NPR: Art Enhancing Computer Graphics

Technical Report TR-2004-17
Laura G. Tateosian, Christopher G. Healey

Knowledge Discovery Lab
Department of Computer Science, North Carolina State University
Raleigh, NC 27695-8207

Email: lgtateos@unity.ncsu.edu

Abstract

Nonphotorealistic rendering is a field in computer science in which scientists apply artistic techniques to enhance computer graphics. This paper addresses the interrogatives what, how, and why, about NPR. The discussion expands on *what* NPR is and what kinds of projects are being done in NPR, specifically it focuses on three issues: two large problems in NPR, simulating pen-and-ink illustration and simulating painting, and last the application of NPR to visualization. Exploring these topics thoroughly provides some specific answers to *how* these effects are accomplished. Throughout the paper various motivations for using NPR are revealed, including the application of NPR to visualization (as evidence of *why*). Our lab is interested in applying NPR techniques to visualization, so the paper concludes with some conjecture on how to verify the efficacy of this goal.

1 Introduction

"Knowledge and techniques long used by artists are now being applied to computer graphics to emphasize specific features of a scene, expose subtle attributes, and omit extraneous information, (giving) rise to a new field. [GG01]" Gooch and Gooch are referring to the field of nonphotorealistic rendering (NPR). As the name implies, scientists in NPR create algorithms to display information in styles other than realism. Figure 1 gives a brief illustration of the Goochs' remark. In Figure 1b artistic enhancements were applied to the standard volume rendering in Figures 1a, resulting in a more informative, nonphotorealistic image, revealing the fibrous complexity of the tomato's structure [RE01]. (Similar enhancement techniques applied to medical imagery are discussed in Section 4).

[KP02]

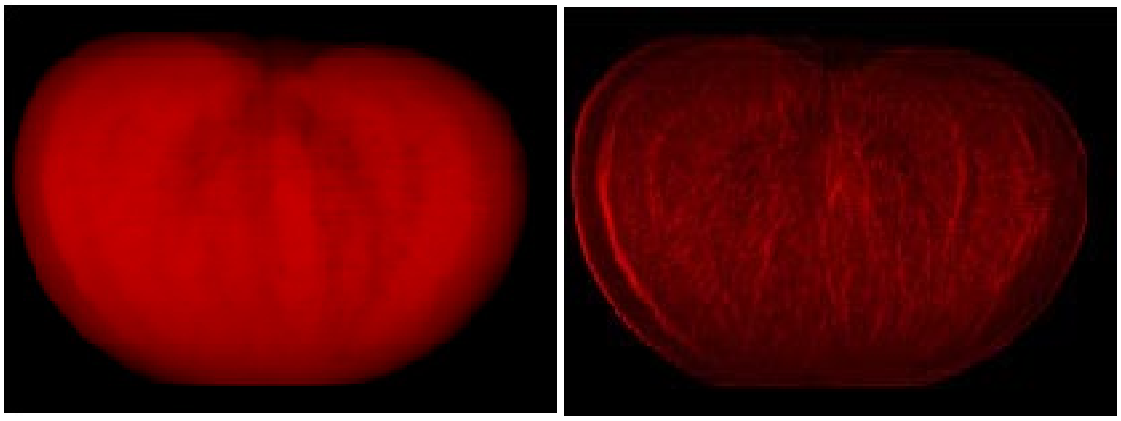

(a) (b)

Figure 1: Volume visualizations of a tomato: (a) standard volume rendering; (b) rendering with silhouette and boundary enhancements.

In general, the process of nonphotorealistic rendering can be thought of as having two stages. First, some kind of input imagery is chosen. Second, the imagery is processed and rendered in some artistic style. The diversity in NPR arises from the choices made in these stages.

Photographic images may spring to mind immediately, as input imagery for drawing program filters, such as embossing or charcoal sketching, which are, in fact, rudimentary NPR systems. Some NPR systems take color photographs or grayscale images (made up of tones of gray) as input, while others work on three-dimensional models or animations or video input. The input for visualization systems, discussed in Section 4, are large sets of scientific data. The input depends on the desired output of the program. "Jigsaw Image Mosaics," for example, takes as input monochromatic shapes and a database of small icons [KP02]. The system tiles the shapes with icons of similar colors as shown in Figure 2. Creative input may be required, depending on how interactive the system is. Though some systems are automated, others are designed as artistic tools, providing an alternative to real canvases and paint brushes, for example, as in Haeberli's "Paint by Numbers" system, described in Section 3, where users can "paint" with a mouse [Hae90]. Alternative forms of user-interaction are being explored. In one system eye-tracking records the user's eye movements to determine how to abstract the image. The image is painted with greatest detail where fixations gather [DS03] [SD02]. DAB uses a haptic paintbrush device to allow the user to "paint" on a virtual canvas [BSMM01].

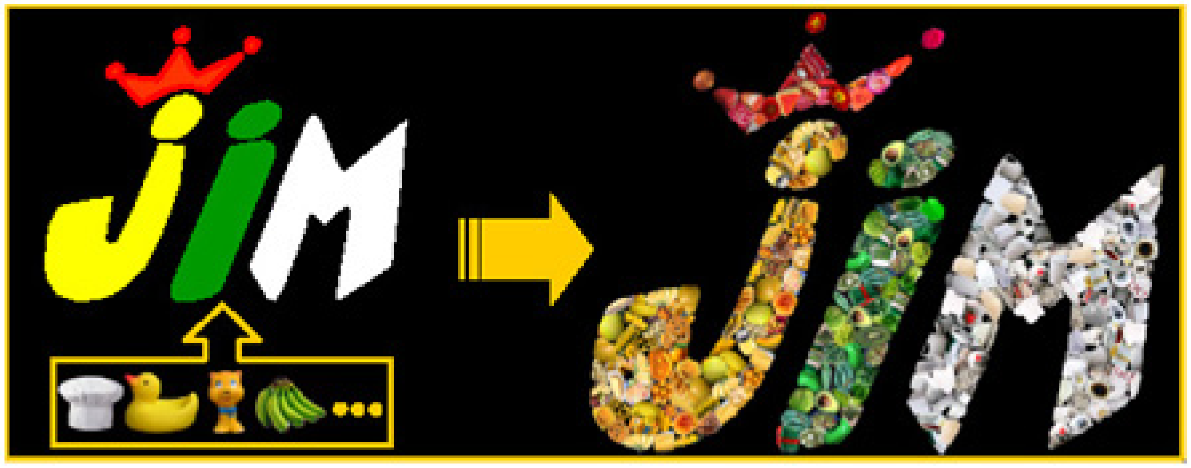


Figure 2: The Jigsaw Image Mosaic uses monochromatic containers and tiles them with small images of similar colors: Left, the letters J, I, and M, are containers; Right, J, I, and M are tiled like a mosaic.

How the input is processed depends not only on the style of art being simulated, but on whether the focus is on simulating the act of creating artwork, the physical behavior of the media, or the visual characteristics of an artistic style. "Paint by Numbers" and the programs described in section 3.1 fall into the first category, composing a collection of brush strokes as an artist does. The second category refers to systems that model the physical properties of the pencil, paper, paint, or whatever artistic materials are being simulated. For example, Sousa and Buchanan created an elaborate system for simulating pencil drawing by studying the microscopic interaction of pencil lead, blenders, erasers, and paper. Systems modeling watercolor and oil paint are described in Section 3.3. Researchers simulating pen-and-ink illustration began with the third approach, compiling a list of pen-and-ink illustration principles based on literature on pen-and-ink [WS94] (see Section 2.1).

The examples in this paper come from three areas in NPR, simulating pen-and-ink illustration, simulating painting, and the application of NPR to visualization, the papers in these areas are diverse collection of sources that facilitates discussion of the field of NPR and the topics described above. A great deal of work has been done in both simulating pen-and-ink and painting, so we can study the problems they have solved; also, the systems are well-developed, so they supply some impressive results. They also provide a basis to discuss the motivations for nonphotorealistic rendering. Research in our lab has investigated the application of nonphotorealistic "painted" visualization. My focus here also reflects my interest in pursuing future projects in this area.

2 Pen-and-Ink Drawing

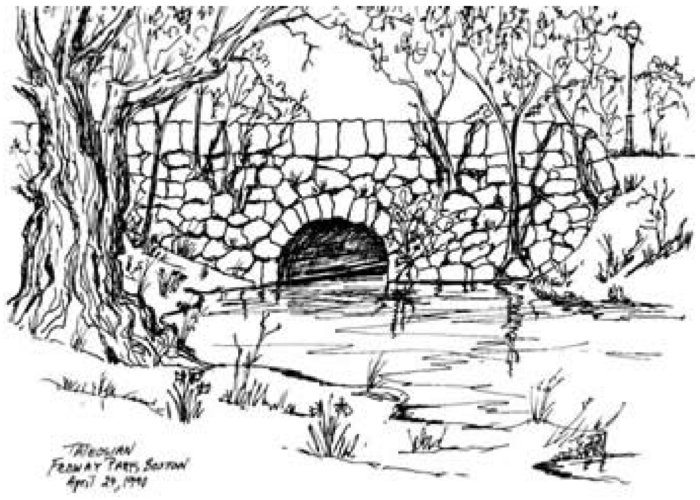


Figure 3: Hand drawn pen-and-ink illustration by artist Sarah Tateosian.

The illustration above shows the basic visual elements of pen-and-ink line drawing: outlining, shading, and texturing. Outlines are used judiciously and expressively and so may be incomplete or imperfect. They often indicate an object's shape rather than explicitly describing every detail. The thickness and direction of outlines may vary depending on the material's properties or the feeling they characterize. Discontinuities may occur in the outlines simply to generate more interest. Since the ink itself doesn't vary in tone, shading is a cumulative effect of multiple strokes laid together. Hatching (loosely drawn patches of parallel lines), for example, is a common shading technique in which darkness is directly proportional to the proximity of the lines. I'm using the third term above, texturing, to describe the lines that indicate the shapes and surface properties of objects. Textures could also be viewed as outlines on a finer scale, as the lines outlining rocks add texture to the bridge in the drawing above or the lines on the tree in the foreground outline the wrinkles on the trunk.

