawing based on a novel stroke representation model, *disk B-spline curve*(DBSC).
- Chapter 7 introduces our improved feature guided texture synthesis algorithm for stylized shading.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

- Chapter 8 concludes this dissertation with summary and future directions.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

Chapter 2

Related Work

Non-photorealistic rendering(NPR) has improved greatly in recent years in assisting artists and illustrators in a wide range of areas including computational art and scientific visualization. Earlier survey on NPR can be found in [65, 31, 113]. Focusing on NPR's application in shapes illustration, we take a narrower view of NPR and review related work of this dissertation with emphasis on feature lines, stylized line drawing and stylized shading.

2.1 Feature Lines

Line drawing is one of the most widely used methods in traditional illustrations. The basic problem of computer generated line drawings is, given a 3D shape, where lines should be drawn? Conveying shapes with feature lines started from the beginning of computer graphics. A number of feature lines have been proposed and applied in non-photorealistic line drawing. However, none of them seem able by themselves to capture all the visually relevant features. A survey can be found in [98], which proposes a classification of feature lines based on the differential properties defined, the order of derivation, and whether the viewpoint is considered. In recent years, the majority of work on feature lines focuses on contours, suggestive contours, and ridge-valley lines.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

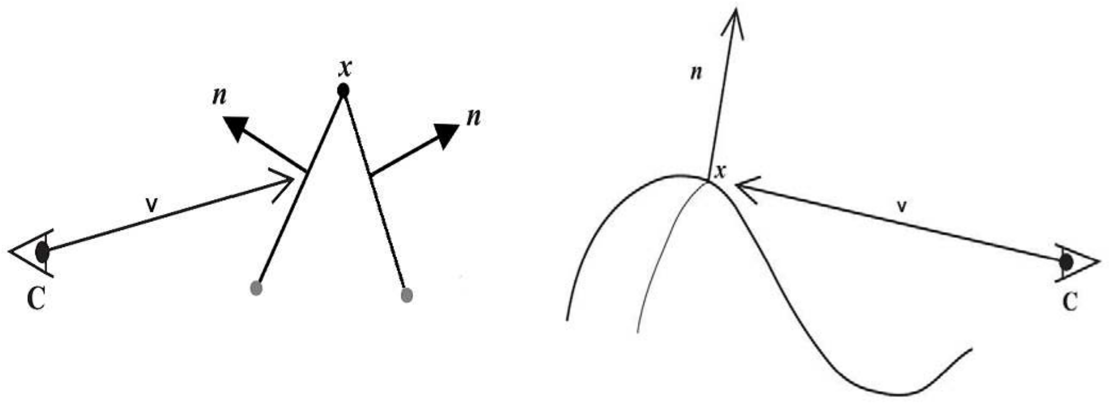


Figure 2.1: (Left)Contours for polygon mesh. (Right)Contours for smooth surface.

2.1.1 Contours

Contours, normally defined as the visible parts of silhouettes, are the set of visible points on the object's surface where the surface normal is perpendicular to the view vector, mathematically, with $\mathbf{n} \cdot \mathbf{v} = 0$. For polygonal meshes, contours are always defined as edges that share a front-facing and a back-facing polygon. The definition of contours for smooth surfaces and polygon meshes are illustrated in Fig 2.1. Contours separate visible and invisible parts of an object, which give the strongest cues of model-to-background distinction.

Contours are view dependant feature lines which change from frame to frame according to the changing of the view point. Extracting contours efficiently is not straightforward. A lot of research has been done in this area. A survey can be found in [52]. According to where the contours are detected and drawn, contours extraction algorithms are classified into three categories: object space algorithms, image space algorithms, and hybrid algorithms.

Object space algorithms. Object space algorithms detect and depict contours in 3D. They get feature lines in precise analytic description which is convenient for stylized line drawing based on contours.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

The straightforward algorithm in object space is the brute-force algorithm which detects contours by traversing all the faces. It is simple in implementation but not efficient enough for large models. Markosian et al. [74] introduce a fast method that trades accuracy for speed based on probabilistic testing. Exploiting the inter-frame coherence of contours, they locate contours using a non-deterministic algorithm. This method achieves real time performance, but it does not guarantee that all contours will be detected. Gooch et al. [32] present a deterministic method for contours detection in object space using a pre-computed Gauss map. By mapping each point on the surface to a point on the Gauss sphere in its normal direction, their method is sped up with no need of checking the front or back facing of each face.

Hertzmann et al. [46] use the concept of dual surface for fast and more exact detection of contours. They define contours as zero sets of smooth surfaces, or contour edges of polygonal meshes. By constructing a dual representation of the mesh in a 4D space, they reduce the problem of finding contours to the problem of intersecting a plane with a surface. Another advantage of this method is that it works for both orthographic projection and perspective projection.

Image space and Hybrid algorithms. Image space algorithms extract contours exploiting discontinuities in the image buffer using image processing methods. The image buffer can also contain geometry information, such as the z-buffer. Hybrid algorithms are slightly different from image space algorithms by modifying surfaces in object space and then find contours using z-buffer. Image space and hybrid algorithms are fast, but they can only achieve contours in pixel precision, and their representation of contours in pixels doesn't facilitate further stylization.

The straightforward method in image space is to detect edges in the rendered image. However, this approach does not always work well on detecting contours as it may fail to separate contours from some other feature lines. Saito and Takahashi [99] suggest

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

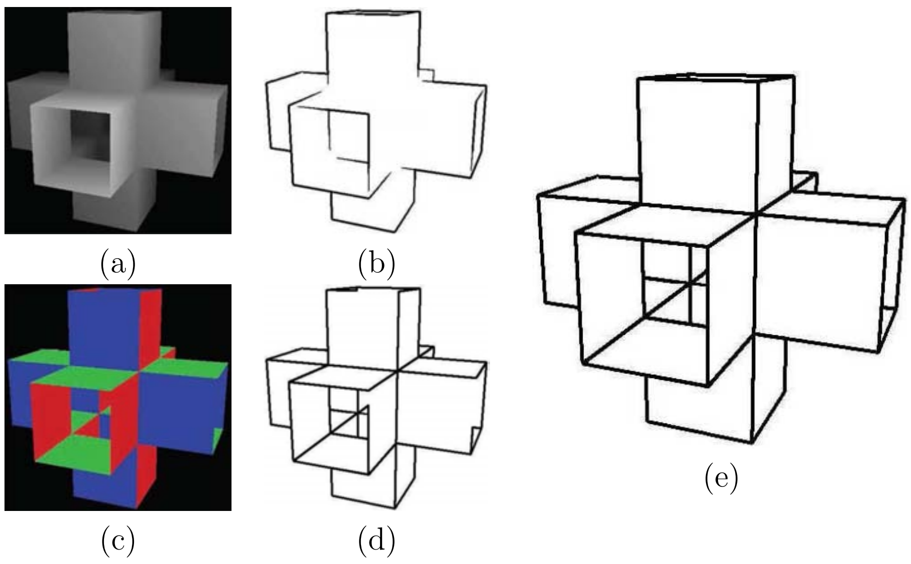


Figure 2.2: Image space contours detection using edge detectors on the z-buffer and the normal buffer. (a)z-buffer, (b)edges extracted from the z-buffer, (c)normal buffer, (d)edges extracted from the normal buffer, (e)combination of edges of both buffers.(Courtesy of Hertzmann 1999)

using the z-buffer instead, and applying edge detector such as Sobel operator on the image buffer which encodes geometry information. Hertzmann [43] extends this method by using a normal buffer. Contours are actually the C^0 discontinuities of the image buffer. This method can also detect C^1 discontinuities. The combination of the C^0 and C^1 discontinuities of the image buffer exhibits pleasing results, as shown in Figure 2.2.

Raskar and Cohen [95] propose a hybrid algorithm for drawing contours using hardware. Their method assumes no prior knowledge or preprocessing of the model, and achieves high performance by leveraging commodity graphics hardware. The method increases the area of intersection at contours. Front-facing polygons are rendered, then back-facing polygons are pulled slightly towards the camera with depth function set to ”Less Than or Equal To” and rendered filled. This algorithm overcomes the precision problem of the depth buffer and provides higher degree of control over the output of the contours. In [94], Raskar improves this method with a one-pass hardware implementation that basically adds borders around each triangle.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

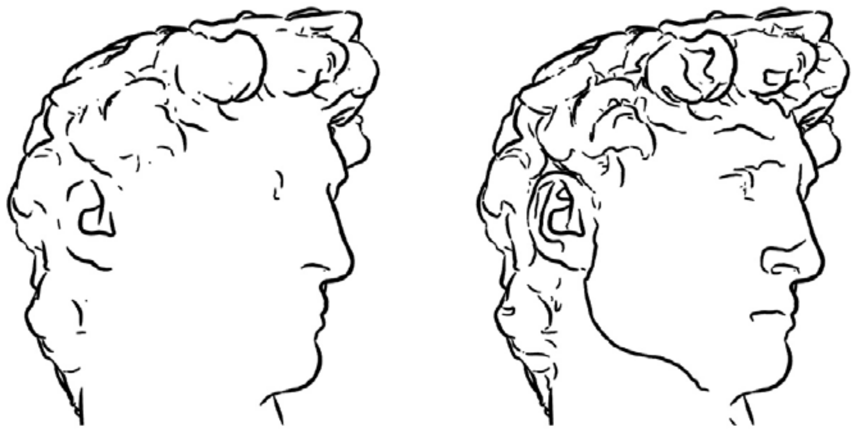


Figure 2.3: An example showing the expressiveness added by suggestive contours. The left image is drawn using contours alone, while the right image uses both contours and suggestive contours.(Courtesy of Decarlo et al. 2003)

2.1.2 Suggestive Contours

Being considered as the extension of the actual contours on surfaces, suggestive contours, a new type of feature lines, are proposed by DeCarlo et al. [16]. Suggestive contours are feature lines drawn on clearly visible parts of the surface, where a true contour would first appear with a minimal change in viewpoint, as shown in Figure 2.3. Thus, suggestive contours are features where a surface bends sharply away from the viewer, yet remains visible. They are almost contours, which may become contours in nearby views. In [16], both image space and object space algorithms for suggestive contours extraction are provided.

Mathematically, suggestive contours are defined with the set of points on the surface at which its radial curvature k_r is 0, and the directional derivative of k_r in the direction of the view vector projected onto its tangent plane, is positive. Defined with viewing conditions, suggestive contours are view dependant feature lines. Before suggestive contours, all other feature lines drawn from general 3D shapes have been feature lines fixed on the surface which can not make natural looking line drawing in conditions with changing

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

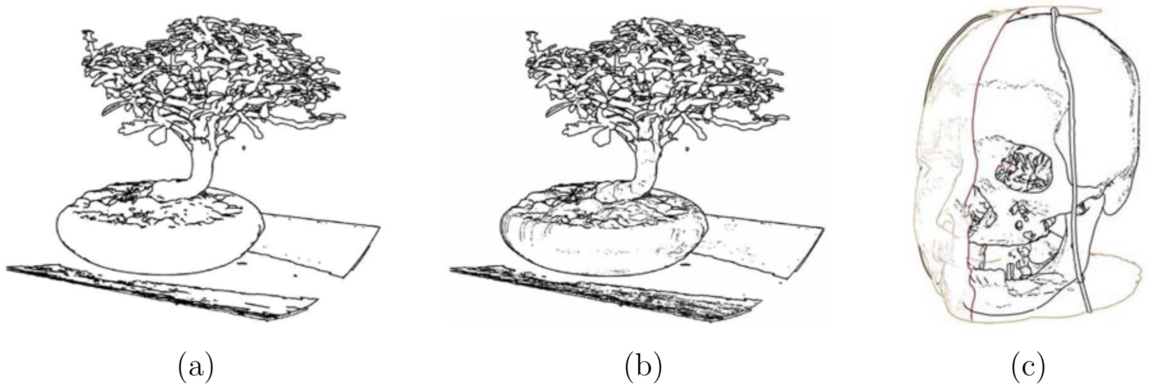


Figure 2.4: Volume illustration. (a)Bonsai with contours only. (b)Bonsai with contours and suggestive contours. (c)Head with skin, bone and cutting plane.(Courtesy of Burns et al. 2005)

view points. Thus, the suggestive contour is the pioneer of view-dependent feature lines which is especially important in animation sequences.

In [15], DeCarlo et al. extend suggestive contours to dynamic and real-time settings. They analyze movement of suggestive contours with respect to changes in viewpoint. They describe practical algorithms for rendering drawings with contours and suggestive contours at interactive rates with temporal coherence and discuss techniques for improving the visual appearance of suggestive contours.

Suggestive contours are also extended to describe volume data by Burns et al. [8]. They propose a volumetric drawing system that directly extracts sparse linear features, such as contours and suggestive contours, using a temporally coherent seed-and-traverse framework. Their method is efficient on large data sets and their resulting imagery is often more comprehensible than standard rendering styles. Volume illustrations generated with this method are shown in Figure 2.4.

Recently, DeCarlo and Rusinkiewicz [17] further improve their work on conveying shapes with sparse feature lines by introducing two new families of lines called suggestive highlights(SH) and principal highlights(PH), based on definitions related to suggestive contours and geometric creases. Given an arbitrary point **p** with normal **n** on a smooth

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

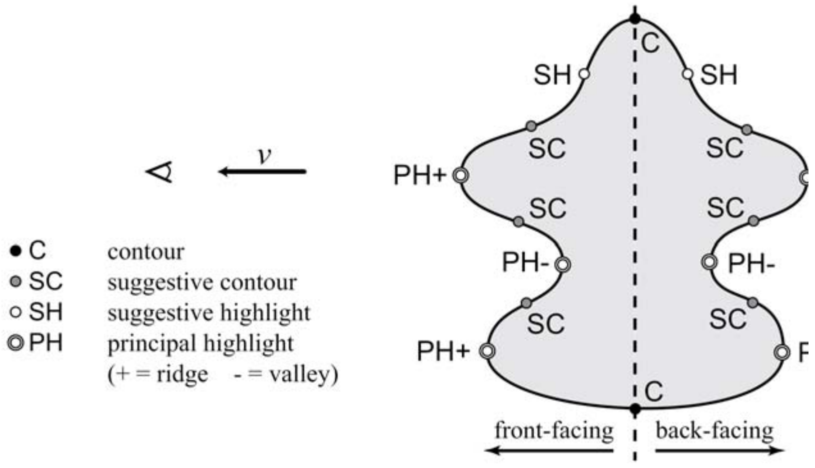


Figure 2.5: Different types of lines given the viewing direction **v** (for clarity this figure uses orthographic projection, hence constant **v**).(Courtesy of Decarlo et al. 2007)

surface S, the view vector **v** is projected onto the tangent plane of **p** to obtain **w**. Principal highlights (PH) are strong positive maxima (or negative minima) of $\mathbf{n} \cdot \mathbf{v}$ in the direction $\mathbf{w}'$ which sits perpendicular to **w** in the tangent plane, while suggestive highlights(SH) are complementary to suggestive contours: they are positive maxima (or negative minima) of $\mathbf{n} \cdot \mathbf{v}$ in the direction **w**. Equivalently, these are locations at which:

$$k_r = 0, \ and \begin{cases} D_{\mathbf{w}} k_r < 0 \ where \ n \cdot v > 0 \\ D_{\mathbf{w}} k_r > 0 \ where \ n \cdot v < 0 \end{cases}$$

These different types of lines are illustrated in Figure 2.5.

When suggestive highlights(SH) and principal highlights(PH) are drawn in white on a gray background, they are naturally interpreted as highlight lines, and complement contours and suggestive contours, as shown in Figure 2.6.

2.1.3 Ridges and Valleys

Traditionally, sharp creases [57, 74, 99] are most frequently used to convey the structure and complexity of an object. This can yield a pronounced improvement in shapes illustration when creases are conspicuous features of an object's shape (as for a cube,

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

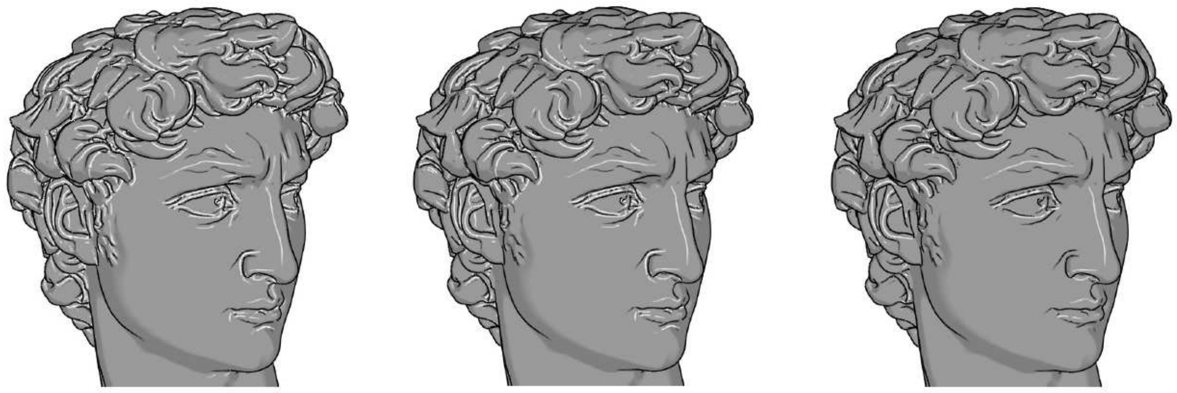

(a)C+SC+PH+SH (b)C+SC+PH (c)C+SC+PH ridges

Figure 2.6: David head with different types of highlight lines (and a toon shader).(Courtesy of Decarlo et al. 2007)

for example). Similarly and more generally, ridges and valleys are defined on smooth surfaces for describing shapes. Ridge and valley lines, also called crest lines, are curves on a surface along which the surface bends sharply. They are ones of the most powerful descriptors for shape variations.

Mathematically, ridges and valleys are points with sharp variation of surface normal, which can be described via extrema of the surface principle curvatures along their corresponding curvature lines. Thus they can be computed with local measures of the surface's differential geometry. Monga et al. [82] assume that the surface is defined locally as a iso-intensity contour and calculate directly the curvatures and characterize the local extrema from the first, second, and third order derivatives of the grey level function. Interrante et al [51] use the same definition of ridge lines as Monga et al, but eliminate the computation of the third order derivative in favor of a very stable and simple approximation that tests for the presence of a local curvature maximum in a sub-voxel region. Ohtake et al. [87] estimate curvature and curvature derivatives of 3D objects using implicit function fits, and extract high-quality ridge and valley lines from these estimations. Curvature based filtering is used to keep the most significant lines.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

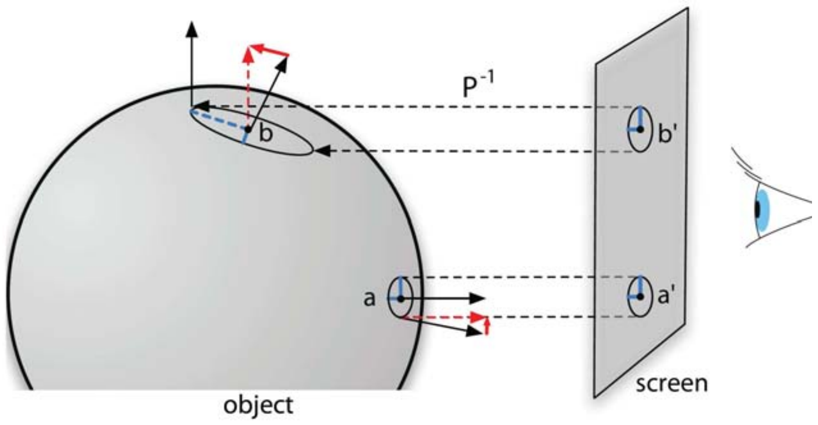


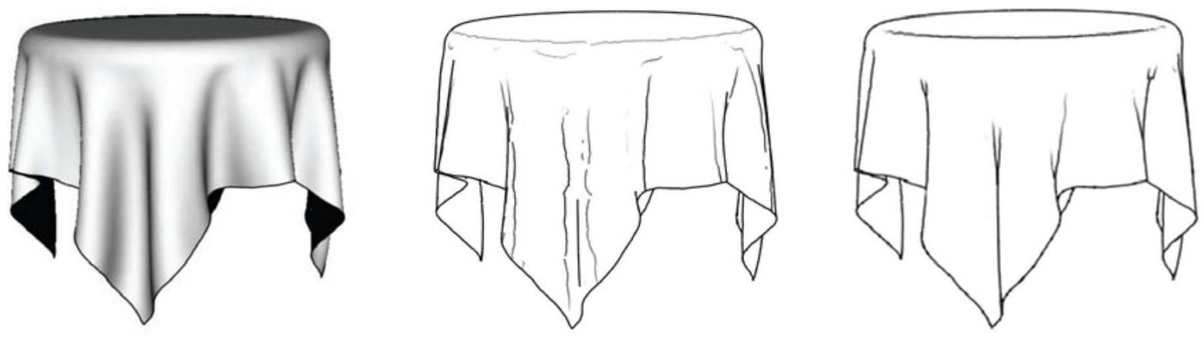
Figure 2.7: Difference between curvature and view-dependent curvature. Curvature is defined as the change in normal as one moves a small distance in direction d along the object. On the other hand, view-dependent curvature is defined as the change in normal as one moves a small distance d along the screen. Note that the view-dependent curvature at point b' is much larger than point a'.(Courtesy of Judd et al. 2007)

Ridges and valleys are fixed feature lines on surfaces which are not suitable for stylized line drawing with changing viewpoints. Recently, Judd et al. [56] introduce apparent ridges which are variants of geometric ridges that include a view-dependent projection. Their work is based on two perceptual observations. First, human perception is sensitive to the variation of shading which is closely related to the surface normal variation. Second, view-dependent lines convey smooth surfaces better. From this they define view-dependent curvature as the variation of the surface normal with respect to a viewing screen plane, and apparent ridges as the locus of points that maximize a view-dependent curvature. The difference between curvature and view-dependent curvature is demonstrated clearly in Figure 2.7. Comparing with some other feature lines, apparent ridges can achieve more perceptually consistent shapes illustration, as shown in Figure 2.8.

2.1.4 Other Feature Lines

Besides contours, ridges and valleys, and suggestive contours, there are also some other kinds of feature lines which are widely used in computer graphics. Border lines are one of

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

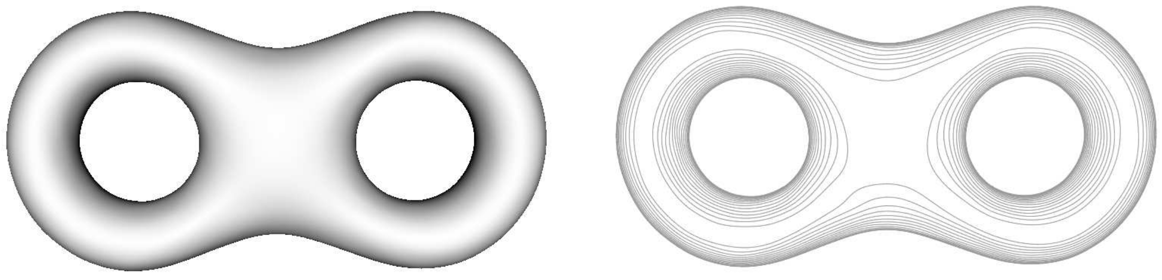


(a)Shaded View (b)Ridges and Valleys (c)Apparent Ridges

Figure 2.8: Tablecloth. Notice how the apparent ridges convey both the smoothness of the rim of the table and the drapery of the tablecloth.(Courtesy of Judd et al. 2007)

these feature lines which appear in models where the surface is open. In case of polygonal meshes, border lines are those edges with only one adjacent polygon.

Isophotes lines and reflection lines are two families of lines which are defined with surface lighting and widely used in visualizing the smoothness of surfaces. Isophotes lines are lines of constant illumination on a surface. They are also the shading boundaries between toon shading regions [64]. An example of isophotes lines is shown in Figure 2.9. Reflection lines are lines reflecting the light source consisting of a set of "light lines" that are placed in space. Traditionally, reflection lines have been used in the process of designing cars.



(a)Shaded View (b)Isophotes

Figure 2.9: Isophotes lines

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

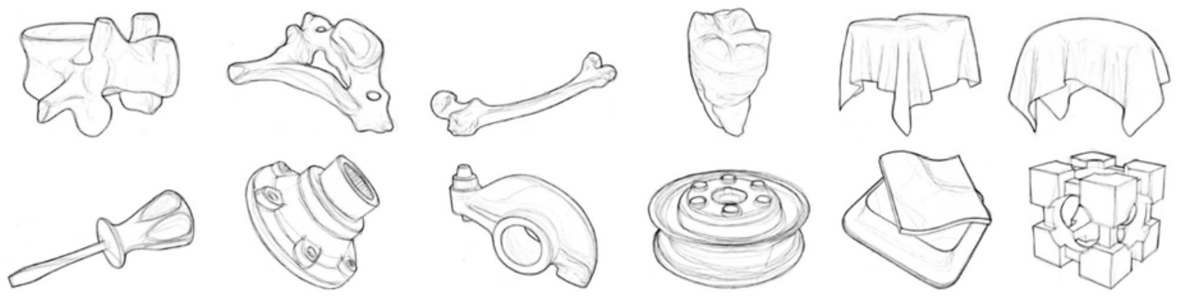


Figure 2.10: *Where people draw lines.* Average images composed of 107 drawings show where artists most commonly drew lines .(Courtesy of Cole et al. 2008)

Besides, there are also some other feature lines [98], including: self intersection lines which are where the surface of the model intersect; parabolic lines which are lines where one of the principal curvatures vanishes; and elevation lines and cutting lines etc.

2.1.5 Where Do People Draw Lines?

Though artists have for centuries studied the principles of how to make line drawings, and a number of feature lines for computer-generated line drawings have been proposed, it is still not clear on *where* to place lines in order to best convey shapes. Aiming to find relationships between the locations of lines drawn by artists and properties of the surface geometry, recently, Cole et al. [13] present very interesting results of a formal study in which artists made line drawings intended to convey specific 3D shapes.

In this study, the artists are asked to draw in two steps: first to draw in a blank area, then to register their drawing to a photorealistic image of the model. Based on these registered drawings, Cole et al. analyze the correlations of locations of lines drawn by an artist with locations of lines drawn by other artists, locations of computer-generated lines, and the underlying differential properties of the surface geometry. Some examples of the average images of the registered drawings are shown in Figure 2.10. The study shows that lines drawn by artists largely overlap one another(75% are within 1mm of another line), particularly along the boundary and occluding contours of the object.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

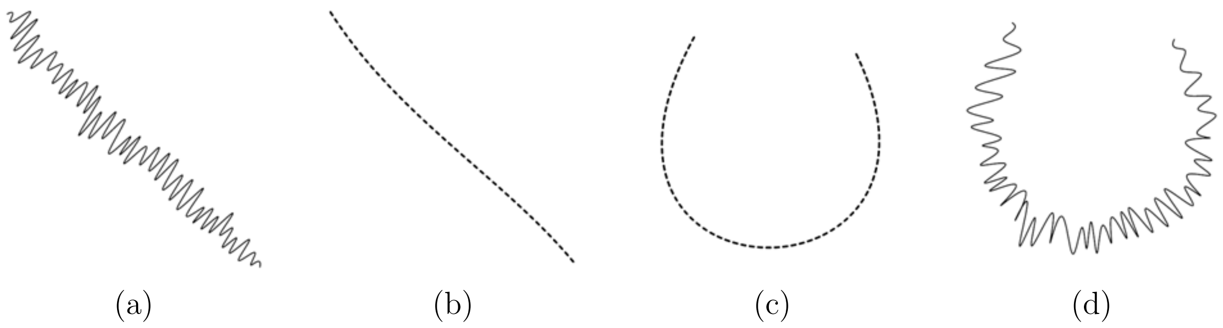


Figure 2.11: Changing the overall sweep of a curve without affecting its character. Given the original curve (a), the system extracts the overall sweep (b). If the user modifies the sweep (c), the system can reapply the detail (d).(Courtesy of Finkelstein et al. 1994)

Most of the lines drawn by the artists in the interior of the object overlap large gradients of the image intensity(PEL is defined based on gradients of intensity, as described in Chapter 3), which correlate strongly with computer-generated lines. 14% of the lines drawn by artists are not well described by any of the local geometry properties. These results provide artists and observers with a precise vocabulary for characterizing where lines on a model are drawn. They will also guide the future development of line drawing algorithms in computer graphics.

2.2 Stylized Line Drawing

One aim of NPR is to mimic handmade visualization styles, automatically or interactively. Normally, artists and illustrators construct their pictures by subsequently adding brush strokes(for an oil painting) or lines(for a pen-and-ink rendition). Thus, it is of utmost importance in modeling these pictorial subunits with their shapes and attributes, and combining them to achieve successful non-photorealistic imageries. In this section, we review some major techniques in representing stylized strokes, and some stylized line drawing systems developed in recent years.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

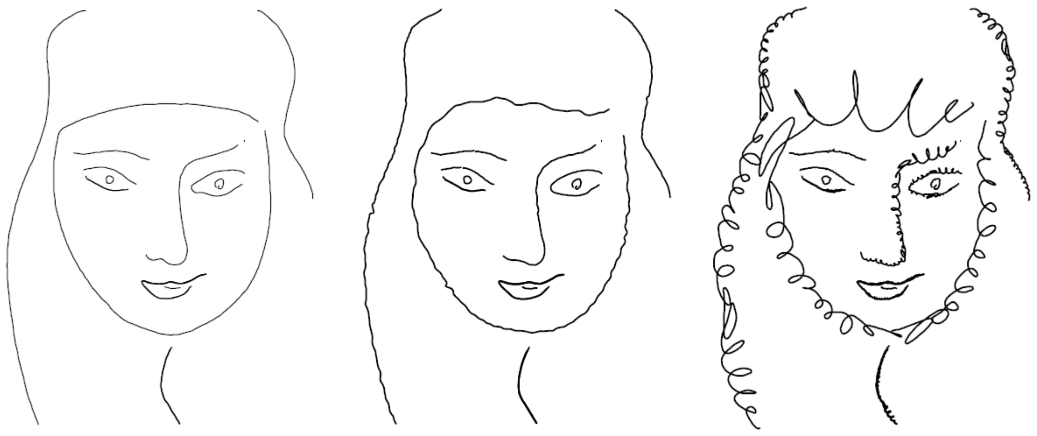


Figure 2.12: Changing the character of a curve without affecting its sweep.(Courtesy of Finkelstein et al. 1994)

2.2.1 Stylized Strokes

Stroke is the fundamental primitive of a line drawing. Each individual stroke has many qualities including path, varying thickness, wiggliness, opacity, and texture etc. These qualities give line drawings much of their character or charm, and can convey feeling as well. There are several categories of techniques in representing stylized strokes.

Multi-resolution Curves. Finkelstein and Salesin [27] propose their multi-resolution curve representation based on wavelets. Their method represents a curve on multiple levels of details and supports a variety of operations which are capable of smoothing a curve; changing the global shape of a curve while maintaining small details; or fitting a curve through a given set of points within a maximum error tolerance. Figure 2.11 shows the changing of the global shape of the curve without affecting its details while Figure 2.12 demonstrates the changing of details of a curve without affecting its global shapes.

This technique offers the most possibilities to influence the geometry of the resulting line. One major drawback, however, is that other visual attributes such as line thickness or line color can not easily be integrated into the model. Hence the use of multi-resolution

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

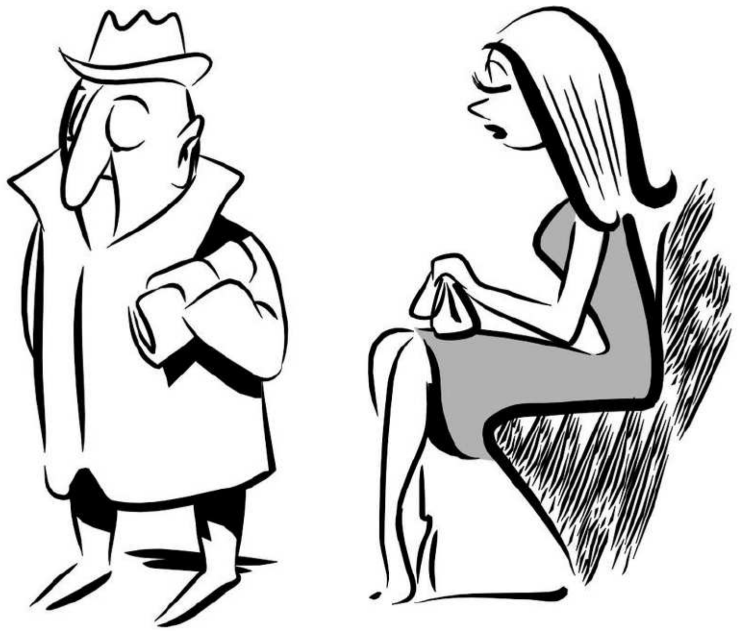


Figure 2.13: Two stylish cartoon figures drawn with skeletal strokes.(Courtesy of Hsu et al. 1994)

curves is well suited for images where the geometry of the line may change but the line width is the same. This is actually the case for pen-and-ink illustration.

Skeletal Strokes. Hsu et al. [49] propose skeletal stroke which is a kind of general brush stroke for changing the shape of pictures as if by bending, shearing, twisting, while conserving the aspect ratio of selected features on the picture. Based on skeletal strokes, a drawing and animation system, 'Skeletal Draw', is developed [48]. Cartoon drawings with skeletal strokes are shown in Figure 2.13.

Many features of skeletal strokes are now standard fare in commercial applications such as Adobe Illustrator. The most challenging task when using skeletal strokes is the design of appropriate strokes for the application at hand. If a library of strokes is available to choose from, the character of an image can be changed rapidly by just selecting a different stroke. And, the major disadvantage of this technique is that information gained from a 3D model is difficult to be mapped onto the stroke, thus limited its applications.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

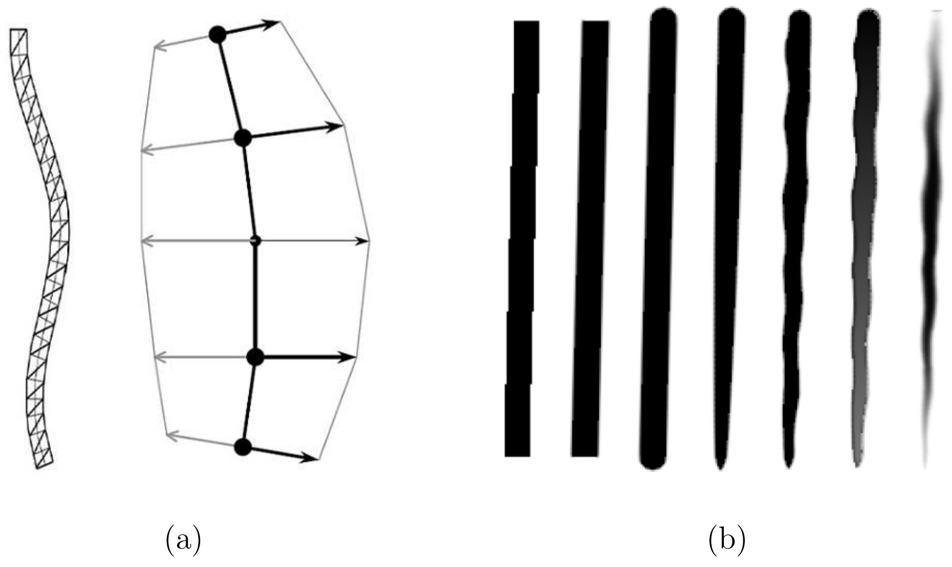


Figure 2.14: Strokes as triangle strips. (a)Constructing strokes by adding width to stroke vertex. (b) The cumulative effects of adding stroke operations, from left to right: raw stroke, antialiasing, taper, flare, wiggle, alpha fade, and texture-mapping. (Courtesy of Northrup et al. 2000)

Strokes as Triangle Strips. Northrup and Markosian [86] describe stylized strokes modeled with long triangle strips which resemble natural media. By leveraging OpenGL's speed and flexibility at drawing triangles, these strokes can be rendered at interactive rates. Stylized strokes represented with triangle strips are widely used efficiently as an important part of interactive rendering systems for 3D models. Furthermore, various styles can be achieved conveniently by texture mapping the triangle strips with current standard graphics interfaces. The stroke model and the cumulative effects of adding stroke operations are shown in Figure 2.14. Since strokes are modeled with piece-wise linear elements, this method has its advantage in high performance. However, to get smoother and more precise strokes, a large number of triangles are needed. This will also slow down the rendering speed of strokes.

Spline based Strokes. Strassmann [112] proposes a method for simulating the effect of ink brushed onto paper in producing sumi-e images. In this method, Strassmann

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

represents strokes with lists of positions and widths. Two cubic B-splines are computed, one for the positions and the other for the pressure values and distance along the stroke. And this results in a series of quadrilaterals representing the body of the stroke. Binh Pham [90] improves Strassmann's method by making greater use of B-splines. He models brushstrokes based on variable offset approximation of uniform cubic B-splines. By being based on B-splines, Pham's system readily lends itself to computer animation. Thierry Pudet [93] further improves his strokes upon Strassmann and Pham. His method works in three steps. First, the digitized trajectory of a stroke is fitted to a Bézier curve. Then, a polygonal approximation of the stroke outline is computed. Finally, the polygonal approximation of the stroke is fitted to a Bézier curve. Comparing with other stroke models, spline based methods represent strokes more precisely and effectively.

2.2.2 Stylized Line Drawing Systems

In recent years, the field of NPR has proposed a variety of techniques to create compelling line drawings. These techniques have also been applied in developing commercial systems, such as Adobe Illustrator, and the in-house NPR system of Disney [117]. In this section, we review some representative stylized line drawing systems with different design and application purposes.

Pen-and-Ink Illustration System. Based on the multi-resolution curve [27] stroke model, Salisbury, Winkenbach and Salesin et al. present pen-and-ink illustration systems for both 2D illustrations and illustrations of 3D models. Salisbury et al. [101] and Winkenbach et al. [131] first describe the principles of traditional pen-and-ink illustration, and present algorithms and data structures for computer generated pen-and-ink illustrations. Salisbury et al. [101, 100, 102] present an interactive image-based system for creating pen-and-ink illustrations. The goal of these works is to provide a non-artist user with high level tools in easily creating illustrations. Salisbury et al. [101] introduce

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

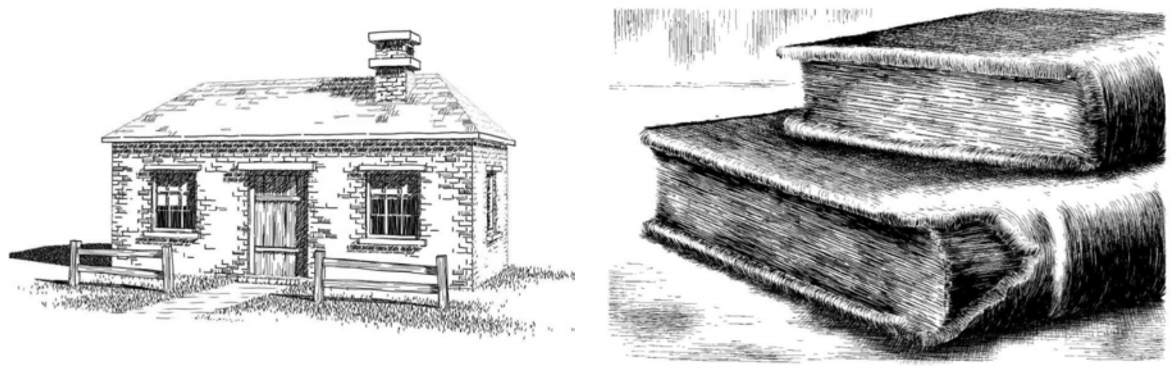

(a) (b)

Figure 2.15: Pen-and-Ink Illustrations. (a)Illustration of 3D shapes.(Courtesy of Winkenbach et al. 1994) (b)Image-based illustrations. (Courtesy of Salisbury et al. 1997)

stroke textures; Salisbury et al. [100] make the resulting illustration scale-independent; while Salisbury et al. [102] render oriented strokes along "painted" vector fields. Winkenbach et al. [131, 132] work in another direction. They introduce two new ideas for creating tone and texture on 3D models: "the prioritized stroke textures" for creating tone and texture on polygonal models, and "controlled density hatching" for free-form surfaces. Illustrations made with these systems are shown in Figure 2.15.

Simulation based Pencil Drawing System. Sousa and Buchanan [7, 109, 108] present a graphite pencil drawing system based on the low-level simulation of the pencil drawing materials and the interaction among these materials. In [109], they present models for graphite pencil, drawing paper, blenders, and kneaded eraser. Their models are based on an observation of how lead pencils interact with drawing paper, and on the absorptive and dispersive properties of blenders and erasers interacting with lead material deposited over drawing paper. Thus, their method can produce realistic looking pencil marks, textures, and tones. In [108], they break the problem of simulating pencil drawing down into four fundamental parts: (1) simulating the drawing materials (graphite pencil and drawing paper, blenders and kneaded eraser), (2) modeling the drawing primitives (individual pencil strokes and mark-making to create tones and textures), (3) simulating the basic

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

rendering techniques used by artists and illustrators familiar with pencil rendering, and (4) modeling the control of the drawing composition. Each part builds upon the others. With all these together, they develop the framework for higher-level pencil rendering systems and tools.

WYSIWYG NPR System. By defining strokes with an extended triangle strips model, Kalnins et al. [57] present a system that lets a designer draw strokes directly on 3D models. The artist chooses a "brush" style, then draws strokes over the model from one or more viewpoints. When the system renders the scene from any new viewpoint, it adapts the number and placement of the strokes appropriately to maintain the original look which imparts a personal aesthetic to the non-photorealistic rendering of the object. One purpose of this system is to provide tools to assist artists in expressing their ideas. Images generated with this system are shown in Figure 2.16 and Figure 2.17. In [58], Kalnins et al. further extend the system to render stylized silhouettes of animated 3D models with temporal coherence.

Programmable Line Drawing System. Inspired by programmable shaders in traditional rendering, Grabli et al. [35] introduce a programmable approach to create line drawing from 3D models. Similar to traditional shaders, the style of a line drawing can be specified by implementing procedures that describe how the feature lines from the 3D model should be turned into strokes. Comparing with interactive drawing systems which depend highly on the proficiency of the users to generate good images, and automatic systems which provide user limited controls with a set of parameters, this system provides artists and programmers more flexibility and power in the specification of pictorial style. The rendering of this system is shown in Figure 2.18.

2.2.3 Others

Besides the above mentioned stylized strokes representation methods and stylized rendering systems, there is still a large amount of research in this area. In this section, two

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

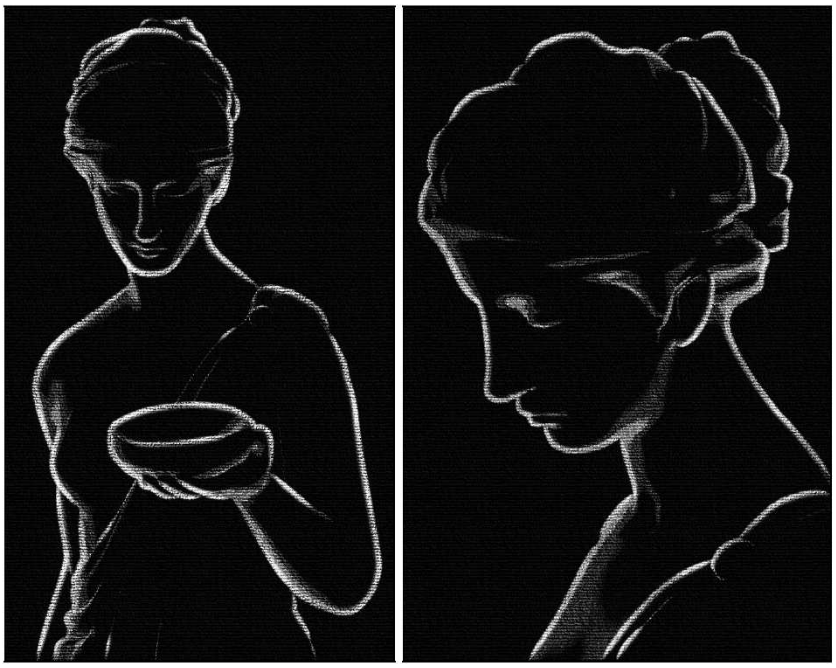


Figure 2.16: Wide silhouette strokes and a subtle toon shader are rendered over a coarse paper texture.(Courtesy of Kalnins et al. 2002)

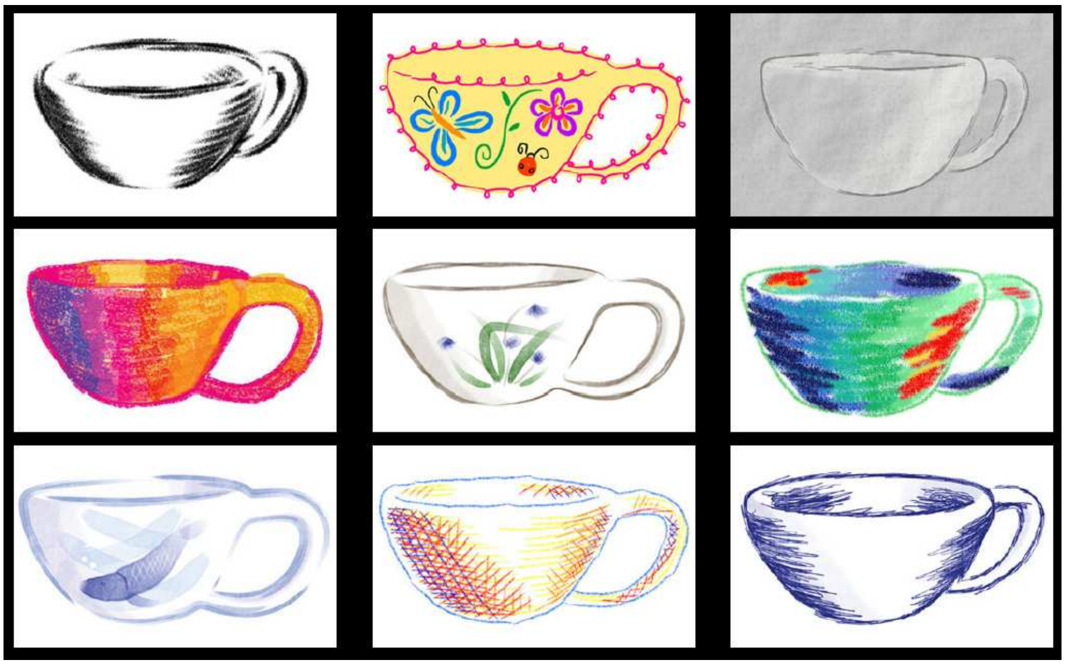


Figure 2.17: Cups rendered in different styles.(Courtesy of Kalnins et al. 2002)

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

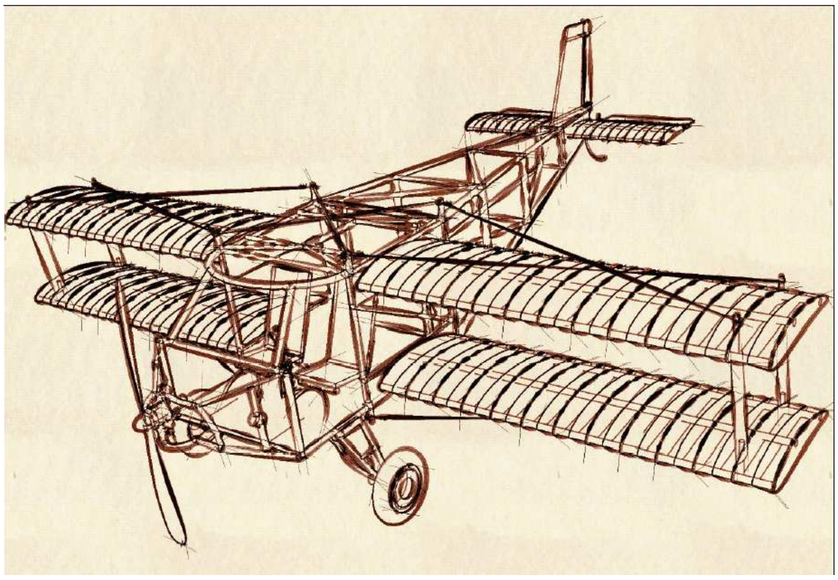


Figure 2.18: Programmable line drawing. A sketchy style based on the use of construction lines.(Courtesy of Grabli et al. 2004)

recent works that are closely related to our work are described. In NPAR 2007, Goodwin et al. [33] propose a method of determining stroke thickness based on the isophote distance. This method improves the quality of strokes and can be conveniently incorporated into various 3D illustration systems. Different from the above mentioned techniques, Lee et al. [66] describe a GPU-based algorithm for rendering a 3D model as a line drawing, based on the insight that a line drawing can be understood as an abstraction of a shaded image. Thus they render lines along tone boundaries or thin dark areas in the shaded images.

2.3 Stylized Shading

A fundamental method for conveying the 3D structure of an object in a 2D image is shading. In traditional illustrations, stylized shading is extremely important in giving strong cues of shapes and structural details. In this section, we review a small set of stylized shading techniques which are closely related in illustrating shapes, including

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

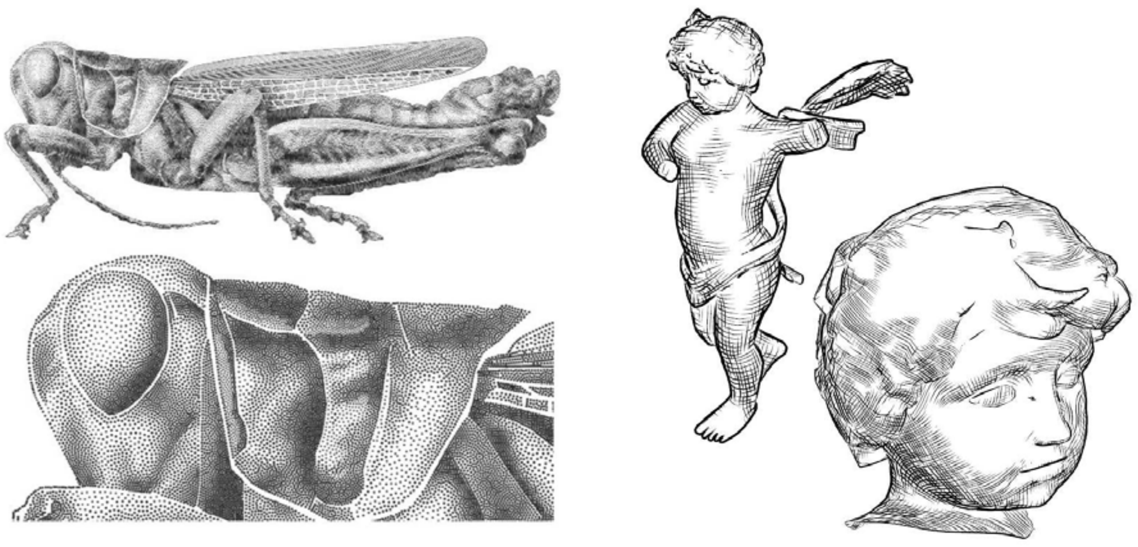


Figure 2.19: Stippling and Hatching. (a)A stippled image of a grasshopper.(Courtesy of Deussen et al. 2000) (b)Line hatching of the Cupid mesh. (Courtesy of Hertzmann et al. 2000)

methods of cartoon shading, stippling, hatching, and learning styles from examples.

Cartoon Shading. Lake et al. [64] first describe the method of toon shading and develop a hard-shading algorithm which is similar to the cel animator's process of painting an already inked cel. Based on toon shading, Anjyo et al. [2] describe techniques to add stylized highlights to a classic toon shader. In [3], Barla et al. introduce "X-Toon" which extend the basic toon shader with view dependant effects by using 2D textures. Recently, Todo et al. [119] introduce a method for locally controllable stylized shading which provide artists the ability to add intentional, but often unrealistic, shading effects. This method demonstrates powerful user controllable cartoon shading for images and animations.

Stippling. Stippling is quite common in traditional, handmade illustrations. With stippling, tone as well as texture are created by the placement of small dots. Deussen et al. [20] generate stipple drawings by first placing stipples roughly on the input image and then relaxing them using Lloyd's algorithm until they are well-spaced. A stippled image

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

of a grasshopper generated with this method is shown in Figure 2.19 (a). Secord [105] improves this method with non-interactive techniques for generating stipple drawings from grayscale images using weighted centroidal Voronoi diagrams. Pastor et al. [89] propose a general framework that produces view-dependent, frame-coherent animations in the stippling style in real time. Differently, Yuan et al. [139] generate stippling in geometry image space.

Hatching. Line art is one of the most common illustration styles. Employing variable-density hatching and complex hatch patterns, artists convey information about shape, texture and lighting. In pen-and-ink illustrations, [131, 101] show how to use stroke textures to convey a certain darkness for shading with pen-and-ink lines. To create hatching renditions from 3D scenes, Elber [26] and Interrante [50] use principal curvature directions for hatching which are proven to be excellent indicators of 3D shapes. Since curvatures can not be reliably computed in many points on a smooth surface, Hertzmann and Zorin [46] make use of the principle curvature directions, and apply an optimization technique to "fill in" the hatching field where it is poorly-defined. Illustration with line hatching generated with this method is shown in Figure 2.19 (b). In [141], Zander et al. improve this method to generate hatching lines from polygonal meshes and render them in high quality either at interactive rates for on-screen display or for reproduction in print. In [92], Praun et al. propose the Tonal Art Map (TAM) to generate real time hatching on arbitrary surfaces with spatial and temporal coherence. Webb et al [126] further improve the real-time hatching scheme leveraging texture mapping hardware. Recently, Yen et al. [137] propose painterly art maps (PAMs) which works particularly well for a rich variety of brush strokes ranging from line art strokes to very complicated ones with significant variations in stroke characteristics.

Learning styles from examples. A great deal of work has been introduced in creating artistic shading styles. Mostly, these methods are specifically tailored to a specific

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

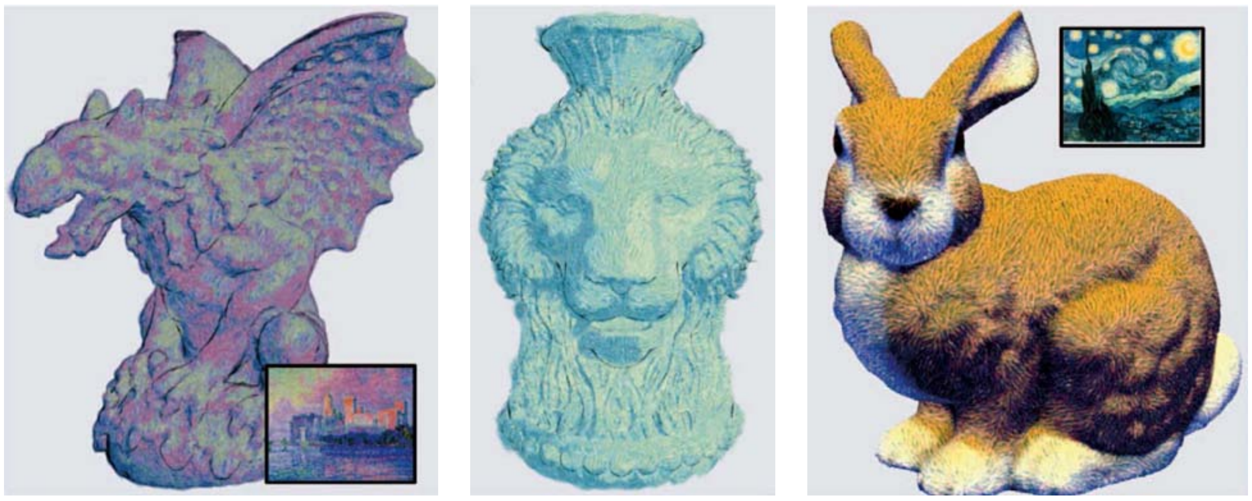

(a) (b) (c)

Figure 2.20: Stylized rendering with PAMs. (a) Rendering with "Palais des Papes Avignon" painted by Paul Signac. (b) and (c) Rendering with "The Starry Night" painted by Van Gogh. Originally, the models in (a) and (b) are uncolored, and the model in (c) is colored.(Courtesy of Yan et al. 2008)

rendering style. On the contrast, example based methods can be used for a very broad range of effects, which are well developed in recent years. By defining artistic shading styles as sample textures, the problem of learning artistic styles from examples can be reduced to the problem of texture synthesis. A major part of these example-based techniques [118, 24, 69, 45, 30, 107] are based on constrained texture synthesis. Representatively, Efros et al. [24] propose the image quilting method. It is applied in stylized shading synthesis with a correspondence map which enforces additional constraints during the texture synthesis process. In the seminal work of Hertzmann et al. [45], a more general image analogies framework is proposed which learns shading styles from image pairs in pixel-wise correspondence. By choosing different types of source image pairs as input, the framework supports a wide variety of "image filter" effects, including traditional image filters, improved texture synthesis, super-resolution and artistic filters etc. In stroke based rendering, the direction of brush strokes is an important factor of painting styles which is not emphasized in the above methods. By defining brush stroke directions

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

with the medial axes of segmented regions, Wang et al. [124] propose an example-based painting method which uses a hierarchical patch-based approach to the synthesis of directional textures. As mentioned in the above, the painterly art maps (PAMs) [137] are also generated from samples of a painted image. Incorporating PAMs with an abstract painterly rendering framework [11] which models shapes with 3D point set represented with a multi-scale segmented sphere hierarchy, stylized images and animations in the style of a given artwork can be achieved. Examples generated with this method are shown in Figure 2.20.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

Chapter 3

Effective Shape Illustration with PELs

Conveying shapes using feature lines is an important tool in visual computing. The existing feature lines (e.g., ridges, valleys, silhouettes, suggestive contours, etc.) are solely determined by local geometry properties (e.g., normals and curvatures) as well as the view position. Strongly inspired by the observation in human vision and perception that a sudden change in the luminance plays a critical role to faithfully represent and recover the 3D information, we adopt the edge detection techniques in image processing to generate line drawings for 3D shape illustration, and present *Photic Extremum Line* (PEL) which emphasizes significant variations of illumination over 3D surfaces. Comparing with the existing feature lines, PELs are more flexible and offer users more freedom to achieve user controllable illustrations and visualization effects. In addition, the user can easily control the illustration of shapes by changing the light position, the number of light sources, and choosing various light models. In this chapter, we compare PELs with existing approaches and demonstrate that PEL is a flexible and effective tool to illustrate 3D surface and volume for visual computing.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

3.1 Introduction

Throughout history and technological evolution, scientific illustration has been widely used to depict the most salient features for scientific data and convey shapes in an accurate and intuitive way. Knowledge of human cognition shows that, so frequently an artistic drawing or painting of a same scene is proved to be both more effective in communication and more pleasing in visual experience than a photograph [47] [123] [110] [32]. This is mainly because of the abstraction nature of various artistic styles. Among various techniques used in scientific illustration, line drawings have been proven to be an effective and commonly-used way to achieve this goal because they have the capability to display information more efficiently by ignoring less important details. By taking advantage of human visual acuity, line drawings can represent a large amount of information in a relatively succinct manner, which enables them to be more expressive than photographs [98].

In computer graphics and visualization, extensive research has been carried out in computer-generated illustration with different types of feature lines. So far, the commonly-used feature lines on 3D shapes, e.g., contours (the external and internal silhouettes), ridge-valley lines [87], suggestive contours [16], are solely defined by local geometrical properties, such as surface normal and curvatures, and/or the view position. However, an important perceptual observation shows that human perception is highly sensitive to edge-like features, i.e., points of high *luminance variation*. It is well-known in psychology that the human perceptual system interprets natural scenes with a wealth of visual cues including luminance, color opponency, and orientation which are proven to play a vital role in human perception and a sudden spatial change in any of these factors gives rise to features [88]. This has led to great success in the research of edge detection in image processing. In 2D images, edges, characterizing the sudden change in the intensity, are

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

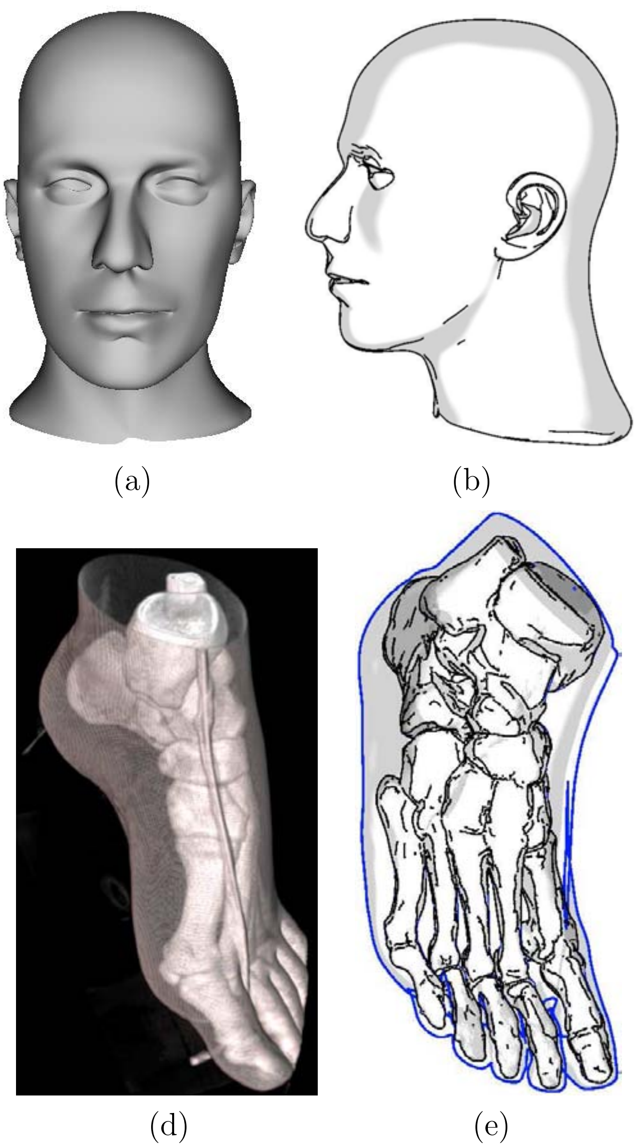


Figure 3.1: Photic Extremum Lines (PELs) convey surfaces and volumes more effectively in a perceptually correct way. Two isosurfaces are extracted from volumetric datasets. The contours of the outer surface are drawn in blue. PELs of the inner surface are drawn in black. Toon shading is used in images shown in the middle.

one of the most important properties of objects since they correspond to the changes in geometric or photometric properties of modeled scenes.

Strongly inspired by the edge detection techniques for 2D images, we present the Photic Extremum Line(PEL), a new type of feature lines which characterizes the sudden

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

change of illumination on 3D shapes. Since the illumination depends on the view position, light models, material, texture, and geometric properties, PELs are more general than the existing feature lines. More importantly, PEL provides the user more freedom to control the desired illustration by manipulating the above parameters. As shown in Fig. 3.1, PELs illustrations are visually pleasing and convey the 3D surface and volume effectively. Although originated from image processing, PEL is different from image space edge detection in that PEL is computed in object space. The pixel-based representation of feature lines in image space will suffer from low precision due to the loss of 3D scene information during rendering, and is not suitable for further processing, such as stylization. In contrast, PEL does not have these constraints since PEL is defined in object space, thus, the user can totally control the illumination by manipulating the light which in turn helps the user to achieve the desired illustration.

Our main contributions are:

(i) We propose the PEL on 3D surfaces, a new type of feature lines which characterizes the sudden change of the luminance. We give the mathematical definition of PEL and demonstrate the effectiveness of PEL with results for both 3D surfaces and volume data.

(ii) We show that PEL is view and light dependent and develop the techniques to determine the optimal setting of light which can help the user to achieve the most desirable illustration.

(iii) We compare PEL with the existing feature lines thoroughly and demonstrate that PEL is a flexible and powerful tool to illustrate 3D shapes for better visualization.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

3.2 Photic Extremum Lines (PELs)

3.2.1 Definition

Edge detection is a well-studied problem in computer vision [9][75]. In 2D images, an edge point is defined as a point at which the gradient magnitude assumes a maximum in the gradient direction. The observation in [140] shows that for a grey-scale image of an illuminated 3D object under general illumination and reflection conditions, the zero-crossings of the second order directional derivative of the image intensity along the direction of the intensity gradient occur near the ridges and valleys of the 3D object. Inspired by the edge detection in image processing, we define Photic Extremum Lines which characterize the local variation of illumination directly on 3D surfaces.

Definition: The PEL is a set of points on the 3D surface where the variation of illumination in the direction of its gradient reaches the local maximum.

Given a smooth surface patch $S : D \subset \mathbb{R}^2 \to \mathbb{R}^3$, assume the illumination function $f : S \to \mathbb{R}$ is C^3-continuous. The gradient of f is given by [21]:

$$\nabla f = \frac{f_u G - f_v F}{EG - F^2} S_u + \frac{f_v E - f_u F}{EG - F^2} S_v \qquad \text{(Eq. 3.1)}$$

where S_u and S_v are the tangent vectors, E, F, and G are the coefficients of the first fundamental form of S. Given arbitrary point $\mathbf{p} \in S$, denote $\mathbf{w}$ the unit vector of the gradient of f, i.e., $\mathbf{w} = \frac{\nabla \mathbf{f}(\mathbf{p})}{||\nabla \mathbf{f}(\mathbf{p})||}$. Then, the PEL is defined as the set of points satisfying:

$$D_{\mathbf{w}}||\nabla f|| = 0 \quad \text{and} \quad D_{\mathbf{w}} D_{\mathbf{w}}||\nabla f|| < 0 \qquad \text{(Eq. 3.2)}$$

The entire solutions of $D_{\mathbf{w}}\|\bigtriangledown f\| = 0$ form closed curves or curves that terminate at the surface boundaries. Thus, the surface can be divided into two regions, i.e., $D_{\mathbf{w}}\| \bigtriangledown f\| > 0$ (colored in green in Fig. 3.2(d)) and $D_{\mathbf{w}}\|\bigtriangledown f\| < 0$ (colored in yellow in Fig. 3.2(d)). PELs (blue curves in Fig. 3.2(b)) correspond to part of the curves where $D_{\mathbf{w}} D_{\mathbf{w}}\| \bigtriangledown f\| < 0$, i.e., the variation of the illumination reaches its local maxima, which effectively convey useful information about the shape.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

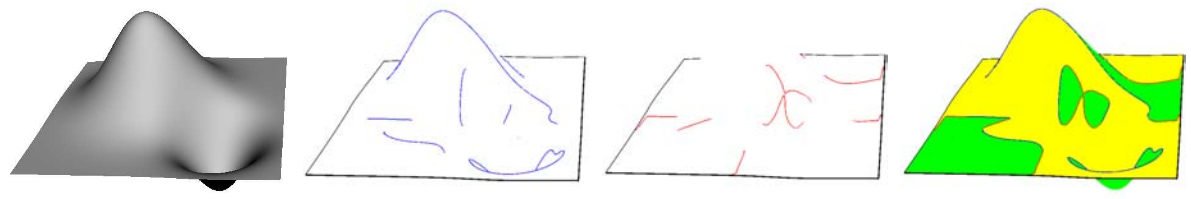

(a) (b) (c) (d)

Figure 3.2: The entire solutions of $D_{\mathbf{w}}\| \bigtriangledown f\| = 0$ form closed curves or curves that terminate at the surface boundaries. These curves divide the surface (a) into two regions, i.e., $D_{\mathbf{w}}\| \bigtriangledown f\| > 0$ (colored in green in (d)) and $D_{\mathbf{w}}\| \bigtriangledown f\| < 0$ (colored in yellow in (d)). PELs (blue curves in (b) and (d)) correspond to part of the curves where $D_{\mathbf{w}}D_{\mathbf{w}}\| \bigtriangledown f\| < 0$, i.e., the variation of the illumination reaches the local maxima. The red curves in (c) and (d) correspond to the points satisfying $D_{\mathbf{w}}\| \bigtriangledown f\| = 0$ and $D_{\mathbf{w}}D_{\mathbf{w}}\| \bigtriangledown f\| > 0$ which do not convey the useful information about the shape. Note that the curves on the hidden surface are removed in (b) and (c).

3.2.2 Lighting Dependency of PELs

Although originated from image processing, the key difference between PEL and edge detection is that the illumination on the 3D surface is totally user-controllable, i.e., the user can achieve the user controllable illustration by manipulating light freely. We will explain this in Sec. 3.2.3 in details.

The illumination of 3D shapes depends on lighting conditions. Among various light models available in graphics, a commonly-used one is Phong specular-reflection model [41]:

$$I = I_{amb} + I_{diff} + I_{spec} = k_a I_a + k_d I_d(\mathbf{n}\cdot\mathbf{l}) + k_s I_s(\mathbf{v}\cdot\mathbf{r})^{n_s}, \qquad \text{(Eq. 3.3)}$$

where $\mathbf{n}$, $\mathbf{l}$, $\mathbf{v}$, $\mathbf{r}$ are the surface normal, light vector, view vector, and reflection vector, respectively. Since Eq. 3.3 is a function of the light and view vectors, the corresponding PELs are view and light dependent. Figure 3.3 shows various PELs illustrations of the smooth surface in different lighting conditions. Figure 3.4 further demonstrates the lighting dependency of PELs illustrations.

PELs convey shapes by emphasizing significant variations of illumination over 3D surfaces. Aiming at 3D shapes illustration with concerns of human perceptual experi-

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

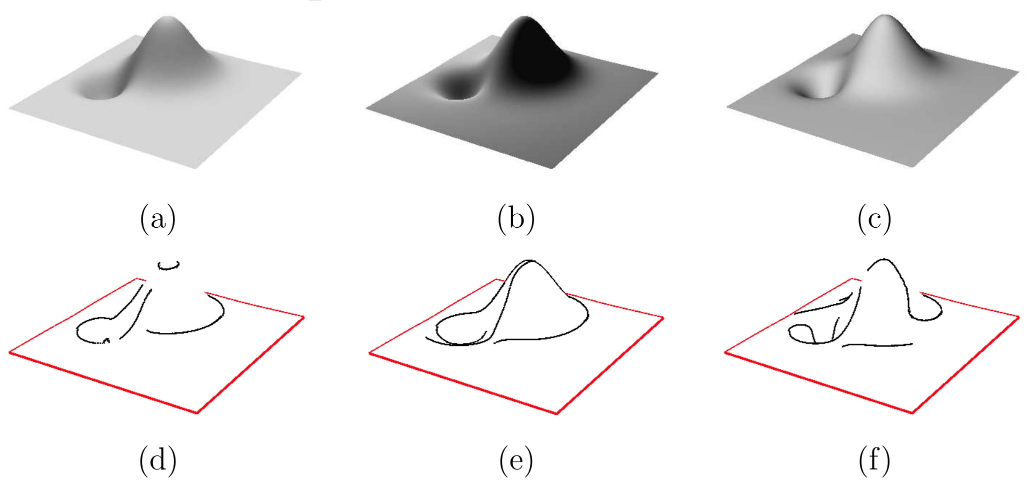


Figure 3.3: PELs illustrations of the smooth surface when applying lights in different directions. The top row shows the shaded smooth surface with lighting from the top, back and front(from left to right). The bottom row shows the corresponding PELs illustrations in different lighting conditions.

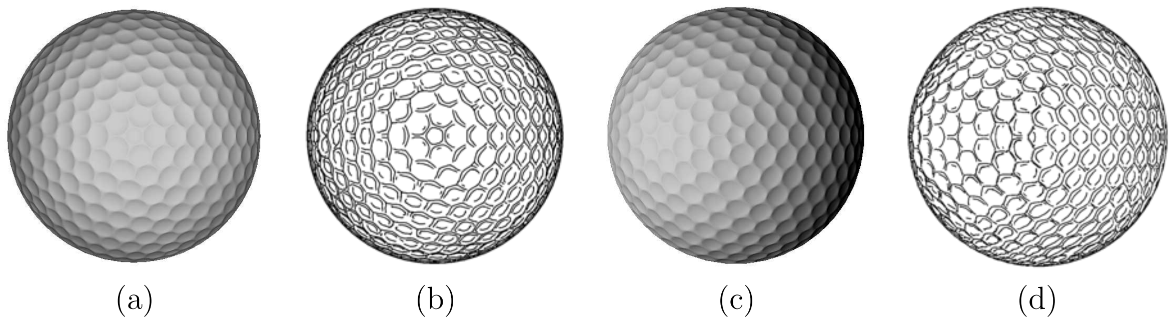


Figure 3.4: Light dependent PELs illustrations. (a) is the shaded golf ball which is lightened from the front. (b) shows the PELs illustration corresponding to (a). In (c), the golf ball is lightened from the left. Corresponding to (c), (d) demonstrates different but perceptually consistent PELs illustration comparing with (b).

ences, the light model should emphasize illumination variations strongly related to shape variations, but suppress those less related or orthogonal to shape variations. In Phong shading model, the ambient light is constant which does not contribute to the variation of illumination. The diffuse light is determined by both shape and light. The variation of the diffuse light is highly affected by the variation of the surface normal. The specular light is dominated by the viewing and lighting conditions, which can be considered an

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

orthogonal factor of the 3D surface. Thus, the specular light may introduce feature lines which are less related to the 3D shape and distracting for 3D shape illustration. At the same time, the power factor of the specular light is more computational expensive. So we ignore the ambient light and the specular light, and consider the diffuse light only,

$$I = k_d I_d (\mathbf{n} \cdot \mathbf{l}) \qquad \text{(Eq. 3.4)}$$

Note that the view vector is not involved in Eq. 3.4, thus, the corresponding PELs are independent of view position. However, the view dependent property is highly desirable in the real-world applications. More importantly, for real-world 3D shapes with complicated geometry and many details, the criteria to locate the feature lines usually vary in different parts and are rather subjective. Thus, it is also highly desirable to provide users more freedom to control the illustration. To satisfy the view dependence as well as the user controllability and performance requirements, we propose the following light model for PEL:

- Main light: we set the main light source using a directional light whose light rays are parallel to the view vector. It is similar to what we call the "head light", which is widely used in the lighting system of vehicles. This setting is based on human perceptual experience. Thus, the light moves when the view point changes. As a result, the illumination and the corresponding PELs are view dependent.
- Auxiliary lights: we use *optional* spot lights to highlight the user-specified areas. The contribution of auxiliary lights to the illumination is local and independent of the view position.

The auxiliary spot lights are also defined with $\mathbf{n}\cdot\mathbf{l}$, but applied locally to user interested regions, which provide the user freedom to control PELs in different areas. Fig. 3.5 illustrates the proposed light model for PEL. Four lights (Fig. 3.5(a)) are used to illustrate

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

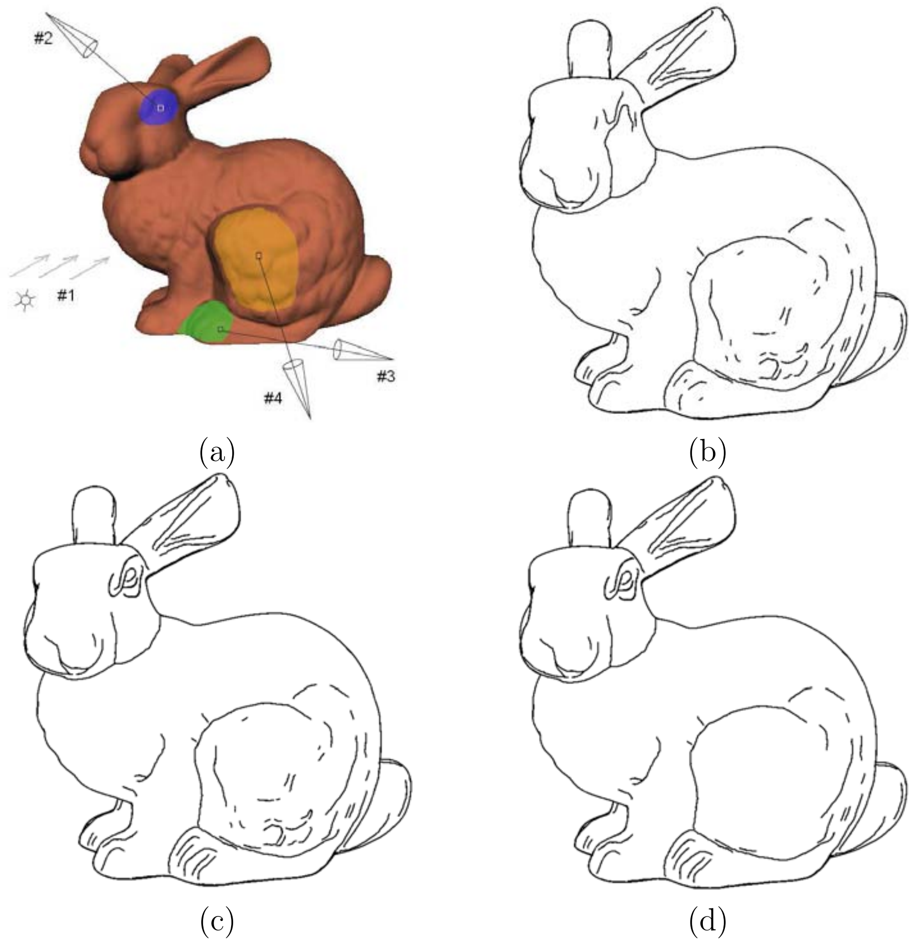


Figure 3.5: Four lights (shown in (a)) are used to extract PELs from the Stanford Bunny. The main light #1 is a directional light whose direction coincides with the camera direction. By using the main light #1 alone, PELs convey the rough shape very well. However, some important lines on the eyes and feet are missing in (b). Then we use auxiliary spot lights #2 and #3 to *increase* the local contrast on eyes and feet to *add* more details (shown in (c)). Note that the Bunny body contains many short feature lines due to the small reliefs. These lines, however, are less favorable by the user. To *remove* these lines, we add another auxiliary spot light #4 to *decrease* the local contrast on the body. As a result shown in (d), most of the small feature lines are removed successfully.

the Stanford Bunny. Light #1 is the main light, whose direction coincides with the view direction. The purpose of the main light is to convey the rough geometry (Fig. 3.5(b)). However, some important lines, such as eyes and feet, are missing. Also, due to some small features, there exist many small PELs which cause distractions and are not helpful

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

to convey the shape. Then, auxiliary lights are used to increase or decrease the local contrast of the specified regions, thus, PELs can be added or removed accordingly. For example, two auxiliary lights #2 and #3 are used to increase the contrast on the eye and foot (Fig. 3.5(c)). Another auxiliary light #4 is used to decrease the contrast on the body. As a result, we achieve better PELs shown in Fig. 3.5(d).

3.2.3 Optimal Lighting for PELs

For simple shapes, one can usually achieve the desired illustration using the main light alone. However, most of the real-world models have complicated geometry and many details, which can not be illustrated using a single light. Thus, we need to use auxiliary lights to improve the illustration. As mentioned above, the key advantage of PEL is that users can fully control the auxiliary lights, such as the number of lights, their intensities, positions, and directions. It is usually a tedious work to manipulate these parameters manually. To minimize the effort of user interaction, we develop a technique to compute the optimal setting of the auxiliary lights automatically.

We assume that the user specifies several regions of interest to be improved. Each region is small enough that we can use just one auxiliary light. Now, the problem can be stated as follows:

Given a user-specified patch $\bar{S} \subset S$, $\mathbf{c} = \frac{\int_{\bar{S}} S dA}{\int_{\bar{S}} dA}$. We want to find a spot light l_{aux} at position $\mathbf{x}$ with light direction $\mathbf{d} = \frac{\mathbf{x}-\mathbf{c}}{||\mathbf{x}-\mathbf{c}||}$ such that the contrast of illumination on $\bar{S}$ is either maximized if one wants to add details or minimized if one wants to remove lines, i.e.,

$$\max \iint_{\bar{S}} ||\nabla f||^2 dA, \quad \textit{to add details}, \qquad \text{(Eq. 3.5)}$$

$$\textit{or} \quad \min \iint_{\bar{S}} ||\nabla f||^2 dA, \quad \textit{to remove details}, \qquad \text{(Eq. 3.6)}$$

where

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

$$\begin{aligned} f &= I_{aux}, \\ I_{aux}(u,v) &= \frac{k_{aux} I_{aux}}{k_c + k_l d + k_q d^2} \mathbf{n}(u,v) \cdot \mathbf{l}, \\ \mathbf{l} &= \frac{\mathbf{x} - S(u,v)}{\|\mathbf{x} - S(u,v)\|}, \\ d &= \|S(u,v) - \mathbf{x}\| \end{aligned}$$

Note that the cut-off angle of the spot light gives rise to the discontinuity of the illumination which in turns would result in unnecessary feature lines. To avoid this, we require that the cut-off angle of the spot light is large enough to cover $\bar{S}$. For each point inside the spot light region (which is larger than $\bar{S}$), two illumination functions $f = I_{main}$ and $f = I_{aux}$ are defined. To compute derivatives of illumination, $f = I_{aux}$ is used for any point $\mathbf{p} \in \bar{S}$, while $f = I_{main}$ is used for any point $\mathbf{p} \in S \setminus \bar{S}$. Therefore, though the overlap of the two functions results in discontinuities, we can guarantee local continuity everywhere by switching between the two functions. Since PELs are defined with local extremum and the illumination is locally continuous for all the points, the auxiliary light will improve the PELs illustration of $\bar{S}$ without introducing unnecessary lines on the boundary.

We would point out that we set the illumination function to be $f = I_{aux}$ for the optimization of the auxiliary light and the extraction of PELs in $\bar{S}$. The optimal auxiliary lights are located in object space. Thus, the optimal setting of the auxiliary lights are not affected by the change of the main light and various viewing conditions. Though the optimization process is not real time, once computed, the auxiliary lights are fixed in the object space. Therefore, additional computational overheads are avoided. More importantly, the auxiliary lights improve the PELs illustration without causing any incoherence in animation with changing viewing conditions.

Fig. 3.6 illustrates the optimal lighting for the Bunny model. Note that the discontinuity of the illumination occurs in the shaded images (Fig. 3.6 (b) and (c)). With

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

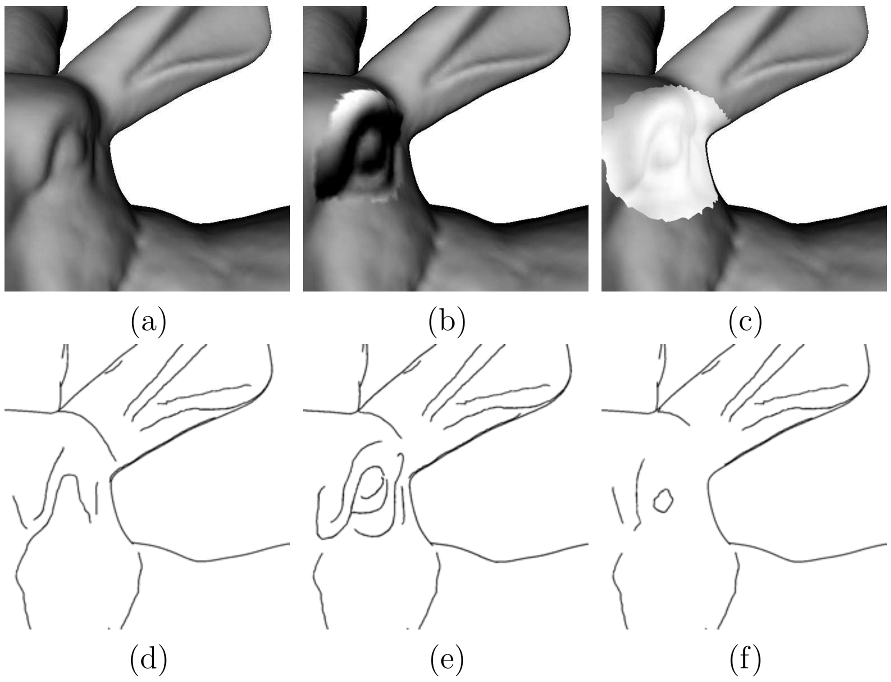


Figure 3.6: An example showing improvement of the PELs illustration on the Bunny eye region using optimal lighting. Using the main light alone, the eye is not illustrated well due to the low contrast around the eye (see (a)). To improve this, we add a spot light which covers the eye and maximizes the contrast (see (b)). As a result, the features are enhanced and the corresponding PELs illustrate the eye very well (see (e)). (c) and (f) show the effect of using a spot light which *minimizes* the local contrast on the eye. As expected, the feature lines are removed accordingly.

the guarantee of local continuity for PELs extraction as mentioned above, user-desirable improved PELs illustration of the bunny eye is achieved without introducing meaningless feature lines on the boundaries (Fig. 3.6 (e) and (f)). It also demonstrates that PELs can *NOT* be extracted from image space. Using any edge detection algorithms, the aforementioned discontinuity of illuminance will definitely result in an edge, which is not acceptable. This also explains one important difference between PEL and the edge detection in image processing. Another example is shown in Figure 3.7 to demonstrate the improvement of PELs illustration with optimal lighting. Note that the auxiliary light of Figure 3.7 lightens the whole face with low intensity which emphasizes the regions of the eyes with more complicated geometrical structures but minimizes the effects of other

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

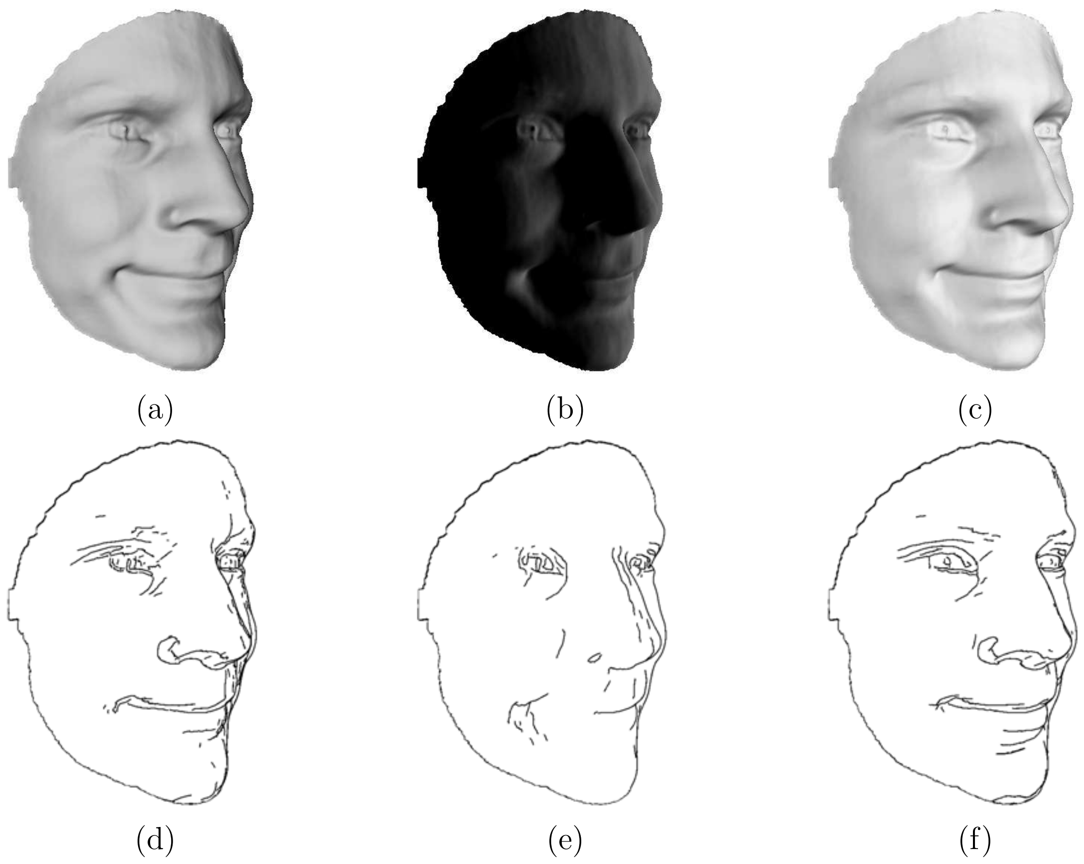


Figure 3.7: Improving PELs illustration with optimal lighting. The PELs illustration (d) is not satisfy with the main light alone as shown in (a). To improve the PELs illustration, an optimal auxiliary light is computed which locates far away at the right-back side of the face model to increase the contrast. The shading and the PELs of the auxiliary light are shown in (b) and (e) respectively. (c) shows the shading with both lights. (f) shows the PELs illustration which has been improved significantly.

regions automatically.

Table 3.1: Comparison of feature lines

Technique	Equation	Derivatives	Dependency
Contours	$\mathbf{n} \cdot \mathbf{v} = 0$	1st order	view
Suggestive Contours	$D_{\mathbf{w}}(\mathbf{n} \cdot \mathbf{v}) = 0$	2nd order	view
Ridges & Valleys	$D_{\mathbf{t_{max}}} k_{max} = 0$ $D_{\mathbf{t_{min}}} k_{min} = 0$	3rd order	N/A
PELs	$D_{\mathbf{w}} \lVert \nabla f \rVert = 0$	3rd order	light & view

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

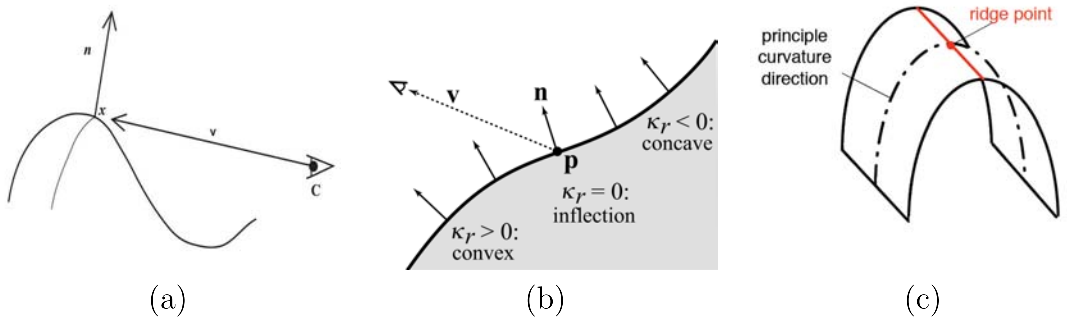


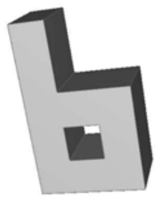
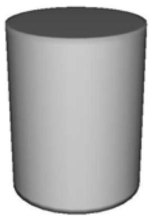
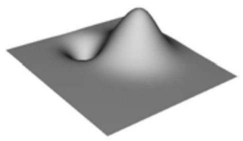
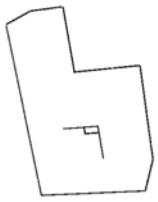
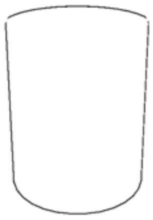
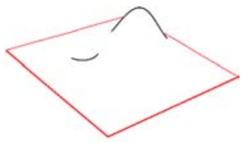
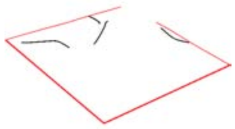
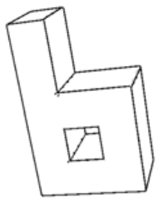
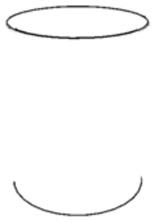
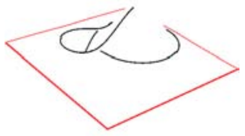
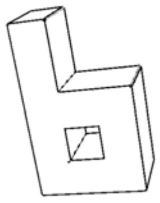
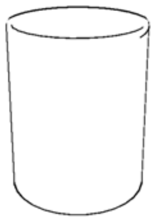
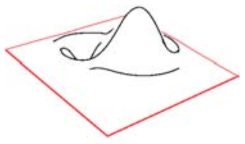
Figure 3.8: Definition of feature points. (a) Contours points are defined where the surface normal is perpendicular to the view vector(by courtesy of Hertzmann). (b) Suggestive contour appears at the inflection point on the radial curve(by courtesy of DeCarlo). (c) Location of ridge line on a surface(by courtesy of Judd).

3.3 Comparisons of PELs with Previous Feature Lines

This section compares PELs with several previously proposed, commonly-used feature lines: contours, suggestive contours, and ridge-valley lines. These feature lines are solely determined by local geometry properties (e.g., normals and curvatures) as well as the view position. PELs, however, depend on view position and illumination (which in turns depends on local geometric properties). Table 3.1 summarizes the properties of contours, suggestive contours, ridges & valleys, and PELs. In [98], a classification of feature lines based on differential properties is proposed, which considers surface normal as a first order differential quantity of surfaces. In this way, suggestive contours are defined with radial curvature which is a second order derivative. Ridge-valley lines are third order derivatives of surfaces. Similarly, in our proposed light model, PELs are defined as second order derivative of $\mathbf{n} \cdot \mathbf{l}$, which turns out to be third order derivative of surfaces as well. Table 3.2 shows the comparisons of PELs with these feature lines for some simple shapes. More detailed comparisons can be found in the following subsections.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

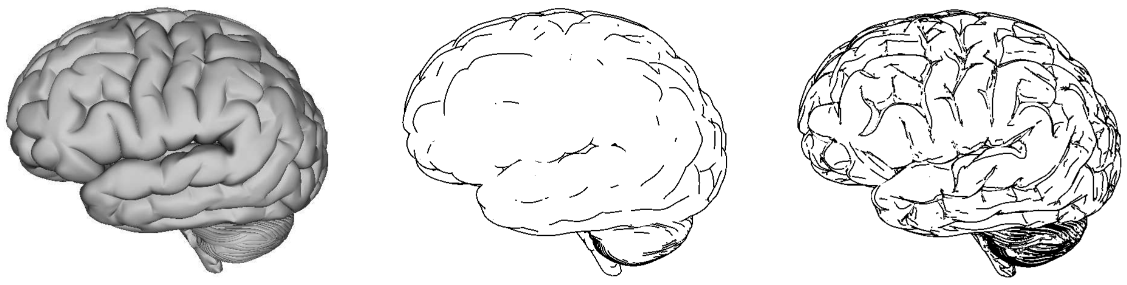
Table 3.2: Comparisons of Feature Lines for Some Simple Shapes

	B-Shape	Cylinder	Smooth Surface
Shaded Model			
Contours			
	⋆⋆⋆	⋆⋆⋆	⋆⋆
Suggestive Contours	NO OUTPUT	NO OUTPUT	
	⋆	⋆	⋆⋆
Ridges and Valleys			
	⋆⋆⋆⋆⋆	⋆⋆⋆	⋆⋆⋆
PELs			
	⋆⋆⋆⋆⋆	⋆⋆⋆⋆⋆	⋆⋆⋆⋆⋆

3.3.1 PELs versus Contours

Contours, one of the widely used feature lines, are lines where a surface turns away from the viewer and becomes invisible. Contours are defined as the set of points with

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library



(a)Shaded model (b) Contours (c) PELs

Figure 3.9: Comparison of PELs with contours (b)shows that contours can not convey the structure and complexity of the interior. (c) shows the expressiveness of PELs.

$\mathbf{n} \cdot \mathbf{v} = 0$, as shown in Fig. 3.8(a) where the surface normal **n** is perpendicular to the view vector **v**. From a generic viewpoint **c**, the contours generator consists of a set of disconnected loops on the surface, while contours are the visible portions of these loops. Although contours show strongest cues with model-to-background distinction, contours alone are quite limited, since they can not capture the structure and complexity of the shape interior. An example is shown in Fig. 3.9, the contours fail to show the interior geometry.

3.3.2 PELs versus Suggestive Contours

Suggestive contours are lines drawn on clearly visible parts of the surface, where a true contour would first appear with a minimum change in view point. Thus suggestive contours are features where a surface bends sharply away from the viewer, yet remains visible. They are almost contours which become contours in nearby views. Suggestive contours are considered as the extension of the actual contours of surfaces. In [16], three mathematically equivalent definitions are given to intuitively characterize suggestive contours. Given an arbitrary point **p** with normal **n** on a smooth surface S, the view vector **v** is projected onto the tangent plane of **p** to obtain **w**. The normal curvature in the direction of **w** is called as radial curvature k_r.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

- **Definition I:** The suggestive contour generator is the set of points on the surface at which its radial curvature k_r is 0, and the directional derivative of k_r in the direction of $\mathbf{w}$ is positive.

$$k_r = 0, \ \mathit{where} \ D_{\mathbf{w}} k_r > 0.$$

- **Definition II:** The suggestive contour generator is the set of minima of $\mathbf{n} \cdot \mathbf{v}$ in the direction of $\mathbf{w}$.

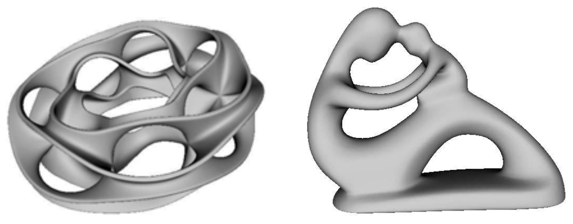
$$D_{\mathbf{w}}(\mathbf{n} \cdot \mathbf{v}) = 0, \ \mathit{where} \ D_{\mathbf{w}}(D_{\mathbf{w}}(\mathbf{n} \cdot \mathbf{v})) > 0.$$

- **Definition III:** The suggestive contour generator is the set of points on the contour generator of a nearby viewpoint(of radial distance less than 90 degrees) that are not in radial correspondence with points on the contour generators of any (radially) closer viewpoint.

Definitions of suggestive contours are illustrated in Fig. 3.8(b). The inflection point $\mathbf{p}$ occurs where the radial curvature k_r changes sign, and the radial curvature k_r is increasing towards the camera(Definition I). At the same time, $\mathbf{p}$ has the surface normal that is the local minimum of $\mathbf{n} \cdot \mathbf{v}$(Definition II). As the viewpoint moves to the left, eventually real contours begin to appear on the surface. They appear when the camera moves below the tangent plane, and for no closer viewpoints(Definition III).

Before suggestive contours, all other feature lines drawn from general 3D shapes have been feature lines fixed on the surface which lack the view-dependent nature of hand-drawn line drawings, thus can not make natural looking line drawing in conditions with changing view points. Suggestive contours are the pioneer of view-dependent feature lines. Similar with suggestive contours, PELs are both view and light dependent feature lines. Suggestive contours extend the actual contours of surfaces in concave regions and

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library


(a) (b)

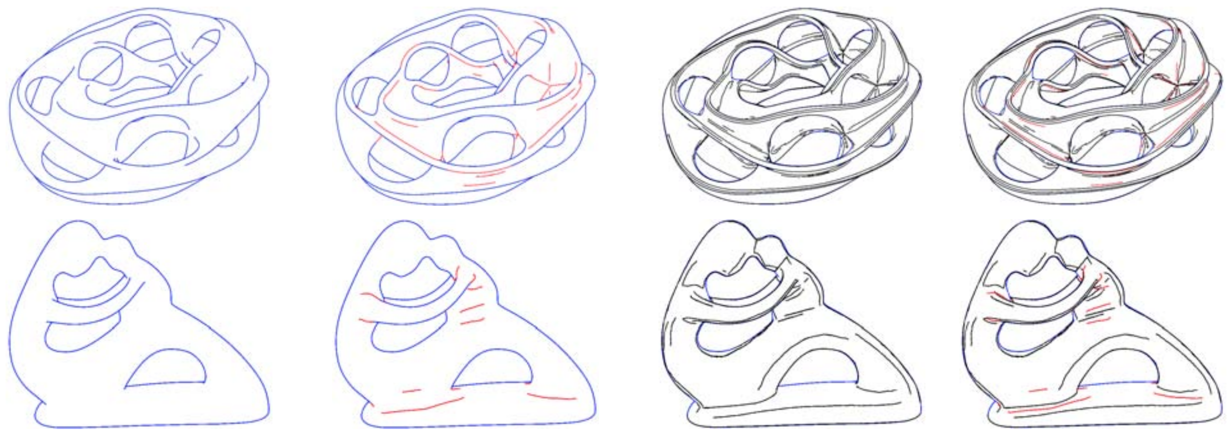

(c) Contours (d) Suggestive contours (e) PELs (f) Combination

Figure 3.10: Comparison of PELs with suggestive contours. Suggestive contours extend contours by adding interior information in **concave** regions, but can not convey salient features in **convex** regions. PELs are defined with the variation of the illumination, thus, can appear anywhere the illumination changes significantly. The combination of PELs and suggestive contours (f) shows that PELs locate in different positions with suggestive contours, including **convex** regions. Contours, suggestive contours and PELs are drawn in blue, red, and black, respectively.

convey the shape in a succinct and elegant way with added expressiveness. However, they can not illustrate salient features in convex regions. Examples are shown in Fig. 3.10 and Fig. 3.12. To compensate for this, DeCarlo *et al.* [16] suggest to set different value of k_r, i.e., given a user-specified value c, compute the points satisfying $k_r = c$. By carefully choosing a positive value c, we can get suggestive contours in certain convex regions, while losing useful suggestive contours in other regions including concave regions as a tradeoff. So for a real-world complicated shape with many convex and concave regions, it is almost impossible to find a globally consistent value c for all regions. Fig. 3.11 illustrates this phenomenon by showing the suggestive contours with three different value

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

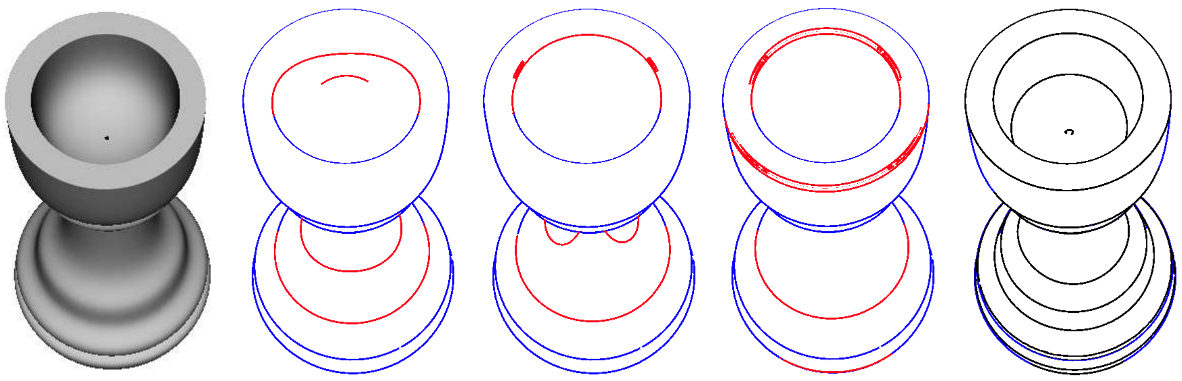

(a) Shading (b) $k_r = -1$ (c) $k_r = 0$ (d) $k_r = 1$ (e) PELs

Figure 3.11: Suggestive contours (c) are the set of points on the surface at which the radial curvature $k_r = 0$. The radial curvature of the convex region is always positive, i.e., $k_r > 0$. Thus, suggestive contours occur only on the concavities. By offsetting the radial curvature $k_r = c$, suggestive contours change accordingly (see (b) to (d)). With a positive offset $k_r = c > 0$, suggestive contours appear on *SOME* of the convex parts. However, it is impossible to choose an appropriate value c such that suggestive contours appear on *ALL* the convex and concave parts simultaneously. PELs (e) do not have such restriction and can appear anywhere where the illumination changes significantly. Contours, suggestive contours, and PELs are drawn in blue, red and black, respectively.