In certain applications, such as architectural drawing, medical textbooks, and instructional manuals, pen-and-ink drawings are frequently preferred over photorealistic images. Architects often employ a sketched first draft because it can convey the character of the design without focusing on details [SS02]. Textbooks and manuals can use black and white images to avoid reproduction artifacts that can be introduced by monitor and printing device color gamuts' inconsistencies. Also, as [LS95] points out, a photograph of an engine is of little use to a mechanic who is already looking at the real thing.

Rather than discussing related work chronologically, I have chosen to focus on some interesting aspects of simulating pen-and-ink; papers are discussed in the context of their relevance to the issues. Although some of the work is cumulative, the order in which it developed is not as important as the problems they were solving. The following sections address how scientists have approached simulating pen-and-ink in general ("Principles in Practice"), as well as some more specific sub problems: simulating stroke textures, rescaling pen-and-ink images, and simulating stippling, a style of pen-and-ink illustration. These topics provide the basis to discuss the most interesting applications in this area.

2.1 Principles in Practice

A research group at the University of Washington made some of the earliest and most significant contributions to pen-and-ink simulation. They approached the problem as a student of art might; they began by studying the principles of traditional pen-and-ink illustration. Then they devised methods consistent with these principles. In one project they created a system that takes polygonal models as input and renders them in pen-and-ink style (see Figures 4) [WS94]. One principle mentioned above is exhibited in this figure. Large areas of consistent texture in Figure 4a are textured only sparingly, i.e., “indication” is used in Figure 4a. The system implements indication by allowing the user to designate detail segments, drawn under the rim of the roof, around the windows, etc. Then the image is drawn with attenuating detail as distance from these line segments increases. Less is more here, not only because fewer strokes need to be used, but the image is made more interesting (compare to Figure 4b in which each blade of grass and each brick is drawn).

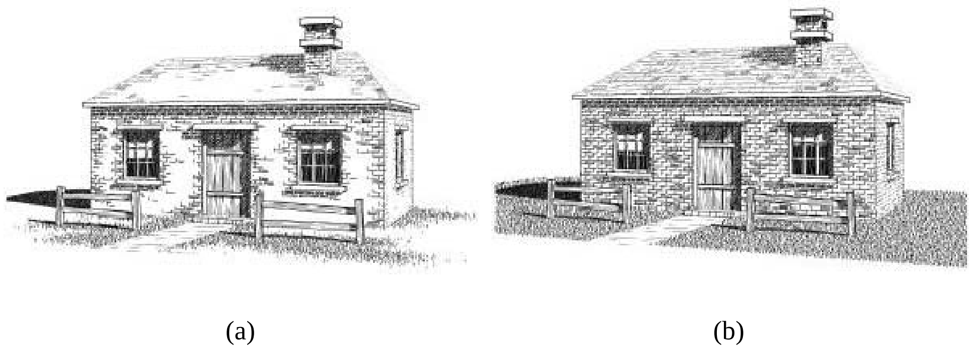

(a) (b)

Figure 4: Pen-and-ink illustrations of an architectural model: (a) the house drawn with “indication” (detail decreases as distance from the detail segments increases); (b) the same drawing without indication. Every brick and blade of grass are drawn.

2.2 Stroke Textures

Generating stroke textures, sets of strokes which control the tone and add texture, is a fundamental subproblem in simulating pen-and-ink. Jodoin et al. focus on hatching and approach the problem as a kind of texture synthesis [JEGPO02]. Given a sample hatching, a manually drawn set of strokes, their system can generate a new set of strokes which is visually similar to the sample. They use a statistical measure of similarity based on a subset of the previous strokes.

The interactive drawing system of Salisbury et al. allows users to select from stored stroke textures or to create them [SABS94]. The stored textures are simply a collection of images which includes each texture drawn in several tone values. Otherwise, the system allows the user to add to this library by creating a new patch from stippling, parallel hatching, or curved strokes.

Another system from the University of Washington uses “orientable textures” [SWHS97]. The system provides a set of stroke texture samples and the user chooses a stroke texture type and a direction field for each region in a grayscale image. The user can edit the tone and the direction field. The system adds strokes from the specified stroke texture set until the desired tone is achieved. The tone is approximated by the system after each stroke is added by comparing the tone image to a blurred version of the illustration pixel-by-pixel. Strokes are positioned at points which have accumulated the least amount of their intended darkness, so that shading occurs consistently. Strokes are drawn so that the control hull follows the target direction field. This means that the curve itself may not pass through the target direction field, but it is a close enough approximation. The raccoon in Figure 5 was created with this system.

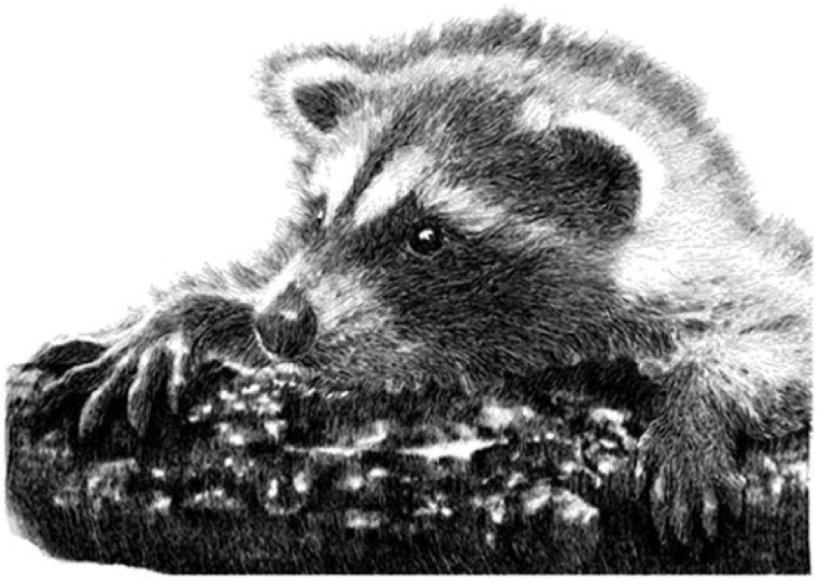


Figure 5: A raccoon created with the orientable textures developed by Salisbury et al.

In [WS96] the authors expanded their system in [WS94] to not only take flat polygonal models as input, but to illustrate free-form surfaces described parametrically. Their “controlled density hatching” system allows them to control texture and tone simultaneously. Individual strokes vary in width along their length based on the distance between the strokes when projected onto the two-dimensional image. This allows for control of tone, not only to enhance the image, but also to avoid unexpected and undesired darkening where strokes grow close together, like in Figure 6a. Figure 6b shows a sphere drawn with controlled density hatching to correct the problem.

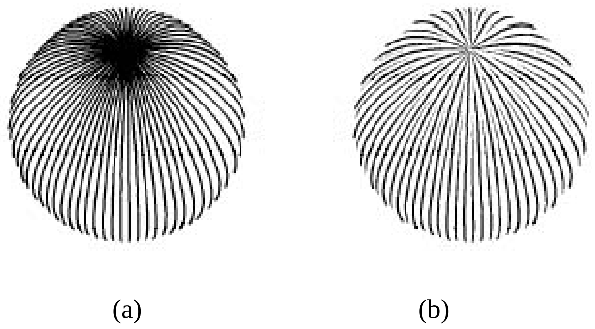


Figure 6: Pen-and-ink spheres: (a) undesired darkness appears where the strokes gather; (b) controlled density hatching solves the problem.

2.3 Rescaling

Enlarging or shrinking images can result in undesired changes in tone and sharpness. In [WS94], Winkenbach and Salesin address this issue by assigning a hierarchical structure to textures, so that the amount of detail drawn varies with the size of the image. Lower priority elements of a texture are not drawn when the image is small, so that the strokes do not become too crowded and appear only dark instead of defining the object. Figure 7 shows a brick texture drawn at different scales. The amount of texture drawn on the bricks decreases with the size and only the brick outlines remain in the smallest image.

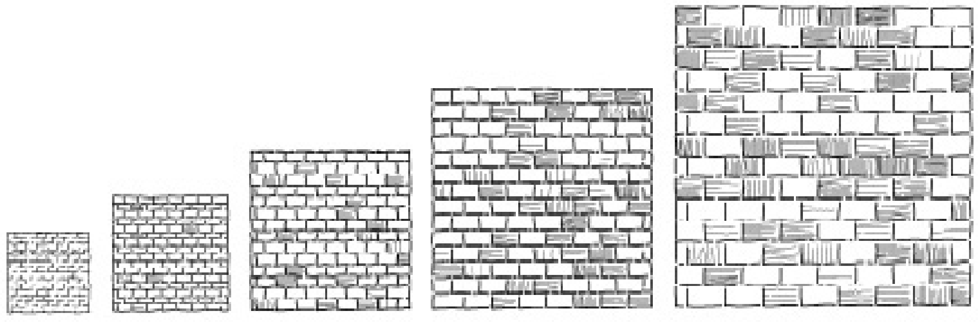


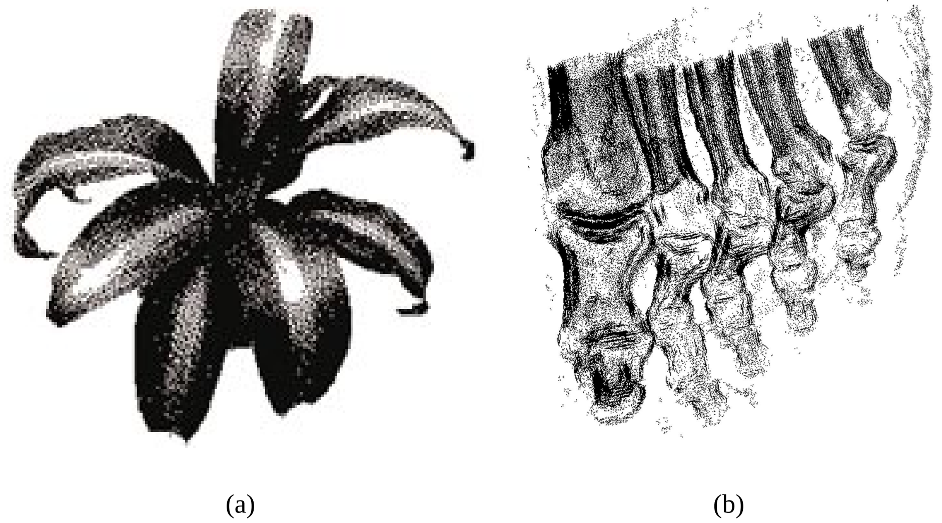
Figure 7: The system developed by Winkenbach and Salesin are made up of prioritized structures. Here the brick textures vary as they are resized.