$c < 0$ (Fig. 3.11(b)), $c = 0$ (Fig. 3.11(c)), and $c > 0$ (Fig. 3.11(d)). In contrast, the proposed PEL method works well on both convex and concave regions (Fig. 3.11(e)). Simultaneously, suggestive contours may also extend contour lines onto smooth surfaces which cause visual distraction, as shown in Fig. 3.12.

3.3.3 PELs versus Ridges & Valleys.

Ridge & valley lines are curves on a surface along which the surface bends sharply which can be intuitively defined as loci of sharp variation points of the surface normal. Mathematically the sharp variation points of the surface normals are described via extrema of the surface principle curvatures along their corresponding lines of curvature. Consider an oriented smooth surface S and denote k_{max} and k_{min} the maximum and minimum principal curvatures with $k_{max} >= k_{min}$. Let t_{max} and t_{min} be the corresponding principal directions, e_{max} and e_{min} be the derivatives of the principal curvatures along their

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

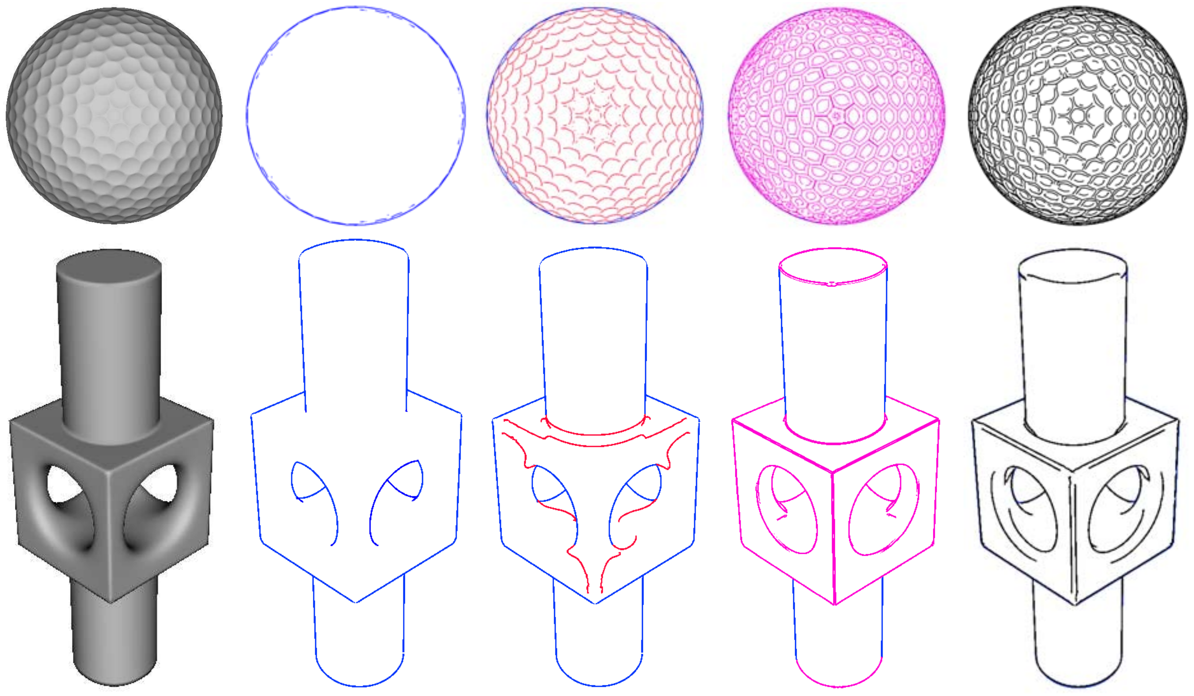

(a) Shaded model (b) Contours (c) Suggestive contours (d) Ridges/ valleys (e) PELs

Figure 3.12: Comparison of PELs with other feature lines. (b) shows that contours can not convey the structure and complexity of the interior. (c) shows that suggestive contours fail to illustrate salient features in convex regions, and suggestive contours extend contours onto smooth surfaces which causes visual distraction. (d) shows that ridge-valley lines with contours convey shapes nicely, but ridge-valley lines are view independent and can not capture features in smooth regions, such as the vertical contours of the cylindrical part of the CSG model. PELs alone in (e) work well in all these aspects. The contours, suggestive contours, ridge-valley lines, and PELs are drawn in blue, red, cyan, and black, respectively.

corresponding curvatures directions with:

$$e_{max} = \partial k_{max}/\partial t_{max}, \quad e_{min} = \partial k_{min}/\partial t_{min}$$

$\mathbf{e_{max}}$ and $\mathbf{e_{min}}$ are not defined at the umbilical points with $k_{max} = k_{min}$. Ridges and valleys consist of perceptually salient non-umbilical points which are extrema of the principal curvatures along their curvature directions. The ridges are defined with:

$$e_{max} = 0, \quad \partial e_{max}/\partial t_{max} < 0, \quad k_{max} > |k_{min}|$$

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

while the valleys are defined with:

$$e_{min} = 0, \quad \partial e_{min}/\partial t_{max} > 0, \quad k_{max} < |k_{min}|$$

The ridge and valley lines are dual w.r.t. the surface orientation: changing the orientation turns the ridge lines into valley lines and vice versa. An example of ridge points is illustrated in Fig. 3.8(c).

Ridge and valley lines are powerful shape descriptors. Ridge & valley lines convey shapes where surface bends sharply. However, visually salient features may also occur in smooth regions such as the vertical contours of the cylindrical part of the CSG model of Fig. 3.12 where ridge-valley lines fail. Comparing with ridge/valley lines, PEL alone can convey features in smooth regions. Another widely recognized limitation of ridge & valley lines is that they are view-independent curves. Thus, they are fixed on the object surface without being able to slide on it when the viewing conditions change, which does not make a natural looking line drawing in animation sequences. It is also demonstrated in Fig. 3.13 that the illustration of ridge & valley lines appears like marks on surface which is quite different from a natural line drawing.

3.3.4 Summary

In summary, contours give strongest cues with model-to-background distinction, but they are quite limited in showing the interior geometry; Suggestive contours extend contours in concave regions, but they can not illustrate salient features in convex regions; Ridges and valleys convey shapes where surface bends sharply, but they are view-independent feature curves fixed on surfaces. Fig. 3.12 gives a comparison of PELs with the other feature lines, which shows that PELs perform well in most of these aspects.

We also compare the feature lines with hand-drawn illustrations done by artists. Similar to the method of [13], artists are asked to draw in two steps: first to draw in a

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

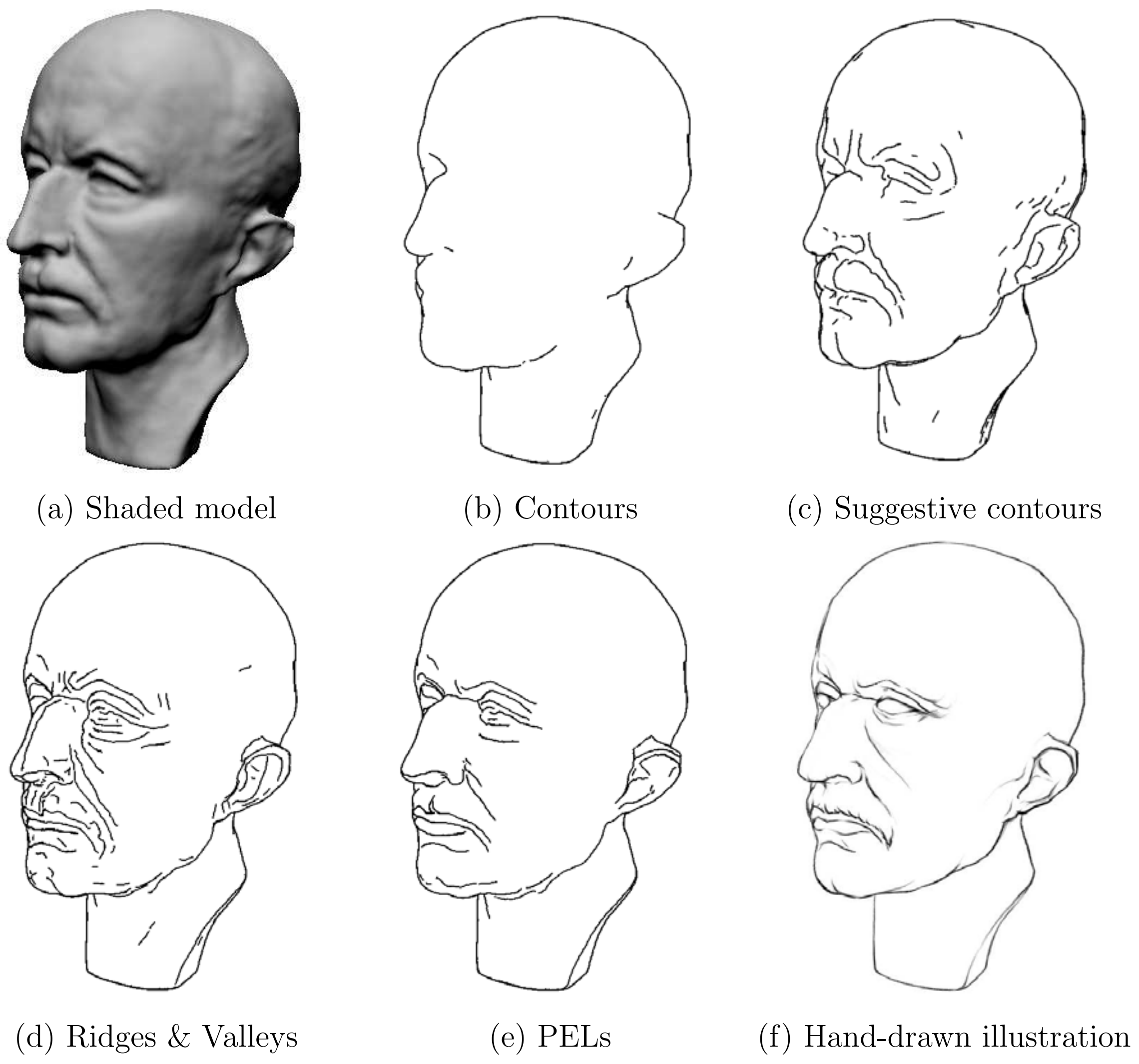


Figure 3.13: The definition of PEL shares similarity with the way of finding a line when artists draw. Thus, PELs illustration is more similar to the hand-drawn illustration than contours, suggestive contours and ridge-valley lines. More detailed quantitative evaluation is given in Section 3.6.1.

blank paper, then to register the drawing to the shaded image of the model. In Fig. 3.13, (f)(by courtesy of Mr. LuPeng) is drawn in this way according to the shaded image (a). Clearly, PEL mimics the hand-drawn illustration better than all the other feature lines. More detailed quantitative evaluation is given in Section 3.6.1. We survey and discuss with professional artists from a cartoon animation studio. Most of the artists agree that our approach on defining PEL based on the significant variation of illumination is quite

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

similar to the way of finding a line where an artist actually draws. This might be the most important reason why the PELs illustration is more similar to the hand-drawn illustration than the other feature lines in Fig. 3.13.

3.4 Comparison of PELs with Apparent Ridges

Carried out concurrently but published a little later than PEL, the apparent ridges [56] is defined as the loci of points that maximize a view-dependent curvature. Based on two important perceptual observations: first, human perception is sensitive to the variation of shading which is closely related to the surface normal variation; second, view-dependent lines better convey smooth surfaces, Judd et al. propose apparent ridges which achieve very promising line drawings. Apparent ridge lines are variants of geometric ridge lines that include a view-dependent projection. Mathematically, the maximum view-dependant curvature q_1 can be defined as:

$$q_1 = \max_{||P(\mathbf{r})||=1} ||S(\mathbf{r})||$$

where $\mathbf{r}$ is a unit vector in the tangent plane, S is the *Weingarten map* or the *shape operator* which is a linear map from the tangent plane to a tangent of the Gauss sphere parallel to the tangent plane, P is a parallel projection which maps points on the surface to points on the viewing screen. Thus, apparent ridges is a variation of traditional ridges which adds view dependency to the traditional definition of curvature by projection. In summary, the main properties of apparent ridges are:

- Similar with traditional ridge and valley lines, apparent ridge lines are defined with the third order derivatives of surfaces.
- Apparent ridge lines are view-dependent feature lines which are determined by local geometric property(curvature) and viewing position.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

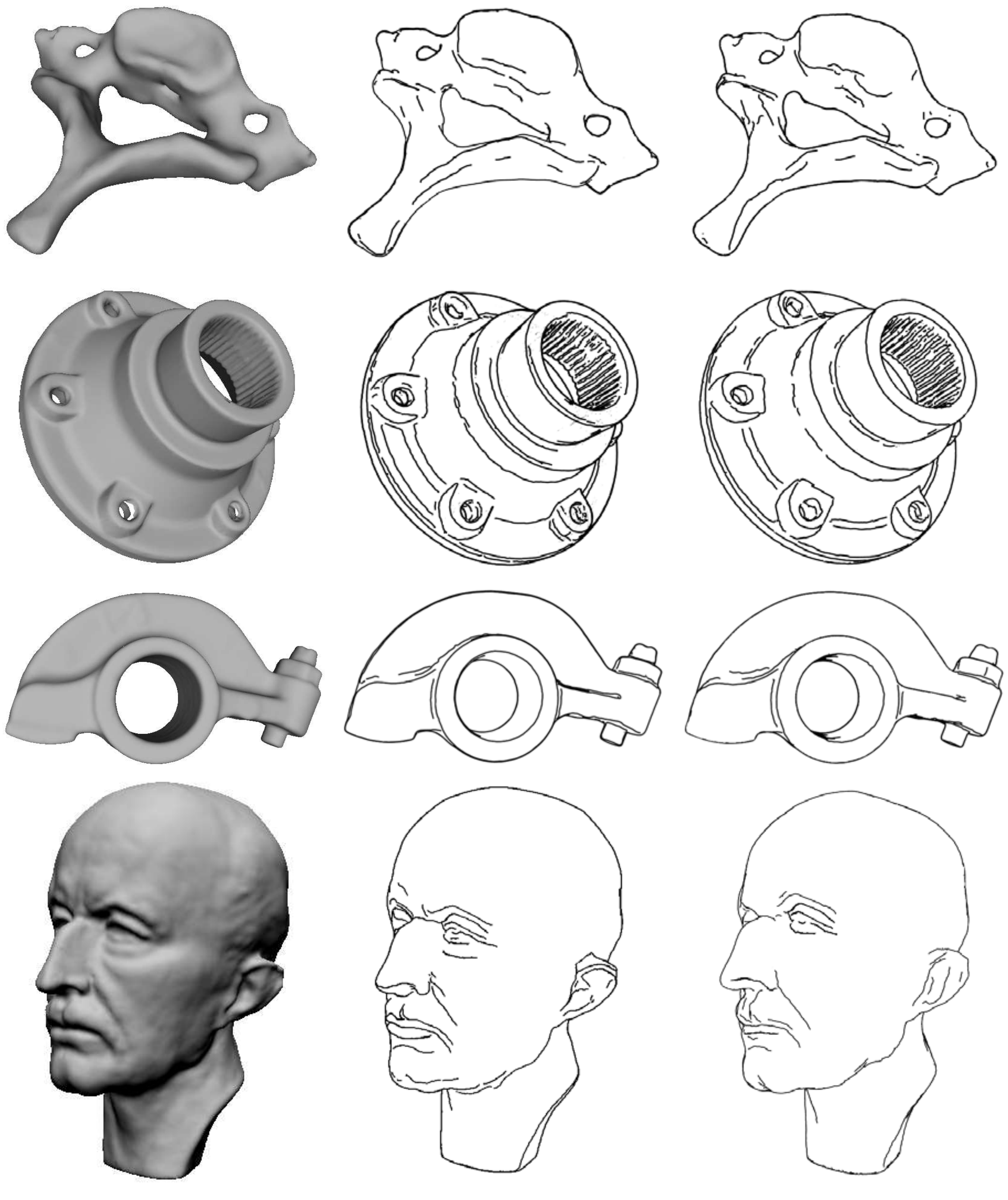


Figure 3.14: Comparison of PELs with apparent ridges. Both identifying the variation of shading on surfaces, the line drawings of PELs and apparent ridges are quite similar in many cases.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

- Apparent ridge lines are lighting independent feature lines which are not affected by lighting and reflectance conditions on surfaces.

PELs and apparent ridges share great similarity in definition. Both PELs and apparent ridges are motivated by the observation that human perception is sensitive to the variation of shading on surfaces. PELs are defined as the loci of points of illumination variation, while Judd et al. consider normal variation as the main cause of illumination variation and define apparent ridges with normal variation(curvature). Theoretically, similar results should be achieved when PELs are computed with headlight only, which is exactly the main light of our proposed light model in Section 3.2.2. Comparison of line drawings with PELs and apparent ridges are shown in Figure 3.14, in which the Max Planck model used in Figure 3.13 and some models recommended in [13] are used for fair comparison. Though the line drawings of PELs and apparent ridges are similar in many cases, they show differences in details. Apparent ridges are strictly defined with the underlying geometry. Thus, lines of apparent ridges follow the structure of the underlying geometry better than PELs, as can be seen in Figure 3.14. However, normal variation is not the only factor of illumination variation on surfaces. By defining feature lines with illumination variation directly, PELs are more flexible in achieving user controllable line drawings. Given a 3D model, the normal variation of the 3D model is fixed, therefore, the apparent ridge lines are also fixed. In case these lines are not completely satisfied(which always happens), PELs provide users more freedom in modifying the line drawing by changing lighting conditions, such as adding auxiliary lights. In our work, we have also developed very convenient tools which allow users to modify lighting on surfaces with simple click and select.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

3.5 PELs for Volume Illustration

Volumetric data are widely used in scientific and medical applications. Volume illustration can be viewed as non-photorealistic rendering applied to volume visualization. The current approaches can be classified into two categories: the first approach enhances the standard volume rendering pipeline with NPR techniques (e.g., [116][96]), while the second approach applies illustrative drawing styles to visualize volumetric data sets directly (e.g., [72][8][84][61][83][103][6]). Visualizing volumetric datasets with feature lines including ridges, valleys, silhouettes, and suggestive contours falls into the second category. Svakhine and Ebert develop an interactive volume illustration system (IVIS) that provides a number of volume illustration enhancements [116]. Kindlmann *et al.* demonstrate that curvature-based transfer functions enhance the expressive and informative power of direct volume rendering [61]. Nagy and Klein render high-quality silhouettes from volumes using GPU [83]. Schein and Elber develop an adaptive algorithm to extract and visualize silhouette from volumetric data by modeling the data with B-spline functions [103]. Burns *et al.* present a volumetric drawing system that directly extracts linear features, such as contours and suggestive contours, using a temporally coherent seek-and-traverse framework [8]. Bruckner and Gröller [6] propose VolumeShop, a dynamic volume illustration system which integrates many non-photorealistic rendering models to interactively alter and explore volume data.

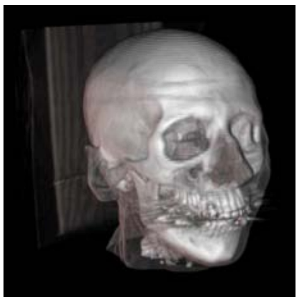
We apply PELs for volume illustration using isosurfaces. An isosurface is a three-dimensional analog of an isocontour. It is a surface that represents points of a constant value (e.g. pressure, temperature, velocity, density) within a volume of space; in other words, it is a level set of a continuous function whose domain is 3D-space. In medical imaging, isosurfaces may be used to represent regions of a particular density in a three-dimensional CT scan, allowing the visualization of internal organs, bones, or other

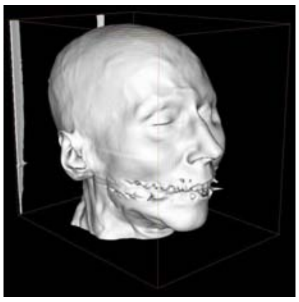
ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

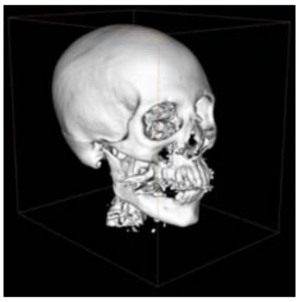
structures. Isosurfacing is one of the most popular and effective methods for visually representing volumetric scalar fields. The visualization of volume datasets with isosurfaces consists of two phases, the isosurface extraction phase and the rendering phase. For volume illustration with PELs, the user interested isosurfaces are extracted in a preprocess phase, and then PELs are applied on the isosurfaces.

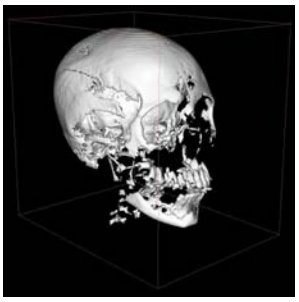
Marching Cubes(MC) [71] is the most popular algorithm for isosurface extraction due to its powerful combination of simplicity, efficiency and robustness of implementation. Numerous methods have been proposed in recent years to improve MC in either the quality of the extracted isosurfaces or the speed. Improved methods for higher quality geometric approximation of isosurfaces are usually categorized into three classes. One general class of methods start with a coarse mesh of the isosurface, and then refine the mesh with edge swapping, vertex repositioning etc [121, 133]. Another class of methods start from one or more seed points and grow triangles in the form of an expanding, or advancing, front [104]. A third class of methods generate an unorganized set of surface samples and create a Delaunay tessellation of the samples [1, 79]. With the rapid improvement of GPU in recent years, real-time isosurface extraction with MC has been implemented in GPU. Based on the geometry shader introduced in the Shader Model 4 [70], Uralsky [120] present how marching tetrahedra can be implemented using GPU, with an implementation given in the NVidia OpenGL SDK-10. Dyken et al. [23] present an implementation of MC on both Shader Model 3 and Shader Model 4 with an extended HistoPyramid structure. Though MC can be implemented in real-time, unfortunately, the output of current techniques is a set of triangles without connectivity information, which limits their application in real-time PELs extraction. In PELs extraction, the computation of third order derivatives on surfaces requires at least 2-ring neighbors of vertices and triangles. It is still very slow to construct a data structure, such as a half edge data structure, to represent the connectivity information of vertices and triangles.

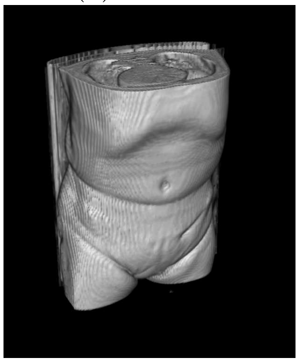
ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

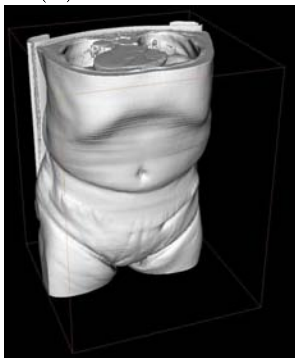

(a) Volume

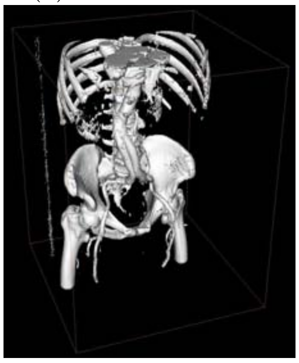

(b) Isosurface_60

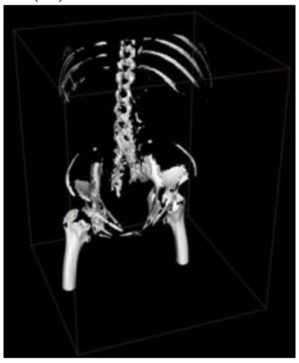

(c) Isosurface_110

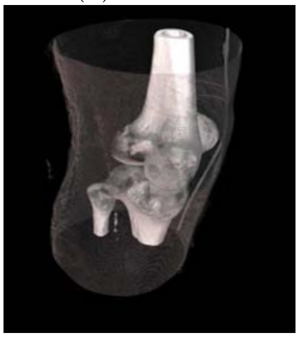

(d) Isosurface_160

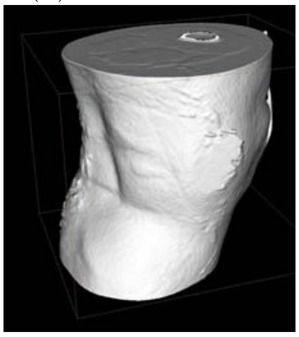

(a) Volume

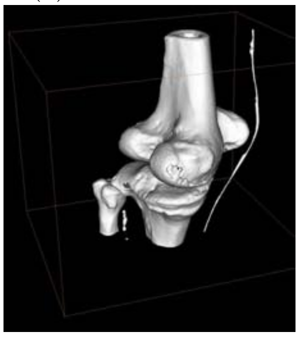

(b) Isosurface_100

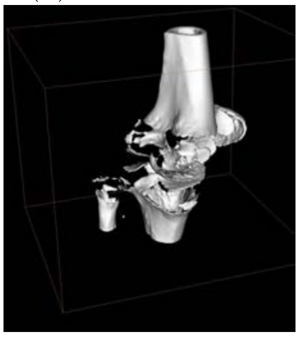

(c) Isosurface_140

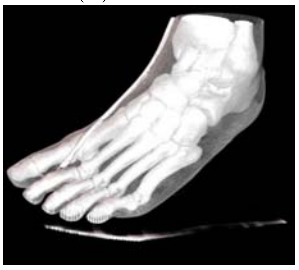

(d) Isosurface_180

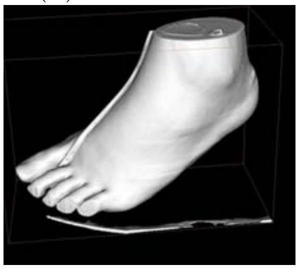

(a) Volume

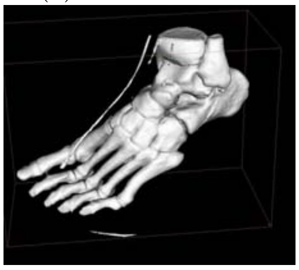

(b) Isosurface_53


(c) Isosurface_71


(d) Isosurface_90


(a) Volume


(b) Isosurface_68


(c) Isosurface_105

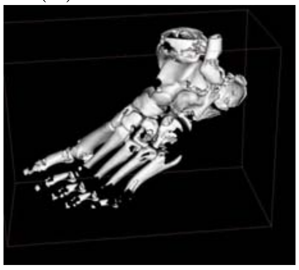

(d) Isosurface_145

Figure 3.15: Volume datasets and isosufaces of different isovalues.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

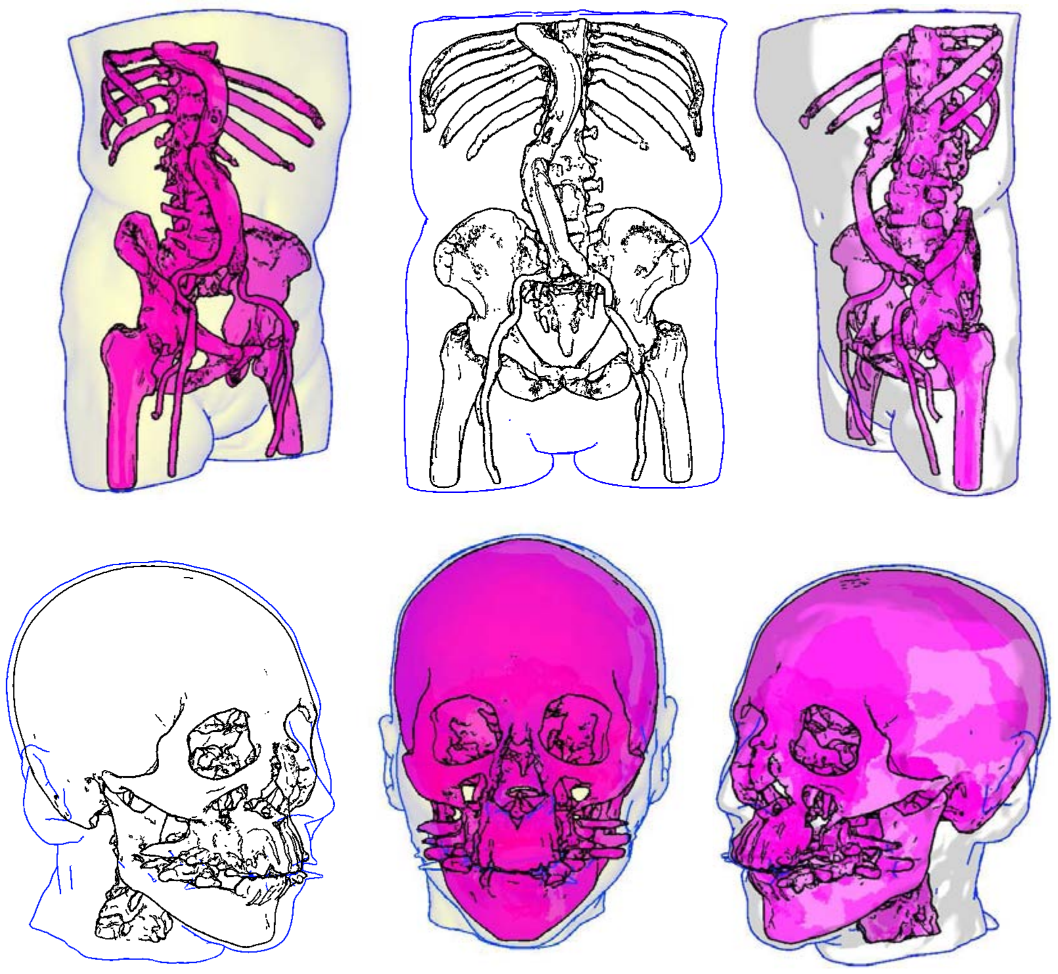
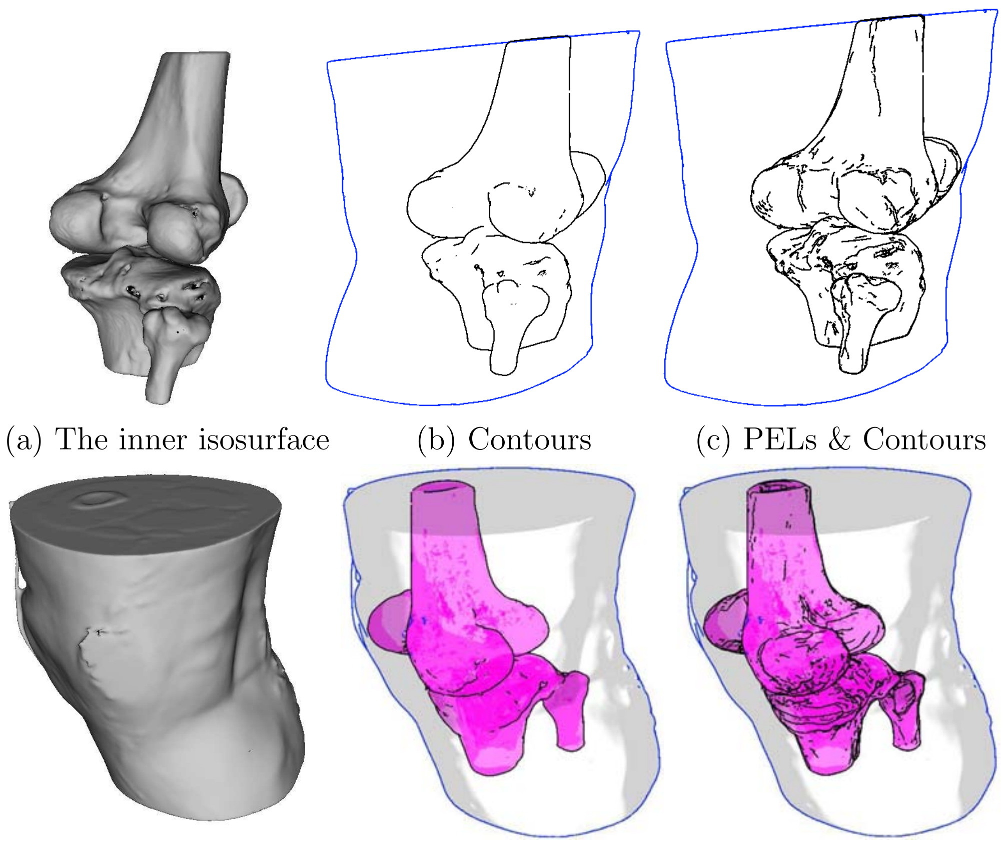


Figure 3.16: PEL for volumetric datasets. Two isosurfaces are extracted from the volumetric datasets. To emphasize the inner structure, the outer isosurface is illustrated with contours only (colored in blue).

Thus, we extract the isosurfaces in a preprocess phase in current stage. In most of our cases, the quality of the extracted isosurfaces of the standard marching cubes method [71] is quite satisfied, and we adopt this method with a half edge data structure for our PELs extraction. Fig. 3.15 shows the volume datasets and isosurfaces of different isovalues for PELs illustrations of this chapter.

We demonstrate PELs for volume illustration with examples. The user first extracts several isosurfaces which are of specific interest. Then our system draws PELs and/or contours on each surface. Usually, users are not equally interested in the different ex-

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library



(a) The inner isosurface (b) Contours (c) PELs & Contours

(d) The outer isosurface (e) Contours (f) PELs & Contours

Figure 3.17: PEL together with contours illustrates the volumetric data much better than using contours alone.

tracted isosurfaces. Therefore, we draw user focus isosurfaces with both contours and PELs for more details while other isosurfaces with contours only. The user can also use various NPR shadings (e.g., toon shading and Gooch shading) to highlight the important details. In Fig. 3.16, two isosurfaces(isosurface_100 and isosurface_140 for the upper row, isosurface_60 and isosurface_110 for the lower row, all these isosurfaces are shown in Fig. 3.15) are rendered. We illustrate the outer isosurface using contours and the inner isosurface using both contours and PELs. Fig. 3.17 demonstrates that one can achieve better illustration by combining PELs and contours than using contours alone.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

3.6 Conclusion

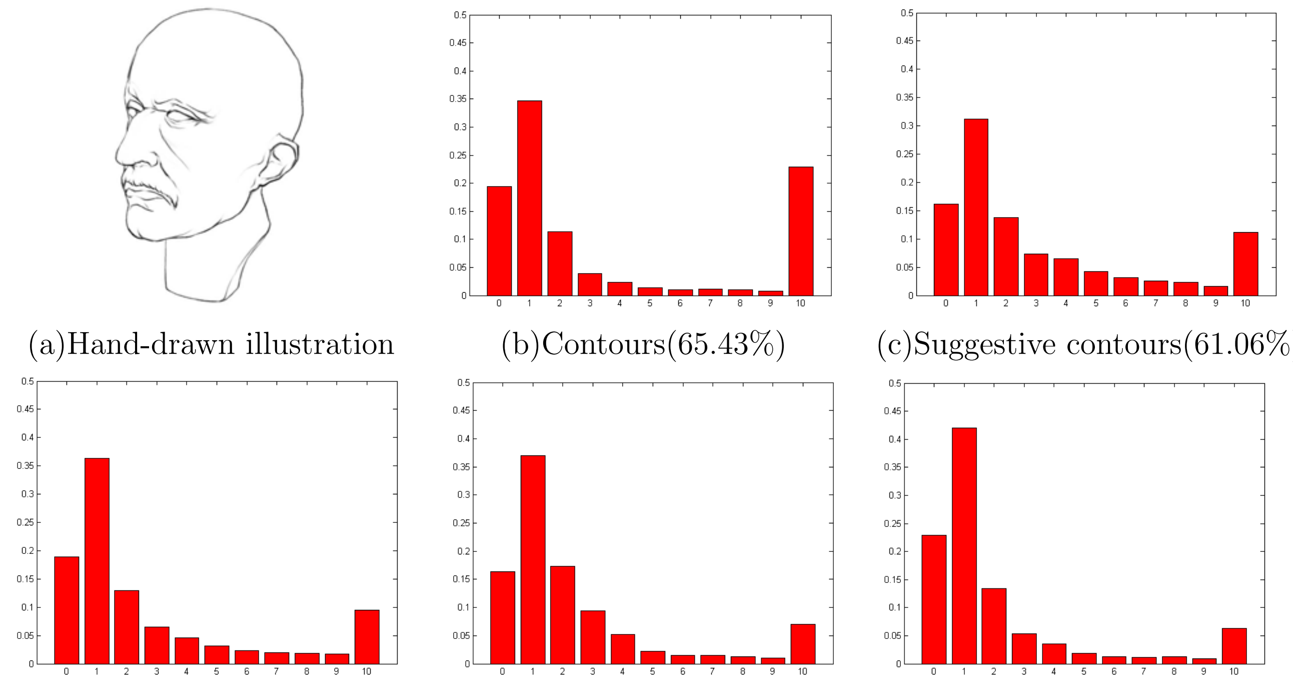
In this chapter, we have presented Photic Extremum Line(PEL), a new type of feature lines for line drawings and scientific illustration of 3D surfaces and volumes. In contrast to the existing feature lines which are solely defined by geometric properties and/or viewing positions, PEL characterizes the significant variations of illumination on 3D surfaces directly. PEL offers the user many degrees of freedom, such as light positions, the number of lights, and various light models, to achieve user controllable illustrations. We have also made comprehensive comparisons between PEL and the existing approaches. Through a wide array of experiments, we have shown that PEL enables a flexible and effective tool to illustrate 3D surfaces and volumes and reveal most important insights towards better line drawings and visualization.

3.6.1 Evaluation

The perceptual consistency of a line drawing is highly subjective. In general, objective evaluation studies are difficult to design for NPR imagery, where there is no obvious ground truth. Relatively few papers have been devoted specifically to evaluations of NPR methods. Inspired by the methods of experimental psychology in studying human visual system, recently, Cole et al. [13] present a formal study on where artists draw lines to convey specific 3D shapes. Based on study methods in this work, we give a quantitative evaluation of the perceptual consistency of PEL in comparison with some other important feature lines in this section.

Over hundreds of years, hand-drawn scientific illustrations have achieved a high level of sophistication. Taking inspiration from a long tradition of artistic and illustrative depiction, NPR often tries to imitate long established illustration techniques. Considering hand-drawn illustrations as the representation of the human perception of 3D shapes, we evaluate the perceptual consistency of a computer generated line drawing with its

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library



(a)Hand-drawn illustration (b)Contours(65.43%) (c)Suggestive contours(61.06%)

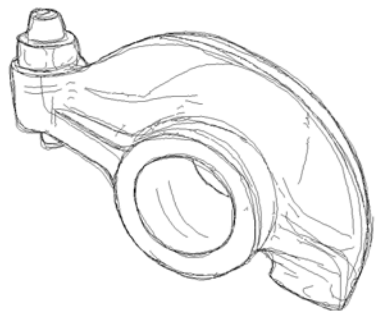
(d)Ridges&Valleys(68.27%) (e)Apparent ridges(70.70%) (f)PELs(78.39%)

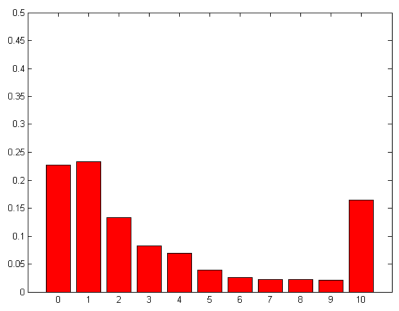
Figure 3.18: Evaluation of similarity between the hand-drawn illustration and computer generated line drawings for the Max Planck model. Histograms of pairwise closest distances between line pixels and the percentages of pixels with distances less than 3 pixels are shown. The PELs and the hand-drawn illustration have 78.39% line pixels overlapping within 3 pixels. Thus, the PELs illustration is more quantitatively similar to the hand-drawn illustration.

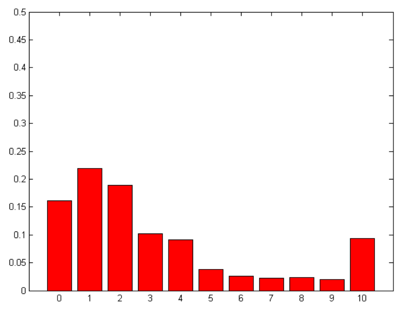
similarity to the hand-drawn illustrations. Following the method of [13], comparisons between drawings are based on overlaps of pixels. This is an approximation of the similarity evaluation that is both simple and robust.

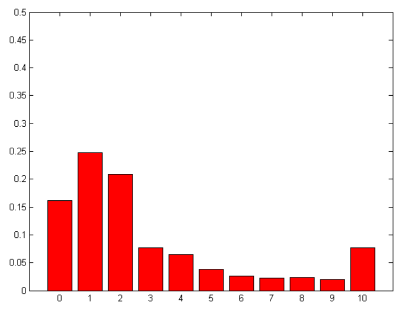
In our experiments, artists are asked to draw the hand-drawn illustrations in two steps: first to draw in a blank paper, then to register the drawing to the shaded image of the 3D model. They are also trained to create line drawings using our software by adjusting thresholds for line strength and applying auxiliary lights for PELs. Thus, the computer generated line drawings with different feature lines for comparison are also created by artists themselves. An evaluation of similarity for the Max Planck model is shown in Figure 3.18 which computes the histogram of pairwise closest distance of line

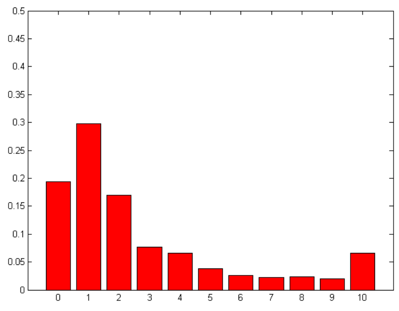
ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library


(a)Hand-drawn illustration


(b)Contours(59.31%)


(c)Suggestive contours(56.97%)


(d)Ridges&Valleys(61.64%)


(e)Apparent ridges(66.17%)

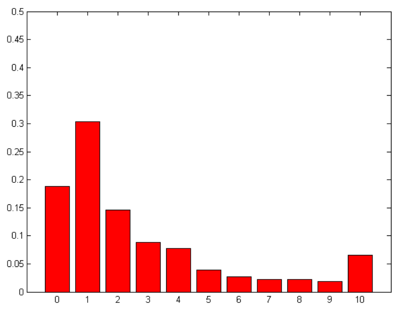

(f)PELs(63.84%)

Figure 3.19: Evaluation of similarity between the hand-drawn illustration and computer generated line drawings with data and the rockarm model provided by Cole et al. [13]. Histograms of pairwise closest distances between line pixels and the percentages of pixels with distances less than 3 pixels are given for comparison, showing that illustrations of apparent ridges and PELs are more similar to the hand-drawn illustration.

pixels for both computer generated line drawings and the hand-drawn illustration. As can be seen in Figure 3.18, all the feature lines have high percentage of overlapping with the hand-drawn illustration within 3 pixels. Of all these feature lines, illustrations of PELs show the best similarity to the hand-drawn illustration. In this example, contours have 65.43% line pixels with distance less than 3 pixels while more than 20% line pixels with distance more than 10 pixels. This also explains clearly that artists always draw contours of objects, and contours alone do not capture interior lines of 3D shapes.

We also evaluate perceptual consistency of line drawings using data set provided in the work of Cole et al. [13]. An evaluation of similarity for the rockarm model is shown in Figure 3.19, in which the hand-drawn illustration of (a) is created with six superim-

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

posed drawings by different artists. Line drawings of other feature lines for comparison are generated with the provided camera and resolution settings. As demonstrated in Figure 3.19, results which are consistent to those of Figure 3.18 are achieved, of which illustrations of apparent ridges and PELs are more similar to the hand-drawn illustration.

3.6.2 Limitation and Discussion

In most of our test cases, PEL performs well in conveying shapes even without auxiliary lights, as demonstrated in our examples. However, results of PELs are not satisfied in some cases, as shown in Figure 3.20. Generally, PEL does not work so well with surfaces with very few or even no features, especially synthetic surfaces. Figure 3.20 (b) illustrates PELs on a torus. The torus can be nicely illustrated by contours. Without the aid of auxiliary lights, PEL gives a closed, ellipse-like curve (colored in blue in Figure 3.20(b)) which locates very close to the silhouettes. Clearly, PEL is not helpful to convey the shape in this case. Besides simple synthetic surfaces, PELs also fail to give satisfied results for some surfaces clustered with high frequency components, such as the small valleys and bumps in the model of Figure 3.20 (c). As can be seen in Figure 3.20 (d), PELs illustrate these small valleys and bumps with double lines, which introduce lines clustering in very small regions. Actually, artists will also use a single line to represent a bump/valley in their drawing process in similar cases. To compensate for this, we provide tools for users to adjust lighting in these areas to eliminate some lines by adding auxiliary lights. However, the selection of interested regions to apply optimized lighting in these small areas is also very time consuming. One of our future works is to refine PEL to overcome these limitations.

It is well known that the key and open problem of line drawing is where to put the lines. Unfortunately, this is a somewhat subjective problem. PELs emphasize this problem by providing user great flexibility in controlling lines by manipulating lights.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

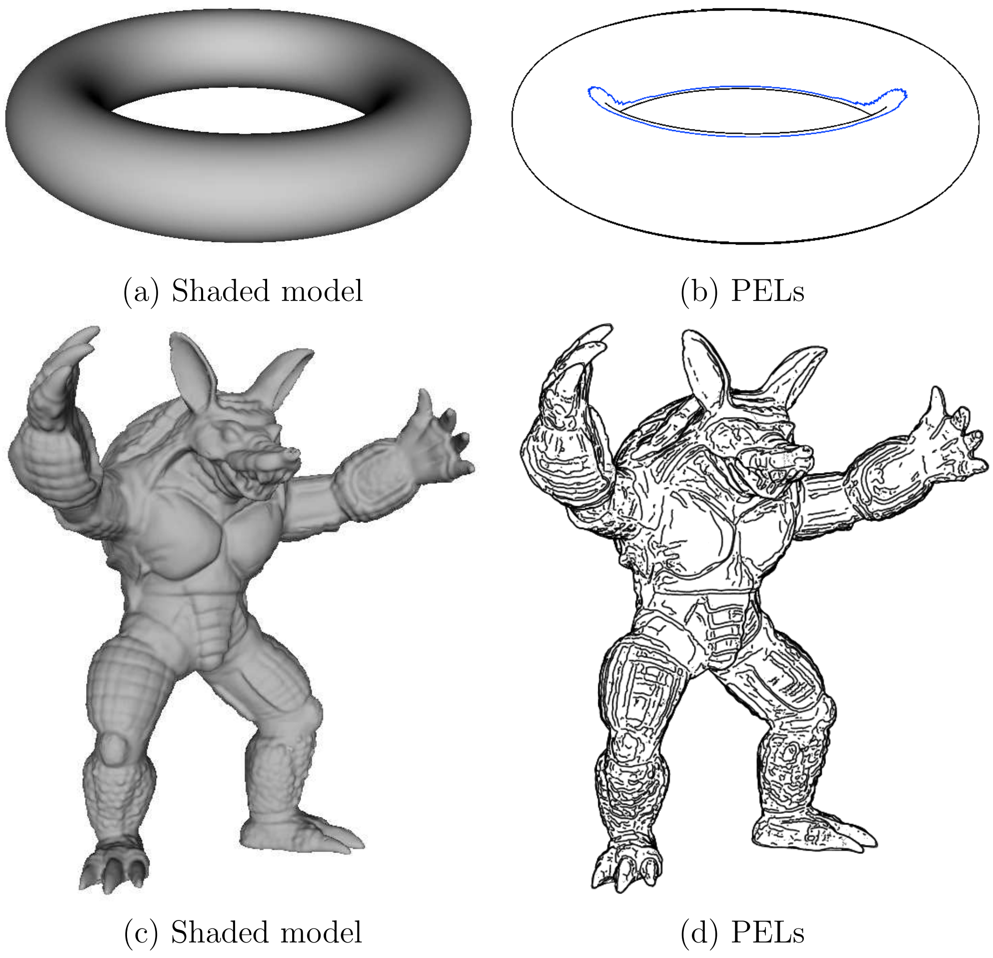


Figure 3.20: Limitations of PELs in conveying shapes.

To minimize the effort of user interaction, we have developed methods to compute the optimal setting of the auxiliary lights automatically. An ideal solution for this open problem is to incorporate high level rules of human perception and advanced techniques of artistic drawing to define feature lines. PELs are defined based on this consideration. In the future, we will further improve our PELs in this direction.

In our current system for volume illustration, PELs are computed on the extracted iso-surfaces of the underlying volume taking advantages of the sparseness of the user interested isosurfaces, which will lead to more comprehensible volume visualization. As mentioned in the previous section, though the iso-surfaces can be extracted in real-time,

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

the construction and representation of the connectivity information of the iso-surface is slow. This makes it a difficult task to achieve real-time interaction. Recently, Burns *et al.* [8] proposed a method to reduce the dimensionality by directly extracting feature lines that lie on iso-surfaces within a volume. One of our ongoing works is to compute PEL from volumetric datasets directly without the need of iso-surface extraction and meshing of iso-surface. We will also work on realtime extraction of PELs for both surfaces and volumes.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

Chapter 4

Algorithms of PELs Extraction

In the previous chapter, we demonstrate the effectiveness of shapes illustration with PELs. PELs can be implemented in different ways. The key of the PELs implementation is the computation of high order derivatives on 3D surfaces. In this chapter, we propose two novel and efficient algorithms for PELs extraction, with which PELs can be conveniently incorporated in various rendering software and systems. The first one is a fast object space PELs extraction algorithm which computes high order derivatives efficiently using methods of discrete differential geometry. It generates analytic PELs that are especially suitable for further process, such as applying artistic styles onto the PELs. The second one is a quite different PELs extraction algorithm in geometry image space. By representing a 3D surface with a regular 2D geometry image, PELs for 3D surfaces are extracted with 2D image processing techniques efficiently.

4.1 Introduction

Feature lines extraction algorithms for 3D surfaces can be divided into three categories based on where the algorithm detects and draws the feature lines: object space, image space, and hybrid algorithms, as those silhouette detection algorithms reviewed in [52] for polygonal models. In this aspect, PELs extraction is similar with silhouette detection,

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

but includes more complicated high order derivatives computation on 3D surfaces. Object space methods detect and depict geometrical features in 3D space, and get feature lines in precise analytic description. However, detecting feature lines in 3D space always requires computation of geometrical quantities on curved surfaces, thus achieving interactive rendering can be challenging. On the contrast, image space methods are normally faster due to the regular structure of the 2D image. Additionally, a wide variety of well developed image processing tools are available when working in image space. This is also a big advantage of image space methods. However, image space algorithms can achieve only pixel or sub-pixel accuracy. They can not preserve fine details of a 3D shape which are smaller than one pixel in size. Furthermore, the projection from a 3D shape to a 2D image during rendering will also cause the lost of some 3D information. Some advanced image space algorithms compensate this in some degree by using geometry buffers such as z-buffer and normal buffer etc. Slightly different from image space algorithms, hybrid algorithms modify 3D surfaces in object space and then find feature edges using geometry buffers.

Though object space algorithms are somewhat slower, their precise analytic description of feature lines is convenient for further processing, such as representing feature lines with strokes and applying styles onto strokes. Thus, feature lines extraction in object space is widely used, especially in game and movie industry. For applications in these areas, triangle mesh is one of the most general ways of object representation. We develop our object space PELs extraction algorithm on triangle meshes. Incorporating current advanced methods in discrete differential geometry for high order derivatives computation, our algorithm achieves interactive rendering speed with meshes in moderate size, as demonstrated in Table 4.1.

To compute the normally time-consuming high order derivatives of 3D surfaces with a regular structure, we present a novel algorithm to extract PELs in geometry-image space.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

Geometry images [36] parameterize and sample arbitrary surfaces onto completely regular structures, simple 2D arrays of quantized points with full geometry information, which is different from geometry buffers of existing image space algorithms that have only partial and limited 3D information. With geometry images, efficient PELs extraction can be easily implemented with existing 2D image processing techniques. Yuan et.al. [139] demonstrate the effectiveness of silhouettes extraction in geometry-image space. With its own technical challenges, PELs extraction in geometry image space shows the feasible potential of combining both the advantages of image space algorithms and object space algorithms. Furthermore, taking advantages of the 2D regularity of geometry images, PELs extraction can be further sped up leveraging modern programable graphics hardware.

4.2 PELs Extraction on Meshes

This section presents the algorithm of extracting PELs from 3D models represented with polygonal meshes. As mentioned above, triangle meshes are widely used in computer graphics, thus we develop our PELs extraction algorithm on surfaces approximated by triangle meshes. PELs are defined with high order derivative of illumination on 3D surfaces. The computation of high order derivatives on triangle meshes are not straight forward. Current methods fall into two categories: patch fitting and discrete differential geometry. Patch fitting methods fit an analytic surface to points either locally [114] [10] or globally [60] [87], then compute the derivatives of the fitted surface analytically. Discrete differential geometry [97] [78] [19] computes the differential properties on polygonal meshes directly, thus more efficient than patch fitting methods. To achieve interactive user controllability of PELs, methods based on discrete differential geometry are adopted in our algorithm.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

Given a triangle mesh S and the illumination f on each vertex, the extraction of PELs on S is as follows:

(i) (Optional preprocessing) Smooth normal $\mathbf{n}$ of surface S.

(ii) Compute the gradient of illumination ∇f for each vertex.

(iii) Compute the directional derivative of $D_{\mathbf{w}}||\nabla f||$ along the gradient direction $\mathbf{w}$ for each vertex.

(iv) Detect the zero-crossing and filter out the ones which are the local minima, i.e., $D_{\mathbf{w}}D_{\mathbf{w}}||\nabla f|| > 0$.

(v) Trace the zero-crossings to get the PEL.

Details of each step are introduced in the following sections.

4.2.1 Preprocessing

The purpose of preprocessing is to reduce the noise of illumination. In our proposed light model for PELs, both the main diffuse light and the auxiliary spot lights are functions of $\mathbf{n} \cdot \mathbf{l}$. Thus, the noise of illumination is mainly caused by the noise of geometry, i.e., the surface normal. We apply the bilateral filter [28] to process the vector field $\mathbf{n}$ on S. For each vertex $\mathbf{v}$, denote its neighborhood by $N(\mathbf{v})$. The bilateral filter is defined as:

$$\hat{\mathbf{n}}(\mathbf{v}) = \frac{\sum_{\mathbf{p}\in N(\mathbf{v})} W_c(\|\mathbf{p}-\mathbf{v}\|)W_s(\rho(\mathbf{v},\mathbf{p}))\mathbf{n}(\mathbf{p})}{\sum_{\mathbf{p}\in N(\mathbf{v})} W_c(\|\mathbf{p}-\mathbf{v}\|)W_s(\rho(\mathbf{v},\mathbf{p}))}$$

with

$$\rho(\mathbf{v},\mathbf{p}) = \frac{\arccos(\|\mathbf{n}(\mathbf{v})\cdot\mathbf{n}(\mathbf{p})\|)}{\|\mathbf{p}-\mathbf{v}\|},$$

where W_c is the closeness smoothing filter with parameter σ_c:

$$W_c(x) = e^{-\frac{-x^2}{2\sigma_c^2}},$$

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

$\rho(\mathbf{v}, \mathbf{p})$ mimics the relative curvature of $\mathbf{v}$ and $\mathbf{p}$, and W_s is the feature preserving filter with parameter σ_s:

$$W_s(x) = e^{-\frac{-x^2}{2\sigma_s^2}},$$

which penalizes large variation of the feature field. In practice, we find this step is usually helpful to improve the robustness of our algorithm for the scanned models. We follow Fleishman *et al.*'s method [28] to set the parameters σ_c and σ_s: the user selects a point on the mesh where the surface is expected to be smooth, and then a radius r of the neighborhood of the point is defined. Set $\sigma_c = r/2$ and σ_s to be the standard deviation of $\rho(\mathbf{v}, \mathbf{p})$ in the selected neighborhood.

The advantage of smoothing surface normal is that it needs to be computed only once, which avoids computational overheads and guarantees achieving PELs illustration in real time for meshes of moderate size. Another option is to smooth the illumination f on the surface, which has to perform the extremely expensive smoothing operation in each frame once the lighting or viewing condition is changed.

4.2.2 Computing the Derivatives

The definition of PEL involves the third order derivatives of the surface illumination, thus the key is to compute the derivatives efficiently and robustly. Similar to Rusinkiewicz's method to estimate curvatures and their derivatives [97], we compute the per-vertex derivatives by averaging adjacent per-face derivatives.

Given an arbitrary scalar function g defined on the mesh M, i.e., $g : M \to \mathbb{R}$, we consider a triangle $T \in M$ with vertices $\mathbf{v}_i \in \mathbb{R}^3$ and the associated scalar value $g_i = g(\mathbf{v}_i) \in \mathbb{R}$, $i = 1, 2, 3$. The per-face coordinate system is defined in the plane perpendicular to the face normal. Three vertices of T in the per-face coordinate system are denoted by $(x_i, y_i) \in \mathbb{R}^2$, $i = 1, 2, 3$. The per-face gradient ∇g is computed as [78]:

$$\nabla g = \begin{pmatrix} \frac{\partial g}{\partial x} \\ \frac{\partial g}{\partial y} \end{pmatrix} = \frac{1}{d_T} \begin{pmatrix} y_2 - y_3 & y_3 - y_1 & y_1 - y_2 \\ x_3 - x_2 & x_1 - x_3 & x_2 - x_1 \end{pmatrix} \begin{pmatrix} g_1 \\ g_2 \\ g_3 \end{pmatrix}$$

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

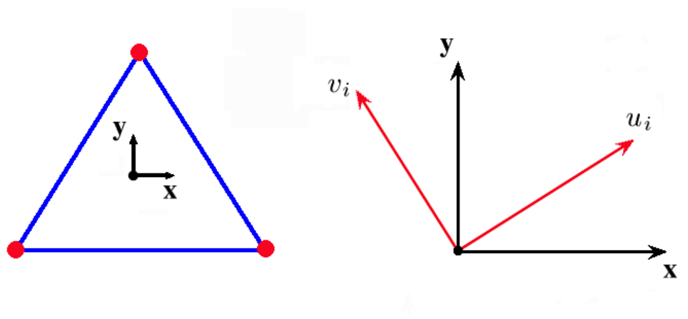


Figure 4.1: The vertex and face coordinate system.

where

$$d_T = (x_1y_2 - y_1x_2) + (x_2y_3 - y_2x_3) + (x_3y_1 - y_3x_1)$$

which is twice the signed area of T.

The gradient of each vertex is considered the weighted average of the gradient of its adjacent faces. The vertex coordinate system (u, v) is defined in the plane perpendicular to the vertex normal. Note that the vertex and face coordinate systems are usually different. Thus coordinate system transformation must be applied before computing the contribution of the per-face derivatives to the per-vertex derivatives [97]. We first rotate the local vertex coordinate system to be coplanar with the local face coordinate system (see Fig. 4.1). Then, given a point $(u_i, v_i) = u_i\mathbf{u}_i + v_i\mathbf{v}_i$ in the vertex coordinate system, we compute the per-vertex gradient in the local face coordinate system with

$$\nabla g(u_i, v_i) = \begin{pmatrix} \mathbf{u_i} \cdot \nabla g(x, y) \\ \mathbf{v_i} \cdot \nabla g(x, y) \end{pmatrix}$$

After the per-vertex gradients for all the adjacent faces of the vertex are computed, they will be accumulated with weighted averaging [97]. Following the method in [78], we use the Voronoi area weight w_i, which is set to be the portion of the area which lies closest to the vertex. Finally, the per-vertex derivative is computed as:

$$\nabla g(u, v) = \sum_i w_i \nabla g(u_i, v_i)$$

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

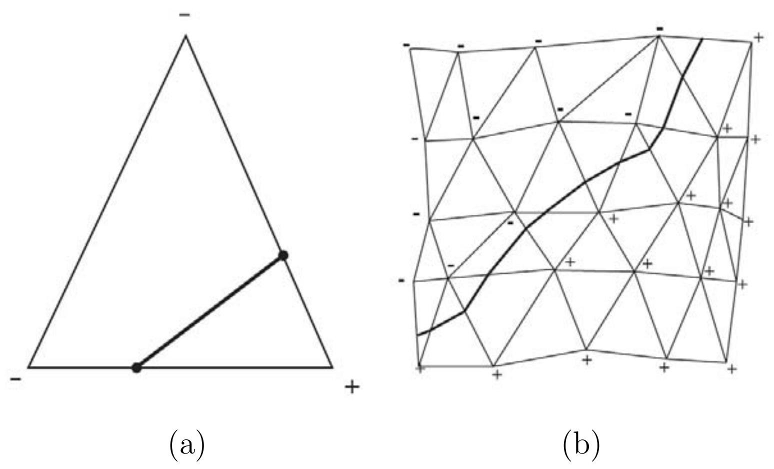


Figure 4.2: Tracing PELs on mesh. (a) PEL points are computed on each edge by linear interpolation. A line segment is created by connecting the points. (b) Line segments are connected to form a piecewise-linear curve.(by courtesy of Hertzmann [43])

4.2.3 Tracing PELs

Once we compute the directional derivatives of variation of illumination at each mesh vertex $\mathbf{v}$, we are ready to check whether the mesh edge $[\mathbf{v}_1, \mathbf{v}_2]$ contains the zero-crossing. Let $h(\mathbf{v}) = D_{\mathbf{w}}\|\nabla f(\mathbf{v})\|$. If $h(\mathbf{v}_1)h(\mathbf{v}_2) < 0$, we use linear interpolation to approximate a zero-crossing on edge $[\mathbf{v}_1, \mathbf{v}_2]$

$$\mathbf{p} = \frac{|h(\mathbf{v}_2)|\mathbf{v}_1 + |h(\mathbf{v}_1)|\mathbf{v}_2}{|h(\mathbf{v}_2)| + |h(\mathbf{v}_1)|}$$

and consider $\mathbf{p}$ a PEL vertex. Following the method of silhouette tracing in the course note of [43], if two PEL vertices are detected on edges of a triangle, we connect them by a straight line segment. Line segments are connected to form a piecewise-linear PEL curve, as shown in Fig. 4.2.

Similar to [87], we also measure the strength of a PEL by the integral of $\|\nabla f\|$ along the line

$$\int \|\nabla f\| ds \approx \sum_i \frac{\|\nabla f(\mathbf{p}_i)\| + \|\nabla f(\mathbf{p}_{i+1})\|}{2} \|\mathbf{p}_i - \mathbf{p}_{i+1}\|.$$

We define a threshold T to filter out noisy PELs with strength less than T. This threshold can be specified by the user or estimated with statistical methods.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

4.2.4 Experimental Results

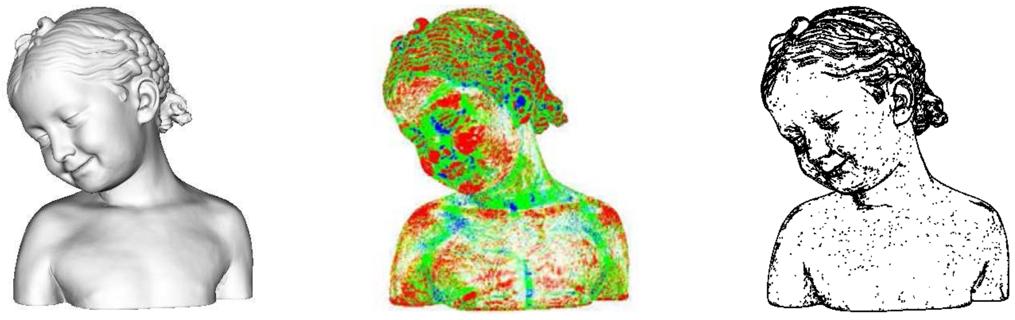
We conduct our experiments on a workstation with a 3.2GHz Xeon processor, NVIDIA Quadro FX 3450 display card with 256M memory, and 3GB of RAM. With our unoptimized code, 0.170 to 1.684 seconds per frame can be achieved with test surfaces ranging from 44K to 320K triangles, as shown in Table 4.1.

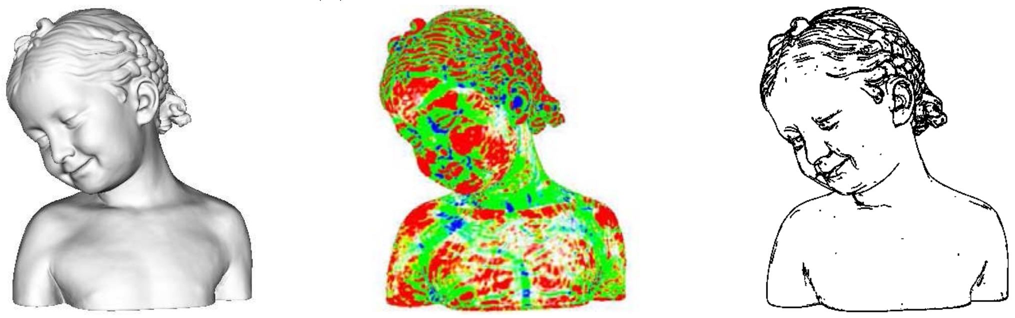
Table 4.1: Performance of PEL illustration

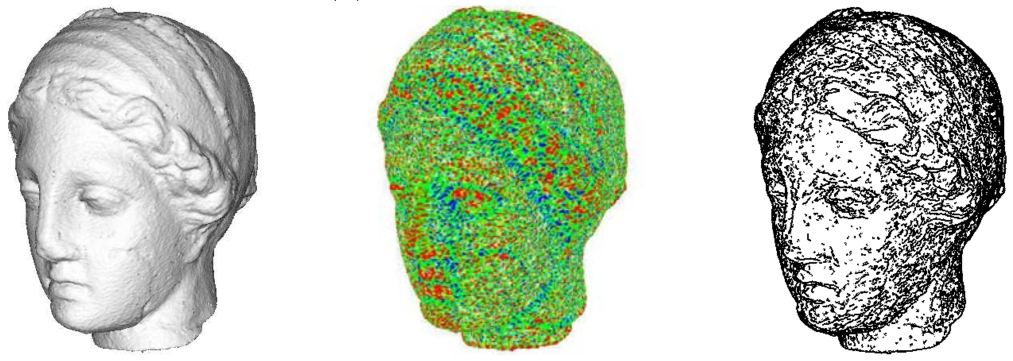
Model	Triangles	Seconds/Frame
Synthetic Surface (Fig. 3.2)	44K	0.170
Max-Planck (Fig. 3.13)	98K	0.350
Bunny (Fig. 3.5)	144K	0.515
Foot (Fig. 3.1)	150K	0.650
Sculpture (Fig. 3.10)	200K	0.869
Head (Fig. 3.16)	320K	1.684

Smoothing normals of surfaces with bilateral filter is optional in our algorithm. It improves the PELs illustrations for models with noise, such as those directly generated with 3D scanners. PELs are defined with third order derivatives of surfaces which are extremely sensitive to noise. As demonstrated in Figure 4.3, the PELs illustrations of the Bimda and the Venus model before smoothing contain a lot of distracting short lines, while the PELs illustrations of the models after smoothing improve greatly. In computer assisted geometry design, gaussian curvature map is always used to illustrate the smoothness of a 3D surface. Thus, we also show the gaussian curvature maps of the models before and after smoothing, encoded in colors, in the middle column of Figure 4.3 to demonstrate the effectiveness of the smoothing operation. The bilateral filter is computational expensive. The timing of smoothing normals of surfaces with bilateral filter is not included in Table 4.1. This is because the smoothing is required only for noisy models, and it will be performed only once in the preprocess. Thus, it won't affect the overall performance of the whole system.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library


(a) Bimda before smoothing


(b) Bimda after smoothing


(c) Venus before smoothing

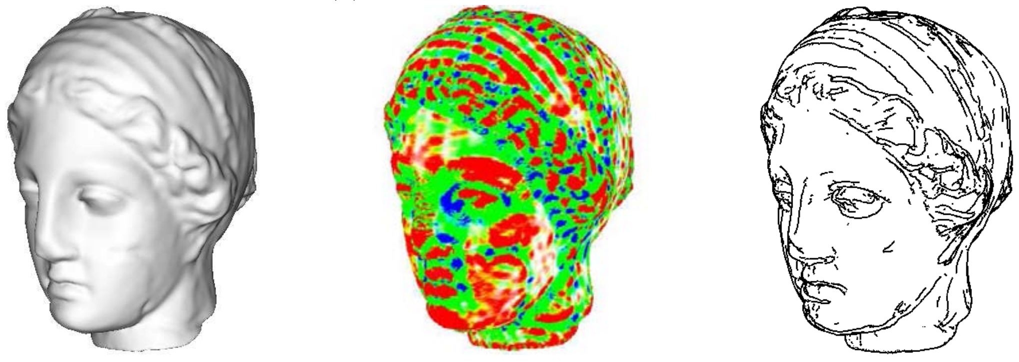

(d) Venus after smoothing

Figure 4.3: Smoothing normals of surfaces improves PELs on noisy models greatly. The shaded models(Left), Gaussian curvature maps(Middle) and PELs(Right) of the Bimda and the Venus model are shown for comparison.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

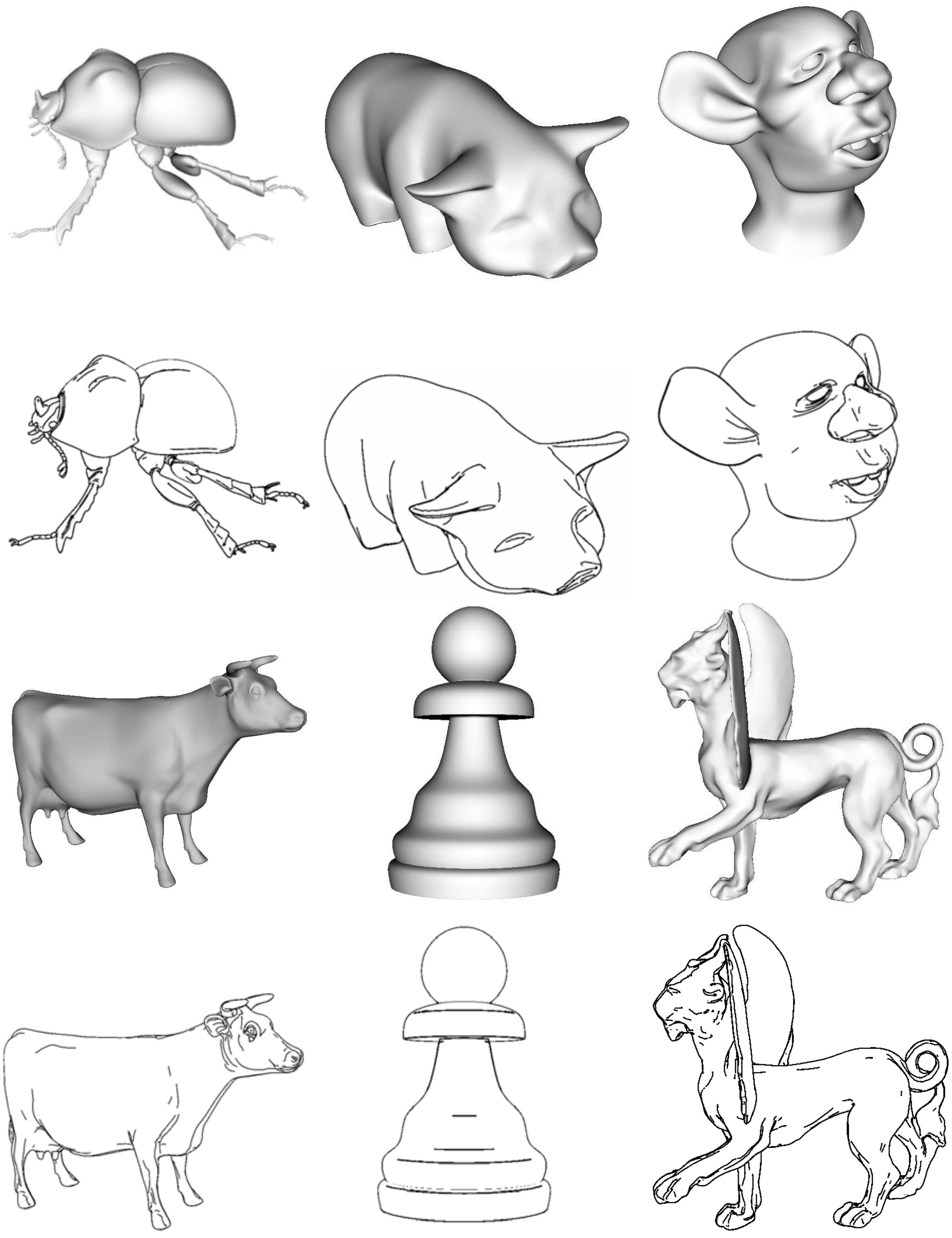


Figure 4.4: Conveying 3D shapes using PELs: six examples.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

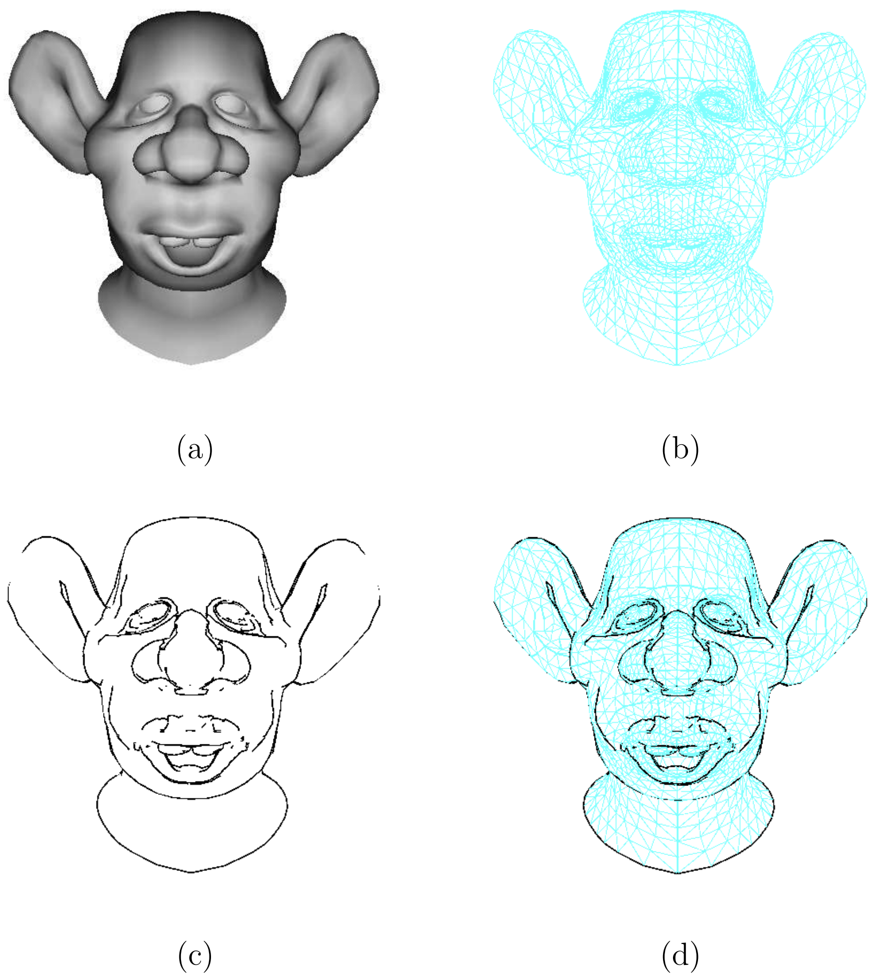


Figure 4.5: Illustration of PELs on rough models, showing that the extraction algorithm is robust which can achieve good results on rough models. (a) Shaded model. (b) Model rendered with wire frame. (c) PELs. (d) PELs on the wire frame.

Fig. 4.4 shows some examples of PELs on surfaces. For more precise high order derivatives computation on surfaces, most of our experimental results are generated with refined models with large numbers of triangles, ranging from $44K$ to $320K$. Our algorithm of PELs extraction on meshes is also robust for rough models. An example is shown in Figure 4.5. In this example, though the triangles are not so dense(5.8K triangles), good PELs illustration can still be achieved.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

4.3 PELs Extraction in Geometry-Image Space

In contrast to most feature lines extraction algorithms which work in object space, image space or a hybrid of both, in this section, a novel algorithm which extracts PELs in a geometry-image domain is proposed. PELs extraction in geometry-image space can be done in two steps. First, global conformal parameterization is applied to the an input model to generate the corresponding geometry image. Then, PELs are computed in the geometry-image domain by applying 2D image processing techniques.

4.3.1 Geometry Images

Surface geometry is often modeled with irregular triangle meshes which causes the computation of geometrical properties on 3D surfaces difficult and slow. Thus we need the process of remeshing to approximate the surface using a mesh with more regular connectivity. On the contrast, 2D images have highly regular structure and connectivity. Taking advantage of the regularity of the 2D grid, high efficiency can be achieved and a wide range of off-the-shelf image processing tools have been developed. A geometry image [36] is obtained by first parameterizing the 3D model and then resampling it based on the regular parameterization grids [38, 55, 37]. Hence, a geometry image resembles a conventional image in terms of its storage regularity. As demonstrated in Figure 4.6, the global conformal parameterization of 3D models is shown in the middle column, with texture mapping to illustrate the global conformal property of the parameterization. Based on the global conformal parameterization, regular 2D grid is generated for resampling. The illuminated geometry images are shown in the right column. Two directional lights are applied to show the whole models more clearly in the geometry images, with one from the front and the other from the back. As pointed out in [36], for a complicated surface, cuts must be found to open the surface into topological disks for a global con-

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

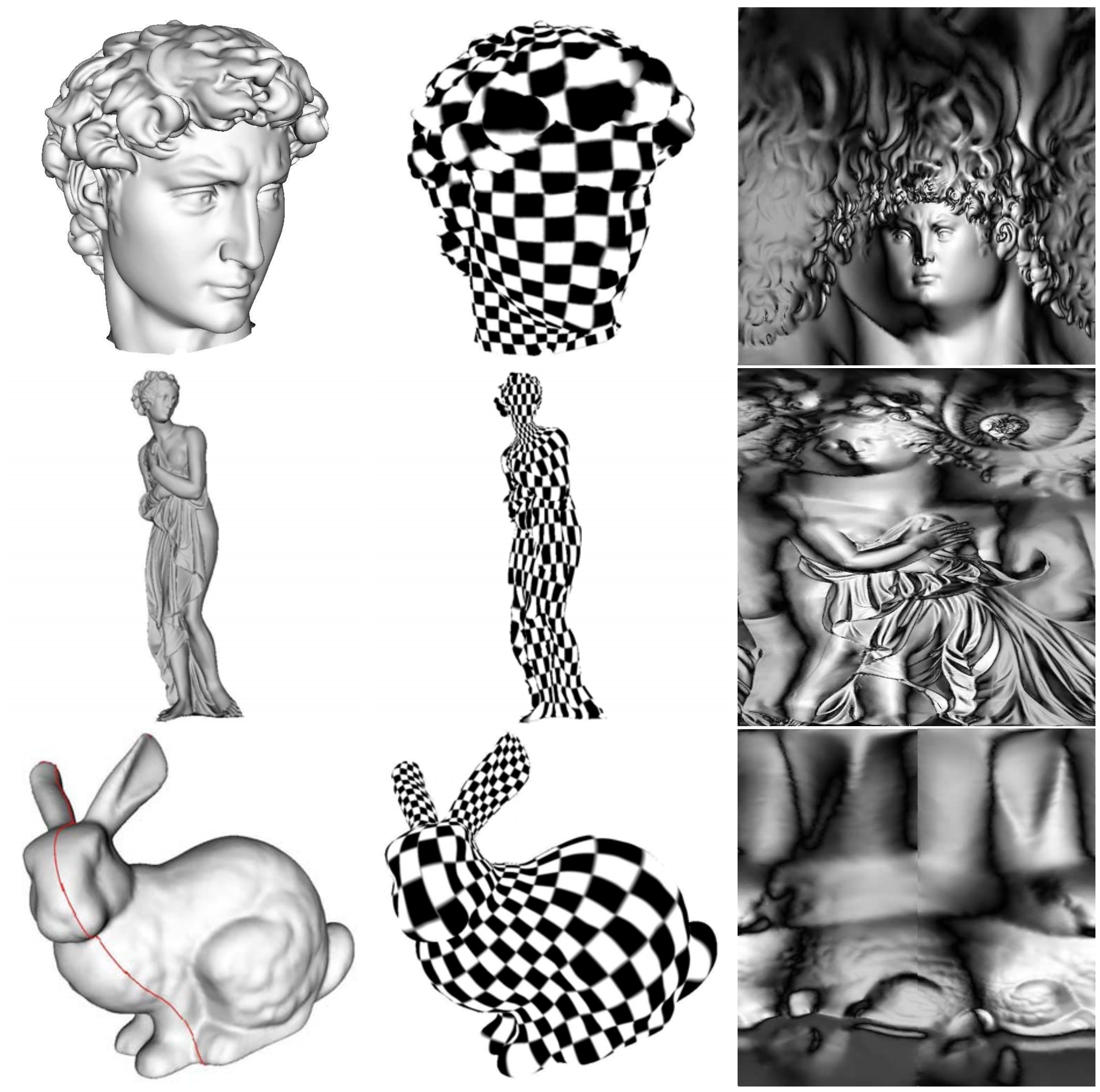


Figure 4.6: Geometry Images. (Left)shaded models. (Middle)texture mapping with global conformal parameterization. (Right)Illuminated geometry images. The bunny is cut into two patches.

formal parameterization. In our examples, the bunny is cut into two patches, as can be seen in Figure 4.6, the cut is represented with a red line.

Working in geometry-image space has combined benefits of working in image space and object space. Geometry images allow us to take advantage of the regularity of 2D

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

images and yet still have full access to the 3D geometry information. We consider geometry images a bridge between 2D and 3D. The main advantages of working in geometry image space are:

- By shifting 3D operations from geometries to 2D images, a wide range of image processing techniques can be applied on 3D surfaces directly.
- Taking advantage of the regular structure of the 2D geometry-image, it is easier to leverage the programmability of modern commodity graphics hardware for 3D geometry processing.

Applying 2D image processing techniques in geometry image space for 3D geometry processing is intuitive for a wide range of applications. Specifically, we apply edge detection techniques of images for our PELs extraction.

4.3.2 Geometry Images Based PELs Extraction

There are two important factors which might affect the result of applying 2D image processing techniques in geometry image spaces. One is the resolution of the geometry image. The geometry image is generated by resampling the original surface. The resolution is correspondent to the sample rate. If the image resolution is not high enough, highly curved shapes in the geometry may be missed. In our implementation, a convenient user interface is designed for users to decide which level of geometry details to be preserved in the geometry image during the process of geometry image generation. The other important factor is the distortion of the geometry image. The global conformal parameterization affects scaling. It has already minimized distortion, but it will still introduce isotropic distortion. We compute a distortion factor to compensate for this effect. In our method, the distortion factor is approximated as the ratio of the local surface area in the 3D space to the parameterized 2D area in the geometry image.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

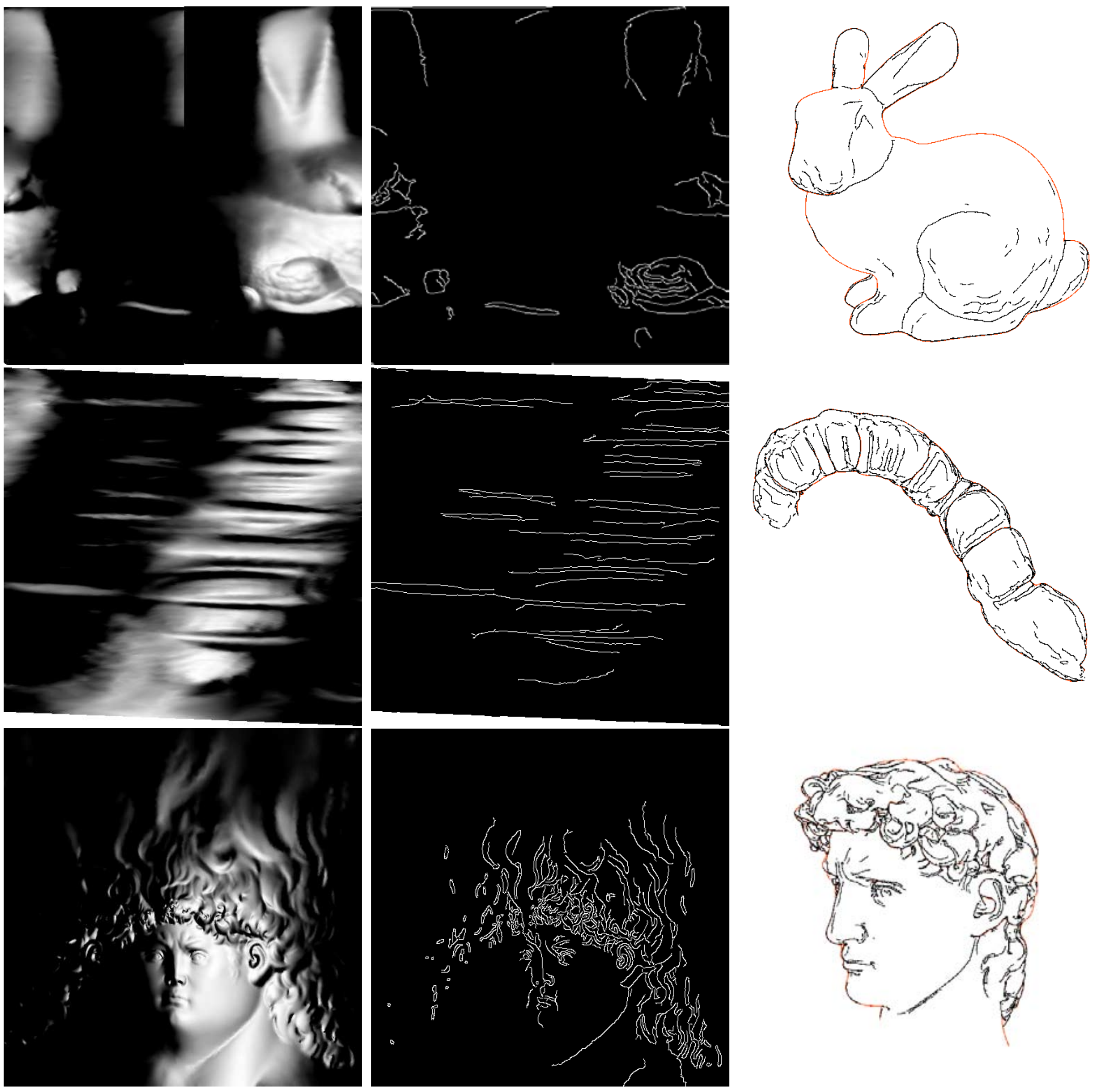


Figure 4.7: PELs extraction in geometry-image space. (Left)Illuminated geometry images. (Middle)PELs of geometry images which are generated with 2D image precessing techniques. (Right)PELs of surfaces by mapping PELs of geometry images back onto the 3D surfaces. Contours are drawn in red for the 3D surfaces. The two patches of the bunny are processed separately.

According to the definition of PEL, specifically, if $S = R^2$, PEL defines points which are zero-crossings of the second order derivative of the image intensity function in the direction of the greatest magnitude of the image gradient. This is similar to the ridges

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

and valleys of images defined in [40]. But, applying edge detector of [40] on a rendered image of a projected surface for PELs extraction is not precise, since the projection from 3D to 2D image plane can not preserve some 3D information, such as the depth. Furthermore, the neighbors of a pixel in the projected image is not always its neighbors in 3D space. Geometry images won't cause these problems. By applying edge detector of [40] on a geometry image, more precise PELs can be achieved. Examples are shown in Figure 4.7, in which the PELs are generated for geometry images with the above mentioned 2D image processing technique, and then traced and mapped back onto the 3D surfaces. We generate PELs illustrations by rendering line primitives mapped back on the original 3D surfaces. In this way, the visibility problem of lines is solved, and we can also avoid the blurring problem of lines which may be caused by directly texture mapping the geometry images back onto the 3D surfaces. Note that the bunny is cut into two patches which are processed separately.

4.4 Discussion and Conclusion

In this chapter, we introduce two novel algorithms for PELs extraction. One is the object space PELs extraction algorithm for triangle meshes which computes high order derivatives on 3D surfaces with advanced discrete differential geometry methods. The other is the geometry-image space algorithm which extracts PELs by applying 2D image processing techniques. With these methods, PEL may become a convenient tool to be incorporated in geometry processing and rendering systems.

Compared with existing feature lines, PELs are more flexible which offer users more freedom to achieve user controllable line drawings. Based on the PELs extraction algorithms introduced in this chapter, especially the object space algorithm for triangle meshes, we have developed a user-friendly prototype system for the easy control of PELs illustrations. Most existing feature lines control the line drawing only with a global

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

threshold. PELs also define such a threshold to filter out noisy lines, as shown in Section 4.2.3. Furthermore, PELs are lighting dependent feature lines which can be controlled locally by modifying the lighting condition in user interested regions. This is achieved by applying different number and types of light sources, and changing the position or direction of each light source. Users can add or remove PELs in interested regions by adding or deleting auxiliary lights, as demonstrated in Figure 3.5, in which four auxiliary lights are used. In most of our experiments, spot lights are used as auxiliary lights. They can also be point lights and directional lights. An example of using directional lights to improve PELs is shown in Figure 3.7. Providing users with more parameters gives users more freedom to control the line drawing. However, it will also cause problems in finding the best combination of parameters. In our case, users need to try a lot to find the best position and direction of each auxiliary lights. To make the system more user friendly, we compute the optimal light setting including light position and direction automatically with optimization methods, as illustrated in Section 3.2.3. In summary, using our system, users can design PELs by simply controlling a global threshold and selecting interested regions to change PELs with simple click and select, as demonstrated in the accompanied video(http://www3.ntu.edu.sg/home/Xuexiang/papers/PELs/PELs.avi).

To validate our system, we have professional artists from a cartoon animation studio who have no technical background use our system. Given a very brief tutorial as in the video, the artists are able to create most of the examples in Chapter 3 and Chapter 4 in less than 15 minutes. By applying different auxiliary lights to interested regions in their own way, they also generate quite different line drawings for the same model. In many cases, to further improve the results, artists need to dither or manually modify the positions and directions of the auxiliary lights after they are automatically computed in the system.

Performance is also a decisive factor of a PEL illustration system, because PELs are user controllable feature lines which means the PEL system should response to the

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

user operations interactively. PELs are defined with high order derivatives on surfaces. Generally, this is computed with patch fitting which is too computational expensive for an interactive system. We develop our object space PELs extraction algorithm based on discrete differential geometry techniques which achieves interactive rendering speed successfully for models of moderate size, as demonstrated in Table 4.1. To further speed up our algorithm for real-time rendering, shifting the computation to GPU might be considered. However, this is still a very challenging problem in current stage. Both the bilateral filter and the derivative computation need at least 2-rings neighbors of the vertices and triangles. This neighborhood information on an irregular grid(the mesh) is difficult to be encoded efficiently. Even after we have collected this information in the GPU memory, the bilateral filter and the derivative computation requires a large number of searches which is also very expensive on the GPU. Current research in this area tries to organize data in some regular data structure in GPU, such as octree and kd-tree [142, 143]. Similar to the ideas of these methods, we propose our geometry-image space algorithm. Geometry image is the re-sampling of the surface in a regular 2D grid which might help in the utilization of GPU for the computation of PELs. In current stage, the global conformal parameterization for geometry images generation is still computational expensive, which can only be done in a pre-process phase. One of our future works is to improve both our object space and geometry-image space algorithm for real-time rendering.

PELs extraction in geometry image space is intuitive, but it still has many technical challenges. Firstly, to process 3D geometry in geometry-image space, complicated shapes always have to be cut into patches. The way of finding the cuts varies with applications. And this will introduce seams between patches. Wang et.al. [125] propose a novel method of mapping the geometry onto a polycube, as shown in Figure 4.8. The polycube has rectangular structures everywhere except a very small number of corner points. The

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

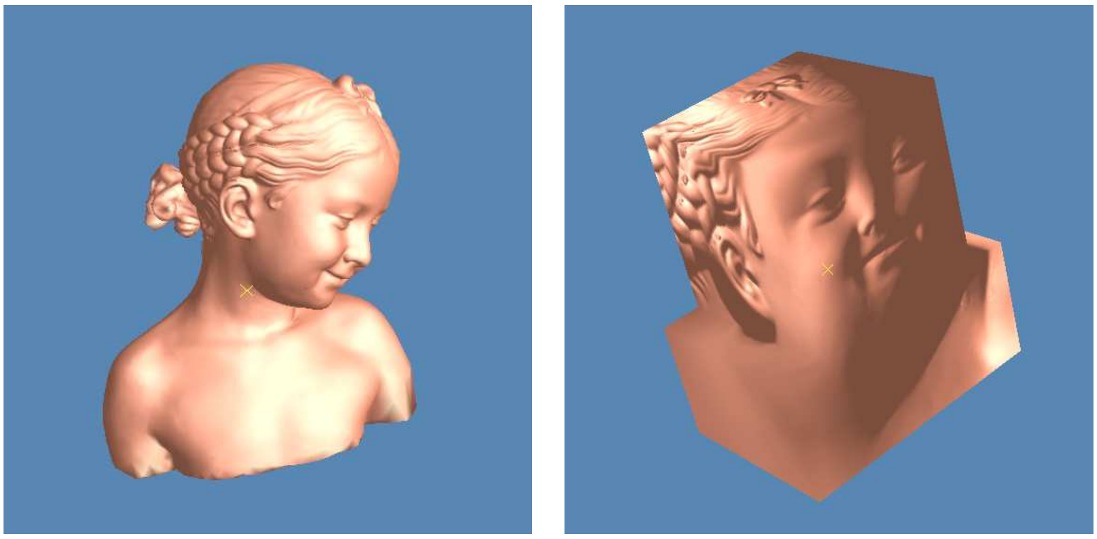

(a)Shaded model (b)The polycube map with shading

Figure 4.8: Polycube map.

advantages of polycube map is that it doesn't need cuts and won't introduce seams between patches. 3D geometry processing in polycube map space is one of our important research directions though still a lot of work need to be done in this area. Secondly, our current solution for the resolution problem of the geometry image are not general enough. To preserve higher level of geometry details, bigger geometry image is required. In [139], the resolution of the geometry image is intuitively set to $m \times m$ where $m = \lceil c\sqrt{n} \rceil$ with n being the number of vertices and c being a constant between 2-5. In our experiments, geometry images of 512×512 give good results. This introduces a tradeoff between the geometry precision and the computational efficiency. Recently, Yao et al. [136] propose a post-processing utility, adaptive geometry image(AGIM) which achieves adaptive resolutions geometry images without T-vertices. One of our on-going works is to adapt this novel method to improve our 3D geometry processing in geometry image space. Furthermore, the distortion of the geometry image is also an obstacle of applying existing 2D image processing tools directly, which requires modification of existing image processing algorithms with special consideration on this issue.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

Besides 3D surfaces and volume datasets, PELs may also be extended to convey shapes of point clouds. The extension is not straightforward, since in most cases, normals and connectivity are not provided in point data sets and their reconstruction is a non-trivial problem, especially when sharp features must be preserved. Most existing methods focus on finding crease lines which are described as the discontinuity of normals of point sets. Demarsin et al. [18] compute point normals using PCA and segment the points into clusters based on normal variation. A minimum spanning tree is constructed between clusters which is used to find feature lines. Gumhold et al. [39] build a Riemann graph of points and use covariance analysis to compute weights that flag points as potential points of feature lines. PELs extraction on point clouds need to compute the normal, gradient of illumination, and directional derivative of gradient magnitude of each point. The above mentioned techniques can be adapted to handle these computations. Furthermore, the direct visibility of feature lines is not considered in most of the previous works. Katz et al. [59] propose a fast operator HPR that determines the visible points in a point cloud without reconstructing the surfaces, which gives cues for the direct visibility of PELs of our on-going work on PELs extraction on point clouds.

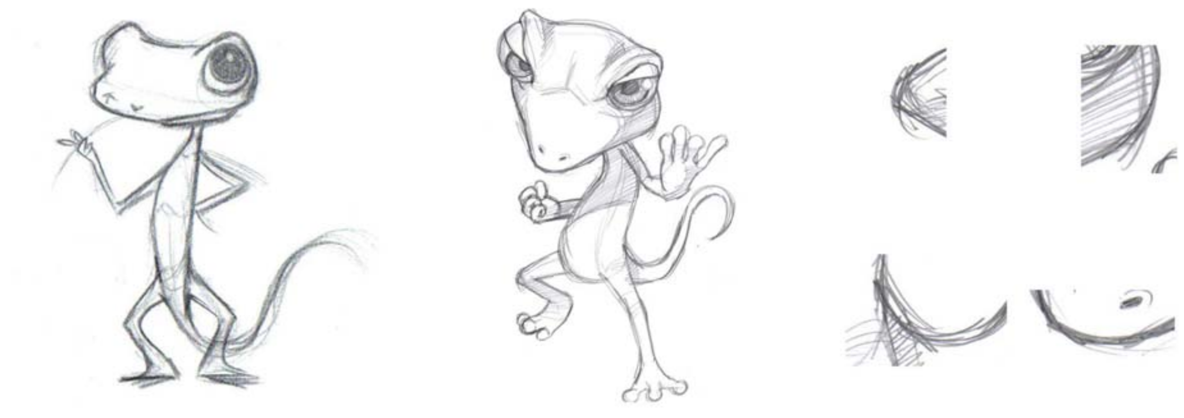
ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

Chapter 5

Conveying Shapes with NPR Sketching

Sketching is a traditional artistic style for conveying shapes with special beauties. The drawing process of sketching displays successive approximation: shapes are first coarsely rendered, then successively more precise strokes are overlaid to improve the rendition. The approximation of feature lines in sketching involves multiple overlapping strokes that are relatively longer in regions of low curvature and shorter in high curvature areas, yet unimportant high-curvature details are omitted. Rendering feature lines with strokes of constant length does not capture the feel of a sketch, and a simple strategy of breaking strokes at curvature maxima is not effective and robust enough in handling unimportant details and noise. In this chapter, we address the problem of breaking sketching strokes with a novel approach which incorporates principles of perceptual grouping with global segmentation algorithms. The strokes generated by this approach qualitatively resemble those produced by artists, and the successive approximation effect seen in sketching can be simulated by employing our approach at a succession of scales (increasing the number of clusters).