Salisbury et al. maintain the tone of a pen-and-ink drawing by storing the underlying grayscale image, resizing this image, and drawing a new set of strokes [SALS96]. When the resolution is changed or the image is enlarged, areas with sharp discontinuities can become blurred. In the underlying grayscale image, pixels on the edge of contrasting areas may contain an average of these areas' grayscale values and hence the edge in the pen-and-ink image becomes fuzzy as this area is enlarged. To preserve sharpness, a list of discontinuity edges that mark sharp discontinuities in the image is created and the image is resampled. Tonal values are determined for a pixel by inspecting its neighbors. Distant neighbors have less influence than close ones. Distance is affected by the intervening discontinuity edges. Some neighbors may even be unreachable because of discontinuity edges and so will have no influence on the tone for this pixel.

2.4 Stippling

Stippling is a style of pen-and-ink illustration which uses very few, if any, lines; instead, dots define and shade objects. To avoid unintended patterns or tonal variations, stipple placement should be locally irregular, though bunching and sparseness should be deliberate. These are surprisingly difficult results to achieve by hand. Computer stippling simulation systems need to determine placement and in some cases the size of the stipples.

The common struggle between speed and image quality arises. Adrian Secord describes two methods; one that can take 20 minutes to produce high quality stipplings of grayscale images; the other can stipple animations [Sec02]. The first method initially distributes a set of stipple points. Then the Voronoi diagram partitions the space into cells, each containing one stipple. The center of mass of each cell is then found. Finally, each stipple is moved to the center of mass in its cell and the resulting arrangement of stipples is the stipple drawing. The grayscale tone of the underlying region is used to calculate the centers of mass, so that the stipples become tightly packed in darker areas and sparser in light areas. The plant in Figure 8 was created with this method. The second method pieces together patches of stipple samples. The samples vary tone by varying stipple density and are selected to match the values of the underlying grayscale image. This method achieves interactive rates, but the quality is lower and frame-to-frame incoherence causes shimmering.



(a) (b)

Figure 8: Computer-generated stipple drawings: (a) plant stippled with Secord's high quality method (20,000 stipples); (b) stippled volume visualization with interior enhancement and silhouette curves created by Lu et al.

The speed of point rendering makes stippling an attractive means to render volume information. Grigoryan and Rheingans used multi-colored stippling, to show uncertainty on the surface of growing tumors [GR02]. This property is important for cancer researchers to see, since the growth of tumors is not well understood. Sparsely placed points create visual fuzziness, conveying uncertainty intuitively. Lu et al. stipple-renders volumes, as well, using a more traditional black and white pen-and-ink stippling where stipples are carefully placed and sized to control the local and overall tone [LME$^+$02]. Preprocessing distributes an initial set of stipples and computes gradient properties. Then the volume is drawn with a subset of the stipples in this pre-generated list to preserve frame-to-frame coherence. The number of stipples per voxel is the product of the precomputed maximum number per voxel and the enhancement factor. The size of the stipples in a voxel is the user-specified maximum size times the voxel gradient's magnitude. The enhancement factor is the product of several enhancement values: the boundary and silhouette factor increases the number of stipples in these areas; the resolution factor increases the stipples used as the image moves closer to the viewpoint; the use of the distance factor causes more distant parts to contain fewer points; the use of interior enhancement causes areas further from the center to be more transparent; and finally, lighting enhancements only allow stipples to be drawn in voxels oriented away from viewer. Silhouette curves can also be added to sharpen the outlines and interior curves of objects. Figure 8 shows some volumes rendered with some of these enhancements.

3 Painting

Haeberli presented the idea of using a list of paint strokes to create nonphotorealistic imagery, in effect, simulating the traditional process of painting [Hae90]. His system takes a photographic source image or a 3-dimensional scene and allows the user to interactively paint the scene by clicking to indicate stroke locations. Each click creates a stroke whose characteristics, with the exception of color, are chosen by the user. A stroke is specified by its location, color, shape or style, size, and direction. The system is called “Paint by Numbers” because the color of the stroke is determined by the color of the underlying source image at that location. Stroke styles include curved, pointillist, and polygonal. The mouse speed or the keyboard can be used to control the stroke's size. Stroke direction can be controlled by the user or by gradients. The system provides enhancement techniques often used by artists such as increasing saturation, exaggerating edges between dark and light areas by making dark areas darker and light areas lighter, and adding noise to flat colored regions.

Haeberli's work raises several questions about simulating painting; the sections below discuss how various scientists have addressed these questions:

1. How can the brush stroke composition paradigm be extended to produce more sophisticated paintings, to require less user interaction, or to accept different forms of input imagery? (Section 3.1)

2. How can the brush strokes be made more expressive or appear more realistic? (Section 3.2)

3. What other approaches apart from composing lists of brush strokes, can be taken to simulate painting? (Section 3.3 and Section 3.4)

3.1 Brush Stroke Compositions

Haeberli's work provided a starting point for many other painting systems which use stroke lists and stroke attributes, position, color, shape, style, and size. Subsequent systems differ in the manner in which these attributes are chosen and vary greatly in their results. They expand on the basic idea he presented by exploring new ways to paint with less user-interaction, new ways to make the strokes more expressive, and by allowing for different types of input, such as animations and videos. Several such systems are discussed here.

Meier created painterly renderings of animation using lists of brush strokes [Mei96]. The main challenge of nonphotorealisticly rendering animation is coherence, since painting strokes should appear to be randomly distributed, but randomness in animation causes an undesirable popping effect. Here a set of particles distributed on the surfaces of objects in the images determine the positions of the strokes. The system uses the geometry, surface attributes, and lighting of a scene to determine the color, orientation, and size of each stroke. An image of transparency values mapped onto the stroke determines the stroke style. The system allows the user to make decisions about the lighting, color, and brush strokes characteristics, then the system positions and draws the strokes automatically. The results are Impressionistic style paintings like the one shown in Figure 9a. Here abstraction allows the image to portray haystacks without rendering each thread of hay.

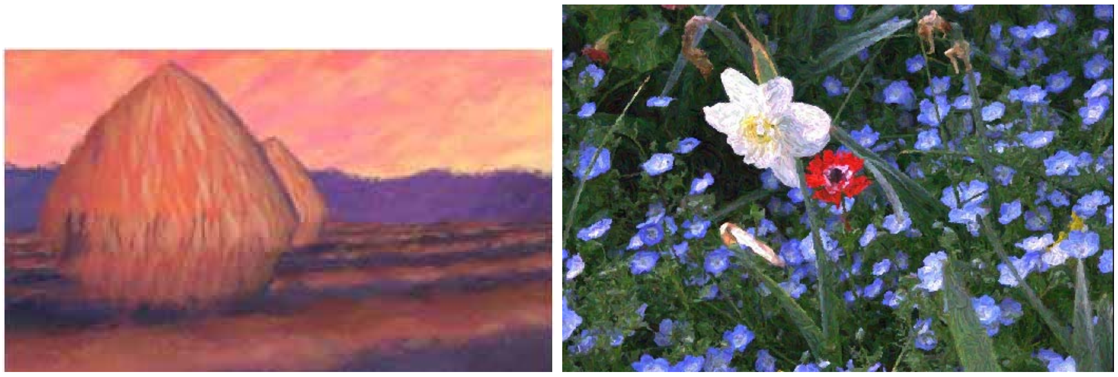

(a) (b)

Figure 9: Computer-generated paintings based on different types of input: (a) based on three-dimensional model animations (by Meier); (b) based on video sequences (by Litwinowicz).

Litwinowicz describes a technique for "painting" video sequences in Impressionistic style (see Figure 9b) [Lit97] . The system creates a list of strokes, storing position, color, orientation, radius, and length attributes for each stroke, until the image is covered. Edge detection seeks outlines in the image and strokes are clipped to conform to object shapes. Optical flow methods are used to move strokes from frame to frame. In some areas strokes are pushed together, causing overcrowding, which slows processing. Thus, some strokes are eliminated where their centers grow too close together. Other regions can become too sparsely covered, as new visual elements come into the scene so new strokes are added.

Hertzmann’s automated painting uses curved brush strokes applied in a series of layers [Her98]. A rough rendition is painted first and layers with progressively finer detail are laid on top, as artists sometimes paint undercoating to define the basic shapes of elements in the scene and add finer detail in subsequent layers. Given a reference image and a set of brush radii, the system paints one layer for each radius. The highest radius, which produces the coarsest image, is used first. A Gaussian blurred version of the reference image is compared with the canvas. Then strokes are placed in the current layer where the difference exceeds the approximation threshold. The strokes are anti-aliased cubic B-splines whose control points are positioned by the luminance gradients of the underlying image. A number of style parameters can be varied to create different renderings of the same input image. The lizard in Figure 10a is painted in Impressionistic style by using a large approximation threshold. Increasing the curvature filter exaggerates the curvature of the strokes to create an Expressionistic style like in Figure 10b. The color wash look in Figure 10c was created by increasing the paint opacity.