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library



(a) (b) (c)

Figure 5.1: Examples of real sketches, with stroke details(c)

5.1 Introduction

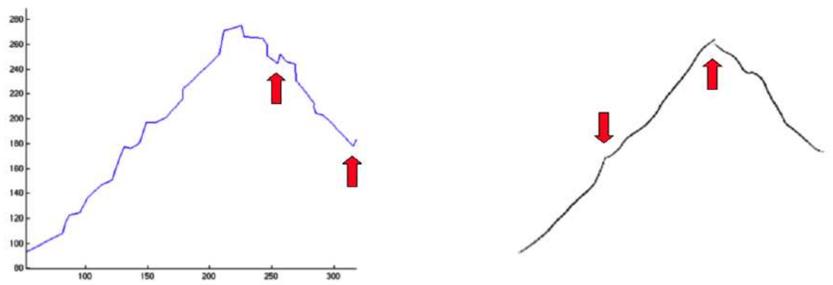
5.1.1 Observation of Traditional Sketching

Sketching is a traditional artistic drawing style which is also widely used in a wide range of other drawing styles at the initial stage in capturing shapes of scenes and objects. ”Sketching” describes a drawing that is not a final, perfect result. In a sketch the drawing process is somewhat evident, with approximation of feature lines and successive approximation in stroke placement often being visible. Although this approximation might be considered in abstract to be a sort of ”rendering error”, in fact, the breaking of strokes and the character of the individual stroke in a sketch can be quite beautiful, and may be a greater component of the art than the actual subject depiction.

In this chapter, we seek to partially emulate the strokes in sketching. Typically these strokes are made lightly and at relatively high speed, and a single feature line is successively approximated with multiple strokes. Because rapid hand movements cannot curve sharply, strokes are often broken at points of curvature. Figure 5.1 shows examples of characteristic sketching stroke effects.

These effects can be approximated with simple and local techniques, such as breaking feature lines at curvature maxima, and perturbing strokes with noise. Although the

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library



(a) (b)

Figure 5.2: Breaking strokes at curvature maxima is not ideal for sketching. (a) shows curvature maxima of the test data set which results in unsatisfied stroke segmentation. (b) demonstrates greatly improved strokes achieved with our perceptual stroke segmentation method. The test data set is rendered with three strokes. It is split at perceptual salient points rather than at curvature maxima, or into roughly equally sized strokes in the absence of distinguished features (left slope).

resulting drawing is likely to be adequate for many purposes, it is unlikely to be an ideal example of a sketch: simply breaking the feature lines at curvature maxima is not always effective, since not all maxima are equally important and not all high curvature points are perceptually salient, as shown in Figure 5.2. This is doubly true in common cases where the original lines have some artifactual curvature and gaps resulting from the discrete raster (e.g. vectorization algorithms that locally are limited to stepping in horizontal, vertical, and diagonal directions), or from tracing feature lines of a polygonal rather than spline model. While the algorithmic artifacts alone can be addressed by fitting approximating splines to the feature lines, the splines may also smooth important feature points, obscuring the question of where to split the feature lines. Figure 5.2 demonstrates that breaking strokes at curvature maxima is not ideal. Note that our perceptual stroke segmentation method breaks strokes at somewhat perceptual salient points. In a large region without distinguished features, feature lines are broken into roughly equally sized strokes which is also an important character of sketching strokes.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

5.1.2 Related Work and Our Contribution

To convey shapes with stylized line drawing, feature lines(e.g., contours, suggestive contours, PELs etc.) are first extracted and linked into long strokes. The strokes can then be rendered in carefully designed styles. A lot of researches have been carried out on issues including feature lines extraction and styles of strokes, but no enough attention has been paid to the problem of breaking strokes in artists' way.

The literature on line drawing algorithms is large and includes issues of stroke placement and orientation (e.g. [102]; [42]). [44] provides a survey focusing on stroke placement issues (see also [31] and [113]). Examples of recent developments include [111], which selects a small number of salient contours that are then carefully rendered in an ink style, and [68], which uses image statistics and regional texture to tune pencil-like shading (hatching) to suit different regions. Closely related to our work, [81] describes a stochastic optimization process for sketching in which a truncated Fourier basis provides a prior of admissible deformations on stroke shape, and edge gradients provide the data. Less probable strokes are rendered with less pressure and smaller width. While [81] chooses only one stroke per non-overlapping window on the image, a minor modification of their process to repeatedly sample strokes at a single location could generate sketching effects with overlaid strokes. However, their process is somewhat expensive, requiring on the order of tens of minutes even without repeated sampling.

Our contribution is to note that the stroke segmentation problem in sketching can be considered as exactly that - an instance of a segmentation or clustering problem. The proximity and orientation of feature lines samples can be compared, with sufficiently similar samples being naturally grouped into strokes. This approach opens the problem to a variety of clustering and segmentation algorithms that have been developed in recent years, including the spectral clustering methods such as [85], [106]. We employ the normalized cuts algorithm (Section 5.3), which is further adapted for our sketching strokes

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

segmentation incorporating principles of perceptual grouping for its affinity measurement. We demonstrate that it is a flexible method which can produce the coarse-to-fine successive stroke segmentation and approximation as seen in real sketching.

5.1.3 Overview of Our method

Our approach of producing a NPR sketch consists of the following steps:

(i) feature line extraction and tracing.

(ii) determining visible feature line segments.

(iii) stroke segmentation.

(iv) spline approximation for each segment.

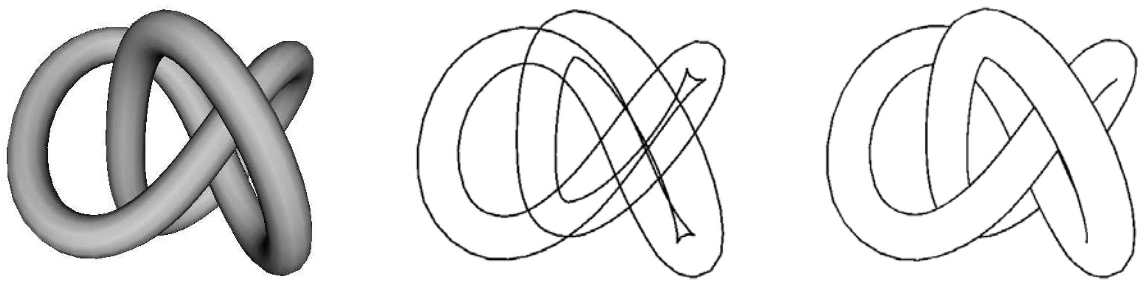
(v) stroke rendering.

This is also the general framework of stylized line drawing. Feature lines extraction is introduced in previous chapters. The contribution of this chapter is in step three (stroke segmentation). In the following sections, we describe step two and step four briefly in section 5.2 and section 5.5. Stroke segmentation is described in two sections. Section 5.3 introduces some related clustering and segmentation algorithms with focus on the normalized cuts algorithm. Section 5.4 describes our affinity measure incorporating perceptual similarity for normalized cuts. The rendering results are demonstrated in section 5.6.

5.2 Stroke Visibility

Once we have extracted feature lines from the 3D model, the next step is to determine which portions of the feature lines are visible. The z-buffer cannot directly compute visibility for strokes because the stylized stroke based on the feature line would generally

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library



(a) shaded model (b) all feature lines (c) visible feature lines

Figure 5.3: Determining visible strokes with ID image.

interpenetrate the surface. We adopt the method of [86], which resolves visibility using an ID reference image. Briefly, the ID reference image is created by rendering the scene into an off-screen buffer with each feature line and each triangle drawn in a unique color. Next we iterate over all the pixels in the reference image and build a list L of feature lines that contributed at least one pixel. We thus remove from consideration feature lines that make no contribution to the pixels of the current frame. Then we scan-convert along each feature line in L to determine which portions of it show up in the ID reference image, and hence are visible. Isenberg et al. [53] improve this method for determining visible silhouettes with z-buffer. They look at the 8-neighborhood of a point to determine its visibility, since a discontinuity must exist in z-depth for each silhouette point. This work well for silhouettes, but feature lines such as PELs won't introduce the required discontinuity in z-depth. An example of visible strokes extracted with ID image is shown in Figure 5.3.

5.3 Normalized Cuts

Image segmentation is often approached by clustering data into coherent ”clusters” based on location, pixel color, and other attributes. Some methods base on points and treat

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

points as vectors in some feature space for segmentation, such as classical k-means algorithm and Expectation Maximization algorithms. In recent years, a new family of clustering algorithms, graph-based algorithms , has been introduced. It considers segmentation as a graph partitioning problem, such as the min-cut/max-flow based graph cuts algorithms [62]. Compared to point-based algorithms, the graph-based algorithms have more flexibility to define constraints or weights on segmentation. Spectral clustering methods such as [85], [106], cut graph by inspecting the eigen-decomposition of a matrix derived from the graph. These methods define edge weights between nodes with an affinity matrix which provides a framework with great flexibility for defining segmentation constraints. Some spectral clustering algorithms, such as max flow, only minimize the cut and make no guarantees about the relative affinity between nodes within a segment. Normalized cuts [106] improve these methods by optimizing the graph cut with relaxation method which minimizes the ratio of the value of a cut to a segment's measure of self-similarity.

For graph-based segmentation, we first set up a weighted graph $G = (V; E)$, and then set the weight on the edge connecting two nodes to be a measure of the similarity between the two nodes, which is represented with an affinity matrix W. We will discuss our carefully designed perceptual similarity measure in Section 5.4. Given the affinity matrix W, segmentation proceeds by considering a graph with $W_{i,j}$ being the weight between nodes i and j. The overall goal is to cut the graph such that the weights of the cut edges are small while the interior weights of the resulting subgraphs are simultaneously large. This general formulation describes a variety of clustering algorithms. As described in [29], the intuition behind spectral methods is as follows: a good cluster will have strong weights in the affinity matrix, and elements of the cluster will be strongly associated with the cluster, so the objective $y^T W y$ (with y a vector of weights giving the association of each element with a proposed cluster) will be large for a good cluster. Maximizing this

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

subject with $||y||$ remaining constant gives

$$y^T W y + \lambda(y^T y - 1)$$

with the result that y is the leading eigenvector of W. Values of y larger than a threshold indicate membership in the strongest cluster. This procedure can be iterated on the remaining (unchosen) samples to find additional clusters.

It is known that the basic spectral clustering scheme just described has a tendency to produce extremely small clusters, since the sum of weights to a small group of nodes will also tend to be small. The normalized cuts algorithm [106] addresses this by normalizing the cost of the cut weights by the total weight mass from that cluster, thereby taking cluster size into account. The approach of normalized cuts is to cut the graph into two connected components such that the cost of the cut is a small fraction of the total affinity within each group. This can be formalized as decomposing a weighted graph V into two components A and B, and scoring the decomposition with:

$$Ncut(A, B) = \frac{cut(A, B)}{assoc(A, V)} + \frac{cut(A, B)}{assoc(B, V)} \qquad \text{(Eq. 5.1)}$$

where:

$$cut(A, B) = \sum_{u \in A, v \in B} w(u, v)$$

$$assoc(A, V) = \sum_{u \in A, t \in B} w(u, t)$$

Here, w represents weight between nodes. This score will be small if the cut separates two components that have very few edges of low weight between them and many internal edges of high weight. Our purpose is to find the cut with the minimum value of this criterion.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

This is a combinatorial optimization problem which is still difficult to solve in this form. Eq. 5.1 can be further rewritten as:

$$min_x Ncut(x) = min_y \frac{y^T(D-W)y}{y^T Dy} \qquad \text{(Eq. 5.2)}$$

where D is the degree matrix, a diagonal matrix defined with:

$$D_{ii} = \sum_j W_{ij}$$

and $y(i) \in \{1, -b\}$ with:

$$y^T Dl = 0$$

l is an $N * 1$ vector of all ones. The values of $y(i)$ are used to distinguish between the components of the graph: if the i'th component of y is 1, the corresponding node belongs to the component, otherwise, $y(i) = -b$.

By relaxing y to take on real values, we can minimize Eq. 5.2 by solving the generalized eigenvalue system:

$$(D-W)y = \lambda Dy \qquad \text{(Eq. 5.3)}$$

The last question is which generalized eigenvector to use. It turns out that the smallest eigenvalue is guaranteed to be zero, so the second smallest eigenvalue is appropriate. The best way to determine this eigenvector is to perform the transformation $z = D^{1/2}y$ to get:

$$D^{-1/2}(D-W)D^{-1/2}z = \lambda z$$

One step further, we have:

$$Nz = D^{-1/2}WD^{-1/2}z = \mu z$$

N is sometimes called the normalized affinity matrix.

All together, data set segmentation with normalized cuts consists of the following steps:

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

(i) Given an input data set, set up a weighted graph $G = (V; E)$ and set the weight on the edge connecting two nodes to be a measure of the similarity between the two nodes.

(ii) Solve $(D - W)y = \lambda Dy$ for eigenvectors with the smallest eigenvalues.

(iii) Use the eigenvector with the second smallest eigenvalue to bipartition the graph.

(iv) Decide if the current partition should be subdivided and recursively repartition the segmented parts if necessary.

In our experiments, we adopt the the n-way variant of normalized cuts [138] for our stroke segmentation.

5.4 Perceptual Stroke Segmentation

Normalized cuts gives a powerful mathematical framework for global segmentation. The successful application of normalized cuts for a clustering problem still depends heavily on the measure of similarity of samples in some feature space. In this work, we propose a perceptual stroke segmentation method incorporating principles of perceptual grouping with normalized cuts.

5.4.1 Principles of Perceptual Grouping

According to theories of visual perception, especially conclusions of the Gestalt psychologists [34], there are series of factors that lead to perceptual grouping, including:

- **Proximity:** samples that are nearby tend to be grouped.
- **Similarity:** similar samples tend to be grouped together.
- **Common fate:** samples that have coherent motion tend to be grouped together.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

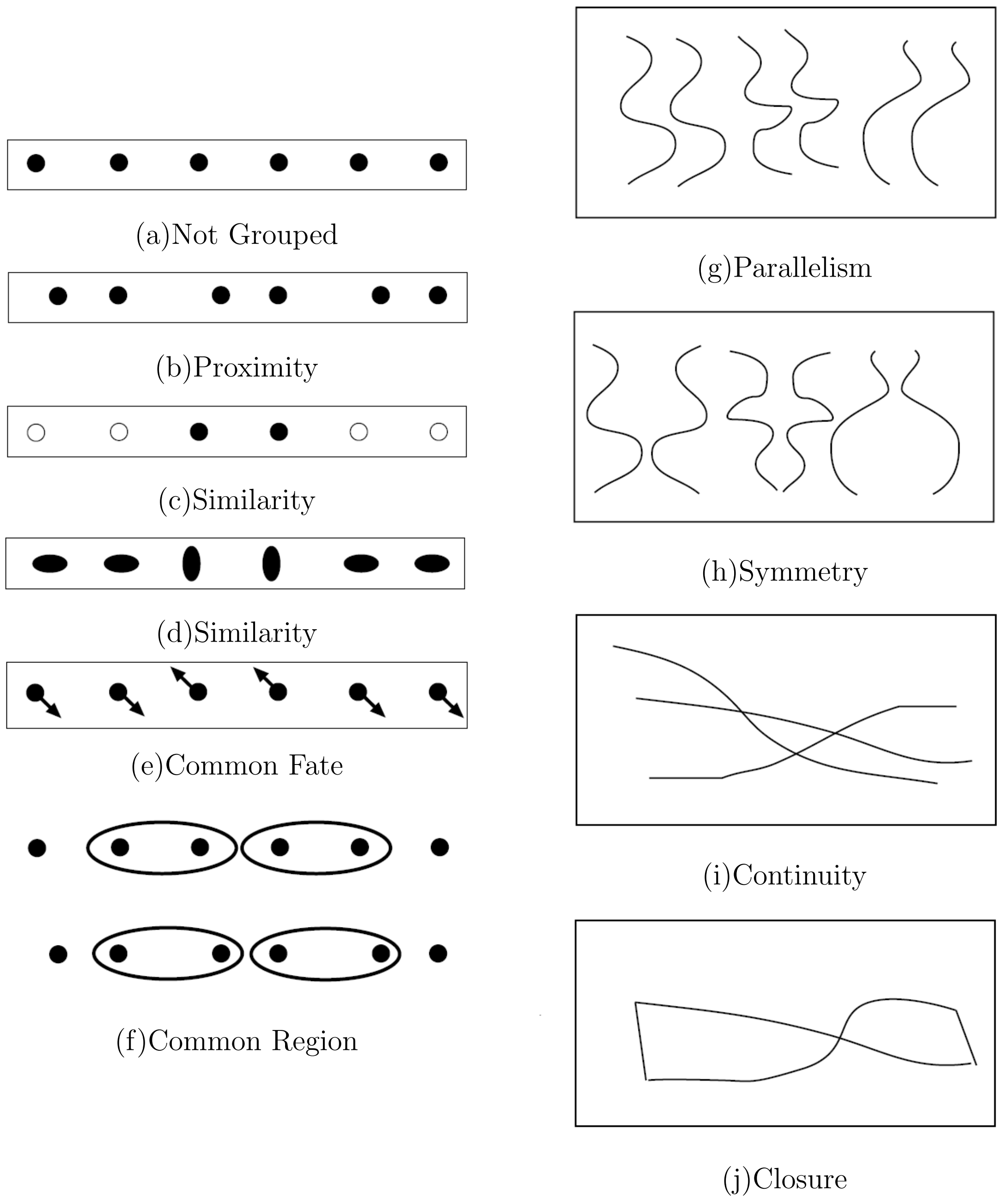


Figure 5.4: Examples of Gestalt factors that lead to grouping. Figures from Gordon, Theories of Visual Perception [34].

- **Common region:** samples that lie inside the same closed region tend to be grouped together.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

- **Parallelism:** parallel curves or samples tend to be grouped together.
- **Closure:** samples or curves that tend to lead to closed curves tend to be grouped together.
- **Symmetry:** curves that lead to symmetric groups are grouped together.
- **Continuity:** samples that lead to "continuous" – as in "joining up nicely", rather than in the formal sense – curves tend to be grouped together.
- **Familiar Configuration:** samples that, when grouped, lead to a familiar object, tend to be grouped together.

These are also called Gestalt factors. Some examples of Gestalt factors are demonstrated in Figure 5.4.

5.4.2 Measure of Perceptual Similarity

Based on the observation of the drawing process of a trained artist, we carefully design our measure of similarity with more consideration on the above mentioned factors of perceptual proximity, closure and continuity. In normalized cuts and other graph cut segmentation methods, the measure of similarity is represented with an affinity matrix W. The affinity matrix is constructed with each row and column corresponding to a numbered node in the graph. In our case, each sample of the feature lines is defined a node. The pairwise affinity $W_{i,j}$ gives the strength of the estimated similarity between any two samples i and j, which can also be considered the possibility of two samples to be linked into the same stroke segment. We get $W_{i,j}$ according to discrepancy of distance and orientation between the two samples. We experiment two slightly different methods of similarity measurement according to the same principles of perceptual grouping.

The first method is inspired by the voting field in tensor voting and perceptual grouping methods [77]. The definition of the pairwise affinity $W_{i,j}$ is illustrated in Figure 5.5.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

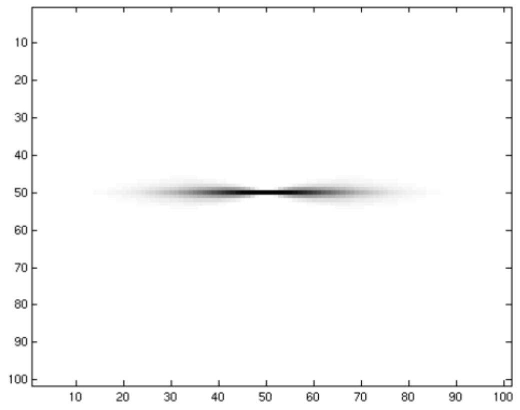


Figure 5.5: Shape of the pairwise affinity, decaying from a horizontally oriented sample at center (black indicates stronger). The affinity falls off with distance, but also allows more orientation discrepancy with increasing distance.

This characteristic "figure-8" shape (Figure 5.5) generally falls off with distance and orientation discrepancy, but it also allows relatively more orientation discrepancy at increasing distance, to allow for smooth curvature of the stroke. This function is computed by projecting the location of the sample j onto the line defined by the orientation of the sample i, and then forming the measure involving the distance along the orientation line d_o for the sample i, the projection distance d_p from the sample j to that line, and the dot product of the two orientations $o_i \cdot o_j$:

$$W_{i,j} = (o_i \cdot o_j)exp(-\frac{(d_o + \alpha d_p/d_o)^2}{\sigma^2}) \qquad \text{(Eq. 5.4)}$$

where α, σ are constants that tune the orientation selectivity and overall scale. The d_p/d_o term causes the orientation selectivity to reduce with increasing distance.

The second method is based on the observation that artists prefer curve strokes in reasonable length during the process of drawing. As shown in Figure 5.6, according to the perceptual constraint of co-curvilinearity, we favor a curve with smaller total curvature to link the two samples. Actually a circular arc is preferred because the variety of curvature

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

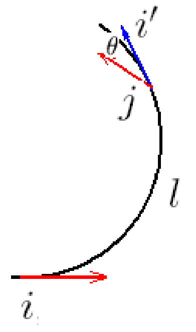


Figure 5.6: Pairwise affinity. The pairwise affinity $W_{i,j}$ is defined with discrepancy of distance and orientation between samples i and j.

along the curve is zero. Thus, we define distance between two samples with the arc-length of the circle, which is l in Figure 5.6. Orientation discrepancy is defined with the angle between the tangent of the curve and the direction of the sample. As illustrated in Figure 5.6, orientation discrepancy between i and j equals to θ between i' and j. By representing θ with the dot product of orientations of the two samples o_i and o_j, the pairwise affinity $W_{i,j}$ can be computed with:

$$W_{i,j} = (o_i \cdot o_j)^2 exp(-\frac{l^2}{\sigma^2}) \qquad \text{(Eq. 5.5)}$$

We add more weight to the orientation factor with $(o_i \cdot o_j)^2$ in this method.

The result of the normalized cuts segmentation depends heavily on the affinity matrix. Comparison of stroke segmentation with normalized cuts but different ways of affinity definition is shown in Figure 5.7. Using a simple weighted sum of proximity and orientation results in a segmentation that incorrectly groups samples on similarly oriented but separate lines which are not well defined individual strokes, as demonstrated in (b). (c) computes the affinity matrix with projection distance and orientation, which results in some very short segments. (d) achieves the best result with totally 28 segments. (Some distinct clusters are colored similarly due to reuse of the color labels.)

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

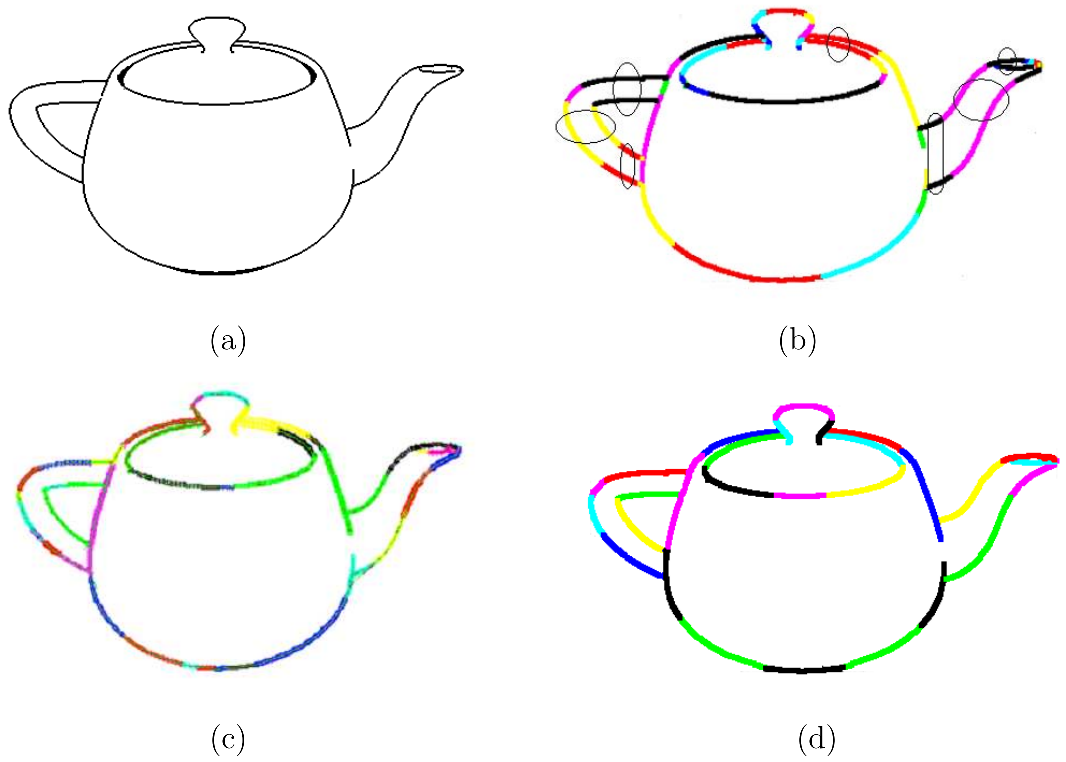


Figure 5.7: Comparison of stroke segmentation with normalized cuts but different affinity definition. (a) Feature lines. (b) Segmentation resulting from affinity defined as a weighted sum of distance and orientation discrepancies. Parallel but separate lines on the handle and spout are grouped together. (c) Segmentation resulting from affinity defined with Eq. 5.4. (d) Segmentation resulting from affinity defined with Eq. 5.5. (Because of the reuse of color labels, separate lines which belong to the same segment are marked as shown in (b).)

5.5 Spline Fitting Stroke Segments

After stroke segmentation, we get stroke segments with no salient feature points. But so often, these stroke segments may still contain artifacts such as stepping effects, noise and gaps etc. We therefore apply stroke fitting before rendering. An example is shown in Figure 5.8. (b) shows strokes before stroke fitting while (c) demonstrates the improved strokes after stroke fitting.

More generally, stroke segments represented with data x can be fit with an approxi-

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

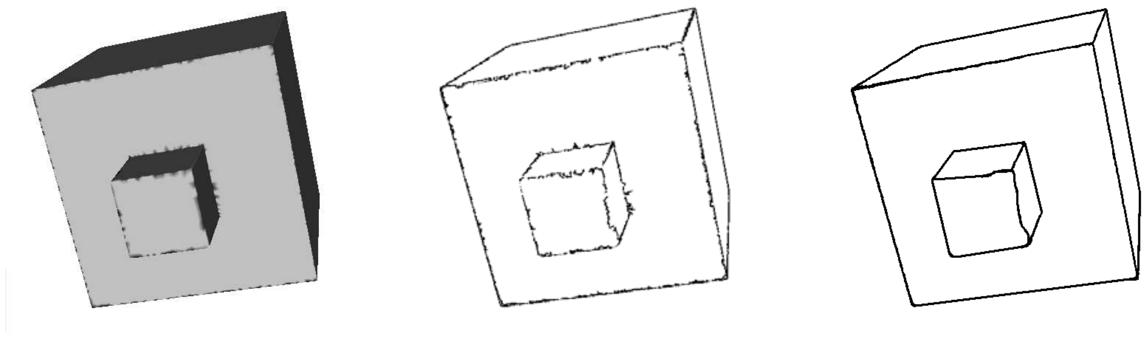

(a) shaded model (b) before stroke fitting (c) after stroke fitting

Figure 5.8: Line drawing with stroke fitting. (b) shows stroke segments containing artifacts such as stepping effects, noise and gaps. (c) shows the improved strokes after stroke fitting.

mating spline y. The spline y is formulated as:

$$min_y||S(y-x)||^2 + \lambda y^T C^T C y$$

where S is a "selection matrix" selecting only the elements of y corresponding to the locations of the data x, and C is a matrix containing the finite-difference approximation to the second derivative. This can be solved for y as a linear system. Further more, this formulation allows a natural formulation of the characteristic stroke "overshoot" often seen in sketching, by having the fit spline to extend beyond the data. In the past the overshoot effect has been obtained simply by extending the line using the slope of the last segment. This sudden transition to an extended straight segment may look unnatural, however, if the the rest of the stroke is significantly curving. In our formulation the curvature in the overshoot region decays more smoothly(Figure 5.9).

Continuing the curvature into the overshoot region can be accommodated easily, by using $||C(y)-c||^2$ in place of $||Cy||^2$, with c being a desired curvature vector (extrapolated from an average of Cx). We believe however that the overshoots in real sketches tend to approach straight lines and so curvature continuation was not employed here.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

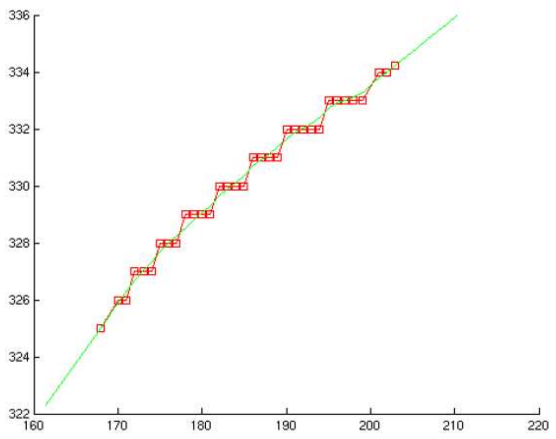


Figure 5.9: Spline fitting stroke segment. A stroke segment with stepping artifacts is fit with an approximating spline with intentional overshoot.

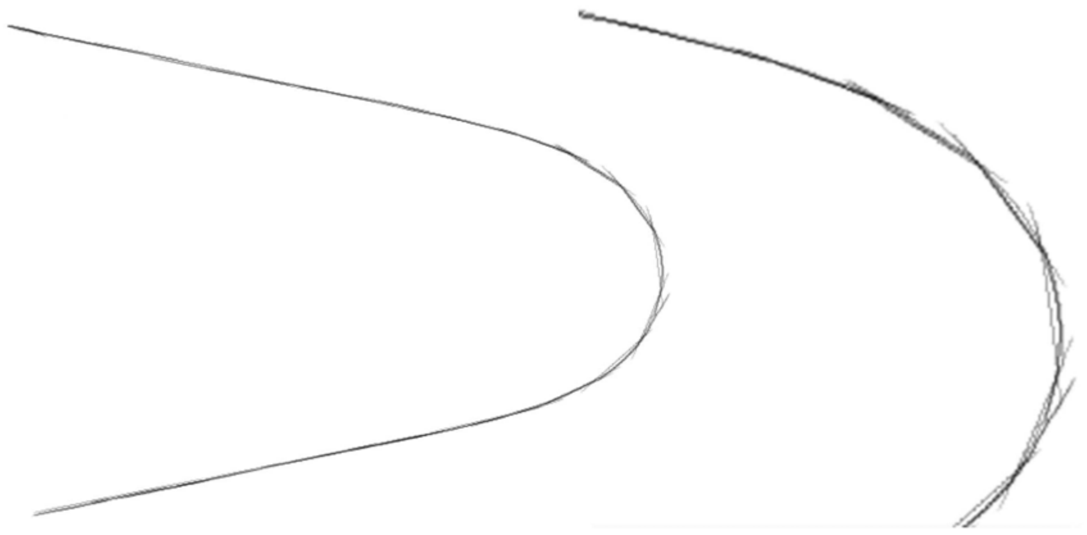
5.6 Results and Discussion

5.6.1 Experimental Results

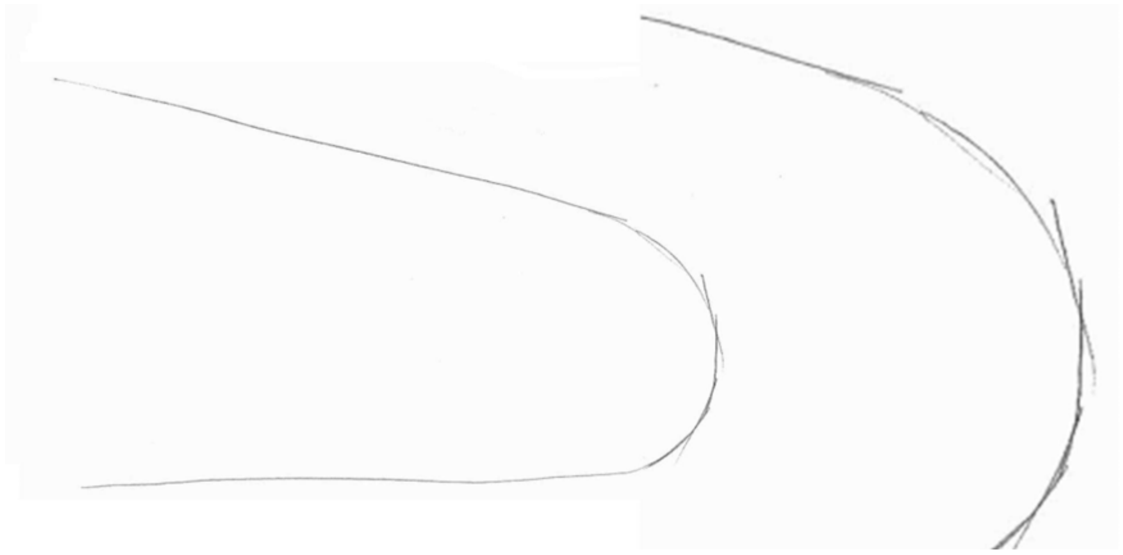
Although we have experimented with more textural stroke rendering styles, all the figures of this part are rendered with simple anti-aliased lines with a constant but adjustable opacity. Design parameters consist of the opacity, the maximum overshoot (overshoot on each stroke is chosen at random up to the maximum), and the usual NPR small random perturbation of stroke vertices (as in [86] for example), added to increase the hand-drawn look. We demonstrate our results in this section with focus on strokes segmentation and sketching simulation, line drawings in a wide range of other styles will be introduced in the following chapters.

Figure 5.10 shows examples of our stroking approach. Note that the stroking effect is rather subtle but can be seen clearly when the figures are enlarged. Strokes generated with our method simulate real sketching strokes with overlaid strokes, stroke overshooting and good texture. Generally, in sketching, artists often lightly outline major structures first

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library



(a) Stroke generated with our method.



(b) Stroke produced by an artist.

Figure 5.10: Sketching strokes simulation. Comparing with strokes produced by an artist(bottom), stokes generated with our method(top) capture the feeling of real sketching strokes, further details can be found in the enlargements(right).

and then overlay additional structures and details. This successive refinement strategy can be quite effectively simulated with our approach, by varying the number of clusters and overlaying the results. The result figures make use of this technique, for example, both images of Figure 5.11 overlays the segmentations obtained with 60, 70, 80, and 90

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

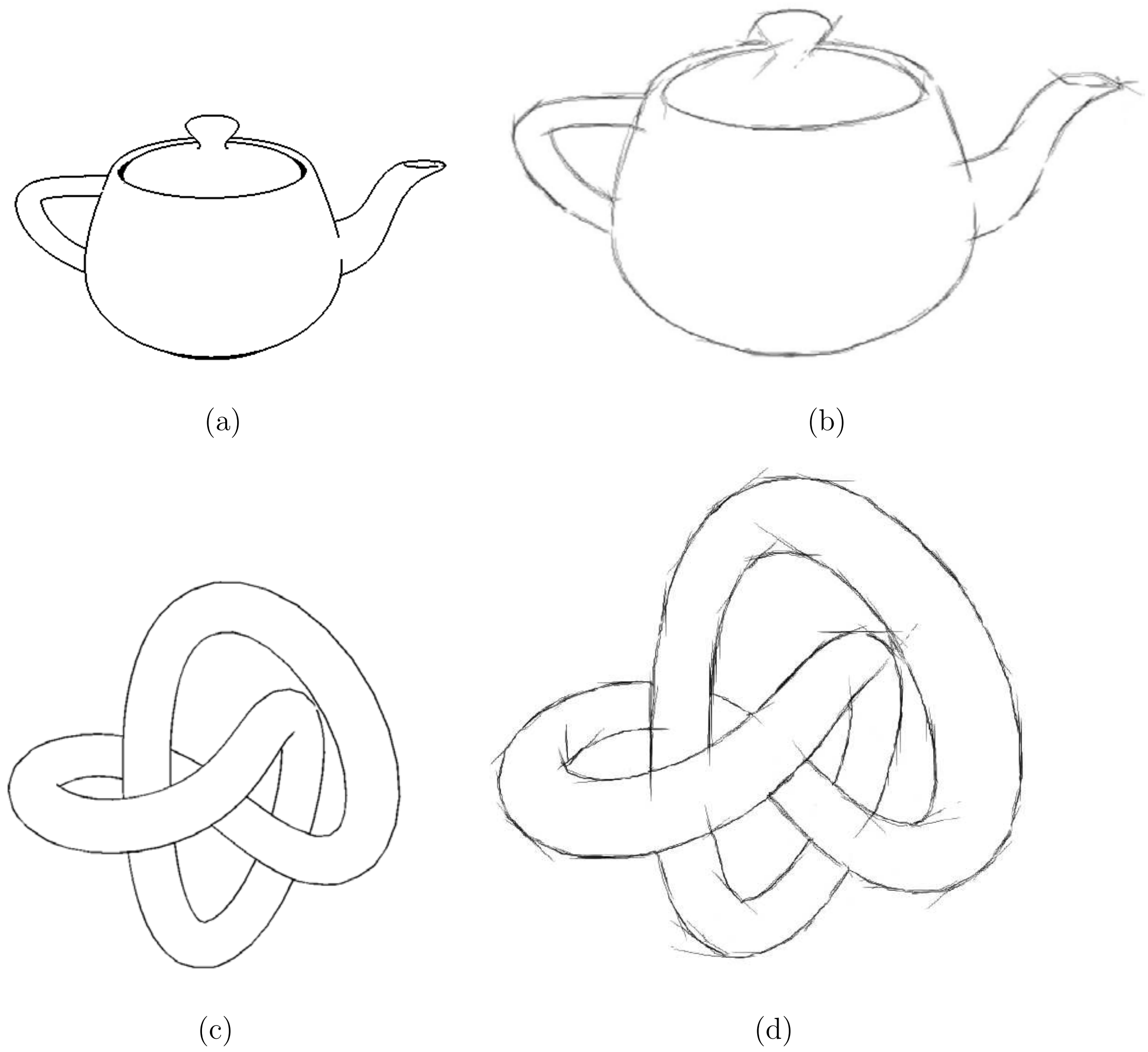


Figure 5.11: Teapot and knot each rendered with four different segmentation levels, and increased overshoot in the knot figure. Enlarge to see line quality. Several slight direction changes in the overshoots at T-junctions are visible in the knot figure; these are due to a few samples in the vicinity of the junction being incorrectly clustered. This could be avoided by tuning the clustering affinity, or by processing each feature edge individually (if the preprocess of feature lines extraction provides this information)

clusters. More examples are given in Figure 5.12 and Figure 5.13. In these examples, the successive approximation effect of sketching is also resembled with the overlaying of stroke clusters. At the same time, we increase the overshooting of strokes to emphasize

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

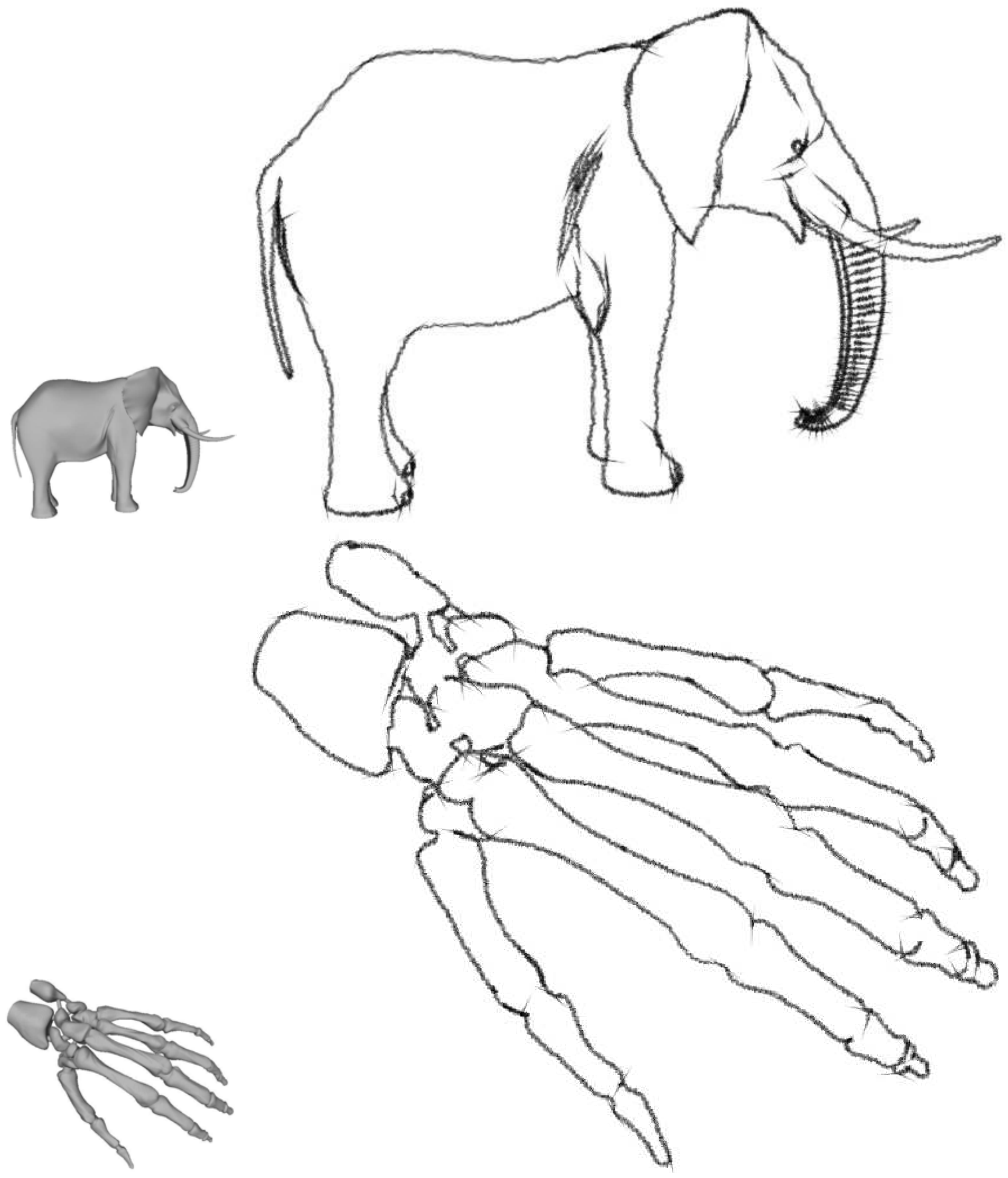


Figure 5.12: NPR sketching with overlaying of stroke clusters.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

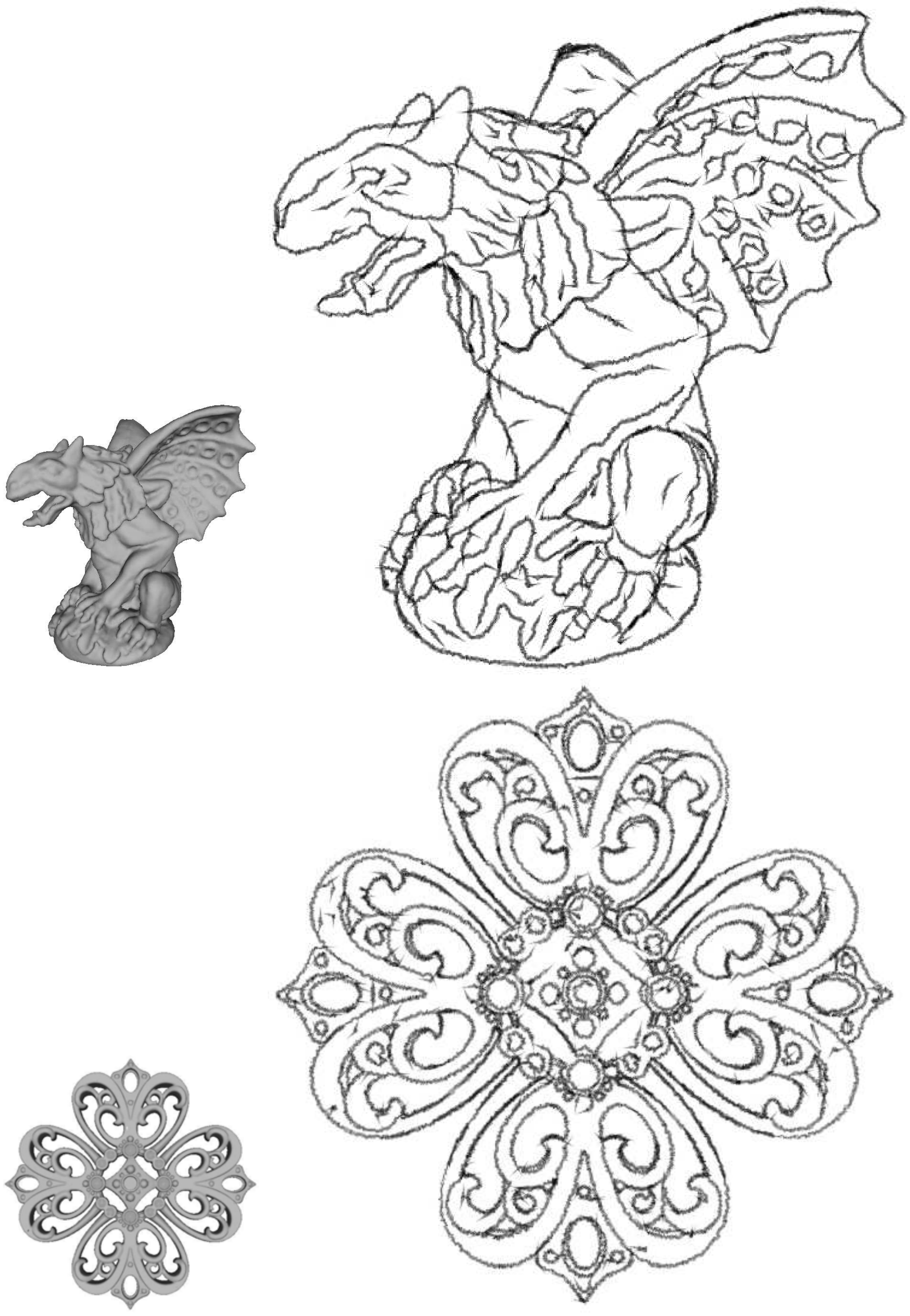


Figure 5.13: NPR sketching with overlaying of stroke clusters.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

the start and end of some single strokes, such as some strokes of the sketching of the hand in Figure 5.12.

5.6.2 Evaluation

In this chapter, we emulate sketching based on observation of some basic properties of traditional sketching. To achieve perceptual consistent and visual pleasing NPR sketching, we develop algorithms incorporating some widely accepted visual perception rules. However, the perceptual consistency of a NPR sketching is highly subjective, and it is quite difficult to objectively evaluate the visual similarity between a NPR and a hand-drawn sketching. Different from the evaluation of PELs, the visual similarity of NPR sketching is more difficult to quantitatively evaluate since appropriate measurable variables for the sketching style are not well defined. This type of evaluation is always done in the field of qualitative research which involves the gathering and use of qualitative data, such as data of interviews. Recently, following study approach of qualitative research, Isenberg et al. [54] present an observational study which compares hand-drawn and computer-generated pen-and-ink illustrations. Based on study methods in this work, we give a qualitative evaluation of the perceptual consistency of our NPR sketching in this section.

Our qualitative evaluation involves discussion, voting and scoring. We conduct an observational study with two groups of people to examine how they understand and assess computer generated NPR sketching. The first group is professional users, including seven artists from a cartoon animation studio(2 female, 5 male, all with a good skill in drawing). The second group is common users, including thirteen researchers from a team working on computational arts(5 female, 8 male, all with good background in computer graphics). First, the participants discuss together to figure out some important features of sketching. Then, they vote separately for some given images to choose those convey

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

3D shapes better. During this process, they are also required to score the given images for comparison.

Though styles of traditional sketching are not precisely defined, some common properties are always observed in many artworks. The participants are active in the discussion, while the professional users are more knowledgable in this area. Some important features of traditional sketching agreed by the participants are figured out to evaluate the computer generated sketching:

- Strokes should show good pencil texture with discernible pencil particles.
- Simple long strokes are seldom used, which are always successively approximated with multiple overlapping strokes.
- Overshooting of strokes is apparent, especially in highly curve areas, because of fast movement of hands when drawing.
- Strokes will not follow the geometry strictly, the overall ”sketching” is quite beautiful.

Then, the participants are required to choose line drawings which convey 3D shapes more nicely based on two important considerations: one is whether the image illustrates the shape correctly; the other is whether the image is visually pleasing. Two sets of images are passed to the participants, line drawing without style and the NPR sketching. The shaded images are also given for their reference. Four examples shown in the previous section are chosen for evaluation: the simple stroke of Figure 5.10(*IMG1*), the knot of Figure 5.11(*IMG2*), the hand of Figure 5.12(*IMG3*), and the gargoyle of Figure 5.13(*IMG4*). The result of the voting is given in Table 5.1, showing that more participants consider NPR sketching conveys 3D shapes better than line drawing without styles. We discuss with the participants after the voting, and find that common users

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

focus more on stroke texture while the professional users focus more on the successively approximated strokes.

Table 5.1: Voting for NPR Sketching

	IMG1	IMG2	IMG3	IMG4
Professional Users	4(57.14%)	5(71.43%)	4(57.14%)	3(42.86%)
Common Users	10(76.92%)	10(76.92%)	9(69.23%)	9(69.23%)
Total	14(70.00%)	15(75.00%)	13(65.00%)	12(60.00%)

To further compare the computer generated NPR sketching with hand-drawn sketching, the participants are also required to score the provided NPR sketching images. The scores range from 0.0 to 1.0, with a higher score representing a NPR sketching being more similar to a hand-drawn one. The average scores of the two groups of users are given in Table 5.2. The result demonstrates that the common users are more satisfied with the computer generated NPR sketching comparing to the professional users.