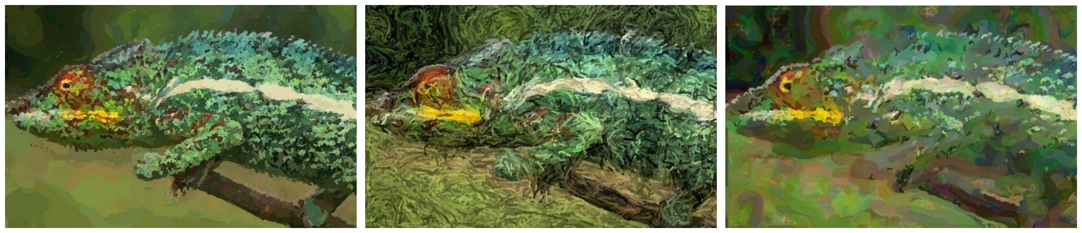
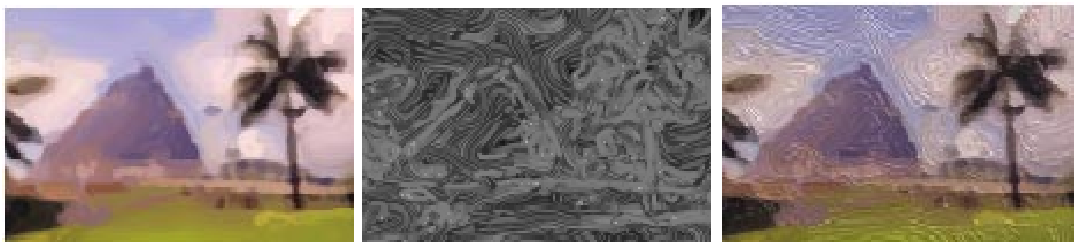


Figure 10: The lizard is painted in three styles with Hertzmann’s curved strokes: (a) Impressionist lizard; (b) Expressionist lizard; (c) Color wash lizard.

Standing close to an oil painting hanging in a museum, you may notice that the surface of the paint is not flat. Chunks of paint dry just as the brush left them when the strokes were painted. These undulations in the paint catch the light and subtly add to the overall effect of the image viewed from further away. Painting programs often simulate the texture of each individual stroke with a texture mapped to the surface of each stroke [Mei96], [Lit97], [Her98], [GCS02]. In [Her02] Hertzmann described a method that improves on this technique, by simulating the effect of paint moving as new strokes are painted over existing strokes. The system renders a set of curved brush strokes as described in [Her98]. Each stroke is given a texture map, also considered the height map of the stroke; the grayscale values of the texture map are used as height values (with black = zero height and lightest gray = highest). As the strokes are laid, a height value is computed at each pixel of the stroke as the sum of the stroke’s height value at that pixel and a fraction of the number of strokes already laid there. The resulting height map for the image is then lit with Phong shading. Figures 11a-c show a painted version of a photographic input image, a height field, and the resulting image.

Using the underlying source image values of the pixel at the center of the stroke is a simple approach to assigning the parameters of a brush stroke in a simulated painting. Shiraishi and Yamaguchi describe a method which incorporates more information from the image by considering the surrounding neighborhood, as well [SY99] [SY00]. The system uses an image



(a) (b) (c)

Figure 11: Three steps in Hertzmann's fast paint texture system: (a) paint the image; (b) create a height field image; (c) bump-map the image and light to create the textured result shown here.

moment function in two contexts to calculate the parameters of a rectangle that approximates an object in a grayscale image. First, it is used to calculate the size of a set of proposed brush strokes. For each pixel in the input image, the system creates a grayscale difference image by calculating the difference between the color of a square of the input image centered at that pixel and a monochrome square of the color of that pixel. The lightest areas in the difference image are closest in color to that pixel. The dimensions of a rectangular stroke that approximates the shape of these areas are calculated by the image moment function. The collection of rectangles are then used to calculate a set of possible stroke locations. The difference image is found at each of these locations. Then at each of these locations, the image moment function is used again to find a stroke and its parameters: length, width, orientation, and location (color is set to that of the location used to generate it). This final stroke list is sorted by size and strokes are drawn in order from largest to smallest, so that small strokes are on top, showing fine detail. The size of the grayscale difference image, i.e., the size of the neighborhood to which a point is compared, is an s-by-s square, where s is a user-selected variable that also controls the maximum stroke size (See Figure 12a in which the dog is painted with s = 25 and 11048 strokes.)

Another way of taking a more global approach is described by Gooch et al. in [GCS02]. The system works by segmenting an input image into regions, as an artist might do with a pencil, and finding the medial axes or backbones of the segments to determine how to paint them. The strokes are textured B-spline curves or textured sets of quadrilaterals, depending on the method used to approximate the medial axes. The system allows for underpainting to be used and provides paint mixing between the layers (see Figure 12b).

3.2 Brush Strokes

The early systems above used a fairly rudimentary textured rectangular brush stroke, mostly focusing on how to choose brush stroke attributes and how to compose the image with the list of brush strokes, while ignoring the nuances of strokes which in real paintings are affected by how much pressure the artist applies, how much paint is on the brush, and other factors. The systems described below focus on the simulation of the brush strokes and addressing these

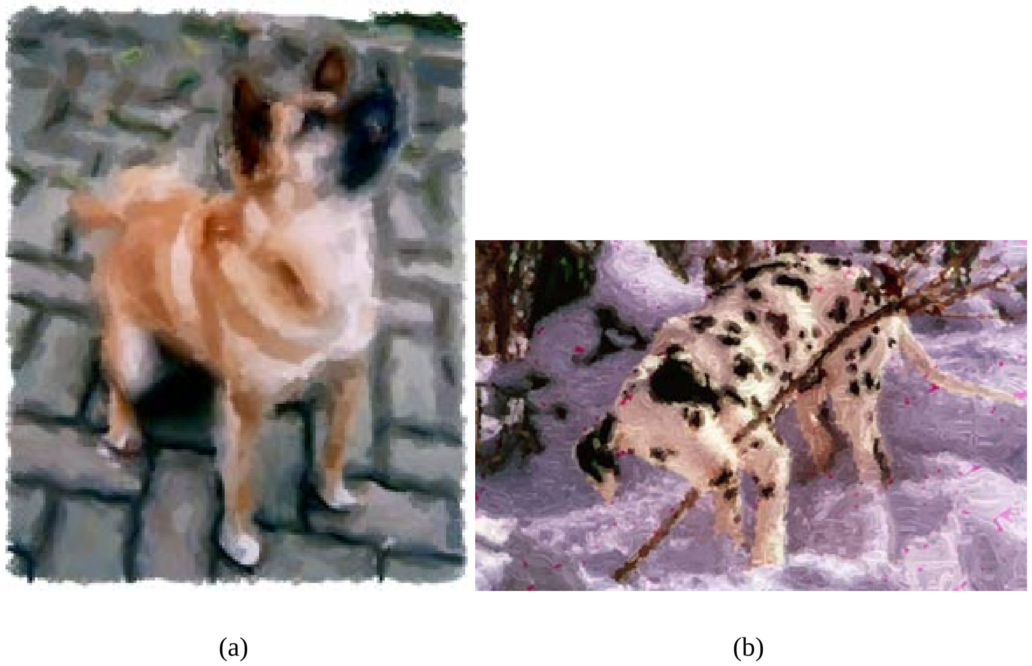


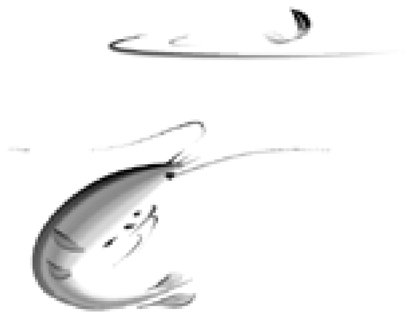
Figure 12: Dogs painted with two region-based methods: (a) image moment dog by Shiraishi and Yamaguchi with maximum stroke size 25 and a total of 11048 strokes; (b) medial axes dog with a pink undercoating by Gooch et al.

issues.

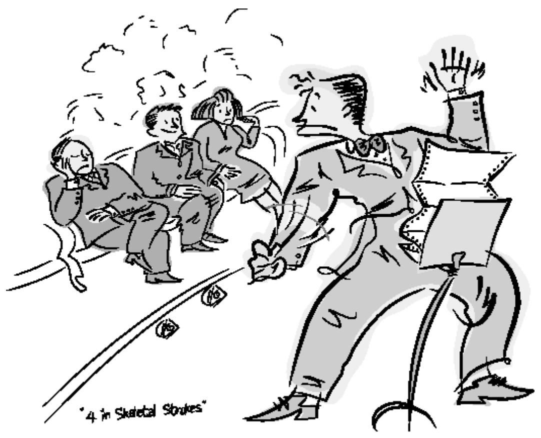
In [Str86], Strassmann describes a technique for simulating brush strokes of a style inspired by a bokbotsu sumi-e, a Japanese painting technique. The style features scenes composed of only a few brush strokes of black ink applied to light canvas (See Figure 13a). The system allows a user to create grayscale brushstrokes by specifying control points and pressure values at each point. Control points specify positions that affect the shape of the curve. Pressure values represent how much pressure the painter is applying to the brush as it moves through the stroke and so a pressure value controls the width of the stroke at that point. The system is object-oriented with brush, bristle, stroke, dip, and paper objects. After each stroke is drawn the virtual brush is dipped back in the paint and the dip object stores the amount of paint on the brush. The amount of ink on the bristles decreases as the stroke is drawn which can cause the tone to vary across the stroke. To render the stroke, the system uses a cubic B-spline curve which approximates the position values and interpolates the pressure values along the curve. Quadrilaterals are computed along the curve's spline and these are then anti-aliased. Various effects can be created by varying the parameters such as the grayscale value across the stroke, the amount of ink each bristle contains, and the pressure, which can vary bristle contact or spread the bristles. Texture mappings can also be applied, creating various types of effects. Rendering each stroke takes one to two minutes.