Table 5.2: Scores for NPR Sketching

	IMG1	IMG2	IMG3	IMG4
Professional Users	0.538	0.602	0.564	0.541
Common Users	0.726	0.803	0.762	0.748
Average	0.660	0.733	0.693	0.676

The qualitative evaluation is still quite limited in explaining the perceptual consistency of NPR sketching. The results show that users perceive differences among line drawings without styles, computer generated NPR sketching, and hand-drawn sketching. But it also shows that computer generated NPR sketching do have many properties of hand-drawn sketching, and is more visually consistent and pleasing in conveying 3D shapes in comparison with line drawing without styles. Furthermore, the results of the evaluation give intuitive cues in improving the computer generated NPR sketching, such as developing better pencil textures and applying more variational strokes.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

5.6.3 Discussion

One may question whether a global approach striving for optimality (normalized cuts have an indirect connection to optimality) is really needed. Skilled artists are certainly not using only local information in drawing strokes: they can observe and remember the whole scene, and extensive prior practice provides knowledge of how the whole should be decomposed into pieces. In fact many practiced forms of human movement appear to optimize physical quantities (see [4] and references therein) and require training to do so (e.g. learning to walk, or to ice skate).

The method we have outlined has several limitations. Performance is a possible concern. In our implementation, solving the eigenvector takes from a few seconds to tens of seconds for the drawings shown here (several thousand feature line samples). There is opportunity for acceleration, however, by using the Nystrom approximation, or by simply subsampling the feature lines. A more serious concern is that the number of clusters and affinity distance require tuning. In our experience these parameters have some range of values over which the results are reasonable and vary intuitively with the parameters, however, outside this range the results (with few clusters in particular) sometimes surprised us. Some of the variety of recently proposed spectral and graph-based clustering methods (for example [85]) may address this problem in the future.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

Chapter 6

Stylized Line Drawing based on DBSC Strokes

Scenes and objects are always illustrated with stylized line drawings in a wide range of applications, such as sketching, medical illustration, technical illustrations, and cartoon animations. In most of these applications, the goal of shapes illustration is to achieve more visually pleasing experience.

Feature lines without style can convey shapes precisely and effectively. Stylized strokes based on feature lines make a line drawing more vivid and attractive. After feature lines are extracted and segmented, strokes are constructed with carefully designed styles to depict shapes. In this chapter, we present an effective and flexible framework for conveying shapes with stylized line drawings based on a novel stroke representation model, *disk B-spline curve*(DBSC).

We define stylized strokes with stroke geometry and stroke texture. Stroke geometry depicts the shape of a single stroke, while stroke texture describes various non-geometry properties along the stroke. Strokes based on DBSC, what we call *DBSC strokes*, represent strokes with unique parametric equations. The DBSC stroke model is both a powerful model in defining stroke geometry and a precise and flexible model in representing stroke texture. We demonstrate the flexibility of our DBSC based stylized line drawing framework with pencil drawings generated with physical simulation methods,

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

and procedural stylized rendering in a wide range of traditional styles. DBSC strokes can be applied widely and consistently to both interactive drawing systems and other existing NPR systems which control styles of strokes automatically through a set of parameters.

6.1 Introduction

Stylized line drawing with strokes in rich styles is a key feature of art, comics, and technical illustration. Feature lines without style can convey shapes precisely, while stylized strokes based on feature lines add life and clarity to drawings [76]. Stylized strokes can also emphasize specific parts of an image for the viewer's attention. Generally, stylized strokes can be defined with two major components, stroke geometry and stroke texture. Stroke geometry is normally modeled with a one dimensional backbone and the thickness along its length. Both the backbone and the thickness of the stroke are usually application dependant. In an interactive drawing system, the backbone can be the trace of pen, and the thickness can show cues of the pressure during drawing. In an automatic stylized illustration system for 3D shapes, the backbone can be the feature line extracted from the 3D model, and the thickness can show such as the depth cues of the 3D shape. The stroke texture is mostly defined with a set of visual attributes(e.g. color, transparency) distributed along the stroke. Stroke texture always shows various physical artistic medium types, such as oil paint, ink and graphite pencil. For example, the stroke texture of graphite pencil is actually the graphite particles left behind on the paper by the pencil when drawing. This metaphor of describing stylized strokes with geometry and texture offers great flexibility in emulating artistic drawing with different media in various styles.

With the above mentioned metaphor for stylized strokes representation, most existing techniques define stroke geometry as a curve with sampled thickness, or a path with stroke

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

outline, or even just pixels. Then stroke texture is defined with properties of discrete sample points of the interior of the stroke. These relatively simple stroke geometry models work well in some cases, but they are not effective and precise in representing the interior of the stroke in high resolutions. Comparing with these methods, DBSC stroke model is more powerful in precisely representing all the points of the stroke in any resolution, which is always required in physical style simulation. DBSC is actually a general 2D region representation model which models the whole stroke with a unique parametric equation. As a continuous model, DBSC is resolution independent and smooth, which is high order differentiable everywhere. With its ability of effectively and precisely representing every point of the stroke in any resolution, DBSC strokes can be incorporated into various stroke texture generation methods including physical simulation methods. Advanced features of DBSC as a stroke geometry model with technical details are shown in Section 6.2. The flexibility of our DBSC based stylized line drawing framework is demonstrated with pencil style simulation in Section 6.3, and various other styles with procedural artistic textures in Section 6.4.

Some NPR techniques work interactively such as the WYSIWYG NPR system [57] which presents an approach for interactively "painting" strokes directly on the 3D model. Based on DBSC strokes, we have also developed our interactive painting system [135]. In this work, we focus more on stylized line drawing which constructs DBSC strokes and control the stroke style automatically through a set of parameters.

6.2 DBSC Strokes

The representation of brush strokes is fundamental for computer generated calligraphic lettering, painting and animation. Many efforts have been devoted to this area. Whitted [128] uses a raster image to represent a stroke, which is not efficient in storage and is not flexible for manipulation. Steve Strassmann [112], Pudet [93] and Yap Siong

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

Chua [12] model brush strokes by representing their outlines with linear, parametric curves like Bézier or B-spline. Siu Chi Hsu, etc. [48] propose an approach of representing brush strokes with their centerline and thickness. Similarly, Binh Pham [90] uses uniform B-spline to define the centerline and outline of strokes. But, it is not easy to compute the offsets of interior points of strokes for both methods. Thus, Sara L. Su, etc. [115] define a 2D region of a brush stroke with interval B-Spline. And, in this section, we also propose a novel representation, disk B-spline curve(DBSC) to directly define a brush stroke with a mathematical equation. In comparison with interval spline, DBSC has better properties in continuity and perform much better in representing complex shapes. Some advantages of using DBSC to represent brush strokes are:

- Solid mathematical fundamentals.
- Precise evaluation.
- Able to represent not only the boundary of a 2D region, but also every point in the region, on which various attributes can be defined.
- Flexible manipulation like deformation, morphing, etc.
- Smaller dataset to represent brush strokes in freeform shapes.

6.2.1 Disk B-Spline Curve(DBSC)

6.2.1.1 Definition

- **Disk Geometry**

  A *disk* is defined as a set

$$< C; r > \equiv \{x \in \mathbb{R}^2 | ||x - C|| \leq r, C \in \mathbb{R}^2, r \in \mathbb{R}^+\}$$

  where C is the center of the disk and r is the radius. For a disk and an arbitrary real number a, the following operations are defined:

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

(i)

$$a < C; r > = < aC, |a|r >$$

(ii)

$$< C_1; r_1 > + < C_2; r_2 > = < C_1 + C_2; r_1 + r_2 >$$

Then any finite summation can be defined as:

$$\sum_{i=1}^{n} a_i < C_i; r_i > = \sum_{i=1}^{n} < a_i C_i, |a_i| r_i >$$

- **Disk B-Spline Curve**

  Let $N_{i,p}(t)$ be the $i-th$ B-spline basis function of degree p with knot vector

$$[u_0, \ldots, u_m] = \{\underbrace{a, \ldots, a}_{\text{p+1}}, u_{p+1}, \ldots, u_{m-p-1}, \underbrace{b, \ldots, b}_{\text{p+1}}\}$$

A DBSC is defined as:

$$< D > (t) = \sum_{i=1}^{n} N_{i,p}(t) < P_i; r_i > \qquad \text{(Eq. 6.1)}$$

where P_i are control points and r_i are control radii.

Since

$$\begin{aligned} < D > (t) &= \sum_{i=1}^{n} N_{i,p}(t) < P_i; r_i > \\ &= \sum_{i=1}^{n} < N_{i,p}(t) P_i; N_{i,p}(t) r_i > \\ &= < \sum_{i=1}^{n} N_{i,p}(t) P_i; \sum_{i=1}^{n} N_{i,p}(t) r_i > \end{aligned}$$

A DBSC can be viewed as two parts: the center curve

$$c(t) = \sum_{i=1}^{n} N_{i,p}(t) P_i$$

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

which is a B-spline curve, and the radius function

$$r(t) = \sum_{i=1}^{n} N_{i,p}(t) r_i$$

which is a B-spline scalar function. Owing to the perfect symmetry property of disks, the curve $c(t)$ constructed from the centers of disks is exactly the centerline of the 2D region represented by the DBSC. Thus, most properties and related algorithms of DBSC can be obtained by working with the B-spline curve and the B-spline scalar function of the two parts of DBSC respectively.

6.2.1.2 Properties of DBSC

- Differentiability

  A DBSC of degree p is at least C^{p-k} order differentiable at a knot of multiplicity k and infinitely differentiable in the interior of knot intervals.

- Local Modification Scheme

  Moving P_i or modifying r_i will affect $< D > (t)$ only in the interval of $\lfloor u_i, u_{i+p+1} \rfloor$.

- Convex Hull Property

  The region represented by a DBSC is contained in the convex hull of the complete set of the circles with radius r_i centered at P_i.

- Affine Invariance

  According to the affine invariance property of B-spline curve and the affine property of disk, a DBSC has affine invariance property.

- Topological Property

  From the topological point of view, a 2D region represented by a DBSC is homotopic to its center curve. Thus an open DBSC without self-intersection represents a 2D region of genus zero, and a close DBSC without self-intersection represents a 2D region of genus one.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

6.2.1.3 Evaluation of DBSC

- Evaluation and Computation of Derivatives

  A point defined by the center curve and its radius at any parameter t can be computed by evaluating the B-spline curve and the B-spline scalar function. Moreover, its derivative can also be obtained by computing the derivatives of the B-spline curve and the B-spline scalar function.

- Degree Elevation

  In shapes modelling, degree elevation is often needed. With a DBSC, this can be implemented by elevating its center curve and radius functions respectively.

- Boundary Evaluation

  To display the boundary of the region represented by a DBSC, we can regard the boundary as the envelope of a one-parameter family of circles. For any parameter t, with the center at

  $$x(t) = \sum_{i=1}^{n} N_{i,p}(t) x_i$$

  and

  $$y(t) = \sum_{i=1}^{n} N_{i,p}(t) y_i$$

  the radius

  $$r(t) = \sum_{i=1}^{n} N_{i,p}(t) r_i$$

  then

  $$(x - x(t))^2 + (y - y(t))^2 = r^2(t)$$

  According to envelope theorem

  $$\begin{cases} F(x,y,t) = 0 \\ \frac{\partial F(x,y,t)}{\partial t} = 0 \end{cases}$$

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

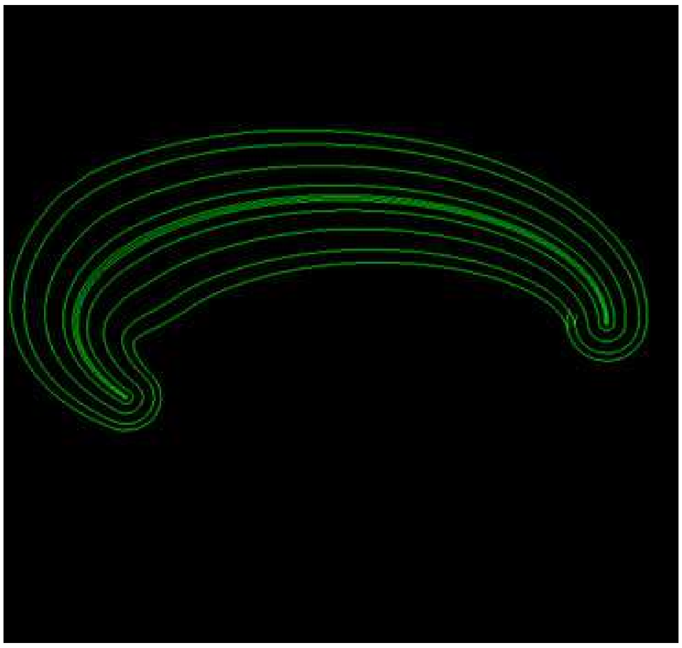


Figure 6.1: Variable offsets of DBSC which are computed by scaling the control radii of the DBSC.

the second equation is obtained:

$$-(x - x(t))x'(t) - (y - y(t))y'(t) - r(t)r'(t) = 0$$

For any t, we can compute $x(t)$, $y(t)$, $r(t)$ and their derivatives $x'(t)$, $y'(t)$, $r'(t)$. Now let $X = x - x(t)$, $Y = y - y(t)$, $C = r(t)$, $D = x'(t)$, $E = y'(t)$, $F = r'(t)$, we get the following equation group:

$$\begin{cases} X^2 + Y^2 = C^2 \\ DX + EY = -CF \end{cases}$$

This is an equation group of a circle and a line intersection. By solving the equation group, we get two solutions:

$$\begin{cases} X = \frac{-CDF \pm CE\sqrt{D^2+E^2-F^2}}{D^2+E^2} \\ Y = \frac{-CDF \mp CD\sqrt{D^2+E^2-F^2}}{D^2+E^2} \end{cases} \quad \text{(Eq. 6.2)}$$

There are two kinds of points on the boundary. When a DBSC is open and the radius at its end is not zero, the end part is an arc whose center is the end point of

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

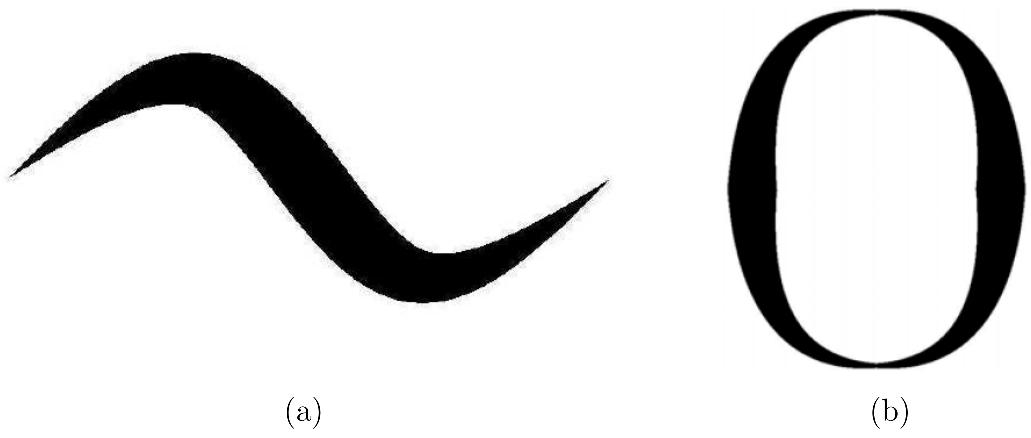


Figure 6.2: Brush strokes representation with DBSC. (a) is a brush stroke represented with an open DBSC. (b) is a brush stroke represented with a closed DBSC.

the center curve. The start and end points on the arc can be computed from the above computing envelope with Eq. 6.2. Therefore, any point on the boundary of the disk B-spline region can be obtained. With the above derivation, variable offsets can be computed by scaling control radii as shown in Figure. 6.1. This is extremely important in achieving good triangulations of DBSC for visualization, and flexible stylization of brush strokes based on DBSC.

6.2.1.4 Representation of brush strokes with DBSC

DBSC is a general model for 2D region representation. A brush stroke can be considered a special 2D region. Our experiments demonstrate that DBSC is a flexible and effective tool for artistic strokes representation. Two types of brush strokes are shown in Figure 6.2, which are represented with either an open DBSC or a closed DBSC. More examples including DBSC stroke based paintings are given in Section 6.2.4.

6.2.2 Generation of DBSC Strokes

By modelling brush strokes with the DBSC stroke model, a new DBSC stroke can be generated with various modelling methods starting from a point set or an existing DBSC

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

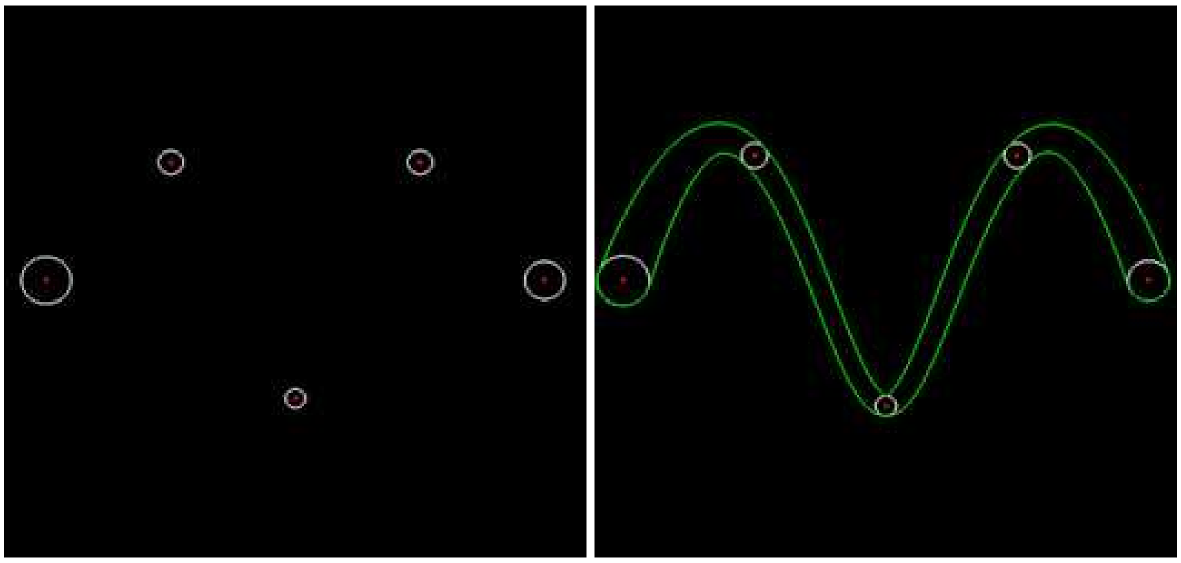

(a)Control Disks (b)Region represented with DBSC

Figure 6.3: DBSC generated from control disks with interpolation.

stroke. Since DBSC can be considered the center B-spline curve plus the radius B-spline scalar function, in this section, we introduce the DBSC strokes generation with interpolation, approximation and deformation methods.

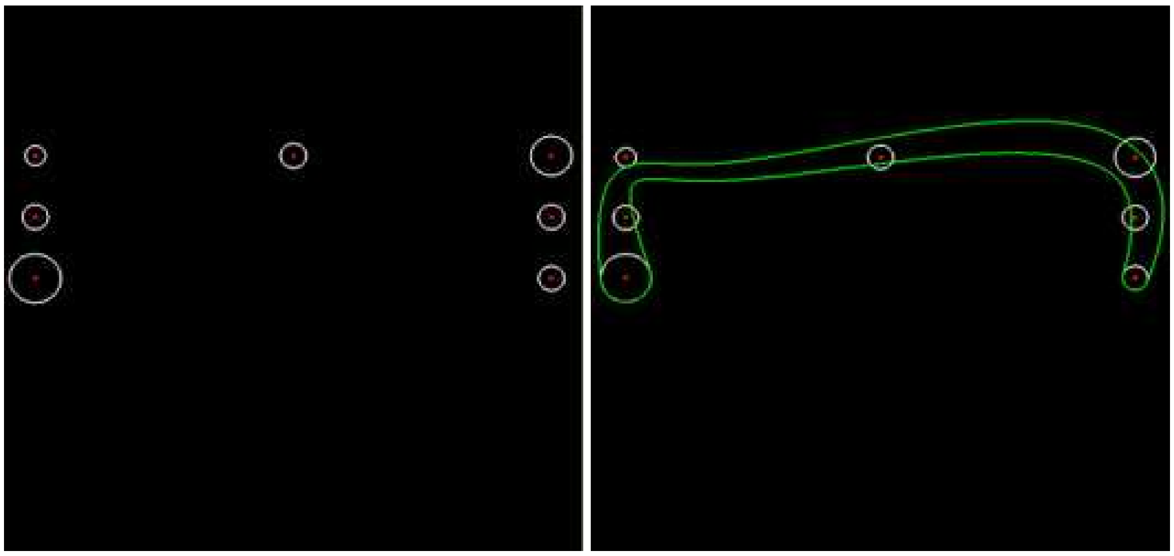
- **Interpolation**

  Given a group of points $\{Q_i\}$, $i = 0, \cdots, m$ on a center curve and their corresponding maximum distances $\{c_i\}$, a DBSC, either open or close, whose center curve passes these data points $\{Q_i\}$ and whose maximum radius is $\{c_i\}$, can be obtained by B-spline curve interpolation and B-spline scalar function interpolation, as illustrated in Figure 6.3.

- **Approximation**

  When a group of points $\{Q_i\}$, $i = 0, \cdots, m$ on central curve and their corresponding maximum distances $\{c_i\}$ are given, a Disk B-Spline curve whose center curve approximates these data points $\{Q_i\}$ and whose maximum radius is $\{c_i\}$, can be obtained by approximation. Here we can use B-Spline curve approximation method

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library



(a)Control Disks (b)Region represented with DBSC

Figure 6.4: DBSC generated from control disks with approximation.

and B-Spline scalar function approximation method to obtain DBSC approximation. Figure 6.4 is an example to illustrate this method.

- **Deformation**

  In geometric modelling, deformation is one of the most important modelling methods. Here we implement the deformation of a DBSC by deforming its center curve and scaling its control radii. Figure 6.5 gives an example to illustrate the flexibility of deformation where the leftmost image is the original textured shape/stroke represented with DBSC which is deformed into a sequence of DBSC strokes on the right.

6.2.3 Visualization of DBSC Strokes

By applying the method of evaluating the center B-spline curve and the radius B-spline scalar function of DBSC in two orthogonal local coordinate directions, every point in the region represented by DBSC can be computed. Therefore, we can triangulate, stylize

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

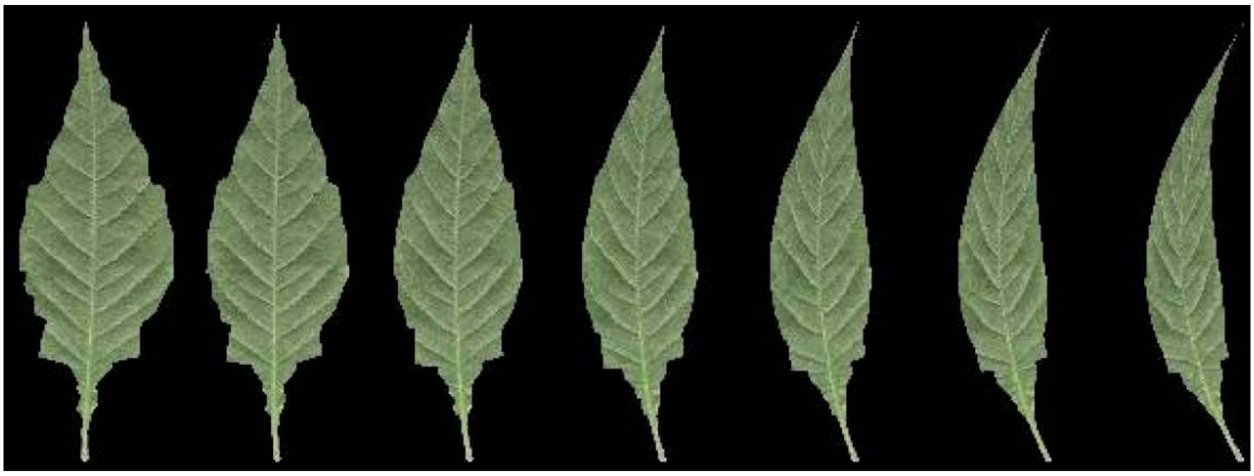
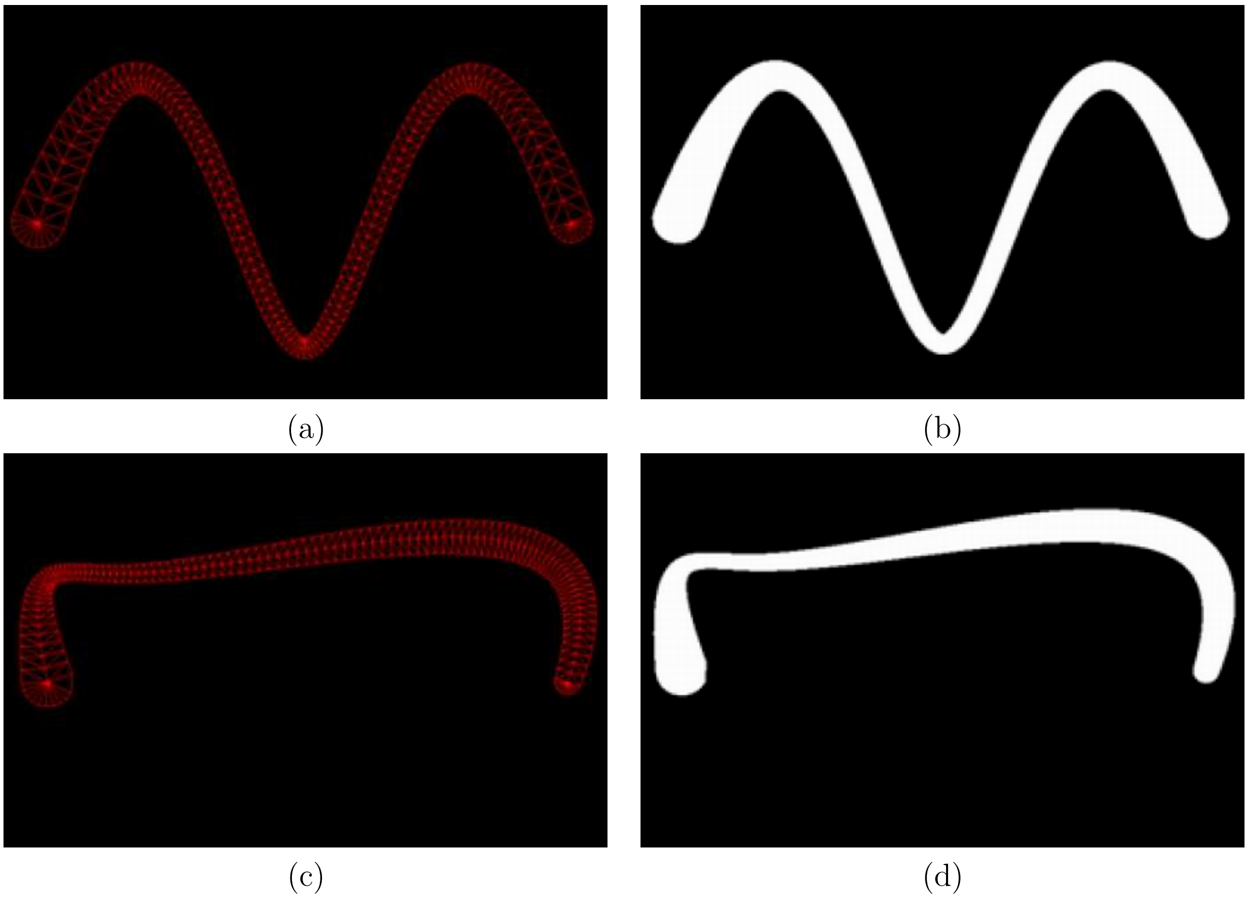


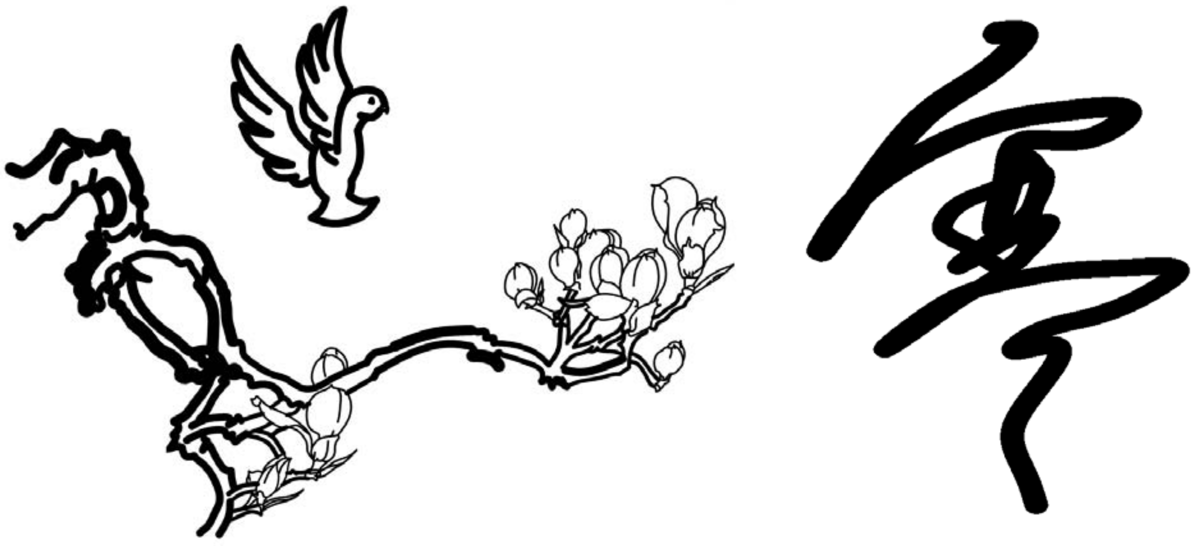
Figure 6.5: Deformation of DBSC.



(a) (b) (c) (d)

Figure 6.6: Triangulation and visualization of DBSC strokes.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library



(a)Chinese painting (b)Chinese calligraphy

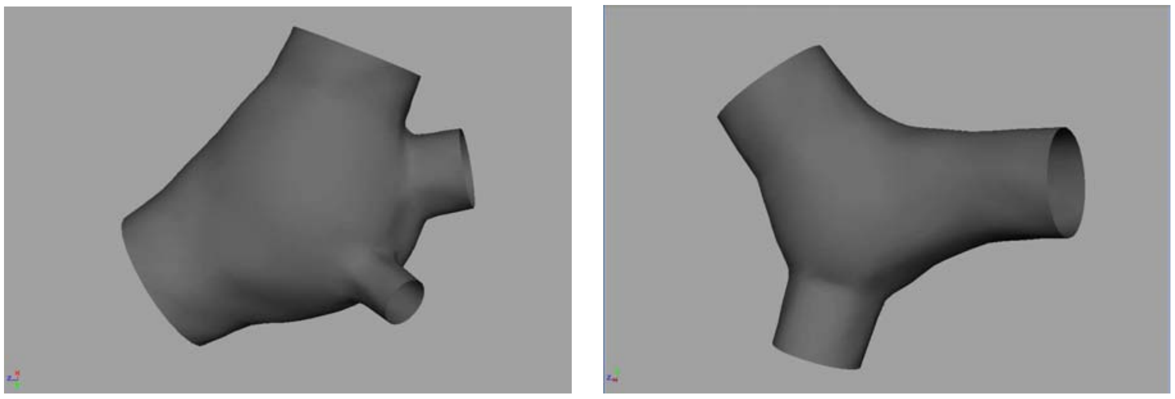
Figure 6.7: DBSC strokes for Chinese painting and calligraphy.

and render DBSC strokes at any resolution. An example of regular triangulation and visualization of DBSC strokes is shown in Figure 6.6. More examples and various styles of DBSC strokes rendering are given in the following sections.

6.2.4 Results and Discussion

In this section, we describe our novel approach of representing brush strokes with DBSC, together with details of construction, evaluation and visualization of DBSC strokes. DBSC is a powerful and flexible tool for brush stroke representation. Using a group of DBSC strokes, complex paintings and Chinese calligraphy can be represented, as demonstrated in Figure 6.7. Since every point of a DBSC stroke can be evaluated precisely and effectively, attributes and patterns such as scalar field and vector field can also be defined on the DBSC stroke. This provides a flexible framework for various styles of strokes rendering, including the physical simulation of certain types of artistic medium. Further more, by applying operations such as transformation, deformation and morphing etc. to the center curves and the control radii of DBSC strokes, animation sequences can

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library



(a) (b)

Figure 6.8: Modeling bifurcation structures of vascular with BBSC.

be created conveniently. This can also be combined with physical dynamic techniques to achieve more realistic animations. Various styles of strokes including graphite, crayon and hairy brush, and the above mentioned methods for DBSC based cartoon animation have been applied to develop our interactive painting and animation system [135].

Besides being a general 2D region representation model with great flexibility in modelling brush strokes, DBSC can also be generalized to 3D space. By extending 2D disks to 3D balls, and center B-spline curves to skeletons in 3D space, we get Ball B-spline Curve(BBSC) for 3D model representation. We have applied BBSC in modelling the vascular network of human body. Figure 6.8 demonstrates that complex multiple-branch structures of the vascular network can be represented with smooth BBSC models.

6.3 DBSC Based Pencil Style Simulation

Physical style simulation methods can achieve high quality stroke texture. Significant approach on the simulation of pencil drawing which is based on the analyzing of the real world process is proposed by M. C. Sousa and J. W. Buchanan [7, 109, 108]. As a powerful stroke geometry model, DBSC stroke can compute every point of the stroke precisely and effectively. In this section, we demonstrate the flexibility of the DBSC

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

stroke by incorporating it with the pencil simulation algorithm for DBSC based pencil style simulation.

6.3.1 Simulation of Pencil Materials

Taking into account the microscopic level of materials, and interaction between the wood-encased graphite pencil and the drawing paper, M. C. Sousa and J. W. Buchanan [7, 109, 108] try to capture the essential physical properties and behaviors of pencil drawing. The basis of their method is the low-level simulation of the pencil drawing materials and the interaction among these materials, including the simulation models for the graphite pencil, the drawing paper, the blender and the kneaded eraser. The main aspects of their method are:

Graphite Pencils: The pencil model has two main aspects, the degree of pencil hardness and the kinds of sharpened points. Pencil hardness is graded in nineteen degrees ranging from 9H (hardest) to 8B (softest). A pencil point is defined by tip shape and pressure distribution coefficients over the point's surface. Sharpening a pencil in different ways changes the shape of the contact surface between the pencil and the paper. Sharply pointed, it gives a line as fine and clean-cut as that of the pen; bluntly pointed, it can be used much like the brush. Pressure distribution coefficients are values between 0 and 1 representing the percentage of the pencil's tip surface that, on average, makes contact with the paper.

Drawing Papers: Papers are made in a great variety of weights and textures. Paper textures for pencil work (categorized as smooth, semi-rough, and rough) have a slight roughness (”tooth” or grain) that enables lead material (graphite, clay, and wax particles) to adhere to the paper. The paper texture is modelled as a height field which can be either procedurally generated or digitized from a paper sample.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

Each paper location (x, y) accumulates lead material. The amount of material depends on the pencils that have crossed the location.

Pencil and Paper Interaction: Lead material is left on paper through friction between the lead and the paper. The amount of lead material depends on the pencil tip shape, the pressure applied to the pencil, and the pencil hardness. In addition to depositing lead, a pencil stroke may alter the texture of the paper by destroying its grains.

Blender and Eraser: The blender and eraser model enhances the rendering results producing realistic looking graphite pencil tones and textures. This model is based on observations on the absorptive and dispersal properties of blenders and erasers interacting with lead material deposited over drawing paper.

To apply the above pencil simulation model to synthesize the pencil texture of the interior of the DBSC strokes, we focus more on the pencil model, the paper model and the pencil paper interaction model.

6.3.2 Pencil Rendering of 3D Models

Based on the models for pencil materials simulation, M. C. Sousa and J. W. Buchanan [108] try to develop a general framework for pencil style rendering of 3D polygonal models. The framework can be broken down into four fundamental parts: (1) simulating the drawing materials, (2) modeling the drawing primitives, (3) simulating the basic rendering techniques used by artists and illustrators, and (4) modeling the control of the drawing composition. Their method draws 3D objects in outlines with pencil stroke primitives.

In [108], a pencil stroke S consists of a number of line segments, a path, and a character function. The path $P(t) : [0..1] \rightarrow R^2$ results from using a curve to approximate the line

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

segments. Bézier curves and B-Splines are used to approximate these line segments. The character function is defined with:

$$C(t) = (C_w(t), C_p(t), C_{ps}(t), C_{pdc}(t), C_{fd}(t), C_\alpha(t), C_\beta(t)) \qquad \text{(Eq. 6.3)}$$

The seven character parameters are waviness $C_w(t)$, pressure $C_p(t)$, point shape $C_{ps}(t)$, pressure distribution coefficients $C_{pdc}(t)$, finger distance $C_{fd}(t)$, pencil slanting $C_\alpha(t)$, and wrist/arm movements $C_\beta(t)$. Thus pencil style 3D outlining can be achieved by applying pencil stroke primitives onto the object outlines.

6.3.3 DBSC Based Pencil Drawing

DBSC stroke model defines smooth and precise stroke geometry while pencil style simulation algorithm achieves high quality pencil stroke texture. DBSC stroke model is flexible which can be integrated effectively with the pencil style simulation algorithm for better pencil strokes. In this section, we demonstrate the effectiveness of this combination.

In systems of [109, 108], high quality pencil textures are demonstrated based on the above mentioned observational models. However, in these systems, only an intrinsic stroke geometry model which is more suitable for interactive drawing systems is used. DBSC stroke is more powerful in modelling stroke geometry. For DBSC based pencil drawing, we represent the stroke geometry with DBSC strokes, and fill the stroke interior with pencil texture generated with the observational models. The complemental application of both techniques can achieve improved shapes illustrations in pencil style. Note that the observational models requires a lot of parameters, which might be confusing for an end user to design a preferable pencil rendering. Based on DBSC, we intend to simplify the simulation model and minimize the number of required parameters.

In [109, 108], the drawing paper is modeled with a height field. The pencil stroke primitive is represented with a path $P(t)$ and a character function $C(t)$ where t is the

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

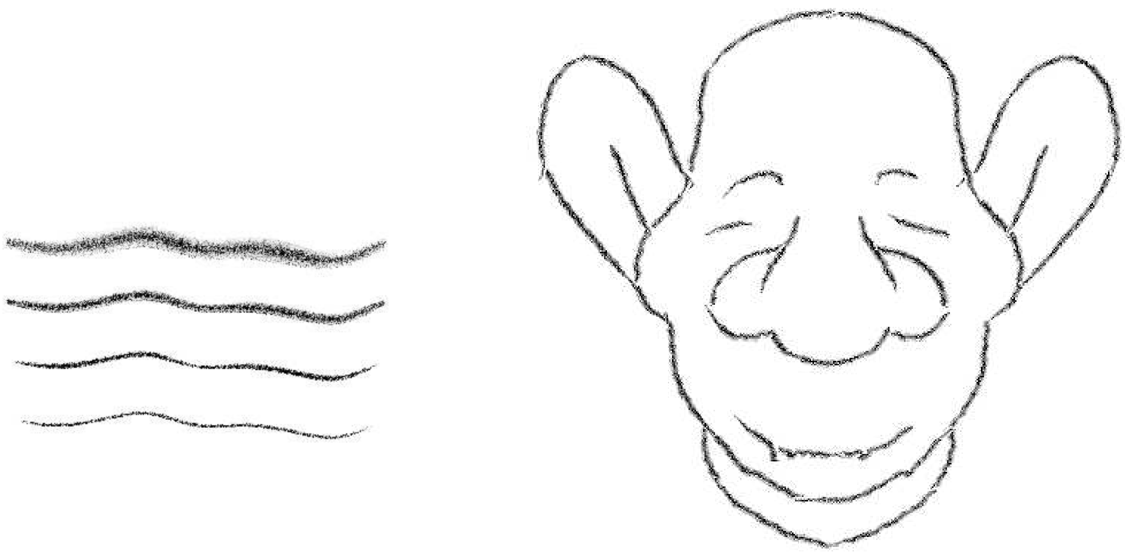

(a) (b)

Figure 6.9: DBSC based pencil simulation. (a) is an example of strokes with different width. (b) is rendered with 2B pencil to draw the 3D object in outlines.

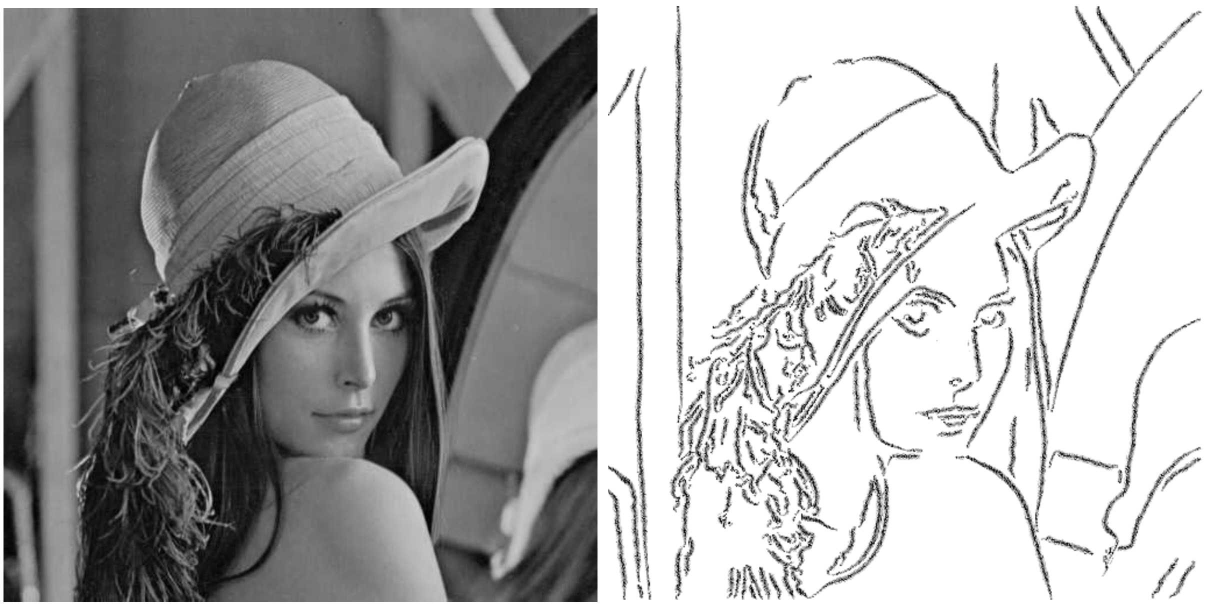

(a) (b)

Figure 6.10: DBSC based pencil drawing for images. (b) is rendered with 2B pencil with feature lines extracted from the source image (a).

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

parameterization of the stroke along the path, as defined in Eq. 6.3. By defining pencil stroke with DBSC stroke model, the path $P(t)$ is now the B-spline central curve of DBSC. The boundaries and the interior points of the DBSC stroke can also be computed. Then we redefine the character function $C(t)$ to generate the pencil texture, which is actually the intensity distribution of pixels within the stroke.

The character function Eq. 6.3 has seven parameters which are difficult to control by the end user. Based on DBSC, we refine the character function with two more intuitive parameters: a more flexible pencil tip shape parameter, and a pressure coefficient parameter for each pixel of the stroke. With the central line and boundaries of the DBSC stroke, we reconstruct and represent the stroke with a list of sequential quadrangles. These quadrangles can be considered the pencil tip shapes varying along the path. For each quadrangle, we have chosen the pressure distribution coefficient to be 1 in the middle and 0.2 at the boundary to simulate the effect of a real pencil stroke that is darker in the center. The in-between interior points are computed analytically and then rendered based on the reflected intensity I_k computed from the pencil-paper interaction process [109]. DBSC based pencil simulation can be applied to both 3D shapes and 2D images. Examples are shown in Figure 6.9 and Figure 6.10. In Figure 6.10, pencil drawing is generated with feature lines extracted from the source image.

During the process of stroke rendering, a new parameter, pixel distribution coefficient d_{rnd}, is introduced. It modifies positions of the stroke points with an offset of d_{off} pixels. It is a random value between $-d_{rnd}$ and $+d_{rnd}$. For a stroke point with coordinates $(x,\ y)$, two random values, $d_{offx} \in [-d_{rnd},\ +d_{rnd}]$ and $d_{offy} \in [-d_{rnd},\ +d_{rnd}]$, are computed. Thus, the final stroke point coordinate is $(x + d_{offx},\ y + d_{offy})$. Effects of using different pencils and d_{rnd} values are shown in Figure 6.11. The introducing of non-zero d_{rnd} makes strokes less uniform and more "fluffy".

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

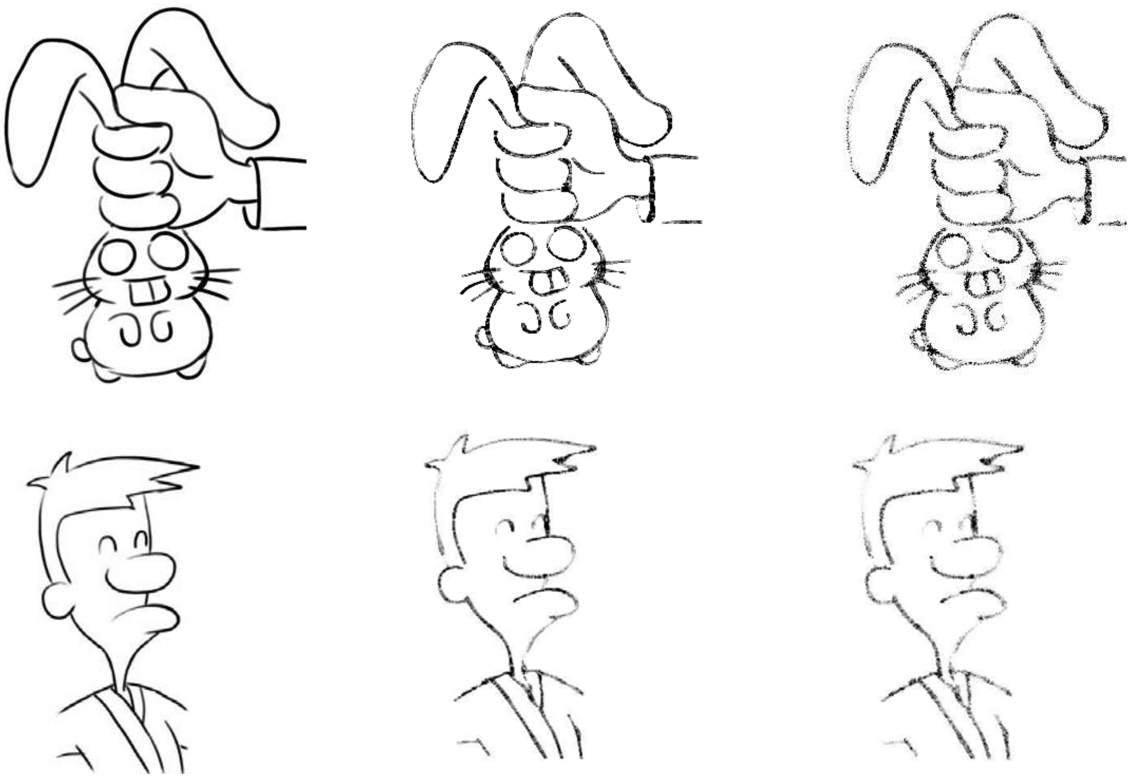

(a) Original (b) $d_{rnd} = 0$ (c) $d_{rnd} = 1$

Figure 6.11: DBSC based pencil drawing. The introducing of d_{rnd} makes strokes less uniform and more ”fluffy”.

6.4 Procedural Stylized Line Drawing

Besides its application in physical style simulation, as a general stroke model, DBSC works well in representing ink drawings with expressive strokes. It is also flexible to be incorporated in other stroke texture generation methods to synthesize various styles of artistic strokes. In this section, two categories of stylized line drawings are introduced: the expressive ink drawings of 3D shapes which focus more on stroke geometry, and the line drawings emulating traditional artistic media which focus more on stroke texture. We also discuss the issues of breaking strokes for stylized line drawing with big brush strokes, such as Chinese painting.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

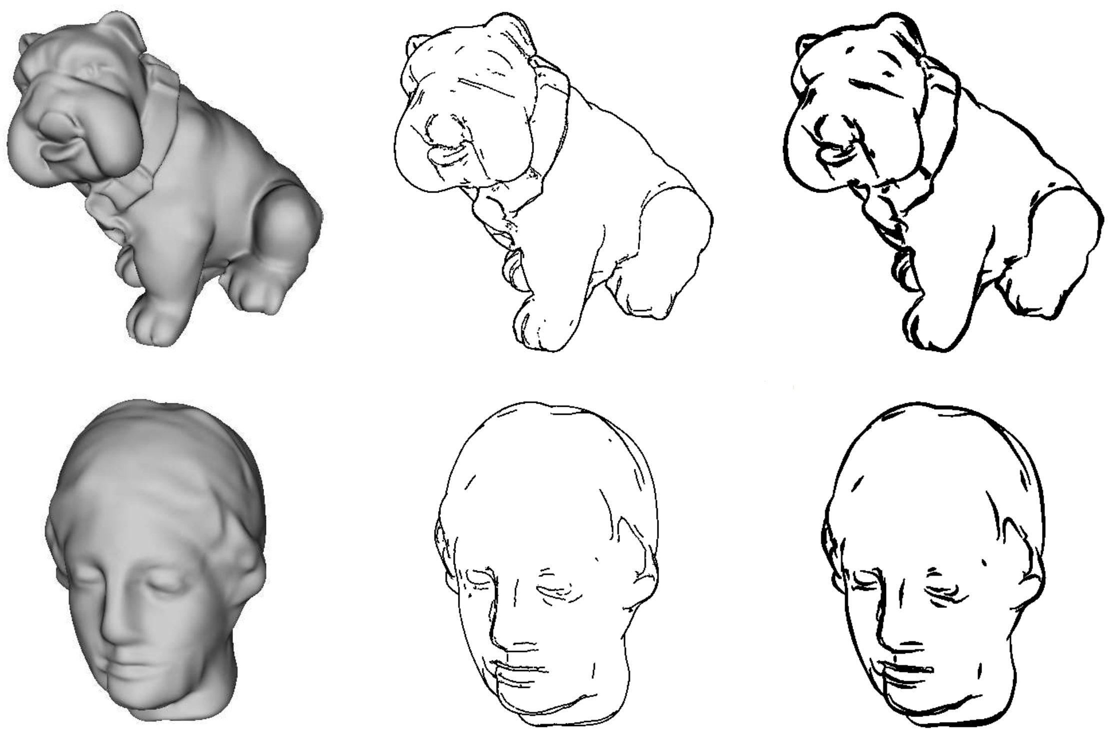


Figure 6.12: Ink drawings with expressive strokes. Comparing with plain line drawings in the middle, more vivid line drawings with DBSC strokes are achieved on the right.