In 1991, Pham created another Sumi-e brush stroke system designed to be more intuitive for the user [Pha91]. Instead of specifying control points, most of which the curve does not pass through, the user chooses knots, which the curve does pass through. Usually, control points are used to specify B-spline curves, since the equations that define the curves depend on these points. Pham uses an inversion algorithm to convert the knots into control points for a curve. The system then draws a stroke centered about this curve by drawing quadrilaterals between offset curves until the user specified thickness is achieved. Transformations can be performed on the control points to create animations.

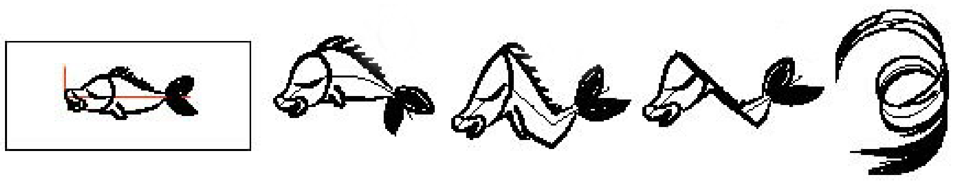
The Hsu and Lee created “skeletal strokes” [HL94], so called because a basic stroke unit is any arbitrary image which can be deformed by manipulating the backbone and thickness, shown in Figure 13c where the reference image is a fish and the backbone is curved, bent, and twisted. The strokes are expressive, because anchors can be affixed to different parts of the stroke, so that when the backbone is manipulated, some portions of the stroke stretch independently. Also, strokes can consist of substrokes. The cartoon in Figure 13b was created with skeletal strokes.



(a)



(b)



(c)

Figure 13: Images from two brush stroke programs: (a) Sumi-e style shrimp painting by Strassmann; (b) cartoon style drawing by Hsu and Lee using skeletal strokes; (c) a fish skeletal stroke and its backbone (in red) on the left and deformed using the backbone in the other images.

3.3 Physical Models

By composing collections of brush strokes, Haeberli and the scientists cited in Section 3.1 approach simulation as an artist might. Alternatively, some scientists address the problem by approximating the physical properties of the painting materials such as watercolor paint and paper as described in the first paragraph below and oil paint in the second.

Curtis et al. took this approach to simulate watercolor painting by modeling watercolor paper and the flow and interaction of multiple glazes of pigment mixed with water [CAS$^+$97]. They studied the makeup of watercolor paint and the techniques that watercolor artists use to create a model consisting of three layers: one inside the paper, called the "capillary layer," one just on top of the paper called the "pigment-deposition layer," and one just above the surface of the paper, the "shallow-water layer." The simulations are achieved by modeling the movement of water and pigment in the shallow-water layer and the transfer of pigment to the deposition and capillary layers. A variety of effects can be achieved with watercolor paint, depending on the wetness of the paper, how much pigment is on the brush, the composition of the paint, and other factors. Flow effects, can be seen on the pear, for example where, wet green paint was virtually applied to wet yellow paint already on the paper. Wet-in-wet makes the paint bleed. Figure 14a was created by their automatic watercolorization system. Another effect, edge darkening can be seen in the real watercolor in Figure 14b where a wet brush was applied to dry paint to create the dark edges on the potato.

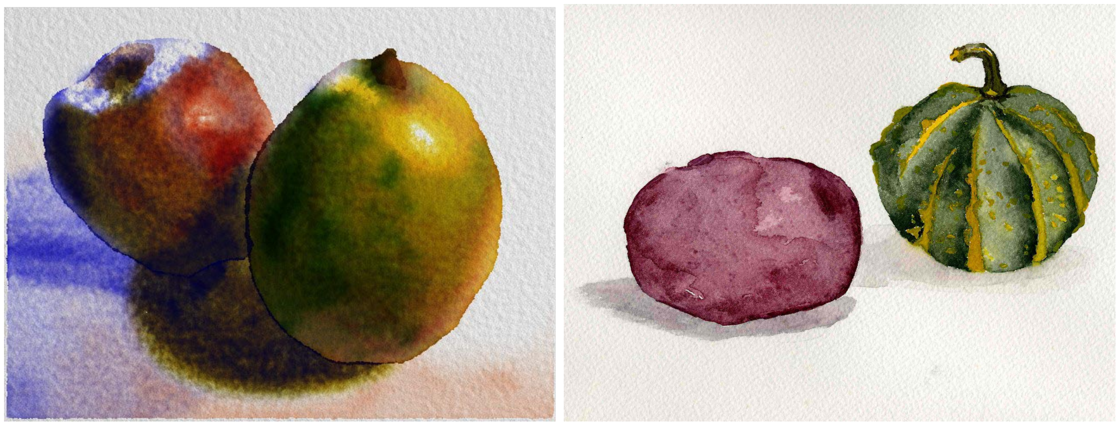

(a) (b)

Figure 14: Watercolor examples (a) automatic watercolorization (by Curtis et al.); (b) real watercolor created by Laura Tateosian.

Cockshott et al. model oil painting with three components: paint particles, an intelligent canvas, and a painting engine [CPE92], [Coc91]. Paint particles have attributes such as color and concentration. The canvas is a 2-dimensional array of cells which hold paint, like an egg carton. The cells are so-called because the ideas of cellular automata, large collections of finite automata, are the basis for the canvas. Each cell in the canvas has information about its attributes, such as its volume, its absorbency, and how much paint it contains. Since oil painting is often done on a canvas propped on an easel, gravity affects the outcome. In this system, if a cell is full, paint may flow from that cell to another. Which cell it flows into depends on that cell's preferred direction of flow. The paint engine controls the interaction between the paint particles and the canvas. The system deposits paint particles onto the canvas as a brush is moved across it. The results are somewhat unpredictable which is considered serendipitous, since accidents are an aspect of real painting.

3.4 Texture Synthesis

Texture can be perceived at different levels in a painting. Each of the following could be thought of as a texture in itself: the ridges left by a paint brush in the paint on a single brush stroke; the blades of grass painted in a field in a painting; or even, the entire painting itself. Texture synthesis is traditionally used to cover surfaces in repeating patterns (e.g., bricks, gravel). I'll begin discussing how scientists apply texture synthesis to simulation with the most intuitive of the three interpretations of textures described above, the texture on brush strokes, and follow with examples of the others.

Haeberli's "Paint by Numbers" system and the systems in Section 3.1 above use a texture mapped rectangle to simulate a brush stroke footprint. Haeberli and Segal proposed the use of texture maps as a fundamental drawing primitive and use texture maps to create air-brush strokes [HS93]. The article suggests that texture mapping can be used to create any conceivable brush stroke, even multi-colored ones.

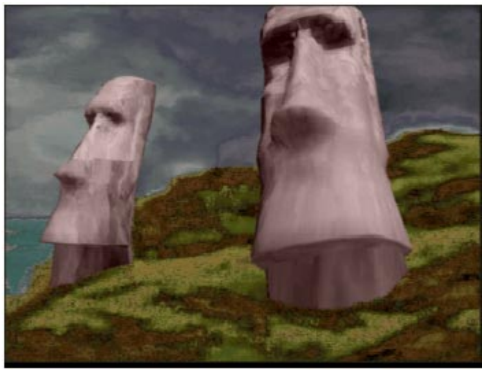


Figure 15: The texture in the sky, sea, and grass was generated using texture spectrums. The texture on the heads was created using Lewis' method for pre-existing texture.

A system that would be appropriate for applying the grass texture in the field is described by Lewis (in fact, see the grass texture in Figure 15) [Lew84]. The system is based on the fact that each texture has a corresponding frequency domain and the spectrum which visualizes the frequency domain is unique. An expression representing a texture can be converted to an expression representing its frequency domain using a Fourier transform and vice versa using an inverse Fourier transform. The author paints a spectrum and applies an inverse Fourier transform. The resulting texture sample convolved with sparse white noise produces a texture field. The convolution method can also be applied to a pre-existing texture to generate a random texture field. In order to do so, a window of the sample must be chosen so that it resembles the result of an inverse Fourier transform on a spectrum. To design a texture the user applies one of the methods above and specifies parameters of the noise. The textures can then be blended with the color or texture already on the surface. With some experience the user can create appealing paintings. The sky and hillside in Figure 15 textured with the spectrum method and the stone heads were textured with pre-existing textures (as seen on the left head, discontinuities can occur using this method).

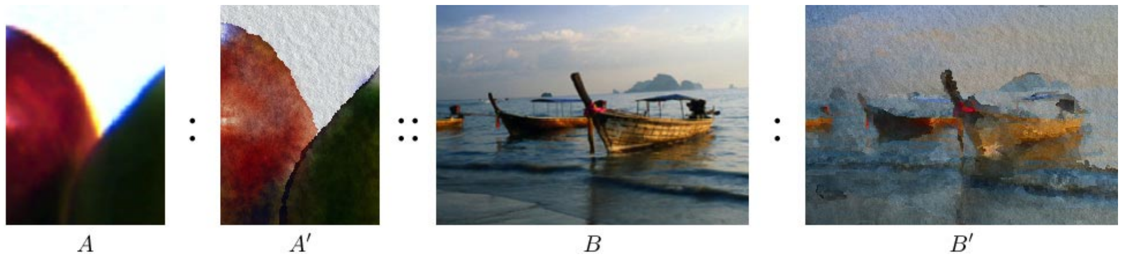


Figure 16: Hertzmann's image analogy system created B' based on B and the relationship between A and A', so that A is to A as B is to B'.

Hertzmann et al. view an entire painting as a texture [HJO+01]. Their system considers the relationship between an image and any filtered version of the image as a texturing. The driving concept is called an "image analogy", because given images A, A', and B, where A' is a filtered version of A, the system can generate B' such that A is to A' as B is to B'. For example, in Figure 16 the system was given A, a three-dimensional model and A', a watercolorization of A, and the photograph B, and it generated the watercolor version of B, B'. The system works by first creating different resolution versions of A, A', and B. Each pixel contains several channels of information, including RGB values and luminance, which are used in pixel comparison. The channel used in the comparisons depends on the style of filtering being simulated. At each level of resolution, the system takes images A, A', and B and the current version of B' and finds the pixel in the source pair that best matches the pixel being synthesized by inspecting its existing neighbors, the corresponding pixel in B' and that pixel's neighbors. Though a devoted system may be capable of generating an individual style more effectively, the image analogy is a general model capable of simulating oil painting, watercolor painting, traditional texture synthesis, embossing, or any arbitrary filter.

4 Nonphotorealistic Visualization

Visualization is an area in computer graphics, where scientists create images to represent collections of data. Challenges include choosing salient and compatible visual features and devising effective mappings from data attributes to visual features. Simulating artwork can be thought of as a kind of visualization, where a source image or model is being visualized in some stylized fashion. But scientific visualization is mostly concerned with visualizing large multivariate scientific datasets to facilitate analysis.

Scientists in visualization have begun using ideas from the field of nonphotorealistic rendering to enhance visualizations or to inspire new visualization techniques, creating nonphotorealistic visualizations (NPV). Artistic techniques, such as using abstraction to eliminate unimportant distractions, and sharpening details to draw attention to important areas, can help to convey information more effectively. Also, application of artist techniques may create more aesthetically pleasing images which may engage the viewer's attention and encourage extended exploration. To develop their visualizations, some scientists are turning to literature on fine arts techniques as well as nonphotorealistic rendering work or even visiting art galleries to observe master artwork.

David Laidlaw describes his observations from a field trip with his students to a gallery in [Lai01]. He remarks that paintings offer information at multiple scales; i.e., viewing painting from different distances yields different understanding at each viewpoint. A painting by Van Gogh, for example, when viewed from afar, appears as one overall texture, but closer viewpoints reveal the nuances of each brush stroke. He also notes that time has an affect on the viewer's perception of painting. Some features are obvious immediately, while other aspects may become apparent upon sustained observation. Laidlaw et al. applied artistic techniques to develop new mappings from data attributes to visual features. Examples are shown in Figures 17 and 18 and described below.

Figure 17 is the visualization of a mouse spinal cord. Laidlaw et al. apply the idea of multi-scale paintings to visualization [LAK$^+$98]. From a distance, a glance at the image gives the impression of a butterfly shape in the middle, surrounded by a darker area. The interior is the gray brain matter and the surrounding area is the white matter. On closer inspection, some of the elliptical shapes have a striped texture, a feature mapped to the rate of diffusion. To scientists studying the effects of Encephalomyelitis, a disease which attacks the white matter, the differentiation between white and gray matter is important and the visualization of multiple tensor image attributes in the same image can help them understand the relationship between these factors and the progress of the disease. The authors use the analogy of underpainting and brush strokes. The underpainting is a luminance image showing the overall structure of the anatomy. The brush strokes are "painted" on top of this and the visual features such as shape, color, transparency, orientation, and texture vary, just as in the painting programs described above. But here the visual features are representing the diffusion tensor data.

The brush stroke and underpainting analogies are also employed by Kirby, Marmanis, and Laidlaw to create the flow visualization shown in Figure 18 [KML99]. The image visualizes six data values at each point of an experimental two-dimensional flow past an airfoil. First a gray primer covers the surface. Then an underpainting, ranging from blue to yellow shows clockwise or counter clockwise vorticity (the rotation component of the flow). A layer of elliptical brush strokes with varying sizes and orientations is mapped to the deformation rate and direction. Transparency and texture of the ellipses both show the vorticity magnitude. Finally, a layer of arrow-shaped brush strokes are sized and directed to represent velocity magnitude and direction. The mapping is chosen to display velocity and vorticity prominently, while still showing deformation of the fluid elements simultaneously. In other words, the visualizations focus attention with visual cues, as an artist might do in a painting.

Interrante proposes another artistic approach to creating visualizations by employing the appeal of natural textures, such as weaving honey combs and woven cloth [Int00]. Interrante observes that natural textures lend themselves to displaying multiple attributes, because of their intrinsic variety, their visual appeal, and how humans receive them with less stress than other more traditional texture visualization patterns. Figure 19, a prototype example created by Interrante, visualizes US agricultural data with a weaving texture, a pebble texture, and color.

Skog et al. are focusing on creating visualizations to be used by the general public to display

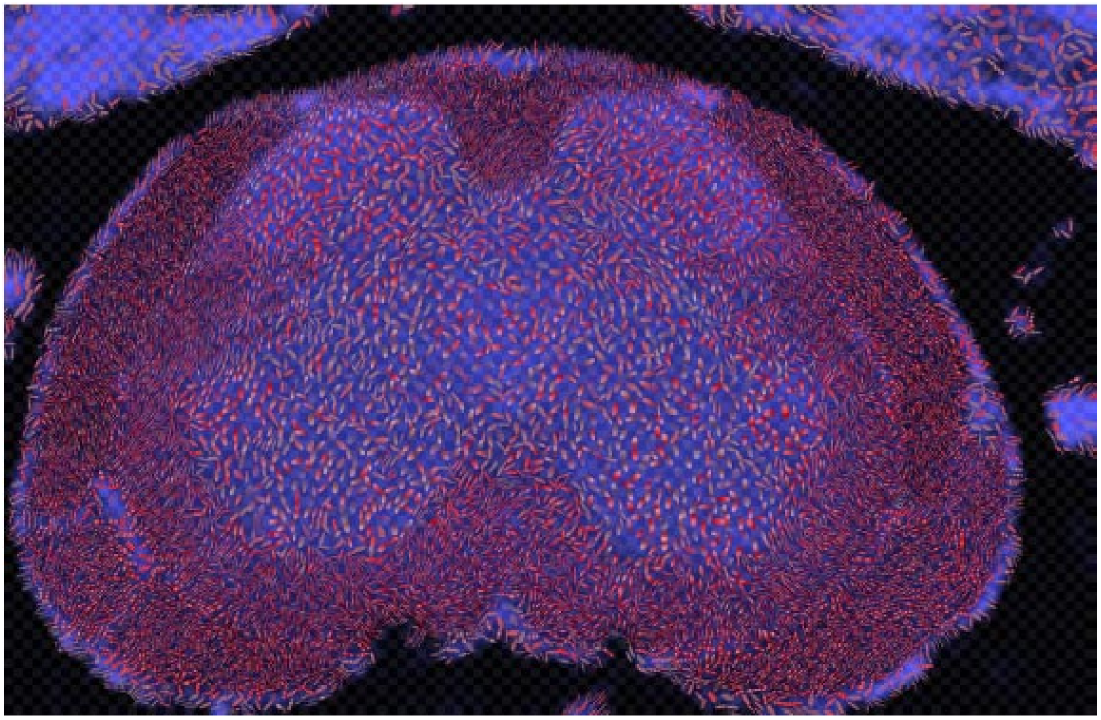


Figure 17: Diffusion tensor data from a healthy mouse spinal cord visualization from Laidlaw et al., visualizing anatomical image (underpainting lightness), ratio of largest to smallest eigenvalue (stroke length and width and transparency), principle direction in the horizontal plane (stroke direction), principle direction in the vertical plane (stroke red saturation), and magnitude diffusion rate (stroke texture frequency).

information in public places [SLH03]. The visualizations where inspired by the painting style of Dutch artist, Piet Mondrian, probably chosen for its sleek simplicity. As seen in Figure 20 the basic design uses blocks of color and lines to break up the canvas. Several applications were tested, including email, world weather, local forecast, and bus traffic. In the bus traffic example, shown in Figure 20, each bus was represented by a colored square. The position, size, and color of each square are mapped to bus schedule information. User studies indicated that the mappings may need to be revised, but the fundamental idea met with approval. Though users need to learn the mapping initially, as an alternative to the traditional timetable, visualizations such as this may more directly address users' demands for "information at a glance," as one user put it.

A volume visualization example was discussed in Section 2.4 that used a stippling technique. Rheingens and Ebert demonstrate another application of NPR to volume visualization. In their work we see that applying artistic techniques to volume visualization can be a very effective means of enhancing images so that various aspects can be discerned more readily [ER00] [RE01]. The result of mapping data values of a CT scan of an abdomen with a standard transfer function are shown in Figure 21a. Silhouette and boundary enhancements achieved by varying opacity depending on gradient magnitude, as in Figure 21b reveal the internal structure of the organs. Another effect created by decreasing the opacity of features oriented towards the viewer creates an outline sketch of the structure, seen in Figure 21c. Regional enhancement

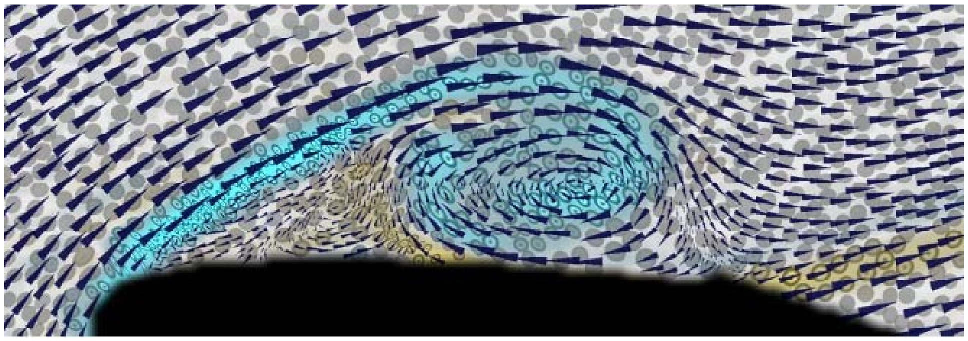


Figure 18: Experimental flow past an airfoil visualization from Kirby et al. visualizing velocity (arrow direction), speed (arrow area), vorticity (underpainting/ellipse color (blue=cw, yellow=ccw) and ellipse texture contrast), diffusion rate (log(ellipse radii)), divergence (ellipse area), and shear (ellipse eccentricity).