6.4.1 Expressive Strokes

Ink drawing can convey very rich information with simple black strokes. It is expressive, appealing and widely used in traditional art and scientific illustration. Without stroke texture, the shape of a stroke is the key of its expressiveness. A plain line drawing can convey 3D shapes with their features precisely, but it can not capture the liveliness of hand drawn images and illustrations. Ink drawing with expressive strokes make a line drawing more vivid with their pleasing aesthetic qualities. Examples comparing plain line drawings with DBSC stroke based ink drawings are shown in Figure 6.12.

Though mostly viewers may not pay much attention to the quality of a single stroke, the entire feeling of the illustration will be different when it is expressed with different strokes. Figure 6.13 shows 3D shapes illustrated with DBSC strokes with different thick-

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

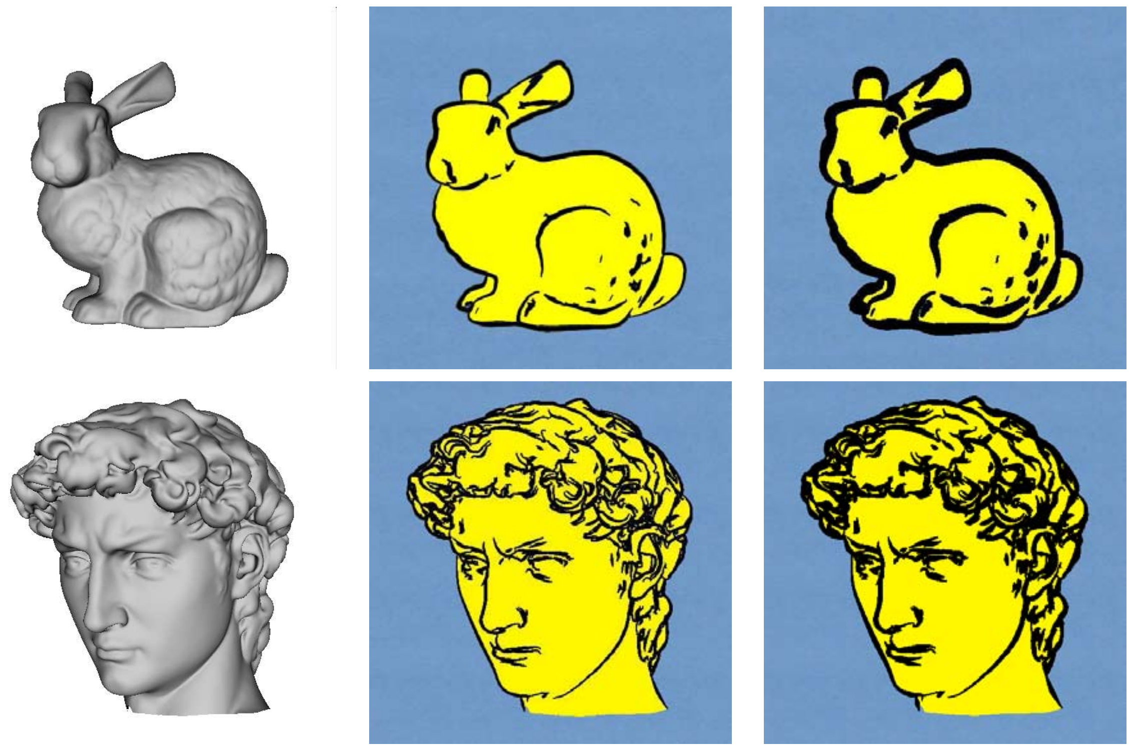


Figure 6.13: Cartoon style 3D shapes illustration with strokes with different thickness.

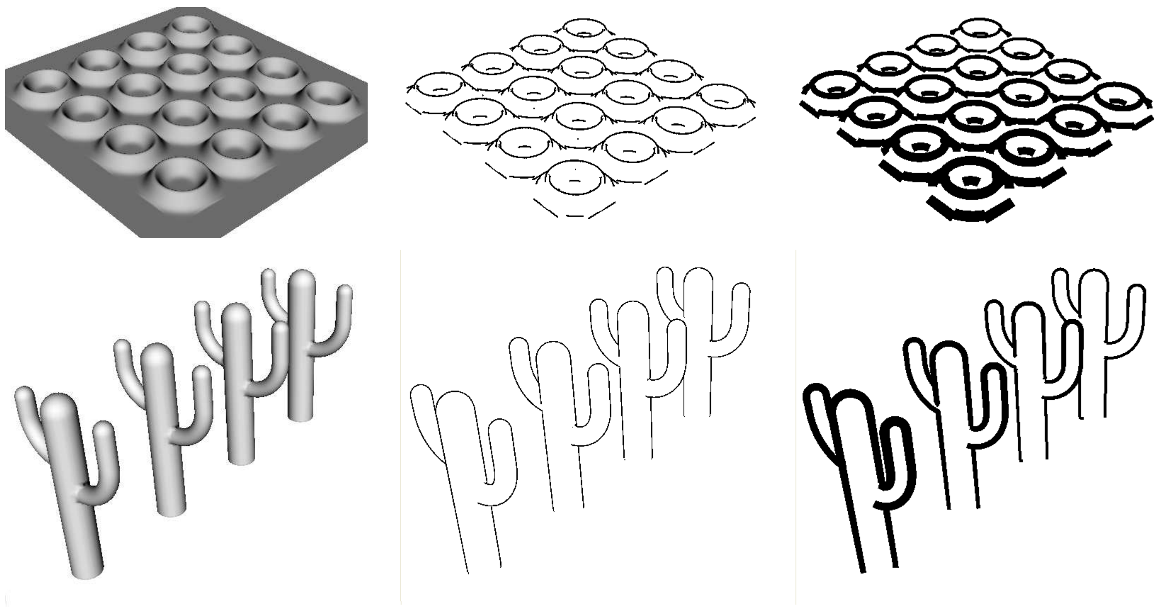


Figure 6.14: Expressive strokes for scene illustration. Comparing with the plain line drawings, illustrations with depth-based thickness of strokes are more comprehensive.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

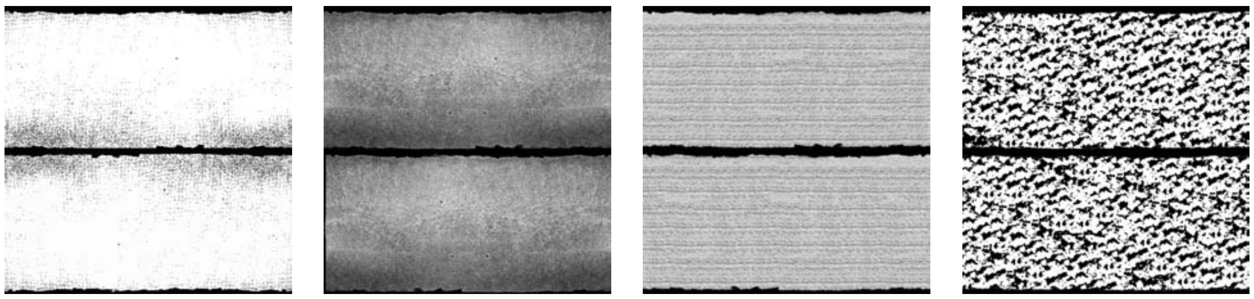

(a)dry brush (b)wet brush (c)thick brush (d)charcoal

Figure 6.15: Stroke textures.

ness. Additionally, the 3D shapes are drawn with color shading and paper background to achieve a cartoon style rendering. Ink strokes in combination with cartoon style shading are widely used in current movie industry. It also improves the expressiveness of the ink drawings greatly.

In scientific illustration, depth-based thickness of strokes is always used to convey overall depth relationships among different parts of a scene. By using thicker strokes to illustrate objects nearer to the viewer, while thinner strokes for objects further away, more comprehensive illustration of a 3D scene can be achieved. An example of illustration with thickness of strokes inversely proportional to object depth is given in Figure 6.14.

6.4.2 Artistic Media with Stroke Textures

In this section, we show line drawings emulating traditional artistic media based on DBSC strokes. Several categories of artistic media are emulated, including thick media such as oil painting; wet media such as water coloring; and dry media such as graphite pencil. Some stroke textures are shown in Figure 6.15, including textures for dry, wet and thick materials, together with the texture for charcoal emulation.

Based on DBSC, strokes are parameterized first, and then stroke textures of Figure 6.15 are used as alpha maps in our method to simulate various artistic media. We use alpha blending of OpenGL in our implementation, with the replace mode for thick

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

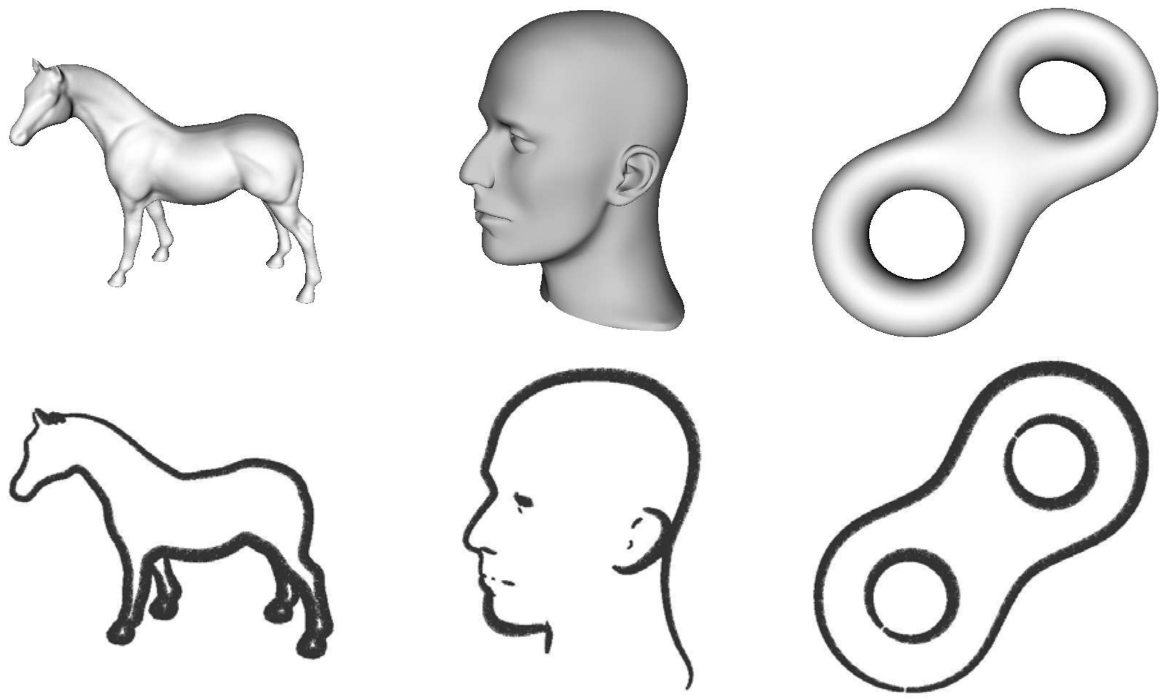


Figure 6.16: Brush painting with the dry brush texture.

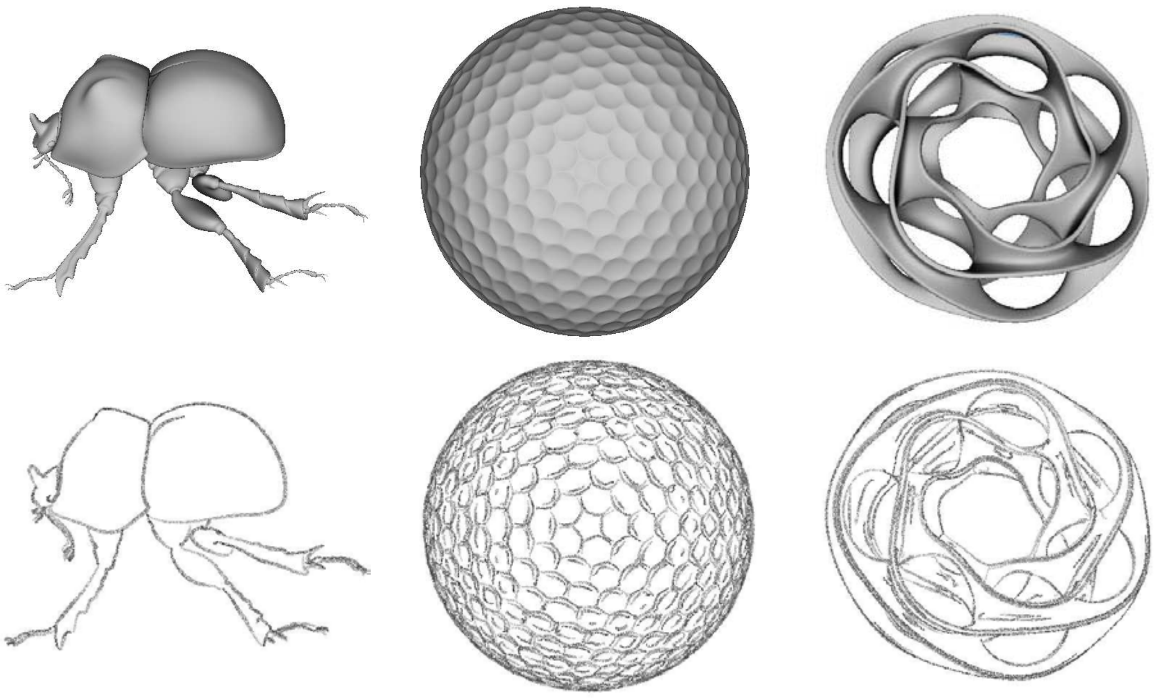
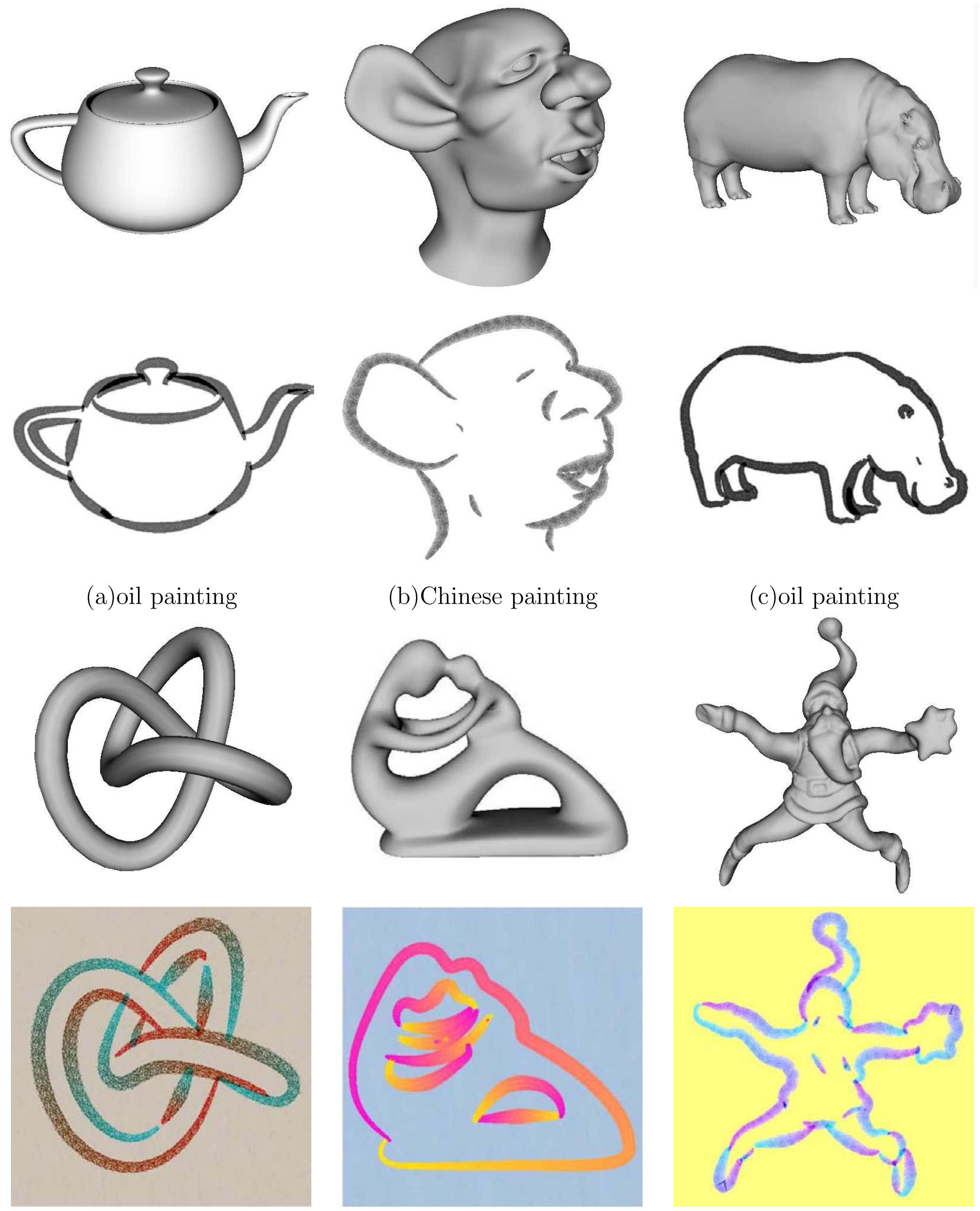


Figure 6.17: Charcoal drawing.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library


(a)oil painting (b)Chinese painting (c)oil painting

(d)color pencil (e)oil painting (f)water coloring

Figure 6.18: Various styles line drawings with DBSC strokes emulating traditional media.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

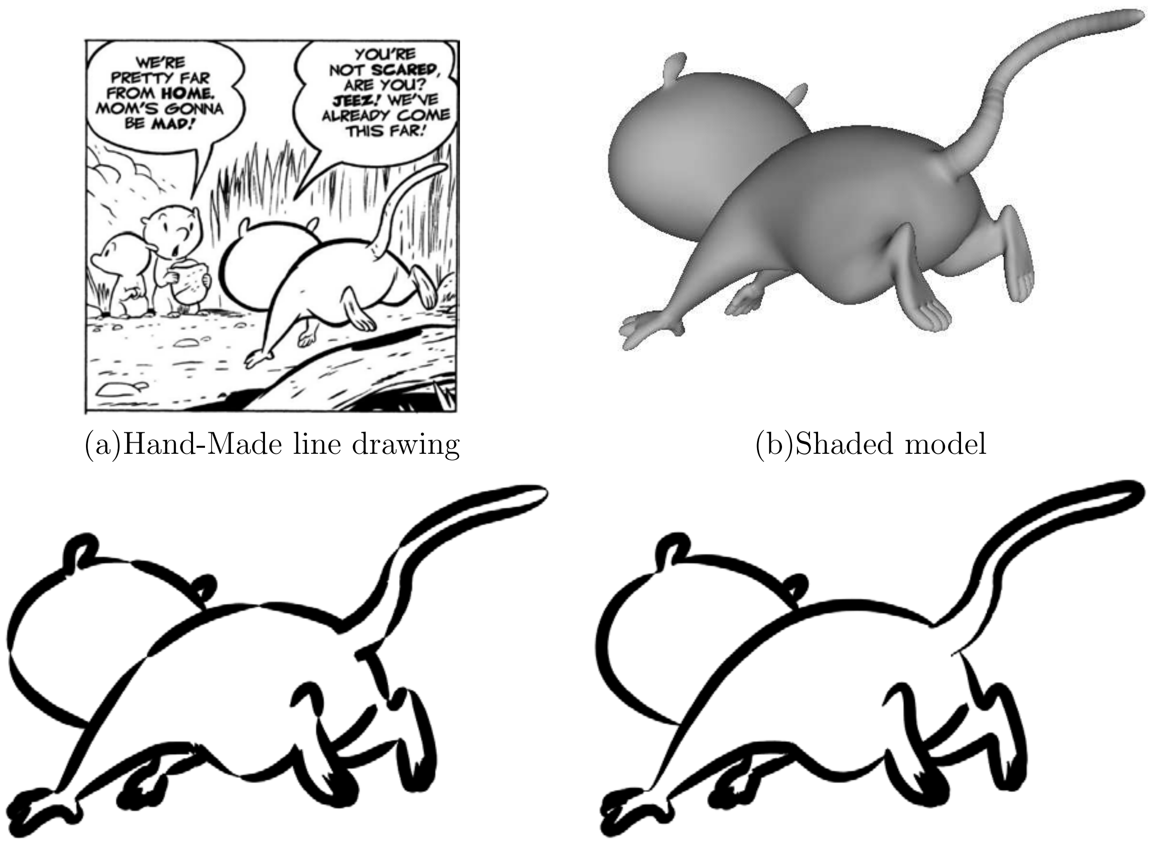
media, the additive blending mode for wet materials, and the minimum blending mode for dry material simulation.

Examples are shown in Figure 6.16, Figure 6.17 and Figure 6.18 respectively. Figure 6.16 shows examples of brush painting with the dry brush texture shown in Figure 6.15 (a). Figure 6.17 shows charcoal drawings with the charcoal texture shown in Figure 6.15 (d). Figure 6.18 shows examples with different types of stroke textures. (b) and (f) of Figure 6.18 illustrate strokes with wet brush, simulating the traditional Chinese painting and water coloring. (a) and (c) are rendered with thick brush texture simulating strokes of oil painting. Compared with (a), a drier brush is applied to render (c). (d) is generated with the charcoal texture to achieve the feeling of a color pencil drawing on a rough paper.

6.4.3 Stroke Segmentation for Stylized Line Drawing

As discussed in Chapter 5, a computer generated line drawing with simple long strokes, or strokes broken at curvature maximum does not capture the liveliness of hand-made sketching. The situation is even worse when the strokes are generated from feature lines with gaps and noise. This is also true for stylized line drawing with big brush strokes, such as Chinese painting. Our stroke segmentation algorithm based on normalized cuts works well for NPR sketching. It also improves the results of stylized line drawings with DBSC strokes. As demonstrated in Figure 6.19, (a) is a hand-made drawing with big brush strokes; (b) is the shaded 3D model created according to the hand-drawn image; (c) is the line drawing with DBSC strokes broken at curvature maximum, the strokes are generated from traced feature lines containing noise and gaps; (d) is the line drawing with segmented DBSC strokes. Compared with (c), strokes of (d) improve greatly after stroke segmentation. And the line drawing of (d) looks more similar to the hand-made line drawing of (a).

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library


WE'RE PRETTY FAR FROM HOME. MOM'S GONNA BE MAD!
YOU'RE NOT SCARED, ARE YOU? JEEZ! WE'VE ALREADY COME THIS FAR!

(a)Hand-Made line drawing

(b)Shaded model

(c)Line drawing without stroke segmentation

(d)Line drawing with stroke segmentation(15 strokes)

Figure 6.19: DBSC Stroke segmentation. Comparing with (c), the line drawing of (d) improves greatly after stroke segmentation. It looks more similar to the hand-made line drawing of (a). (a) is by courtesy of Jeff Smith. The model is by courtesy of Todd Goodwin [33].

Though stroke segmentation improves the quality of the line drawing, it still does not satisfy in capturing many other qualitative properties of hand-made line drawings, such as selective tapering of strokes, variation of the overall stroke thickness in different parts of the drawing etc. As shown in Figure 6.19, there are quite big differences between strokes of (a) and (d). Artists usually draw in many different ways, and may often be inconsistent. Thus, it is extremely challenging in simulating any existing artwork. Instead of breaking long strokes into short ones, Goodwin et. al. [33] compute the thickness of

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

strokes according to the lighting on the surface. In many cases, their results are quite similar to break strokes in bright areas. This method creates promising line drawings and gives very intuitive cues in developing novel algorithms to simulate artistic strokes in real artwork.

6.5 Conclusion

In this chapter, we present our flexible stylized line drawing framework based on DBSC strokes. We demonstrate the effectiveness of the framework and the expressiveness of the DBSC stroke model with physical based pencil simulation, and various types of traditional styles emulated with artistic stroke textures. The DBSC stroke model is effective and versatile which can be easily incorporated in a wide range of existing graphics systems, including interactive drawing systems and automatic rendering systems.

According to our experiments, we find that more investigation is needed on determining the thickness of DBSC strokes. In an interactive drawing system, this may not be a problem since the thickness of strokes can be easily decided with the pressure when the user draws. In our case of an automatic system which generates stylized line drawing from the 3D scene, there is uncertainty in how to decide the thickness of a stroke automatically. We experiment with constant thickness of strokes, procedural thickness of strokes with random variation, and thickness of strokes proportional to depth or curvature etc. All these methods can not fully capture the vivid variation of thickness of strokes in a hand made drawing or painting. Recently, Goodwin et. al. [33] proposed a method of determining stroke thickness based on the isophote distance. This method improves the quality of strokes, but it is still not completely satisfied. In the future, we will try to learn stroke thickness from real paintings with machine learning methods. In current stage, we are focusing on stylized line drawing of static scenes. In a dynamic setting, such as stroke based stylized animations, to maintain the temporal coherence of

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

strokes between frames is a challenging problem which should also be emphasized in our future work.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

Chapter 7

Stylized Shading with Feature Guided Texture Synthesis

Stylized line drawing based on feature lines is a highly abstract description of shapes. In hand made artworks and illustrations, stylized shading is equally important that gives strong cues of shapes and more details. Existing techniques can generate a small set of shading styles, such as stippling and hatching. In real works, the styles of shading are countless and hard to be defined which show the creativity of artists and illustrators. In this chapter, we experiment in emulating a wide range of shading styles with example based rendering methods for shapes illustration.

The target of example based stylized shading can be formulated as follows: Given an example image, e.g. image of real artworks or illustrations, we represent shapes and objects in a synthesized image with the shading style learned from the example. By treating shading styles as sample textures, this can be considered a texture synthesis problem. Aiming at preserving important features of the illustrating shapes in the synthesized image, we present an improved feature guided texture synthesis algorithm in creating stylized shading from the example. In our algorithm, a feature field is introduced to better express shapes and guide the texture synthesis process. Additionally, an improved L_2 neighborhood similarity metric is also proposed. Results and comparisons are given to demonstrate the effectiveness of the improved algorithm.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

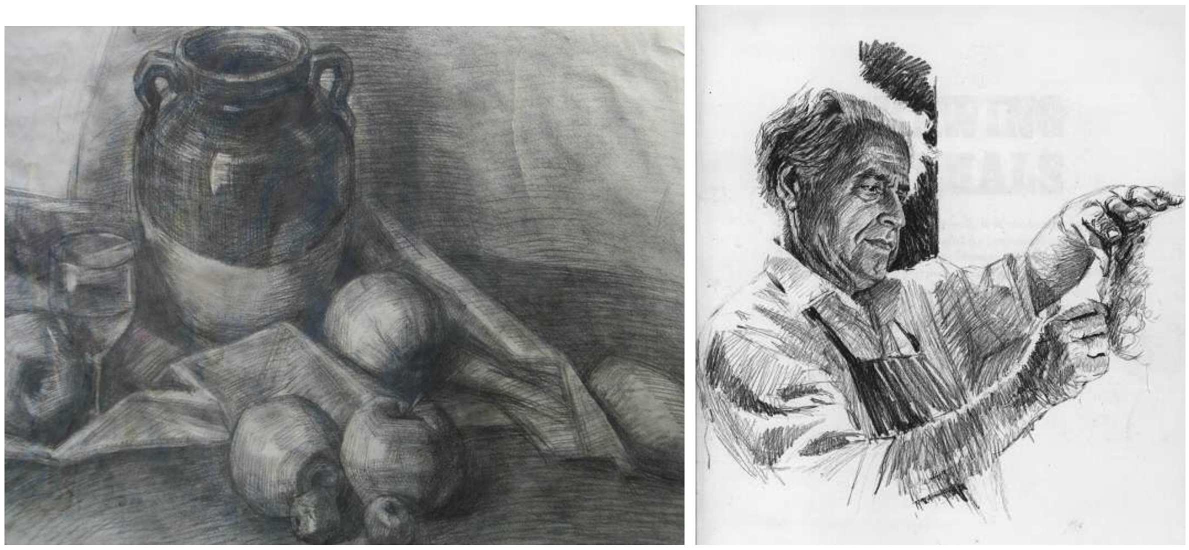


Figure 7.1: Hand-made pencil drawings in which shapes and objects are illustrated with stylized shading using overlapping pencil strokes.

As the first step of our work on stylized shading, our algorithm works in image space i.e. we stylize shapes and objects present in a source image with shading styles from the example image, the source image can be the rendered result of the illustrating shapes or scenes. Further discussion on styles and stylized shading are given in Chapter 8.

7.1 Introduction

In hand made artworks and illustrations, shading styles of shapes and objects are always apparent but hard to be precisely defined. Two hand-made pencil drawings are shown in Figure 7.1, in which shapes and objects are effectively illustrated with stylized shading which is built with overlapping pencil strokes. The use of different pencil strokes, and the different ways of shading with pencil hatching, give strong cues of different styles. But it is still hard to define the different styles precisely since style is somewhat subjective which is not well defined yet even in the area of art. To simulate stylized shading in pencil drawing, some existing systems such as PencilSketch [122] work interactively, and leave artists to make decisions on style issues. Some other techniques work automatically,

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

such as methods based on line integral convolution(LIC) [73] [68], but these methods are limited in specific subset of pencil styles. They are not effective in generating styles as those shown in Figure 7.1. Comparing with these methods, example based rendering methods are more versatile. But there are still fundamental challenges in stylized shading from example. Without a well defined style model, the recognition of shading styles from the example is not precise and often depends on prior knowledge. Most existing methods consider the problem a texture synthesis problem by treating style as a special texture and model style with low-level statistical features. These methods perform surprisingly well in a lot of cases though the fundamental challenges remain.

Texture synthesis generates textures similar to the example. It can generate a large patch of texture under the rule of an underlying small patch of texture without obvious patterns repetition. The similarity of the textures can be modeled with a Markov Random Field(MRF) based on low-level statistical features. There are mainly two classes of texture synthesis methods. Pixel-based methods [14] [25] [127] generate textures pixel by pixel while patch-based methods [24] [69] [63] [134] generate textures patch by patch. Some low-level characters of shading styles can be defined and replicated with texture synthesis. The statistical property of the similarity of the textures is analysis based on the MRF framework in Section 7.2.

Some existing example based rendering techniques [118] [24] [69] [45] [30] [107] get good results in style learning based on constrained texture synthesis. Efros et. al. [24] propose the image quilting method. It is applied in stylized shading synthesis with a correspondence map which enforces additional constraints during the texture synthesis process. In the seminal work of [45], Hertzmann et al. propose the image analogies framework which learns shading styles from image pairs in pixel-wise correspondence. Taking stroke directions into consideration, and defining brush stroke directions with the medial axes of segmented regions, Wang et al. [124] propose an example-based painting

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

method which uses a hierarchical patch-based approach to the synthesis of directional textures. For all these methods, there is a tradeoff between the preservation of features of the illustrating shapes with more details and the preservation of shading style with larger patches of textures. Usually, this tradeoff is handled with global control parameters, as shown in [24] and [45]. This does not work well in many cases. Aiming at better preservation of important features for shapes illustration, based on the image analogies framework [45], we propose an improved algorithm which defines the illustrating shapes with a feature field that guides the texture synthesis process. For the same purpose, an improved L_2 neighborhood distance metric is also introduced. Since pencil drawing is widely used in shapes illustration, in this chapter, we demonstrate the effectiveness of the improved algorithm with results and comparisons of pencil style shading from examples.

The rest of the chapter is organized with the analysis of texture synthesis within the MRF framework in Section 7.2, followed by the introduction of our feature guided texture synthesis method in Section 7.3. Results and comparisons are shown in Section 7.4, with discussion and future work in Section 7.5.

7.2 MRF Analysis of Texture Synthesis

Texture synthesis techniques model texture with low-level statistical features. The statistical property of the similarity of a synthesized texture can be well defined based on the MRF framework. By modeling a texture with a Markov Random Field(MRF), texture synthesis can be illustrated with a Bayesian inference process [80].

According to the constraints of the synthesis process, texture synthesis can be classified in two categories: unconstrained texture synthesis and constrained texture synthesis. An example of unconstrained texture synthesis is shown in Figure 7.2. The input texture example of (a) can be defined as MRF X, and the synthesized texture of (b) can be

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

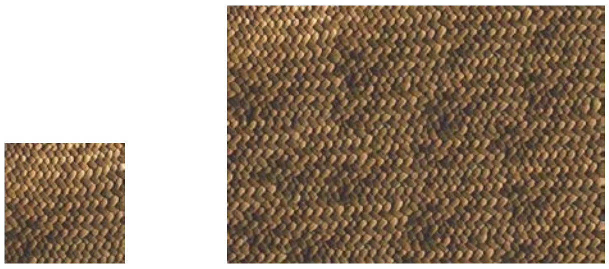

(a)texture example (b)synthesized texture

Figure 7.2: Unconstrained texture synthesis. (b) is synthesized from example (a).

defined as MRF Z.

$$X = \{X_s, s \in S'\}$$

$$Z = \{Z_s, s \in S\}$$

Both S and S' are lattices of sites of the MRF, which are pixels of image (a) and (b) in our case. Thus the possibility of the output Z given the input X can be defined with:

$$P_{Z/X} \propto exp(-\sum_{s \in S}(\min_{p \in S'} D(N(X_p), N(Z_s)))) \qquad \text{(Eq. 7.1)}$$

where $N(.)$ is the neighborhood of the sites and $D(.)$ is the distance. Here, the distance is the color distance among pixels.

In the case of stylized shading with constrained texture synthesis, similar to unconstrained texture synthesis, the style example is defined as MRF X, and the synthesized image is defined as MRF Z. An additional constraint, the source image can be defined as MRF Y.

$$Y = \{Y_s, s \in S\}$$

According to the general Bayesian inference, the posterior possibility $P_{Z/Y}$ can be computed from the prior possibility with:

$$P_{Z/Y} \propto P_Z P_{Y/Z}$$

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

where P_Z is a uniform possibility distribution. Therefore $P_{Z/Y}$ is decided by $P_{Y/Z}$ which can be defined as a different metric between values of cites of Y and Z.

$$\begin{aligned} P_{Y/Z} &\propto exp(-\beta \sum_{s \in S} (Y_s - Z_s)^2) \\ &\propto exp(-\beta \sum_{s \in S} (D(N(Y_s), N(Z_s)))) \\ &\propto exp(-\beta \sum_{s \in S} \min_{p \in S'} (D(N(Y_s), N(X_p)))) \end{aligned}$$

All together, according to the likelihood of $P_{Z/X}$ and $P_{Y/Z}$, the MAP estimation of Z_{MAP} for constrained texture synthesis is equivalent to the minimization of:

$$U(X, Y, Z) = \sum_{s \in S} \min_{p \in S'} (D(N(x_p), N(z_s)) + \beta D(N(Y_s), N(X_p))) \qquad \text{(Eq. 7.2)}$$

Examples of stylized shading with constrained texture synthesis are given in section 7.4. Usually, a global parameter β is used to minimize $U(X, Y, Z)$ to synthesize image similar to the source image. In our feature guided texture synthesis, a discrete function $\kappa(v(x, y))$ which is a feature field generated from the source image Y, is introduced to better preserve important features of the illustrating shapes.

7.3 Feature Guided Texture Synthesis

Focusing on shading style synthesis, our improved feature guided texture synthesis algorithm models style with low-level statistical features and represents illustrating shapes and objects with a feature field generated from the source image. The feature field will guide the texture synthesis process for better preservation of important features of the illustrating shapes. Our algorithm is based on the image analogies [45] framework.

7.3.1 Image Analogies Framework

In image analogies [45], the problem to be solved is defined as:

Given a pair of images A and A' (the *unfiltered* and *filtered source images*, respectively),

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

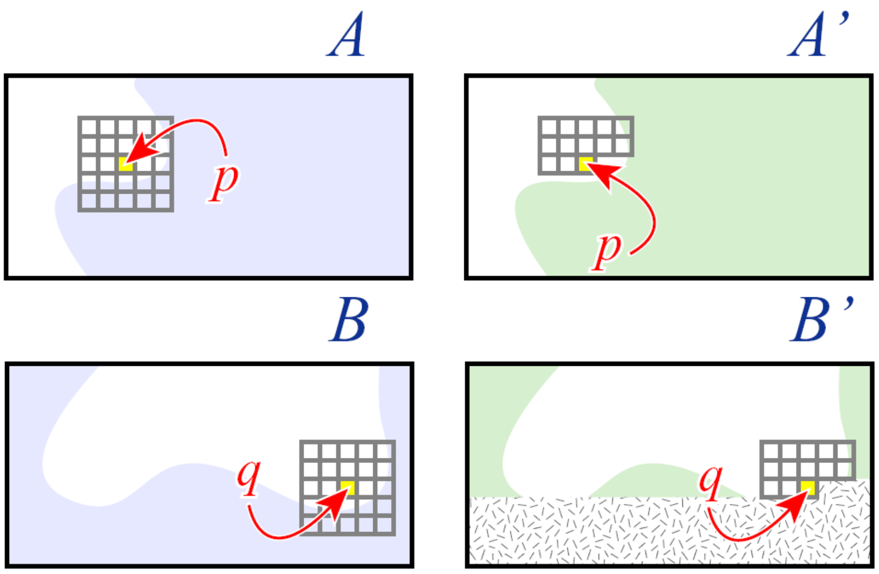


Figure 7.3: Neighborhood matching for image analogies. In order to synthesize the pixel value at q in image B' , pixel q with its neighborhood is considered. We search for the pixel p in the A images that give the closest match. The synthesis proceeds in scan-line ordering in B'.(Courtesy of Hertzmann et al. [45])

along with some additional *unfiltered target image* B, synthesize a new *filtered target image* B' such that:

$$A : A' :: B : B'$$

This is achieved by finding the best matches pixel by pixel. As illustrated in Figure 7.3, for the pixel q of B being synthesized, the pixel p in the source image A that best matches q is found. Then, the pixel in the corresponding position of p in A' is copied to the pixel in the corresponding position of q in B'. The core of this process is finding the best matches with neighborhood. Two different approaches are combined, an approximate search which attempts to efficiently find the closest-matching pixel according to the feature vectors of p, q, and their neighborhoods; and a coherence search which attempts to preserve coherence with the neighboring synthesized pixels. A coherence parameter κ is then introduced to choose between the best approximation matching and the coherence matching. The larger the value of κ, the more coherence is favored over accuracy in

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

the synthesized image. Further more, multiscale (Gaussian pyramid) representations of images are also constructed to match pixels with the size of the neighborhood of pixels varying from large to small.

For our stylized shading based on image analogies, we have the example image A', and the input source image B. The unfiltered image A is generated by applying anisotropic diffusion or similar filter to A' (such as the "Smart Blur" filter of Adobe Photoshop). And the stylized image is B'. Without losing generality, our feature guided texture synthesis algorithm improves in two key features of the matching process of image analogies:

(i) Instead of using a global parameter κ to choose between the approximation matching and the coherence matching, a discrete function $\kappa(v(x, y))$ which is a feature field generated from the input image B, is introduced for per-pixel local control. The feature field guides the synthesis process to achieve better preservation of both the illustrating shapes of the source image and the shading style of the example image in the synthesized image B'.

(ii) An improved L_2 neighborhood distance metric which provides better measures of texture similarity is introduced based on the feature field.

7.3.2 Feature Field Generation

Since the human perceptual system is more sensitive to features such as edges, corners and somewhat salient structures of an image, it is especially effective in representing shapes present in an image using these features. We assume that pixels nearer to these features carry more information of the image. And the amount of information carried by pixels drops off gradually. In this way, we define a distance based feature field to represent the distribution of information in the source image.

The features of the source image can be extracted with various feature detectors, such as the Canny edge detector [9] for line features. By considering feature points as

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

a pre-defined scatter data set, the feature field can be generated with distance-based interpolation. Scatter data interpolation with the radial basis functions(RBF) [91] is a natural choice for our feature field generation. It is a linear combination of nonlinear functions of distances from the feature points. The interpolation is defined as:

$$v(X) = \sum_{k=1}^{n} w_k \phi(\|X - X_k\|) \tag{Eq. 7.3}$$

with:

$$\phi(x) = exp(-\frac{x^2}{2\sigma^2})$$

where X_k is pre-defined feature points, $v(X)$ is the interpolated value of the feature field at point X. σ controls the speed of the drop-off according to the distances. w_k is the weight of X_k, which can be solved with a linear system from the pre-defined values of the feature points.

In our experiments, the RBF interpolation is very sensitive to points clusters, and the computation is extremely expensive with a large set of pre-defined feature points. To solve these issues, we define our interpolation as:

$$v(X) = \phi(w \min_{k=1}^{n} \|X - X_k\|) \tag{Eq. 7.4}$$

Instead of using a linear combination of the distances to all the pre-defined feature points, we use the minimum distance to the nearest one. To speed up, we approximate the Euclidean distance with the 3-4 distance [5]. The 3-4 distance defines the distance of horizontal and vertical neighboring pixels to be 3, and the distance of diagonal neighboring pixels to be 4. The distance field is computed with a two-pass scanning during which a distance propagation is performed to achieve the minimum distance image. The pseudo-code is given in [5]. In Eq. 7.4, w is a user-defined parameter to adjust the approximated distance and counteract the approximation errors. In our experiments, $0.1 =< w <= 0.3$ gives good results. An example is shown in Figure 7.4. The feature points image (b) is

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

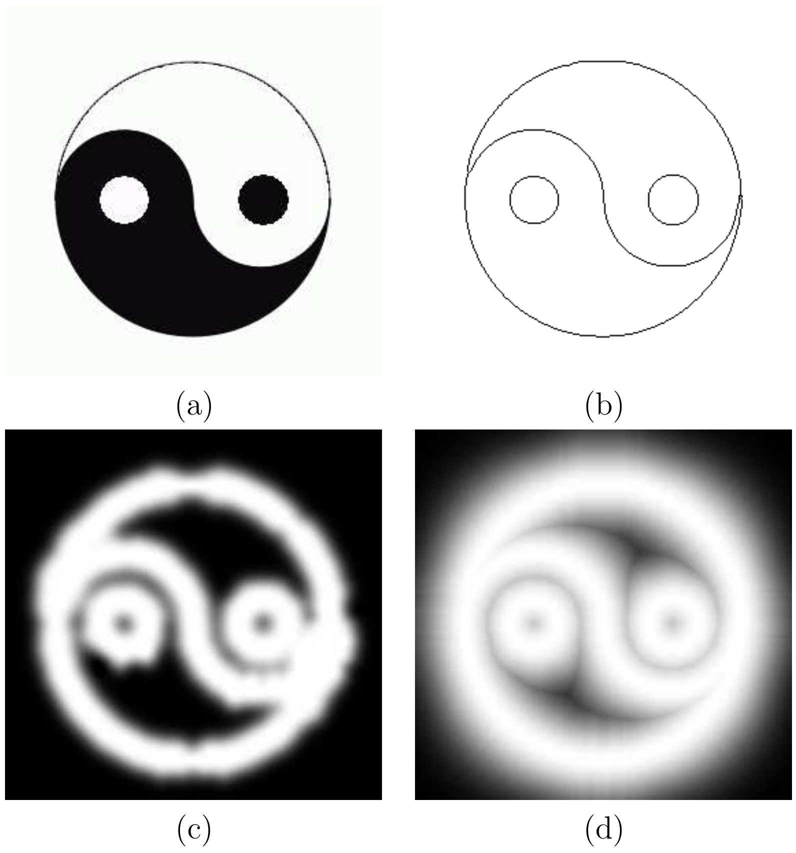


Figure 7.4: Feature field interpolation. (a)the source image. (b)the feature points. (c)the feature field generated with the RBF based interpolation. (d)the feature field generated with the minimum distance transformation.

extracted from the source image (a) with the Canny edge detector. The feature field (c) is generated with RBF interpolation with $\sigma = 7$ (Discussion on the parameter σ of gaussian radial basis functions can be found in neural network literatures). The feature field (d) is generated with our minimum distance transformation, also with $\sigma = 7$, together with $w = 0.1$ for the distance approximation adjustment. Obviously, the feature field (d) gives a much smoother result.

7.3.3 Improved L_2 Distance Metric

For both pixel-based and patch-based constrained texture synthesis, choosing an element(a pixel or a patch) in a new position must subject to two constraints: (1) the

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

measurement of the similarity with the neighboring already synthesized elements. (2) the measurement of the similarity with the underlying source image. Mostly, both the measurements of the similarity are computed with the L2-norm. It is well-known that this summed squared difference is not perfect because each pixel involved might contribute differently to the similarity measurement. Therefore, a weight should be assigned to each pixel for a better similarity measurement. As discussed in the previous section, the feature field $v(x, y)$ generated from the source image defines how 'important' each pixel is in representing the image. Based on the feature field, an improved L2-norm distance metric can be defined with per-pixel weight control as:

$$D(I, I') = \sum v(x, y) \| I(x, y) - I'(x', y') \|^2 \qquad \text{(Eq. 7.5)}$$

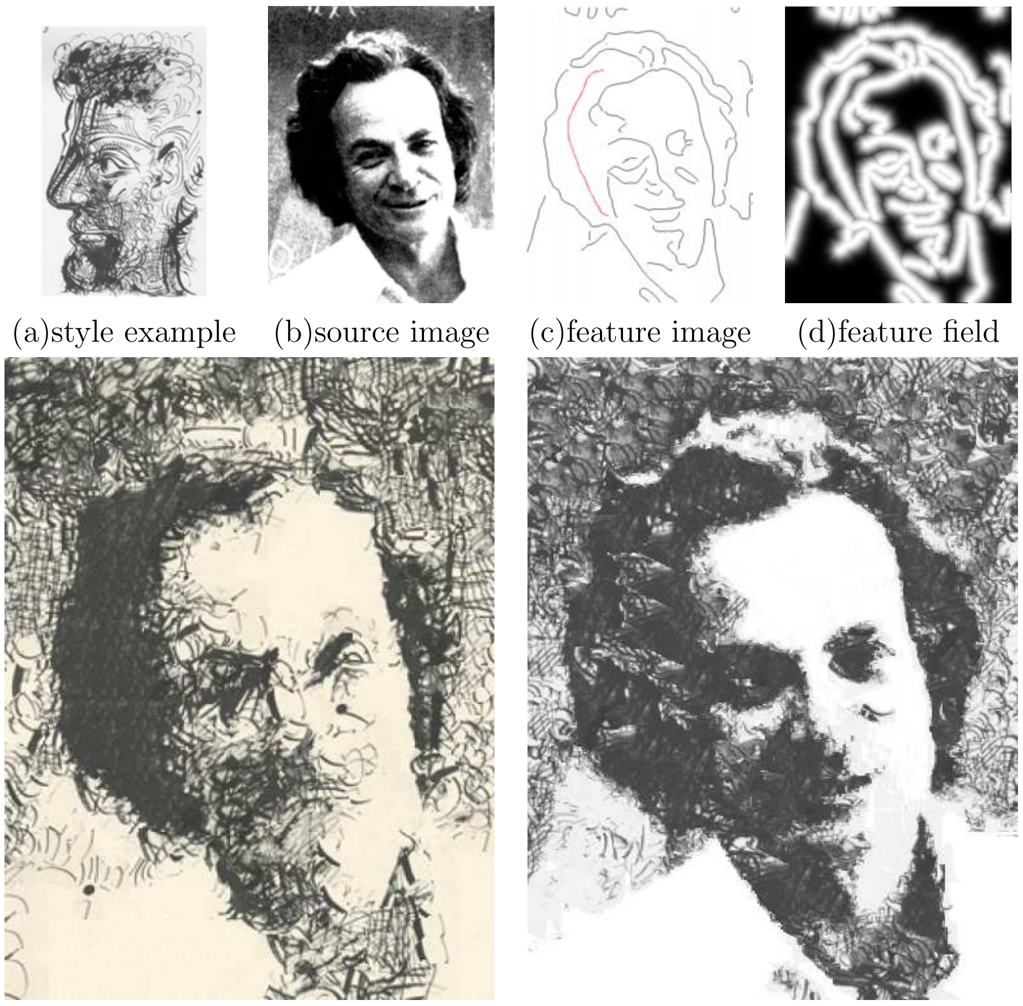
where I and I' are two corresponding patches, and their distance of similarity, $D(I, I')$, is moderated with a weight field $v(x, y)$. This improved distance measurement can be applied to both pixel-based and patch-based texture synthesis with neighborhoods or patches in fixed or various shapes and sizes.

7.4 Results and Comparisons

Among existing example based rendering methods, the image quilting method [24] and the image analogies method [45] are two outstanding and representative methods which achieve promising results in learning artistic shading styles. Based on the image analogies framework, our algorithm improves in preserving important features and communicating shapes better in the synthesized image. Improved results with comparisons are given to demonstrate the effectiveness of our algorithm.

Figure 7.5 compares results of our algorithm with image quilting. Image (e) of image quilting capture the style of pencil sketch of the style example (a), but the face part of the source image (b) are badly blurred. Comparing with image (e), the result of our feature

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library



(a)style example (b)source image (c)feature image (d)feature field

(e)result of image quilting from [24] (f)our result

Figure 7.5: Comparison of feature guided texture synthesis and image quilting for stylized shading, showing that feature guided texture synthesis achieves better results.

guided algorithm, image (f) shows big improvement in both capturing style of the style example (a) and preserving important features of the shape present in the source image (b). Since style is quite subjective, some user intuitive control is critical for example-based stylized shading. Figure 7.5 partially demonstrates the user controllable stylized shading of our algorithm with a user added feature line(in red) in the hair part of image (c) to adjust the hatching result in image (f).