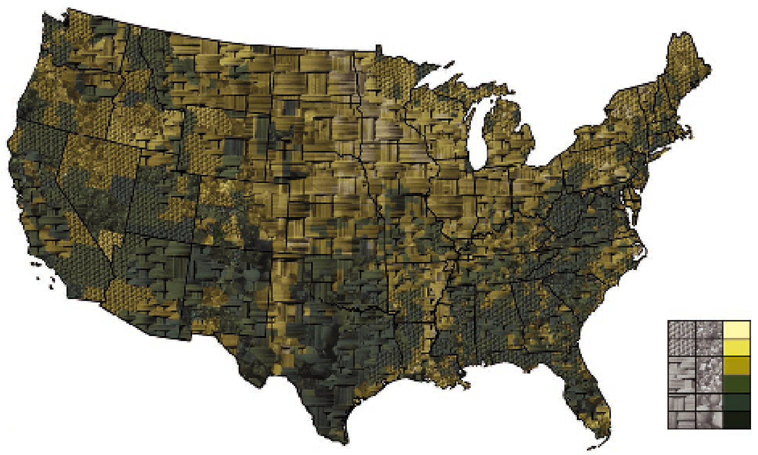


Figure 19: Interrante's agricultural map shows farms as a percentage of land (weaving texture), percent change in number of farms (pebble texture), average age of farm operators (color).

draws attention to the liver in Figure 21d.

In our lab we are also interested in how applying artistic techniques can enhance our visualizations [HETR04]. We created visualizations with visual features based on characteristics of Impressionistic artwork. In the resulting images the visual features of texture mapped brush strokes (size, orientation, and color) are mapped to attributes of large datasets. The placement of the strokes (coverage) is also based on dataset information. In Figure 22 a large multi-dimensional weather dataset is visualized with this system. In this image temperature is mapped to color (dark blue to bright pink for cold to hot), wind speed is mapped to coverage (low to high coverage for weak to strong), pressure is mapped to stroke size (small to large for low to high), wet day frequency is mapped to orientation (upright to flat for light to heavy), and precipitation is mapped to bright highlight strokes on a top layer. Studies showed that the visual features we derived from artistic techniques correspond to ones detected by the low-level human visual system, an encouraging sign for further research in applying artist techniques to visualization.

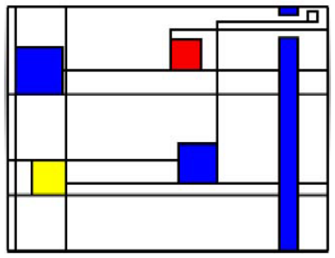


Figure 20: Contemporary art style display for public space (by Skog et al). Here each of the four colored squares represents a bus.

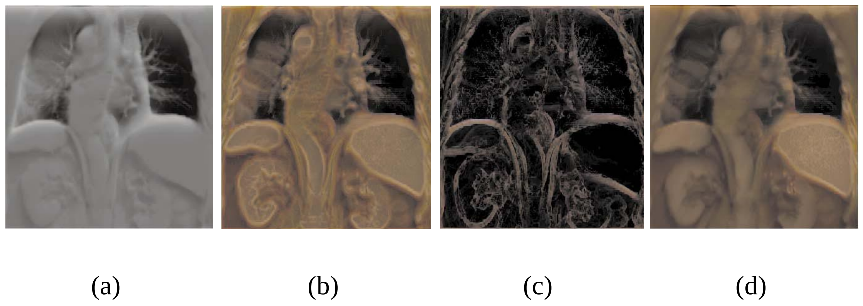


Figure 21: Abdominal CT scans: (a) standard CT visualization; (b) silhouette and boundary enhancements; (c) colored outline sketch techniques; (d) regional enhancement centered on the liver.

5 Conclusions

This survey of NPR demonstrates two general lessons about NPR. First, NPR has a myriad of potential uses. Why not just scan a real painting if artistic digital images are needed? The obvious answer (not everyone can create his own real artwork or come by artwork freely) is not the only one. This survey has described some systems that are designed to be used by artists, providing a new alternative tool for creative expression. Also, the less interactive systems offer an opportunity for less artistically inclined users to produce creative effects and be exposed to a creative realm that they might otherwise avoid. Making artistic images easy to create and share may increase society's creative literacy, and improves individuals' intellectual ability to operate in analytical and creative realms simultaneously. Finally, Section 4 provided one of the most concrete and compelling motivations for continued research in NPR: NPR may provide improved alternative visualization techniques.

Second, NPR is not as easy as it looks. Because artwork often simplifies and abstracts its subjects, a well-executed piece of artwork may lead to the assumption that because it looks harmonious and clean, the process of creating it was simple as well. A good artist makes it look simple. The same might be said of sophisticated nonphotorealistic rendering. But the complexity of the algorithms described in this paper contradicts this notion. A great deal of work goes into achieving an artistic feel, perhaps because some of the beauty and authenticity of artwork comes from imprecision and computers are relentlessly precise.

Musing on the future of NPR, an interesting implication is the resurgence of an intermingling of art and science. This may not be a new renaissance, but it seems that the influx of ideas

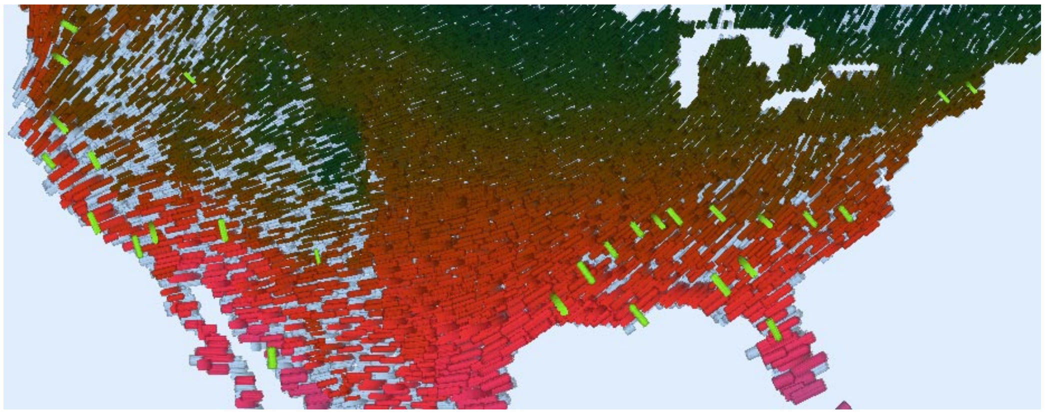


Figure 22: A visualization of January weather conditions in easter US, visualizing temperature with color (dark blue to bright pink for cold to hot), wind speed with coverage (low to high coverage for weak to strong), pressure with stroke size (small to large for low to high), wet day frequency with orientation (upright to flat for light to heavy), and precipitation with bright highlight strokes on a top layer.

from the artistic domains may prove to be a fertile source for understanding and invention in NPR, visualization, and possibly even in other areas of computer science. In the Renaissance, Leonardo Da Vinci possessed an understanding of both science and art and this seemed to improve his abilities in both; his understanding of anatomy, for example, enhanced his drawing abilities and visa versa.

As new artistic media and new uses of old media are discovered, new NPR techniques may follow. For example, contemporary artist, Andy Goldsworthy, creates stunning works of art from natural objects. As mentioned in Section 4, research is already being conducted on using natural textures in visualization. Natural textures and imagery, like Goldsworthy's, might inspire innovative styles in which NPR techniques are partnered with natural phenomenon simulation.

Alternatively, we may view NPR and NPV as a means of discovering new artistic possibilities. Generating images with computers opens the possibility of diverging from imitation and creating new forms of artwork not yet explored or perhaps not even possible with real materials because of physical limitations. Cockshot's painting model, for example, described in Section 3.3 allows for specification of different gravity directions on the same canvas. So strokes can be applied to two cells and the paint will run in two directions simultaneously. Free from the constraints of the physical world, the creative possibilities are boundless. Meanwhile, nonphotorealistic visualizations demonstrate the influence of art on visualization. Visualization may contribute to art the realm, as well. Work by contemporary artist Mark Lombardi, "Conspiracy Art," consists of large charts visualizing the relationships between currents events, such as the links between global finance and international terrorism. In summary, the intermingling of art and NPR or NPV seems to result in a symbiotic relationship.

In our lab we are interested in pursuing some of the open questions on the relationship between art and visualization. The work described in Section 4 has produced some appealing images and expanded the visual feature vocabulary of the visualization community. Our lab has conducted some studies that support the effectiveness of the visualizations we've created, as compared to traditional means of presenting this information. In comparison to a traditional weather map, a visualization like the one in Figure 22 was as effective and more effective in some measures. More studies are needed to compare the effectiveness of these visualizations to traditional glyph-based visualizations. Assuming that these prove to be equally accessible, in the sense that viewers can use them to detect answers to specific questions about the data accurately and efficiently with either visualization, some more interesting questions still remain.

Work in NPR is motivated by the underlying premise that artistic images have advantages over photorealistic images in some circumstances. This leads to the question: Does this premise extrapolate to the realm of visualization? Specifically, do nonphotorealistic visualizations have the advantages over traditional visualizations that we intuitively expect? For example, do they encourage extended exploration, enabling discovery? In fact, this may be a good starting point, since it is readily quantifiable. Then, if the answer is yes, we can follow the path of existing and future NPR research to make our images even more appealing. In our visualizations thus far we have used individually textured rectangles. Two obvious next steps could be using curved strokes and applying Hertzmann's method to create a more convincing oil paint-like overall texture [Her02]. If, on the other hand, the answer is no, we could pursue one of two possible avenues: 1) We could develop different nonphotorealistic visualization methods and iterate again. Do these methods have the same results? If so, we may begin to believe that the hypothesis was invalid and extended attendance is not an advantage of nonphotorealistic visualizations. 2) Otherwise, we might change our focus and ask, if there are other possible advantages of nonphotorealistic visualizations. This report began by making the point that scientists in NPR observe that artistic images have the capacity to "emphasize specific features of a scene, expose subtle attributes, and omit extraneous information. [GG01]" Intuitively, this application of abstraction and focus on detail could enhance visualizations of large datasets. An interesting study might involve a comparison of viewers' eye-movements when viewing the visualizations. We could determine where and when fixations occur on a nonphotorealistic visualization versus a traditional glyph-based visualization.