Figure 7.6 compares results of our algorithm with image analogies. In image (e), the

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

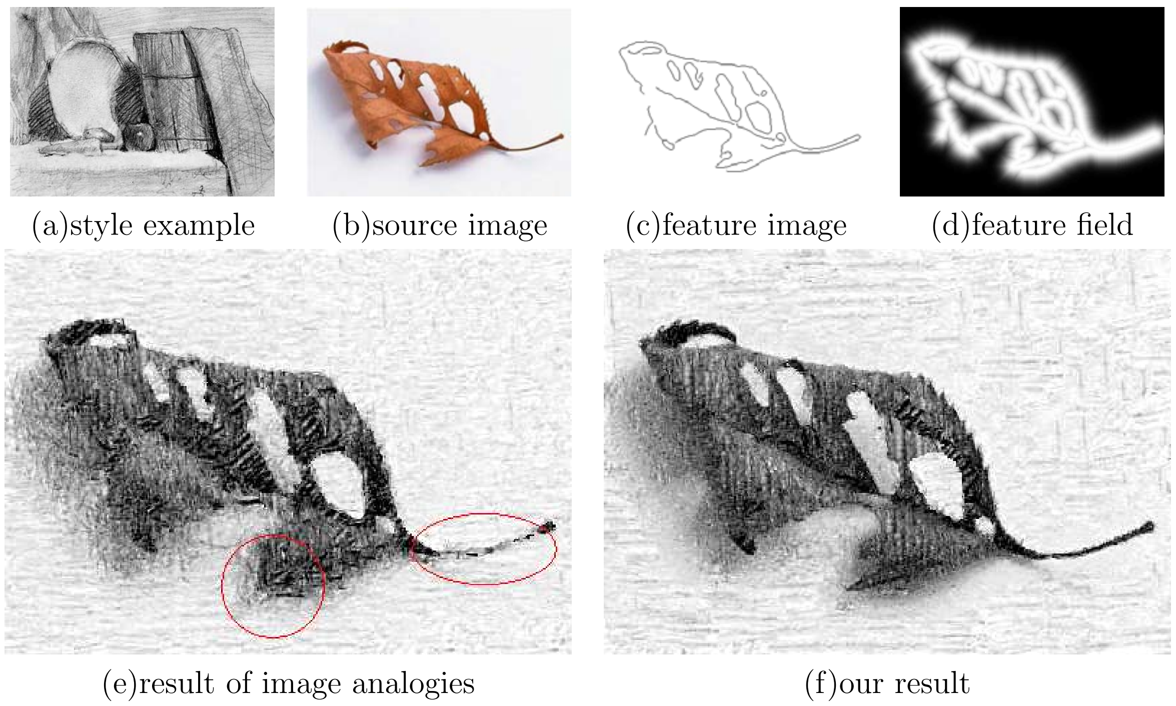


Figure 7.6: Comparison of feature guided texture synthesis and image analogies for stylized shading, showing that feature guided texture synthesis achieves better results.

result of image analogies method, the edge of the leaf and some other parts(as indicated in the image with red circles) are badly blurred, while result of our algorithm, image (f) shows largely improved quality in all of these parts. Since our method is based on the image analogies framework, in this example, both methods use $\kappa = 4$ for better comparison.

Some existing commercial image processing software, such as Adobe Photoshop, provides artistic filters for images stylization. A comparison of our algorithm with Adobe Photoshop in pencil sketch is shown in Figure 7.7. We also compare our algorithm with stylized shading generated with specific techniques. An example is shown in Figure 7.8, image (c) of which is a pencil drawing generated with the line integral convolution(LIC) [68] method that gives a pencil drawing in a specific style different from ours. Our algorithm tries to capture the feeling of a dark pencil drawing on an old paper, as shown in image (d). Note that for both the artistic filters of Adobe Photoshop and

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

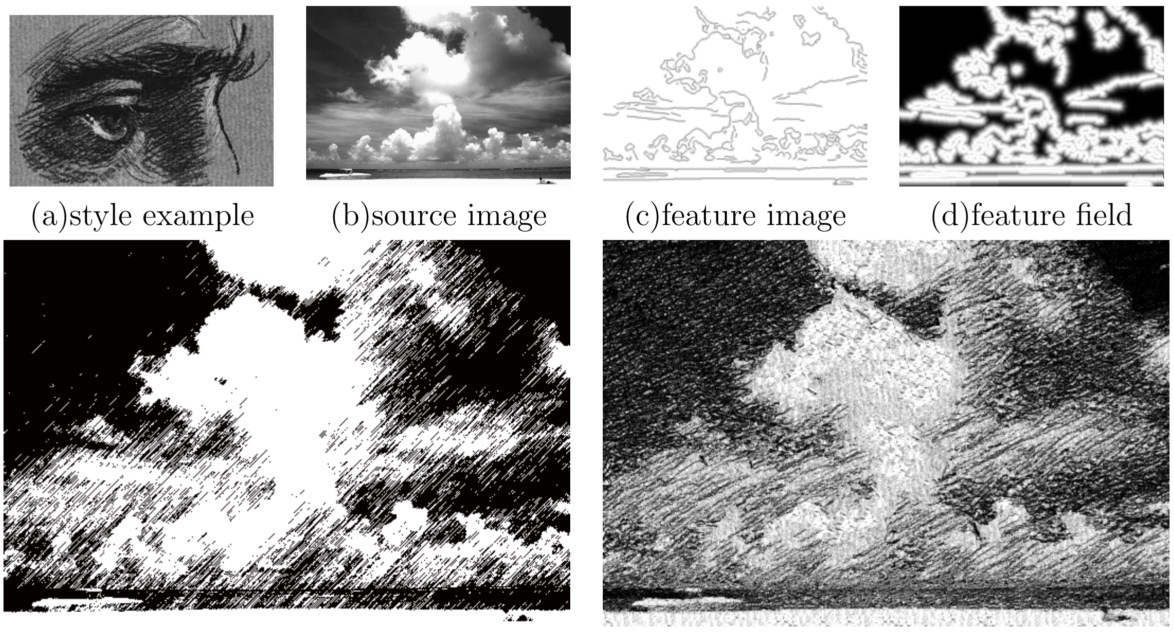
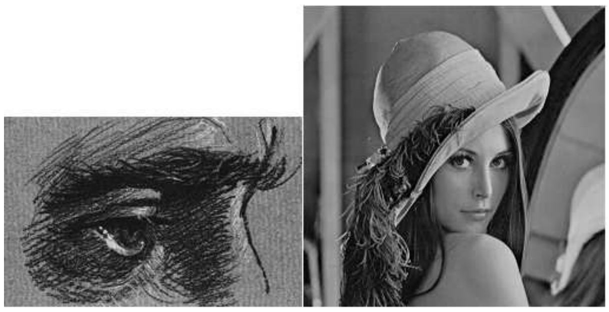


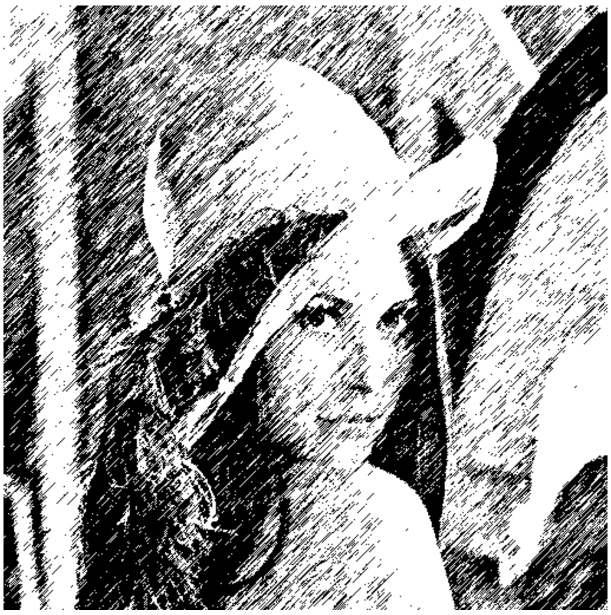
Figure 7.7: Comparison of our algorithm and Adobe Photoshop, showing that our algorithm achieves better result and are capable of learning style from example.

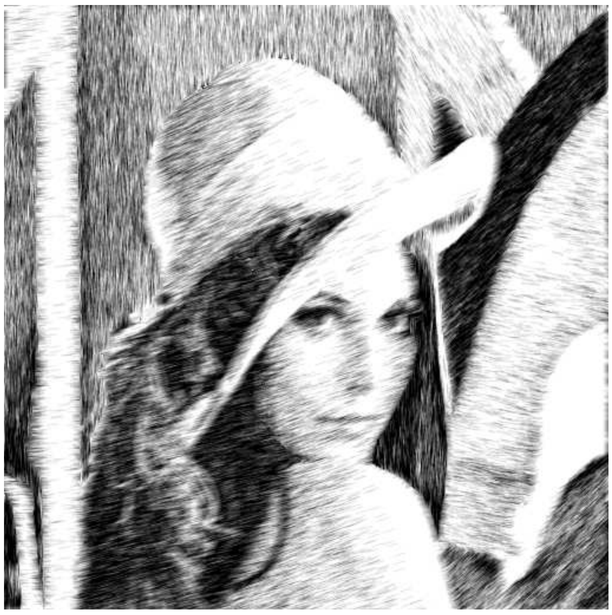
methods like LIC etc., users can only obtain a limited space of styles by adjusting some given parameters. Our algorithm is capable of learning different styles from different examples.

As demonstrated in the previous chapters, stylized line drawing based on feature lines gives a highly abstract description of 3D shapes. On the contrast, stylized shading focuses on more detailed variation of illumination. The combination of stylized line drawing with stylized shading improves the effectiveness of 3D shapes illustration greatly. As demonstrated in Figure 7.9, (c) and (d) are the stylized line drawing and stylized shading alone in pencil style respectively, (e) is a composition of the stylized line drawing and the stylized shading. Compared with (c), (e) presents more interior details; while compared with (d), (e) shows much better global structure. In current stage, our stylized shading algorithm works in image space. Therefore, the pencil hatching might not follow the 3D geometry very well in this example. To convey 3D shapes better with artistic texture, one

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library


(a)style example and the source image


(b)result of Adobe Photoshop


(c)result of LIC method [68]

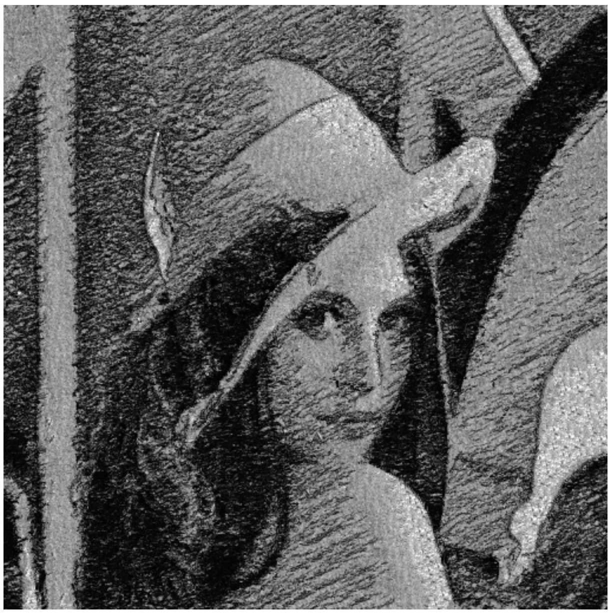
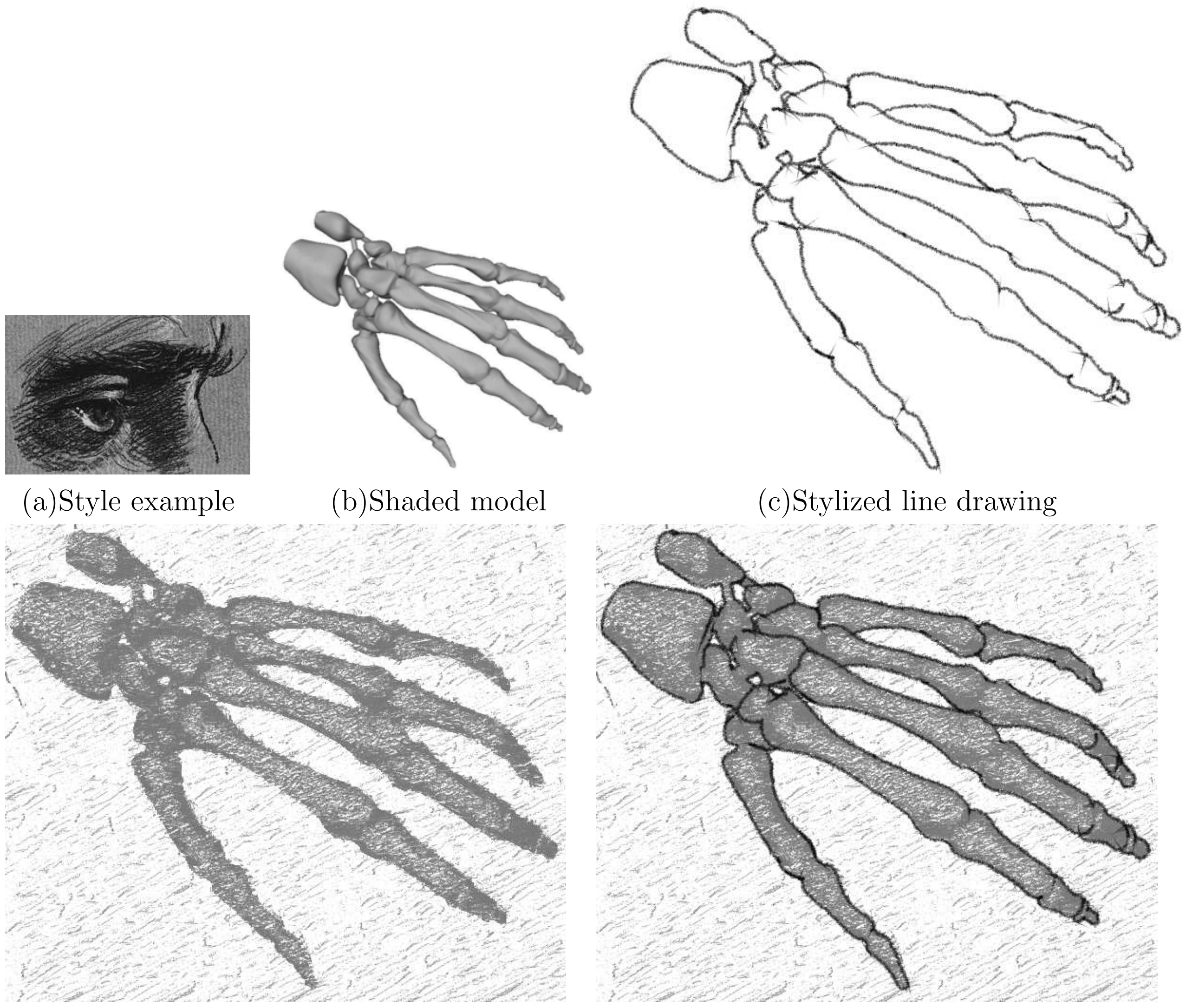

(d)our result

Figure 7.8: Comparison of pencil drawings generated with Adobe Photoshop, LIC method and our algorithm. LIC method gives pencil drawing in a specific style while our algorithm captures the feeling of a dark pencil drawing on an old paper which is similar to the style example.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library


(a)Style example (b)Shaded model (c)Stylized line drawing

(d)Stylized shading (e)Composition

Figure 7.9: Combination of stylized line drawing and stylized shading for 3D shapes illustration.

of our future works is to develop a quite different algorithm to generate stylized shading directly on 3D surfaces. To further improve the effectiveness of the composition, we are also working on a line drawing algorithm which synthesizes stylized lines from the input examples. In summary, the combination of stylized line drawing and stylized shading conveys shapes effectively and gives very intuitive cues in the direction of developing novel algorithms in conveying 3D shapes.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

7.5 Conclusion

In this chapter, we propose an improved feature guided texture synthesis algorithm for example-based stylized shading. Comparing with existing example based texture synthesis methods, our algorithm achieves improved results in representing and communicating shapes and objects with stylized shading in the synthesized image. Aiming at better preservation of important features for shapes illustration, a feature field generated from the source image and an improved L_2 distance metric for similarity measurement are introduced and incorporated in the texture synthesis process.

As the first step of our work on example based stylized shading, our algorithm works in image space based on texture synthesis techniques. Texture synthesis methods model shading style with low-level statistical features. Unfortunately, artistic styles are mostly defined with higher-level, larger-scale features involving prior human knowledge, and texture synthesis methods are still quite limited in capturing these styles. Among existing methods, Yen et al. [137] and Wang et al. [124] define styles as sample textures that the user can select from the example, and Wang et al. [124] segment the source image into regions before applying textures. This improves the texture synthesis result greatly. And the method of applying textures following a direction field of the segmented regions can also be adapted for feature preservation. To solve the fundamental problem of style learning, a practical model for artistic style definition should be developed. This is also our long term future work. More discussions are given in Chapter 8.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

Chapter 8

Conclusions and Future Work

8.1 Summary

Over centuries, artists and illustrators have developed techniques to effectively convey shapes in very rich forms with applications in a wide range of areas such as scientific and medical illustration, technical and archaeological illustration, and storytelling. In recent years, NPR techniques have improved greatly to assist artists and illustrators with computer generated imagery. However, it is more than often that hand drawn illustrations are still the only choice in modern industry. This motivates us to investigate more effective techniques.

Though a number of feature lines have been proposed, they don't seem able by themselves to capture all the visually relevant features. Thus the first aim of our work is to obtain new feature lines which are more perceptual consistent and flexible in conveying shapes. Strongly inspired by the observation in human vision and perception that a sudden change in the luminance plays a critical role to faithfully represent and recover 3D information, we adopt the edge detection techniques of image processing to generate line drawings for 3D shape illustration, and present *Photic Extremum Line* (PEL) in Chapter 3 which emphasizes significant variations of illumination over 3D surfaces. PELs offer users more freedom and control to achieve their desired illustrations.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

In Chapter 4, two novel algorithms are introduced for PELs extraction. One is the fast object space PELs extraction algorithm for triangle meshes. The other is the algorithm which extracts PELs in geometry-image space with 2D image processing techniques. With these methods, PEL may be incorporated into various rendering systems.

Sketching is a traditional illustration style which conveys shapes with successive approximation. Architects often trace over computer rendering of their initial designs to achieve a sketchier effect. Based on the observation that the approximation of feature lines in sketching involves multiple overlapping strokes that are broken in an artistic way, in Chapter 5, we emulate sketching strokes with a novel approach which incorporates principles of perceptual grouping with a global segmentation algorithm to break long strokes into short ones. The strokes generated by our approach qualitatively resemble those produced by artists, and the successive approximation effect seen in sketching is effectively simulated by employing our approach at a succession of scales.

In many applications, one of the most important goals of illustration is to achieve visually pleasing line drawings, such as cartoon. Feature lines can convey shapes precisely. Yet, stylized strokes based on feature lines make a line drawing more vivid and attractive. For these applications, in Chapter 6, we present an effective and flexible framework for conveying shapes with stylized line drawings based on a novel stroke representation model, *disk B-spline curve*(DBSC). We demonstrate the flexibility of our stylized line drawing framework with pencil drawings generated with physical simulation, and line drawings with strokes in a wide range of traditional styles generated with procedural methods.

In illustrations, stylized shading is equally important which gives strong cues of shapes and more details. In Chapter 7, by modeling shading styles with low level statistical features, we present an improved feature guided texture synthesis algorithm to create shading from examples in artistic styles.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

8.2 Future Work

Up to date, a great number of illustration styles have been developed by artists and illustrators. These styles show the creativity of artists and illustrators, conveying shapes effectively and impressively. In current stage, only a very small set of styles are nicely emulated with computer, and a lot of artists still think that computer generated illustrations have a mechanical feeling, more or less. This is why hand-drawn illustrations are always the only choice for high quality. Our work on stylized line drawing and stylized shading leads us to seek for a general model to represent styles. It turns out extremely technical challenging, since the pictorial style is highly subjective which is not precisely defined even in the area of art. In the future, we expect to explore deeper in this area to design and develop a system which can effectively learn illustration styles from examples. Subsequently the next step is to improve the various parts of our work(e.g. feature lines, stylized line drawing, polycube map space rendering, example-based rendering), and integrate these techniques into an effective illustration system. More details are given in the rest of this section.

8.2.1 Style Learning from Example

The core of emulating traditional illustration styles with style learning from examples is to find a more general pictorial style definition which can be represented with computer. In fact, neither terms for pictorial style nor general agreement on what constitutes pictorial style have been defined or agreed. In art, the style has a wide range of meaning. It may be defined with the whole cultures such as style of oriental painting and western painting, or defined with periods such as style of Baroque art which is stilted, affected and degenerated. It may also be defined with individual artists such as style of Van Gogh. Further more, each category of style definition may also vary widely.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

In computer graphics, Willats and Durand [130] try to define pictorial style by comparing it with styles of modern linguistics, in which the notion of style is associated with the use of specific linguistic structures. This suggests that, as in language, style in pictures is also defined with distinctive combinations of pictorial devices or structures characteristic of a particular culture or period or the work of a particular artist. According to these, Willates [129] describes the rules of presentation in terms of three broad categories: (i) the *drawing* or *spatial systems*, which map spatial relations in the scene into corresponding spatial relations on the picture surface; (ii) the *denotation systems* which map scene primitives into picture primitives; and (iii) the *mark systems* which map picture primitives into physical marks. Durand [22] adds the fourth category: the *attributes system* which defines the attributes and properties of elements that make up the denotation and mark systems. This intuitive style definition is however very coarse-grained, and requires great effort to implement and realize with computer programs.

In practice, some existing techniques define style with low level statistical features(e.g. the local distribution of color or illumination of pixels), and model style with Markov Random Field. As shown in our experiments on stylized shading with texture synthesis in Chapter 7, this does not work well with most artistic styles with higher-level, larger-scale features such as broad, coherent brush strokes. Mostly, the definition of these artistic styles involves prior human knowledge. Thus, in the future, we plan to find more practical style definition and develop more efficient data structures for style representation in computer graphics.

With a practical style model, the following steps for style learning from examples include feature selection for style representation, feature extraction, and applying machine learning algorithms for style learning based on features. The challenge is that these tasks are application dependent which tightly related to the style model. This involves research in many areas including computer graphics, computer vision and machine learning.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

8.2.2 Improvement and System Integration

In the aspect of engineering, one of the most important goals of our research is to build tools to help artists and illustrators to improve their efficiency of developing graphics related products. Towards this goal, we have developed prototype systems. The next step is to improve and integrate techniques we have proposed into an effective illustration system.

In the future, we will improve our PEL definition to incorporate high level rules of human perception and extend PEL for direct volume rendering instead of PEL for iso-surfaces of volumes. For our PELs extraction algorithm in geometry-image space, the next step is to extend it to polycube map space. For our stylized line drawing based on feature lines, two issues need to be tackled in the near future. One is a more optimal scheme for artistic stroke segmentation and the other is a better way of determining the thickness of our DBSC strokes automatically. We are currently at the initial stage of our example based stylize shading. Effective style learning from examples for shapes illustration will be carried on with our long term research on style definition and representation.

To integrate our techniques into an effective illustration system, we need a well defined work flow and an excellent user interface. An important issue of the integration is the speed. It is possible to extract PELs in real time, and the stylization of DBSC strokes for a wide range of styles can also be performed in interactive rate. However, the generation of the geometry image or the polycube map, the segmentation of strokes, and the stylized shading with constrained texture synthesis are still far from the requirement of real-time or interactive rendering in current stage(e.g. the polycube map can be generated only offline). In a NPR system, interactivity between artists and the system is extremely important for its effectiveness. Thus, we need further improvement on performance in integrating these subsystems.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

References

[1] N. Amenta, S. Choi, T. K. Dey, and N. Leekha. A simple algorithm for homeomorphic surface reconstruction. In *SCG '00: Proceedings of the sixteenth annual symposium on Computational geometry*, pages 213–222, New York, NY, USA, 2000. ACM.

[2] K. Anjyo and K. Hiramitsu. Stylized highlights for cartoon rendering and animation. *IEEE Comput. Graph. Appl.*, 23(4):54–61, 2003.

[3] P. Barla, J. Thollot, and L. Markosian. X-toon: an extended toon shader. In *International Symposium on Non-Photorealistic Animation and Rendering (NPAR)*. ACM, 2006.

[4] J. E. Bobrow, B. Martin, G. Sohl, E. C. Wang, F. C. Park, and J. Kim. Optimal robot motions for physical criteria. *Journal of Robotic Systems*, 18(12):785–795, 2001.

[5] G. Borgefors. Hierarchical chamfer matching: a parametric edge matching algorithm. In *IEEE Trans. Pattern Analysis and Machine Intelligence 10.*, pages 849–865, 1988.

[6] S. Bruckner and M. E. Gröller. Volumeshop: an interactive system for direct volume illustration. In *IEEE Visualization 2005*, pages 671–678.

[7] J. W. Buchanan and M. C. Sousa. Observational model of blenders and erasers in computer-generated pencil rendering. In *Graphics Interface (GI'99)*, 1999.

[8] M. Burns, J. Klawe, S. Rusinkiewicz, A. Finkelstein, and D. DeCarlo. Line drawings from volume data. In *SIGGRAPH 2005*, pages 512–518.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

[9] J. Canny. A computational approach to edge detection. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 8:679–698, 1986.

[10] F. Cazals and M. Pouget. Ridges and umbilics of a sampled smooth surface: a complete picture gearing toward topological coherence. In *Rapport de Recherche RR-5294, INRIA, September*, 2004.

[11] M. Chi and T. Lee. Stylized and abstract painterly rendering system using a multiscale segmented sphere hierarchy. *IEEE Transactions on Visualization and Computer Graphics*, 12(1):61–72, 2006.

[12] Y. S. Chua. Bézier brushstrokes. *Computer-Aided Design*, 22(9):550–555, 1990.

[13] F. Cole, A. Golovinskiy, A. Limpaecher, H. S. Barros, A. Finkelstein, T. Funkhouser, and S. Rusinkiewicz. Where do people draw lines? *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 27(3), August 2008.

[14] J. S. de Bonet. Multiresolution sampling procedure for analysis and synthesis of texture images. In *SIGGRAPH 1997*, pages 361–368.

[15] D. DeCarlo, A. Finkelstein, and S. Rusinkiewicz. Interactive rendering of suggestive contours with temporal coherence. In *NPAR '04: Proceedings of the 3rd international symposium on Non-photorealistic animation and rendering*, pages 15–24, New York, NY, USA, 2004. ACM.

[16] D. DeCarlo, A. Finkelstein, S. Rusinkiewicz, and A. Santella. Suggestive contours for conveying shape. In *SIGGRAPH 2003*, pages 848–855.

[17] D. DeCarlo and S. Rusinkiewicz. Highlight lines for conveying shape. In *NPAR '07: Proceedings of the 5th international symposium on Non-photorealistic animation and rendering*, pages 63–70, New York, NY, USA, 2007. ACM.

[18] K. Demarsin, D. Vanderstraeten, T. Volodine, and D. Roose. Detection of closed sharp feature lines in point clouds for reverse engineering applications. In *Proceedings of geometric modeling and processing*, 2006.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

[19] M. Desbrun. Applied geometry: Discrete differential calculus for graphics. *Comput. Graph. Forum*, 23(3):269–269(1), 2004.

[20] O. Deussen, S. Hiller, C. Overveld, and T. Strothotte. Floating points: a method for computing stipple drawings. *Computer Graphics Forum*, 19(3):40–51, 2000.

[21] M. P. Docarmo. *Differential Geometry of Curves and Surfaces*. Prentice-Hall, 1976.

[22] F. Durand. An invitation to discuss computer depiction. In *NPAR 2002*, 2002.

[23] C. Dyken, G. Ziegler, and C. Theobaltand H. Seidel. High-speed marching cubes using histogram pyramids. *Computer Graphics Forum*, to appear.

[24] A. Efros and W. T. Freeman. Image quilting for texture synthesis and transfer. In *SIGGRAPH 2001*, pages 341–346.

[25] A. Efros and T. Leung. Texture synthesis by non-parametric sampling. In *Proceedings of ICCV*, pages 1033–1038, 1999.

[26] G. Elber. Line art illustrations of parametric and implicit forms. *IEEE Transactions on Visualization and Computer Graphics*, 4(1):71–81, 1998.

[27] A. Finkelstein and D. H. Salesin. Multiresolution curves. In *SIGGRAPH 1994*, pages 261–268.

[28] S. Fleishman, I. Drori, and D. Cohen-Or. Bilateral mesh denoising. *ACM Trans. Graph.*, 22(3):950–953, 2003.

[29] D. A. Forsyth and J. Ponce. *Computer Vision: A Modern Approach*. Prentice Hall, 2003.

[30] W. T. Freeman, T. R. Jones, and E. Pasztor. Example-based super-resolution. In *IEEE Computer Graphics and Applications*, pages 56–65, 2002.

[31] B. Gooch and A. Gooch. *Non-Photorealistic Rendering*. A. K. Peters, Ltd., Natick, MA, USA, 2001.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

[32] B. Gooch, P. J. Sloan, A. Gooch, P. Shirley, and R. F. Riesenfeld. Interactive technical illustration. In *I3D*, pages 31–38, 1999.

[33] T. Goodwin, I. Vollick, and A. Hertzmann. Isophote distance: a shading approach to artistic stroke thickness. In *NPAR '07: Proceedings of the 5th international symposium on Non-photorealistic animation and rendering*, pages 53–62, New York, NY, USA, 2007. ACM.

[34] I.E. Gordon. *Theories of Visual Perception.* John Wiley, Chichester, 1997.

[35] S. Grabli, E. Turquin, F. Durand, and F. Sillion. Programmable style for npr line drawing. In *Rendering Techniques 2004 (Eurographics Symposium on Rendering).* ACM Press, june 2004.

[36] X. Gu, S. J. Gortler, and H. Hoppe. Geometry images. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pages 355–361, New York, NY, USA, 2002. ACM Press.

[37] X. Gu, Y. Wang, T. F. Chan, P. M. Thompson, and S. Yau. Genus zero surface conformal mapping and its application to brain surface mapping. *IEEE Trans. Med. Imaging*, 23(8):949–958, 2004.

[38] X. Gu and S. Yau. Global conformal surface parameterization. In *SGP '03: Proceedings of the 2003 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pages 127–137, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.

[39] S. Gumhold, X. Wang, and R. Macleod. Feature extraction from point clouds. In *Proceedings of the 10 th International Meshing Roundtable*, pages 293–305, 2001.

[40] R. M. Haralick. Ridges and valleys on digital images. In *Computer Vision, Graphics, and Image Processing*, pages 28–38, 1983.

[41] D. Hearn and M. P. Baker. *Computer Graphics.* Prentice-Hall, 1997.

[42] A. Hertzmann. Painterly rendering with curved brush strokes of multiple sizes. *Computer Graphics*, 32(Annual Conference Series):453–460, 1998.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

[43] A. Hertzmann. Introduction to 3d non-photorealistic rendering: silhouettes and outlines. In *Course Notes of SIGGRAPH 1999*, 1999.

[44] A. Hertzmann. Tutorial: a survey of stroke-based rendering. *IEEE Comput. Graph. Appl.*, 23(4):70–81, 2003.

[45] A. Hertzmann, C. E. Jacobs, N. Oliver, B. Curless, and D. H. Salesin. Image analogies. In *SIGGRAPH 2001*, pages 327–340.

[46] A. Hertzmann and D. Zorin. Illustrating smooth surfaces. In *SIGGRAPH 2000*, pages 517–526.

[47] E. R. S. Hodges. *The Guild Handbook of Scientific Illustration.* John Wiley and Sons, 2003.

[48] S. C. Hsu and I. H. H. Lee. Drawing and animation using skeletal strokes. In *SIGGRAPH '94: Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 109–118, New York, NY, USA, 1994. ACM Press.

[49] S. C. Hsu, I. H. H. Lee, and N. E. Wiseman. Skeletal strokes. In *UIST '93: Proceedings of the 6th annual ACM symposium on User interface software and technology*, pages 197–206, New York, NY, USA, 1993. ACM.

[50] V. Interrante. Illustrating surface shape in volume data via principal direction-driven 3d line integral convolution. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 109–116, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[51] V. Interrante, H. Fuchs, and S. M. Pizer. Enhancing transparent skin surfaces with ridge and valley lines. In *IEEE Visualization*, pages 52–59, 1995.

[52] T. Isenberg, B. Freudenberg, N. Halper, S. Schlechtweg, and T. Strothotte. A developer's guide to silhouette algorithms for polygonal models. *IEEE Computer Graphics and Applications*, 23(4):28–37, 2003.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

[53] T. Isenberg, N. Halper, and T. Strothotte. Stylizing silhouettes at interactive rates: from silhouette edges to silhouette strokes. *Computer Graphics Forum (Proceedings of Eurographics)*, 21(3):249–258, September 2002.

[54] T. Isenberg, P. Neumann, S. Carpendale, M.C. Sousa, and J.A. Jorge. Non-photorealistic rendering in context: an observational study. In *NPAR '06: Proceedings of the 4th international symposium on Non-photorealistic animation and rendering*, pages 115–126, New York, NY, USA, 2006. ACM.

[55] M. Jin, Y. Wang, S. Yau, and X. Gu. Optimal global conformal surface parameterization. *vis*, 00:267–274, 2004.

[56] T. Judd, F. Durand, and E. Adelson. Apparent ridges for line drawing. *In SIGGRAPH 2007*, 26(3), 2007.

[57] R. Kalnins, L. Markosian, B. Meier, M. Kowalski, J. Lee, P. Davidson, M. Webb, J. Hughes, and A. Finkelstein. WYSIWYG NPR: Drawing strokes directly on 3d models. In *SIGGRAPH 2002*, pages 755–762.

[58] R. D. Kalnins, P. L. Davidson, L. Markosian, and A. Finkelstein. Coherent stylized silhouettes. *ACM Trans. Graph.*, 22(3):856–861, 2003.

[59] S. Katz, A. Tal, and R. Basri. Direct visibility of point sets. In *SIGGRAPH '07: ACM SIGGRAPH 2007 papers*, page 24, New York, NY, USA, 2007. ACM.

[60] J. Kent, K. Mardia, and J. West. Ridge curves and shape analysis. In *Monograph, Department of Statistics, University of Leeds., Leeds LS2 9JT, UK, May, 1996.*, 1996.

[61] G. L. Kindlmann, R. T. Whitaker, T. Tasdizen, and T. Möller. Curvature-based transfer functions for direct volume rendering: methods and applications. In *IEEE Visualization*, pages 513–520, 2003.

[62] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? *ECCV*, (3):65–81, 2002.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

[63] V. Kwatra, A. Schodl, I. Essa, G. Turk, and A. Bobick. Graphcut textures: image and video synthesis using graph cuts. In *SIGGRAPH 2003*, pages 277–286.

[64] A. Lake, C. Marshall, M. Harris, and M. Blackstein. Stylized rendering techniques for scalable real-time 3d animation. In *Proceedings of NPAR 2000*, pages 13–20, 2000.

[65] J. Lansdown and S. Schofield. Expressive rendering: a review of nonphotorealistic techniques. *IEEE Comput. Graph. Appl.*, 15(3):29–37, 1995.

[66] Y. Lee, L. Markosian, S. Lee, and J. F. Hughes. Line drawings via abstracted shading. In *SIGGRAPH '07: ACM SIGGRAPH 2007 papers*, page 18, New York, NY, USA, 2007. ACM.

[67] J. Lewis. Personal communication. In *Boeing Computer Services, Seattle, Washington*, November 1993.

[68] N. Li and Z. Huang. A feature-based pencil drawing method. In *GRAPHITE '03: Proceedings of the 1st international conference on Computer graphics and interactive techniques in Australasia and South East Asia*, pages 135–ff, New York, NY, USA, 2003. ACM Press.

[69] L. Liang, C. Liu, Y. Q. Xu, B. Guo, and H. Y. Shum. Real-time texture synthesis by patch-based sampling. In *ACM Transactions on Graphics (TOG)*, pages 127 – 150, 2001.

[70] B. Lichtenbelt and P. Brown. GL_EXT_gpu_shader4. OpenGL extension registry, 2007.

[71] W. E. Lorensen and H. E. Cline. Marching cubes: a high resolution 3d surface construction algorithm. *SIGGRAPH Comput. Graph.*, 21(4):163–169, 1987.

[72] A. Lu, C. J. Morris, D. S. Ebert, P. Rheingans, and C. D. Hansen. Non-photorealistic volume rendering using stippling techniques. In *IEEE Visualization*, pages 211–218, 2002.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

[73] X. Mao, Y. Nagasaka, and A. Imamiya. Automatic generation of pencil drawing from 2d images using line integral convolution. In *CAD/Graphics*, pages 240–248, 2001.

[74] L. Markosian, M. A. Kowalski, S. J. Trychin, L. D. Bourdev, D. Goldstein, and J. F. Hughes. Real-time nonphotorealistic rendering. *Computer Graphics*, 31(Annual Conference Series):415–420, 1997.

[75] D. Marr and E. Hildreth. Theory of edge detection. In *Proc. Royal Soc. London*, volume 207, pages 187–217, 1980.

[76] S. McCloud. *Making Comics: Storytelling Secrets of Comics, Manga and Graphics Novels*. Harper, 2006.

[77] G. Medioni, C. K. Tang, and M. S. Lee. Tensor voting: theory and applications. In *European Conference on Computer Vision*, 2002.

[78] M. Meyer, M. Desbrun, P. Schröder, and A. H. Barr. Discrete differential-geometry operators for triangulated 2-manifolds. *VisMath*, pages 35–57, 2002.

[79] M. Meyer, R. M. Kirby, and R. Whitaker. Topology, accuracy, and quality of isosurface meshes using dynamic particles. *IEEE Transactions on Visualization and Computer Graphics (Proceedings Visualization / Information Visualization 2007)*, 12(5), September-October 2007.

[80] M. Mignotte. Bayesian rendering with non-parametric multiscale prior model. In *Proceedings of ICPR02*, 2002.

[81] M. Mignotte. Unsupervised statistical sketching for non-photorealistic rendering models. In *ICIP (3)*, pages 573–576, 2003.

[82] O. Monga, S. Benayoun, and O. D. Faugeras. From partial derivatives of 3d density images to ridge lines. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 354–359, 1992.

[83] Z. Nagy and R. Klein. High-quality silhouette illustration for texture-based volume rendering. In *Proc. WSCG*, pages 301–308, 2004.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

[84] Z. Nagy, J. Schneider, and R. Westermann. Interactive volume illustration. In *VMV*, pages 497–504, 2002.

[85] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: analysis and an algorithm. In *NIPS*, pages 849–856, 2001.

[86] J. D. Northrup and L. Markosian. Artistic silhouettes: a hybrid approach. In *NPAR '00: Proceedings of the 1st international symposium on Non-photorealistic animation and rendering*, pages 31–37, New York, NY, USA, 2000. ACM Press.

[87] Y. Ohtake, A. Belyaev, and H. Seidel. Ridge-valley lines on meshes via implicit surface fitting. In *SIGGRAPH 2004*, pages 609–612.

[88] S. E. Palmer. *Vision Science: Photons to Phenomenology.* The MIT Press, 1999.

[89] O. M. Pastor, B. Freudenberg, and T. Strothotte. Real-time animated stippling. *IEEE Comput. Graph. Appl.*, 23(4):62–68, 2003.

[90] B. Pham. Expressive brush strokes. *CVGIP: Graph. Models Image Process.*, 53(1):1–6, 1991.

[91] J. D. Powell. The theory of radial basis function approximation. In *Cambridge University Numerical Analysis Report*, 1990.

[92] E. Praun, H. Hoppe, M. Webb, and A. Finkelstein. Real-time hatching. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, page 581, New York, NY, USA, 2001. ACM.

[93] T. Pudet. Real time fitting of hand-sketched pressure brushstrokes. *Comput. Graph. Forum*, 13(3):205–220, 1994.

[94] R. Raskar. Hardware support for non-photorealistic rendering. In *HWWS '01: Proceedings of the ACM SIGGRAPH/EUROGRAPHICS workshop on Graphics hardware*, pages 41–47, New York, NY, USA, 2001. ACM.

[95] R. Raskar and M. Cohen. Image precision silhouette edges. In *I3D '99: Proceedings of the 1999 symposium on Interactive 3D graphics*, pages 135–140, New York, NY, USA, 1999. ACM.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

[96] P. Rheingans and D. S. Ebert. Volume illustration: nonphotorealistic rendering of volume models. *IEEE Trans. Vis. Comput. Graph.*, 7(3):253–264, 2001.

[97] S. Rusinkiewicz. Estimating curvatures and their derivatives on triangle meshes. In *3DPVT 2004*, pages 486–493.

[98] S. Rusinkiewicz, D. DeCarlo, and A. Finkelstein. Line drawings from 3D models. In *Course Note of SIGGRAPH 2005*.

[99] T. Saito and T. Takahashi. Comprehensible rendering of 3-d shapes. *SIGGRAPH Comput. Graph.*, 24(4):197–206, 1990.

[100] M. Salisbury, C. Anderson, D. Lischinski, and D. H. Salesin. Scale-dependent reproduction of pen-and-ink illustrations. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 461–468, New York, NY, USA, 1996. ACM.

[101] M. P. Salisbury, S. E. Anderson, R. Barzel, and D. H. Salesin. Interactive pen-and-ink illustration. In *SIGGRAPH '94: Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 101–108, New York, NY, USA, 1994. ACM.

[102] M. P. Salisbury, M. T. Wong, J. F. Hughes, and D. H. Salesin. Orientable textures for image-based pen-and-ink illustration. *Computer Graphics*, 31(Annual Conference Series):401–406, 1997.

[103] S. Schein and G. Elber. Adaptive extraction and visualization of silhouette curves from volumetric datasets. *Vis. Comput.*, 20(4):243–252, 2004.

[104] J. Schreiner and C. Scheidegger. High-quality extraction of isosurfaces from regular and irregular grids. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1205–1212, 2006. Member-Claudio Silva.

[105] A. Secord. Weighted voronoi stippling. In *NPAR '02: Proceedings of the 2nd international symposium on Non-photorealistic animation and rendering*, pages 37–43, New York, NY, USA, 2002. ACM.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

[106] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[107] C. Soler, M. P. Cani, and A. Angelidis. Hierarchical pattern mapping. In *SIGGRAPH 2002*, pages 673–680.

[108] M. C. Sousa and J. W. Buchanan. Computer-generated graphite pencil rendering of 3D polygonal models. *Computer Graphics Forum*, 18(3):195–208, 1999.

[109] M. C. Sousa and J. W. Buchanan. Observational models of graphite pencil materials. *Computer Graphics Forum*, 19(1):27–49, March 2000.

[110] M. C. Sousa, A. Gooch, and B. Gooch. Illustrative scientific visualization framework. In *Eurographics Workshop on Computational Aesthetics*, 2005.

[111] M. C. Sousa and P. Prusinkiewicz. A few good lines: suggestive drawing of 3d models. *Comput. Graph. Forum*, 22(3):381–390, 2003.

[112] S. Strassmann. Hairy brushes. In *SIGGRAPH '86: Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, pages 225–232, New York, NY, USA, 1986. ACM Press.

[113] T. Strothotte and S. Schlechtweg. *Non-photorealistic computer graphics: modeling, rendering, and animation.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2002.

[114] G. Stylianou and G. Farin. Crest lines for surface segmentation and flattening. In *IEEE Transactions on Visualization and Computer Graphics 10, 5 (September/October)*, pages 536–544, 2004.

[115] S. L. Su, Y. Xu, H. Shum, and F. Chen. Simulating artistic brushstrokes using interval splines. In *Proceedings of the 5th IASTED International Conference on Computer Graphics and Imaging (CGIM '02, Kauai, Hawaii, August 2002)*, pages 85–90, 2002.

[116] N. A. Svakhine and D. S. Ebert. Interactive volume illustration and feature halos. In *Pacific Graphics*, pages 347–354, 2003.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

[117] D. Teece. Ink line rendering for film production. In *Theory and practice of Non-Photorealistic Graphics: Algorithms, Methods and Production Systems, SIGGRAPH Course Notes*, 2003.

[118] J. B. Tenenbaum and W. T. Freeman. Separating style and content with bilinear models. *Neural Computation*, 12(6):1247–1283, 2000.

[119] H. Todo, K. Anjyo, W. Baxter, and T. Igarashi. Locally controllable stylized shading. In *SIGGRAPH '07: ACM SIGGRAPH 2007 papers*, page 17, New York, NY, USA, 2007. ACM.

[120] Y. Uralsky. DX10:practical metaballs and implicit surfaces. In *GameDevelopers conference*, 2007.

[121] Luiz Velho. Simple and efficient polygonization of implicit surfaces. *J. Graph. Tools*, 1(2):5–24, 1996.

[122] A. H. Vermeulen and P. P. Tanner. Pencilsketch-a pencil based paint system. In *Graphics Interface '89*, pages 138–143, 1989.

[123] I. Viola, M. E. Gröller, M. Hadwiger, K. Bühler, B. Preim, M. C. Sousa, D. S. Ebert, and D. Stredney. Illustrative visualization. In *IEEE Visualization*, page 124, 2005.

[124] B. Wang, W. Wang, H. Yang, and J. Sun. Efficient example-based painting and synthesis of 2d directional texture. *IEEE Transactions on Visualization and Computer Graphics*, 10(3):266–277, 2004.

[125] H. Wang, Y. He, X. Li, X. Gu, and H. Qin. Polycube splines. In *SPM '07: Proceedings of the 2007 ACM symposium on Solid and physical modeling*, pages 241–251, New York, NY, USA, 2007. ACM Press.

[126] M. Webb, E. Praun, A. Finkelstein, and H. Hoppe. Fine tone control in hardware hatching. In *NPAR '02: Proceedings of the 2nd international symposium on Non-photorealistic animation and rendering*, pages 53–ff, New York, NY, USA, 2002. ACM.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

[127] L. Wei and M. Levoy. Fast texture synthesis using tree-structured vector quantization. In *SIGGRAPH 2000*, pages 479–488.

[128] T. Whitted. Anti-aliased line drawing using brush extrusion. In *SIGGRAPH '83: Proceedings of the 10th annual conference on Computer graphics and interactive techniques*, pages 151–156, New York, NY, USA, 1983. ACM Press.

[129] J. Willats. *Art and Representation:New Principles in the Analysis of Pictures*. Princeton:Princeton University Press, 1997.

[130] J. Willats and F. Durand. Defining pictorial style: lessons from linguistics and computer graphics. In *Axiomathes*, 2004.

[131] G. Winkenbach and D. H. Salesin. Computer-generated pen-and-ink illustration. In *SIGGRAPH '94: Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 91–100, New York, NY, USA, 1994. ACM.

[132] G. Winkenbach and D. H. Salesin. Rendering parametric surfaces in pen and ink. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 469–476, New York, NY, USA, 1996. ACM.

[133] Z. Wood, P. Schröder, D. Breen, and M. Desbrun. Semi-regular mesh extraction from volumes. In *VIS '00: Proceedings of the conference on Visualization '00*, pages 275–282, Los Alamitos, CA, USA, 2000. IEEE Computer Society Press.

[134] Q. Wu and Y. Yu. Feature matching and deformation for texture synthesis. In *SIGGRAPH 2004*.

[135] X. Xiao, H. S. Seah, Z. Wu, F. Tian, and X. Xie. Interactive free-hand drawing and in-between generation with disk bspline curves. In *Conference on Multimedia Arts Asia Pacific, MAAP, ISBN:981-05-2187-1*, 2004.

[136] C. Yao and T. Lee. Adaptive geometry image. *IEEE Transactions on Visualization and Computer Graphics*, 2008.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

[137] C. Yen, M. Chi, T. Lee, and W. Lin. Stylized rendering using samples of a painted image. *IEEE Transactions on Visualization and Computer Graphics*, 14(2):468–480, 2008.

[138] S. X. Yu and J. Shi. Multiclass spectral clustering. In *ICCV '03: Proceedings of the Ninth IEEE International Conference on Computer Vision*, page 313, Washington, DC, USA, 2003. IEEE Computer Society.

[139] X. Yuan, M. X. Nguyen, N. Zhang, and B. Chen. Stippling and silhouettes rendering in geometry-image space. In *Proceedings of Eurographics Symposium on Rendering (EGSR'05)*, pages 193–200, color plate 318. the Eurographics Association, 2005.

[140] A. Yuille. Zero crossings on lines of curvature. *Graphical Models and Image Processing*, 45:68–87, 1989.

[141] O. Zander, T. Isenberg, S. Schlechtweg, and T. Strothotte. High quality hatching. *Computer Graphics Forum (Proceedings of EuroGraphics 2004)*, 23(3):421–430, 2004.

[142] K. Zhou, M. Gong, X. Huang, and B. Guo. Highly parallel surface reconstruction. In *Microsoft Technical Report, MSR-TR-2008-53*, 2008.

[143] K. Zhou, Q. Hou, R. Wang, and B. Guo. Real-time kd-tree construction on graphics hardware. In *ACM TOG(SIGGRAPH Asia 2008)*, 2008 to appear.

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

Publications

Journal papers

1. **Xuexiang Xie**, Ying He, Feng Tian, Hock-Soon Seah, Xianfeng Gu and Hong Qin. "An effective illustrative visualization framework based on photic extremum lines(PELs)". *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1328-1335, 2007.

2. **Xuexiang Xie**, Feng Tian and Hock-Soon Seah. "Style learning with feature based texture synthesis". *ACM Computers in Entertainment (CIE)*. Accepted. 2008.

Conference papers

1. **Xuexiang Xie**, Feng Tian and Hock-Soon Seah. "Feature guided texture synthesis for artistic style transfer". *Second International Conference on Digital Interactive Media in Entertainment and Arts*, pp. 44-49, 2007.

2. **Xuexiang Xie**, Feng Tian, Hock-Soon Seah, Zhongke Wu and Konstantin Melikhov. "Pencil drawing for NPR rendering and animation". *International Conference on Computer Animation and Social Agents*, Geneva, July 2006.

3. Konstantin Melikhov, Feng Tian, **Xuexiang Xie** and Hock-Soon Seah. "DBSC-based pencil style simulation for line drawings", *CyberGames'06: Proceedings of the 2006 international conference on Game research and development*, pp.17-24, 2006.

4. John Lewis, Nickson Fong, **Xuexiang Xie**, Hock-Soon Seah and Feng Tian. "More

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

optimal strokes for NPR sketching". *GRAPHITE'05: Proceedings of the 3rd international conference on Computer graphics and interactive techniques in Australasia and South East Asia*, pp. 47-50, 2005.

5. **Xuexiang Xie**, Feng Tian, Hock-Soon Seah and Zhongke Wu. "Texture space stylization for NPR rendering and animation", *Proceedings of Computer Graphics, Imaging and Vision*, pp. 178-184, Beijing, China, July 2005.

6. **Xuexiang Xie**, Feng Tian, Hock-Soon Seah and Zhongke Wu. "Silhouette extraction and stylization for Non-Photorealistic rendering". *Conference on Multimedia Arts Asia Pacific, MAAP, ISBN:981-05-2187-1*, October, 2004.

7. Zhongke Wu, Hock-Soon Seah, Feng Tian, Xian Xiao and **Xuexiang Xie**. "Simulating artistic brush strokes using disk B-spline curves". *Conference on Multimedia Arts Asia Pacific, MAAP, ISBN:981-05-2187-1*, October, 2004.

8. Xian Xiao, Hock-Soon Seah, Zhongke Wu, Feng Tian and **Xuexiang Xie**. "Interactive free-hand drawing and in-between generation with disk B-spline curves". *Conference on Multimedia Arts Asia Pacific, MAAP, ISBN:981-05-2187-1*, October, 2004.