Of course, other important questions may arise as these questions are being answered. Regardless of whether or not the answers to any of these questions are affirmative, they will contribute to the body of knowledge in the visualization community and perhaps in the nonphotorealistic rendering community, as well.

References

[BSMM01] Bill Baxter, Vincent Scheib, C. Ming, Lin, and Dinesh Manocha. Dab: interactive haptic painting with 3d virtual brushes. In *SIGGRAPH 2001 Conference Proceedings,* pages 461–468, 2001.

[CAS$^+$97] C. J. Curtis, S. E. Anderson, J. E. Seims, K. W. Fleischer, and D. H. Salesin. Computer-generated watercolor. In Turner Whitted, editor, *SIGGRAPH 97 Conference Proceedings,* pages 421–430, Los Angeles, California, 1997.

[Coc91] Tunde Cockshott. Wet and sticky: A novel for computer based painting. *PhD Thesis, Computer Science Department Research Report 91/R20*, 1991.

[CPE92] Tunde Cockshott, John Patterson, and David England. Modeling the texture of paint. *Computer Graphics Forum (Proceedings Eurographics '92),* 11(3):217–226, 1992.

[DS03] Doug DeCarlo and Anthony Santella. Stylization and abstraction of photographs. In *SIGGRAPH 2002 Conference Proceedings*, pages 769–776, 2003.

[ER00] David Ebert and Penny Rheingans. Volume illustration: Non-photorealistic rendering of volume models. In *Proceedings Visualization 2000,* pages 195–202, San Francisco, California, 2000.

[GCS02] Bruce Gooch, Greg Coombe, and Peter Shirley. Artistic vision: Painterly rendering using computer vision techniques. In *Proceedings NPAR 2002 Symposium on Non-Photorealistic Animation and Rendering*, pages 83–90, Annecy, France, 2002.

[GG01] Bruch Gooch and Amy Gooch. *Non-Photorealistic Rendering*. A K Peters, Ltd., Natick, Massachusetts, 2001.

[GR02] Gevorg Grigoryan and Penny Rheingans. Probabilistic surfaces: Point based primitives to show uncertainty. In *Proceedings Visualization 2002*, pages 147–153, 2002.

[Hae90] Paul Haeberli. Paint by numbers: Abstract image representations. *Computer Graphics (SIGGRAPH 90 Conference Proceedings)*, 24(4):207–214, 1990.

[Her98] Aaron Hertzmann. Painterly rendering with curved brush strokes of multiple sizes. In Michael Cohen, editor, *SIGGRAPH 98 Conference Proceedings*, pages 453–460, Orlando, Florida, 1998.

[Her02] Aaron Hertzmann. Fast texture maps. In *Proceedings NPAR 2002 Symposium on Non-Photorealistic Animation and Rendering*, pages 91–96, Annecy, France, 2002.

[HETR04] Christopher Healey, J. Enns, L. Tateosian, and M. Rempel. Perceptually-based brush strokes for nonphotorealistic visualization. *ACM Transactions on Graphics*, 2004.

[HJO$^+$01] Aaron Hertzmann, Charles E. Jacobs, Nuria Oliver, Brian Curless, and David H. Salesin. Image analogies. In Eugene Fiume, editor, *SIGGRAPH 2001 Conference Proceedings*, pages 327–340, Los Angeles, California, 2001.

[HL94] S. C. Hsu and I. H. H. Lee. Drawing and animation using skeletal strokes. In Andrew Glassner, editor, *SIGGRAPH 94 Conference Proceedings*, pages 109–118, Orlando, Florida, 1994.

[HS93] Paul Haeberli and Mark Segal. Texture mapping as a fundamental drawing primative. In Michael Cohen, Claude Puech, and Francois Sillion, editors, *Fourth Eurographics Workshop on Rendering*, pages 259–266, Paris, France, 1993.

[Int00] Victoria Interrante. Harnessing natural textures for multivariate visualization. *IEEE Computer Graphics & Applications*, 20(6):6–11, 2000.

[JEGPO02] Pierre-Marc Jodoin, Emric Epstein, Martin Granger-Piche, and Victor Ostromoukhov. Hatching by example: A statistical approach. In *Proceedings NPAR 2002 Symposium on Non-Photorealistic Animation and Rendering*, pages 29–36, Annecy, France, 2002.

[KML99] R. M. Kirby, H. Marmanis, and D. H. Laidlaw. Visualizing multivalued data from 2D incompressible flows using concepts from painting. In *Proceedings Visualization '99*, pages 333–340, San Francisco, California, 1999.

[KP02] Junhuan Kim and Fabio Pellacini. Jigsaw image mosaic. In *SIGGRAPH 2002 Conference Proceedings*, pages 657–664, 2002.

[Lai01] David H. Laidlaw. Loose, artistic "textures" for visualization. *IEEE Computer Graphics & Applications*, 21(2):6–9, 2001.

[LAK$^+$98] David H. Laidlaw, Eric T. Ahrens, David Kremers, Matthew J. Avalos, Russell E. Jacobs, and Carol Readhead. Visualizing diffusion tensor images of the mouse spinal cord. In *Proceedings Visualization '98*, pages 127–134, Research Triangle Park, North Carolina, 1998.

[Lew84] John-Peter Lewis. Texture synthesis for digital painting. *Computer Graphics (SIGGRAPH 84 Proceedings)*, 18(3):245–252, 1984.

[Lit97] Peter Litwinowicz. Processing images and video for an impressionist effect. In Turner Whitted, editor, *SIGGRAPH 97 Conference Proceedings*, pages 407–414, Los Angeles, California, 1997.

[LME$^+$02] Aidong Lu, Christopher J. Morris, David S. Ebert, Penny Rheingans, and Charles Hansen. Non-photorealistic volume rendering using stippling techniques. In *Proceedings Visualization 2002*, pages 211–218, Boston, Massachusetts, 2002.

[LS95] John Landsdown and S. Schofield. Expressive rendering: A review of nonphotorealistic techniques. *IEEE Computer Graphics & Applications*, 15(3):29–37, 1995.

[Mei96] Barbara J. Meier. Painterly rendering for animation. In Holly Rushmeier, editor, *SIGGRAPH 96 Conference Proceedings*, pages 477–484, New Orleans, Louisiana, 1996.

[Pha91] Binh Pham. Expressive brush strokes. *Computer Vision, Graphics and Image Processing*, 53(1):1–6, 1991.

[RE01] Penny Rheingans and David Ebert. Volume illustration: Nonphotorealistic rendering of volume models. *IEEE Transactions on Visualization and Computer Graphics*, 7(3):253–264, 2001.

[SABS94] Mike Salisbury, Sean E. Anderson, Ronen Barzel, and David H. Salesin. Interactive pen-and-ink illustrations. In Andrew S. Glassner, editor, *SIGGRAPH 94 Conference Proceedings*, pages 101–108, Orlando, Florida, 1994.

[SALS96] Mike Salisbury, Corin Anderson, Dani Lischinski, and David H. Salesin. Scale-dependent reproduction of pen-and-ink illustrations. In Holly Rushmeier, editor, *SIGGRAPH 96 Conference Proceedings*, pages 461–468, New Orleans, Louisiana, 1996.

[SD02] Anthony Santella and David DeCarlo. Abstracted painterly renderings using eye-tracking. In *Proceedings NPAR 2002 Symposium on Non-Photorealistic Animation and Rendering*, pages 75–83, 2002.

[Sec02] Adrian Secord. Weighted voronoi stippling. In *Proceedings NPAR 2002 Symposium on Non-Photorealistic Animation and Rendering*, pages 37–43, Annecy, France, 2002.

[SLH03] T. Skog, S. Ljungblad, and L.E. Holmquist. Between aesthetics and utility: Designing ambient information visualization. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 233–240, 2003.

[SS02] Thoman Strothotte and Stefan Schlechtweg. *Non-Photorealistic Computer Graphics: Modeling, Rendering and Animation*. Morgan Kaufmann Publishers, Inc., San Francisco, California, 2002.

[Str86] Steve Strassmann. Hairy brushes. *Computer Graphics (SIGGRAPH 86 Proceedings)*, 20(4):185–194, 1986.

[SWHS97] Mike Salisbury, Michael T. Wong, John F. Hughes, and David H. Salesin. Orientable textures for image-based pen-and-ink illustration. In Turner Whitted, editor, *SIGGRAPH 97 Conference Proceedings*, pages 401–406, Los Angeles, California, 1997.

[SY99] Michio Shiraishi and Yasushi Yamaguchi. Image moment-based stroke placement. In Richard Kidd, editor, *SIGGRAPH 99 Sketches & Applications*, page 247, Los Angeles, California, 1999.

[SY00] Michio Shiraishi and Yasushi Yamaguchi. An algorithm for automatic painterly rendering based on local source image approximation. In *Proceedings NPAR 2000 Symposium on Non-Photorealistic Animation and Rendering*, pages 53–58, 2000.

[WS94] Georges Winkenbach and David H. Salesin. Computer-generated pen-and-ink illustration. In Andrew Glassner, editor, *SIGGRAPH 94 Conference Proceedings*, pages 91–100, Orlando, Florida, 1994.

[WS96] Georges Winkenbach and David H. Salesin. Rendering parametric surfaces in pen-and-ink. In Holly Rushmeier, editor, *SIGGRAPH 96 Conference Proceedings*, pages 469–476, New Orleans, Louisiana, 1996.